



WP3 – ENABLE: Creating LSD Enabler Tools

## D3.6: Industrial Symbiosis platform



This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 873468.

## Document Information

<b>Grant Agreement Number</b>	873468	<b>Acronym</b>	DigiCirc	
<b>Full Title</b>	European cluster-led accelerator for digitization of the circular economy across key emerging sectors			
<b>Start Date</b>	1 <sup>st</sup> May 2020	<b>Duration</b>	32 months	
<b>Project URL</b>	<a href="https://digicirc.eu/">https://digicirc.eu/</a>			
<b>Deliverable</b>	D 3.6 – Industrial Symbiosis platform			
<b>Work Package</b>	WP 3 - ENABLE: Creating LSD Enabler Tools			
<b>Date of Delivery</b>	<b>Contractual</b>	31 <sup>st</sup> January 2021	<b>Actual</b>	15 <sup>th</sup> February 2021
<b>Nature</b>	Report	<b>Dissemination Level</b>	Public	
<b>Lead Beneficiary</b>	CLMS			
<b>Responsible Author</b>	Konstantinos Skianis CLMS			
<b>Contributions from</b>	Yiannis Zorgios CLMS, Vasilis Vasilopoulos CLMS, Theodoros Kalampoukis CLMS, Konstantinos Skianis CLMS			

## Document History

<b>Version</b>	<b>Issue Date</b>	<b>Stage</b>	<b>Description</b>	<b>Contributor</b>
0.1	01/11/2020	Draft	TOC	Konstantinos Skianis CLMS
0.2	31/12/2020	Draft	Architecture, components	Konstantinos Skianis CLMS
0.3	15/01/2021	Draft	Technologies	Konstantinos Skianis CLMS
0.4	30/01/2021	Final	Functionalities and screenshots	Konstantinos Skianis CLMS

## Disclaimer

Any dissemination of results reflects only the author's view and the European Commission is not responsible for any use that may be made of the information it contains.

## Copyright message

© DigiCirc Consortium, 2020

This deliverable contains original unpublished work except where clearly indicated otherwise. Acknowledgement of previously published material and of the work of others has been made through appropriate citation, quotation or both. Reproduction is authorised provided the source is acknowledged.



# Table of Contents

List of acronyms .....	5
1 Executive summary.....	6
2 Introduction.....	7
3 Industrial Symbiosis .....	8
3.1 Main themes of Industrial Symbiosis.....	8
3.2 Business Environment .....	9
4 Operational Characteristics & Requirements for the Industrial Symbiosis tool.....	11
4.1 Operational Characteristics .....	11
4.1.1 Robust, scalable, enterprise-level architecture.....	11
4.1.2 User-friendly and easy to use functionalities.....	11
4.1.3 Built-in standards-based security.....	12
4.1.4 Administration tools.....	12
4.2 Requirements .....	12
4.2.1 Technology requirements .....	13
4.2.2 User requirements .....	13
4.2.2.1 Overall vision.....	13
4.2.2.2 Resources and technologies .....	13
4.2.2.3 Knowledge portal.....	14
4.2.2.4 User registration .....	14
4.2.2.5 Characterization of resources.....	14
4.2.2.6 Case study production .....	14
4.2.3 Resources Management .....	15
5 Technologies.....	16
5.1 Ontologies .....	16
5.2 Databases .....	18
5.3 Knowledge graphs .....	18
5.3.1 Key Characteristics .....	19
5.3.2 Ontologies and Formal Semantics.....	20
5.3.3 Advantages of a knowledge graph .....	20
6 Industrial Symbiosis tool .....	22
6.1 Development platform .....	22
6.1.1 Fast, modern & native apps .....	22
6.1.2 Rich & responsive UI .....	22
6.1.3 Quick integration with APIs.....	23
6.1.4 Built-in security .....	23



## D3.6: Industrial Symbiosis tool

6.1.5 Events.....	24
6.1.6 Comprehensive cache management.....	24
6.1.7 Painless localization .....	24
6.1.8 Unlimited extensibility .....	25
6.2 System architecture .....	25
6.3 Components.....	26
6.3.1 User Interface .....	26
6.3.2 Data layer & resources.....	27
6.3.2.1 The semantic web.....	27
6.3.2.2 Ontologies.....	28
6.3.3 Material Flow Knowledge Graph.....	31
6.4 Users of the Industrial Symbiosis tool .....	32
6.5 Workflow.....	33
7 Web platform functionalities.....	34
7.1 User registration and login.....	34
7.2 Material flow explorer.....	37
7.3 Resources and Processes.....	39
8 Use Cases.....	41
8.1 Use case 1.....	41
8.2 Use case 2.....	41
8.3 Use case 3.....	41
8.4 Use case 3.....	41
9 Conclusion .....	42
10 Bibliography.....	43
11 Appendix – Development documentation .....	44



## Table of Figures

Figure 1: Traditional Supply Chain .....	9
Figure 2: Extended Supply Chain, involving waste treatment.....	10
Figure 3: the Industrial Symbiosis concepts ontology, taken by the eSymbiosis platform. ....	17
Figure 4: an example of a knowledge graph.....	19
Figure 5: The Industrial Symbiosis Platform architecture.....	26
Figure 6: Resources classification. ....	29
Figure 7: Classification of resources by Type. ....	29
Figure 8: Classification of Organic Materials.....	30
Figure 9: An integrated example of a resources ontology. ....	30
Figure 10: an illustration of the exploration process on the IS knowledge graph.....	32
Figure 11: form for the registration process of a user.....	36
Figure 12: user login UI component.....	37
Figure 13: the starting page of the industrial symbiosis tool, with the material flow component. ....	37
Figure 14: the starting page of the industrial symbiosis tool, viewed with an Administrator account. ....	38
Figure 15: selecting and visualizing resources via the knowledge graph. ....	38
Figure 16: another sample of the industrial symbiosis material flow knowledge graph. ....	39
Figure 17: the component which allows for showing the existing resources and adding new ones. ....	39
Figure 18: adding a new material in the industrial symbiosis platform. ....	40
Figure 19: the industrial symbiosis component responsible for showing existing processes and adding new ones..	40
Figure 20: adding a new process for the newly added resource. ....	40



# List of acronyms

**Table 1: acronyms used in the report.**

Acronym	Designation
API	Application Programming Interface
CE	Circular Economy
IS	Industrial Symbiosis
GIS	Geographic Information System
GUI	Graphical User Interface
KPI	Key Performance Indicator
LSD	Large scale demonstration
OGC	Open Geospatial Consortium
RTO	Regional & Technology Organization
SME	Small and Medium Enterprise
UI	User Interface
US	User Story
WMS	Web Map Service



# 1 Executive summary

Industrial Symbiosis (IS) is an innovative environmental practice of the circular economy (CE) that is helping companies to trade resources or waste as feedstock, building communities that are environmentally integrated and efficient. Industrial Symbiosis is market-oriented, creating new businesses and jobs, and saving costs in raw materials and discharge fees. IS also reduces waste across the wide range of industrial activity that includes chemicals, metals, plastic, biomass, electronics, but also municipal waste.

The DigiCirc project aims to boost the circular economy using digital tools, by supporting innovative Small and Medium Enterprises (SMEs) in the development and marketing of solutions based on circular value chains through 3 acceleration programs on the following themes: "Circular City", "Blue economy" and "Bioeconomy". 45 consortia involving the collaboration between 'quadruple-helix' stakeholders (local/regional authorities including permitting authorities, big industry actors, SMEs, Regional & Technology Organization (RTOs), academia, civil society, etc.) will be selected through open calls. DigiCirc offers the consortia four DigiCirc Digital Tools to support them in the development, demonstration and commercialization of their solutions. One of the developed tools is the Industrial Symbiosis tool, which gathers and stores all necessary data on participating companies and their associated resources, accessed via live web information portals. This deliverable is focused on presenting and describing in detail all the functionalities of the Industrial Symbiosis tool.

The intention of the Industrial Symbiosis tool (link: <http://digicirc.clms.io/KnowledgeHub/Index>) is to develop a web-based application which enables users, companies and clusters of entities around the European Union (EU), to participate in industrial symbiosis activities which will improve resource efficiency across the economy. The project particularly aims to engage SMEs, which are conspicuously lacking from participating in circular economy environments.

The Industrial symbiosis tool serves as a centralized gateway of information for interested parties. The tool's main functionality is to allow users to investigate and interact with resources and associated processes or technologies, in an Industrial Symbiosis environment. All the registered resources and processes are stored following a state-of-the-art knowledge graph architecture, which is the key component of the Industrial Symbiosis tool. The knowledge graph acts as a resource/material knowledge database, connecting materials with processes via a graph representation. Inside the graph, nodes represent resources/materials and they can be connected via edges, which represent processes or technologies. Thus, the tool allows the users to view information about material flow; how a material can be transformed to another material via a specific process or technology. The material flow explorer also allows for searching both ends of the flow, meaning that the tool gives the ability to query either the available paths starting or ending from/to a specific material. Additionally, users are able to suggest new materials or processes for extending the tool, and afterwards a user with elevated rights (Administrator or Expert) may add the new knowledge to the system. Moreover, the newly added resources can carry additional information like type, physical form, unit of measurement, description and hazardness. Finally, the processes also allow for information like metadata, reference or any useful notes. The implementation of the Industrial Symbiosis tool can be accessed via a public Github repository: <https://github.com/CLMSUK/DigicircPlatform>.

This deliverable gives an overview of DigiCirc's Industrial Symbiosis tool components, including its architecture, used technologies and ontologies, developed knowledge databases, and designed components. The remaining parts of the deliverable provide in detail the Industrial Symbiosis functionalities, following specific user requirements, via a simple and easy to use web graphical interface. Finally, the deliverable reports use cases that cover the majority of the needs of the participating actors, towards a fully Industrial Symbiosis integration.



## 2 Introduction

The DigiCirc project aims to develop digital platforms to streamline data analysis and processing, and provide relevant datasets of real-world test beds for each accelerator. As such, it allows for beneficiaries to gain a better understanding of material dynamics underlying their businesses, test business hypotheses “in silica” and develop applications to manage logistics and organization in a complex environment – thus lowering learning curves and providing an entry point for further technologies to be implemented and integrated. In order to achieve these goals, a critical digital tool is developed within the project, the Industrial Symbiosis tool.

The Industrial Symbiosis tool (link: <http://digicirc.clms.io/KnowledgeHub/Index>) serves as a centralized gateway for interested parties to find, track and investigate resources flow in a circular economy (CE) environment. The tool is powered by semantic models like ontologies, to model, track and project availability and movement of materials, covering waste and raw materials, products, by-products and processing technologies. The models are used to infer new knowledge, hence allowing the discovery of otherwise ‘missed’ opportunities and they will enable the offering of a flexible ‘matchmaking’ service for CE by allowing also partial matching between waste materials and technologies, hence ameliorating the potential of successful CE collaborations.

The Industrial Symbiosis tool is the component of the DigiCirc project, responsible for organizing the material/resource and processes’ knowledge in a knowledge database.

The tool’s main functionality is to allow users to investigate and interact with resources and associated processes or technologies, in an Industrial Symbiosis environment. All the registered resources and processes are stored following a state-of-the-art knowledge graph architecture, which is the key component of the Industrial Symbiosis tool. The knowledge graph acts as a resource/material knowledge database, connecting materials with processes via a graph representation. Inside the graph, nodes represent resources/materials and they can be connected via edges, which represent processes or technologies. Thus, the tool allows the users to view information about material flow; how a material can be transformed to another material via a specific process or technology. The material flow explorer also allows for searching both ends of the flow, meaning that the tool gives the ability to query either the available paths starting or ending from/to a specific material. Additionally, users are able to suggest new materials or processes for extending the tool, and afterwards a user with elevated rights (Administrator or Expert) may add the new knowledge to the system. Finally, the newly added resources can carry additional information like type, physical form, unit of measurement, description and hazardness. The implementation of the Industrial Symbiosis tool can be accessed via a public Github repository: <https://github.com/CLMSUK/DigicircPlatform>.

This report is organized in the following way. First, we present an overview of a Circular Economy environment, where Industrial Symbiosis consists one of the most popular practices, including a brief introduction to the business and market properties, connected to Industrial Symbiosis. Next, we describe the operational characteristics and requirements of the Industrial Symbiosis tool. Afterwards, we show the required technologies that we used for developing the tool. The next section presents the design and implementation of the Industrial Symbiosis tool in detail. Next, comes the presentation of the tool’s functionalities, with specific use cases that prove the value for the participating parties. Finally, we conclude with a summarized overview of the developed solutions, including useful remarks.



## 3 Industrial Symbiosis

The Industrial Symbiosis tool is aiming to boost the circular economy (CE) using novel data processing approaches, by supporting innovative SMEs in the development and marketing of solutions based on circular value chains on the themes of "Circular City", "Blue economy" and "Bio-economy".

### 3.1 Main themes of Industrial Symbiosis

The main sectors of the Blue Economy include ocean fisheries, tourism, transportation, marine energy, seabed mining for minerals and blue carbon sequestration by restoring mangroves and seagrasses. Restoring, protecting and maintaining the diversity, productivity and resilience of marine ecosystems, will require clean technologies as well as latest technologies such as Robotics, Nanotechnology, Digitalization and Artificial Intelligence (AI) etc.; renewable energy; and circular material flows. Even in the "Circular Economy", the concept of "Sustainability" is important and relevant. In a circular economy, all the economic activities build or rebuild the overall health of the system. It recognizes the importance of the economy needing to work effectively at all scales – for large and small businesses, for organizations and individuals, globally and locally. Thus, the local community is common in both the Blue Economy as well as the Circular Economy. The Circular Economy represents a systemic shift that builds long-term resilience, generates business and economic opportunities, and provides environmental and societal benefits. Hence, it is evident that Sustainability is central to both types of economies.

A circular city, in a short definition, is one that eliminates waste, keeps goods and their ingredients in use and regenerates natural systems. A circular city names new possibilities for designing, planning, manufacturing and accessing goods, buildings and vehicles, and keeping materials in use. This can involve more distributed ways of managing resources, including exchanging or renting goods instead of buying them. It involves reverse logistics to bring materials back into consumer flows instead of banishing them as waste. Ideally, businesses and consumers ultimately will discard nothing and share more, at no cost to their convenience and liberty.

Similarly to circular cities, bio-economy and blue economy refers to the smoother running of our economy with respect to the protection and preservation of our environment on land and sea respectively. These two areas offer as well many possibilities of exploiting circular economy platforms.

Blue Economy includes the Circular Economy but goes beyond it. Both are about radical resource productivity, zero waste and Sustainability. Blue Economy is more comprehensive as it shifts away from core business/core competence that force companies to focus on one industry by considering local economic development as a priority, ensuring that local purchasing power increases and more money circulates regionally. This enables growth without inflation through an increase in local production of goods and services. Blue Economy focuses on the notion that nothing is waste; and it is guided by 23 explicit principles framed by Gunter Pauli. It takes into consideration Global Progress Indicators (GPI) rather than Gross Domestic Product (GDP) which attempts to factor in the degradation of the environment. Blue Economy also emphasizes the "innovativeness" (all inspired by nature with zero waste) to create jobs along with efficiency. There are Models of Success and Sustainability (MOSS) as reflected in Vortex Processing Technology (VPT) - A multi-application Blue Economy Innovation; along with Industrial Vortex Generators (IVG) which change the properties of water crystallizing line particles, see more air bubbles etc.; and the "blow down" water can be re-used without treatment a second time before going to the sewage.

The Industrial Symbiosis tool represents a significant step in the direction of an interconnected circular economy in all the aforementioned areas. It does not only support the exploitation of data for the development of services and products answering varied users' needs and market opportunities in emerging industries of the affected domains (circular cities, bio-economy, blue-economy), it also effectively fosters the emergence of a new Ecosystem of actors across multiple sectors, nurturing their potential to innovate through cross-border/cross-sectoral collaborations



and strengthening their circular business proposition (through access to key resources and expertise) amid strong, global competition. In that regard, one of the main strategic impacts, is a solid contribution to the establishment of new value chains bustling with innovative SMEs that can translate the need to reuse extant resources and materials into competitive and valuable commercial products/applications/services, for the benefit of the economy, the society and the environment. Information technology and use of data can create the required transparency on flows and availability of materials and the associated financial costs. This elevated awareness will become the foundation of fact-based trust – making CE the most profitable business model of the 21st century. The circular economy is beginning to take off primarily because it makes business sense — but also because it engages communities, the government and municipalities as well.

Industrial Symbiosis (IS) is an innovative approach, originating from a Circular Economy environment, that brings together companies from all business sectors with the aim of improving cross industry resource efficiency through the commercial trading of materials, energy and water and sharing assets, logistics and expertise. Industrial Symbiosis is helping companies to trade waste as feedstock, building communities that are environmentally integrated and efficient.

## 3.2 Business Environment

Industrial Symbiosis is widely seen as one of the most effective business concepts and means for mitigating the pollution problems treating waste as feedstock and increasing the company revenue, and eventually observing the natural resources in the planet.

To this end, Industrial Symbiosis (IS) is an innovative environmental practice that is helping companies to trade resources and waste as feedstock, building communities that are environmentally integrated and efficient. The IS enterprises will perform in a modified business environment where win-win partnerships are formed based on the IS ‘waste’ supply chain.

The traditional supply chain is defined as an integrated manufacturing process wherein raw materials are manufactured into final products, then delivered to customers (via distribution, retail, or both). Figure 1 below illustrates the structure of the traditional supply chain.

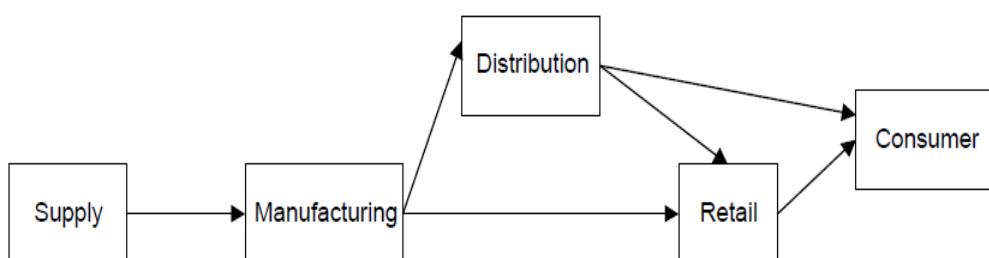


Figure 1: Traditional Supply Chain.

The extended supply chain (Beamon 1999) contains all of the elements of the traditional supply chain, but extends the one-way chain to construct a semi-closed loop that includes product and packaging recycling, re-use, and/or remanufacturing operations.

Figure 2 illustrates the extended supply chain. The traditional supply chain links as solid lines, and the links corresponding to the extended supply chain as dashed lines. The W's enclosed by diamonds represent waste (or disposed) materials.



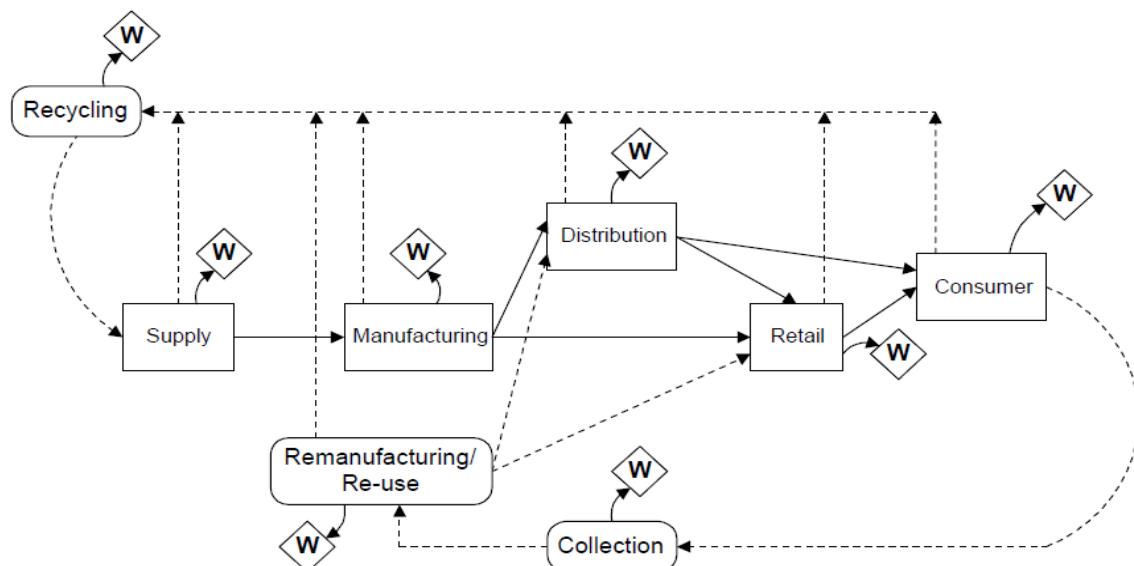
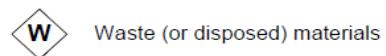
Legend:

Figure 2: Extended Supply Chain, involving waste treatment.

The extended supply chain, while powerful, is considered to be an internal method of an entity for waste treatment and recycling. It is clear that this a single entity and its expertise and facilities are limiting this process. Following the idea of the extended supply chain, Industrial Symbiosis aims to recycle, reuse and exploit wastes as far as possible, involving external parties as well. Living in a digital era, the easiest way to achieve that is by creating web tools that will enable this collaboration between partners.

The DIGICIRC project is aiming to accumulate the Industrial Symbiosis business concept and reach the above goals by:

- Reviewing resources/material streams and industrial cases, in order to develop knowledge models capable of capturing and processing knowledge in an IS environment
- Developing a web platform to enable access to the aforementioned resources and models (via ontologies, knowledge graph-based semantic web services and visualization tools).

# 4 Operational Characteristics & Requirements for the Industrial Symbiosis tool

The Industrial Symbiosis tool is aiming to meet the business objectives of a large number of companies, especially targeting on the procurement functions and enhancing their activities by letting them transform material and waste flows to profit, by increasing their efficiency in managing the industrial waste and remaining resources.

The participating businesses should be assisted with IS focused functionality and decision support tools to determine profitable strategies for sourcing of raw materials, manage distribution of goods, plan and optimize the flow of materials - based on real-time constraints such as capacity, material, and resource availability.

All the above results, within an Industrial Symbiosis environment, and with the use of the developed tool, will find immediate application in dealing with remaining resources or industrial waste, as this exchange enables transformations from liability to asset, taking the characteristics of utilizable raw material.

The resulting operational environment is considered to be, multi-tiered, and sophisticated.

In this part of the deliverable, we first present the main operational characteristics of the Industrial Symbiosis tool. Next, having already described the characteristics, we continue by presenting the requirements gathered for the Industrial Symbiosis tool.

## 4.1 Operational Characteristics

### 4.1.1 Robust, scalable, enterprise-level architecture

The architecture of the Industrial Symbiosis tool is robust and scalable so that it can meet the demands of increased traffic. Robust scalability requires well-planned architectural partitioning that anticipates where major system demands will occur and dedicates appropriate resources to manage those demands.

The Industrial Symbiosis solution aims to use the latest technologies including server virtualization, so that the environment will be constantly monitored, flexible and ‘tailor-made’ to adapt to the constantly changing requirements of the Industrial Symbiosis participants (users) in terms of sizes and functionality.

### 4.1.2 User-friendly and easy to use functionalities

The end-user experience should be easy to customize and brand while not limiting the flexibility for designing the look and feel of the system. It should provide control over:

- Site look and feel
- Showing and searching resources/materials
- Resources categorization



The administrator should be able to modify many of these elements using straightforward tools. These attributes are fundamental to ensuring that the company maintains complete control over the marketplace and all of its functionality.

#### 4.1.3 Built-in standards-based security

The Industrial Symbiosis platform will take all necessary provisions to allow for encrypted communications that use industry standards such as Secure Sockets Layer (SSL), Secure HTTP, and digital certificates, as is common to all Public Portals holding critical information for third party entities.

Additionally, a flexible security model for assigning user rights is required, to allow a system administrator to set security for individual users and groups, allowing different levels of access to the system. Security levels should allow limiting access to specific areas of the site or restrict certain features. This type of security model enables an administrator to develop highly focused groups of markets and users which help eliminate channel conflict and allows various personalized functions to be implemented, e.g. effective targeted marketing to end-users.

All sensitive data and information, along with passwords should be stored in an encrypted database, making it impossible for an intruder to steal them.

#### 4.1.4 Administration tools

It is required that the platform will provide a complete set of tools for data administration. The tools should be full-featured but easy to use and allow remote access of the system.

The administration tools should permit retrieval and maintenance of user information, such as permission status information.

A complete set of reporting tools for generating information on current status of resources and material flow are provisioned.

## 4.2 Requirements

We begin by declaring some general requirements:

- It is important to have the ability to modify the data entry form – to hide fields, make them redundant, add new fields and so forth.
- The ability to change the way a field behaves is also particularly important, in that it offers a measure of potential control over data quality.
- The ability to flag resources as ‘Confidential’ is essential: this means that the resources are managed by a Practitioner, rather than appearing in the publicly available system for management through the normal automated process.
- As a general principle, the ability to delete data/information should be available to all users, regarding their own data.



Next, we present requirements, which apply to specific areas of interest for the Industrial Symbiosis platform.

## 4.2.1 Technology requirements

- Conventional RDMS (e.g., SQL Server) and communication technologies are also employed for storing the all the necessary ingredients of the industrial symbiosis tool.
- The database structure is critical to the success of Industrial Symbiosis platform - the quality of information coming out of the system can only ever be as good as that being entered. The use of dropdowns with fixed choices is therefore essential wherever possible, the skill being in making sure that the drop-downs are well-researched and thorough, without being overbearing or complicated to use. A 'data dictionary' would perhaps support and firm up data entry standards where they rely on free text entry.
- The material knowledge graph needs to be designed in order to be interactive and user-friendly. For the knowledge graph we have used neo4j, which is a state-of-the-art library for developing knowledge graphs.

## 4.2.2 User requirements

The following sections will list the identified user requirements. Following prototyping and testing/evaluation with users the requirements will be updated. The aim and interaction of each user with the Industrial Symbiosis web service is different for the different stages in the Industrial Symbiosis workflow. Therefore four groups of global user requirement have been identified. Next, we list the user requirements for the six stages in the Industrial Symbiosis workflow.

### 4.2.2.1 Overall vision

1. Visitors to the Industrial Symbiosis web site should be able to learn about the concept of Industrial Symbiosis.
2. Secure login-driven user portal, one-stop-shop for information and business opportunities based on stated business resources / feedstock requirements.
3. Ability to carry out 'self-service' Industrial Symbiosis using user-friendly synergy management screens and search facilities, the companies either operate the entire process themselves or with minimum Practitioner assistance.
4. Ability to receive news that matches the technological profile of a Member
5. Evident Practitioner support with contact details to assigned Practitioner on the Member first page after log-in.
6. Simple documents and contact handling.

### 4.2.2.2 Resources and technologies

1. Ability for Members to browse the taxonomy and classification of resources or technologies stored in the Industrial Symbiosis system.
2. Relevant Members should be automatically informed when a new resource or technology matching their search criteria is added to the Industrial Symbiosis tool.
3. The Industrial Symbiosis system should be able to learn what type of resources or technologies are of interest to a Member based on click-count and recommend resources accordingly. In other words, a



Member's click-behavior may differ from their stated 'wants' but perhaps gives some insight into other items of interest.

#### 4.2.2.3 Knowledge portal

1. Links to industry sites and reports from e.g. Resource Recovery Forum, Lets Recycle, Edie, etc. These are assigned to networks of Members by the Practitioner to make them appear on the relevant Member's home page
2. Links to academic/research sites and reports. These are assigned to networks of Members by the Practitioner to make them appear on the relevant Member's home page.
3. Ability for Member to upload documents and link to information.
4. Search facilities.

#### 4.2.2.4 User registration

1. Simple procedure to register to become Member
2. Easy to fill in company information in the system
3. Clear explanation for every field of every company data form that needs to be filled in (online help or off-line document)
4. Ability to mark company information as "confidential", this would mean that they will only be visible to the Practitioner
5. Ability to update profile information
6. Feedback of successful member registration
7. Ability to modify forms and tables

#### 4.2.2.5 Characterization of resources

1. Easy to fill in resource or technology information in the system
2. Clear explanation for every field of every resource or technology form that needs to be filled in (online help or off-line document)
3. Ability to upload data sheets, pictures, etc., relevant to a resource or technology
4. Ability to mark resource or technology information as "confidential", this would mean that they will only be visible to the Practitioner
5. Ability to mark resource as 'haves and 'wants'
6. Ability to update and modify resource or technology information
7. Feedback of successful resource or technology registration
8. Ability to modify forms and tables
9. The ontology should include Member resources and technologies

#### 4.2.2.6 Case study production

1. Ability to customize case study online templates
2. Ability to import data from Industrial Symbiosis system to prepare online case study report [PR]



3. Ability to export data from Industrial Symbiosis system to prepare case study reports in various document formats

#### 4.2.3 Resources Management

An elevated user (Administrator or expert):

- 1) Should be able to add a new resource
- 2) Can define resource composition.
- 3) Can define related resources.
- 4) Can define new processes.
- 5) Can define the potential users to whom its resource is addressed.



# 5 Technologies

In this section we present the necessary technologies, which were exploited for developing the Industrial Symbiosis tool and its functionalities.

## 5.1 Ontologies

Ontologies can be considered as the backbone of the Industrial Symbiosis tool. Ontology is a description of the concepts and relationships that constitute a domain of knowledge as this is understood by a group of people related to this domain. The concepts (classes) that describe the domain are organized into a hierarchical structure (taxonomy) and the properties are used in order to indicate the interrelations among concepts (object properties) or to link them with certain values (Data Properties).

An ontology provides a common vocabulary for a specific domain of knowledge and it also provides the relations among the terms of this vocabulary. Ontologies can be shared and reused. The information that is stored in an ontology can also be used for reasoning purposes (machine interpretation) and lead to the extraction of new knowledge. Industrial Symbiosis includes the knowledge models (ontologies) and the registration portal. In Industrial Symbiosis ontologies will formally represent knowledge as a set of concepts within the IS domain, and the relationships between those concepts. Ontologies will be used for reasoning about the IS entities and resources and will be used to assist many Industrial Symbiosis functions.

The ontologies can serve as a "database" of the information about different types of waste and their respective technologies as well as their properties. The user has the ability to navigate through the ontology in order to classify the relevant waste/technology. The registration portal (ontology interpreter) is an "engine" that will facilitate this navigation and will guide the user through the whole process.

It needs to be stressed at this point, that ontologies are not the goal of the project, but can be seen as a technology facilitator. Although ontologies are considered as one "database" of the Industrial Symbiosis tool, they have to offer much more than a conventional database. Some of their advantages are the flexibility in using synonyms which can help eliminating jargon barriers and avoid misunderstandings. Another advantage of the use of ontologies is the fact that it is much easier to change the represented knowledge compared to a conventional database which would require more time and effort.

Finally, a great advantage is the fact that the information stored in an ontology is machine interpretable which means that it can be read by machines/computers and used for reasoning. This way it is easier to extract more relevant information from the user (as well as information he might not know about) and provide a better and more accurate services.

In parallel domain ontologies existing semantic web service frameworks will be used for describing the profile of industries registered with the platform. Using a semantic description allows the automation of service discovery which is the possible synergies between industries by matching the semantics of their functionalities and non-functional attributes as well the input and output.

The level of granularity at which the service is described is essential in defining the flexibility of the platform. A too detailed description would greatly reduce flexibility of providers in describing the services and closes the scope of functionality while a too high level description although giving flexibility, creates major difficulties in the matching and discovery process. It is therefore essential to create balance for the level of description based on specification of the IS platform.



### D3.6: Industrial Symbiosis tool

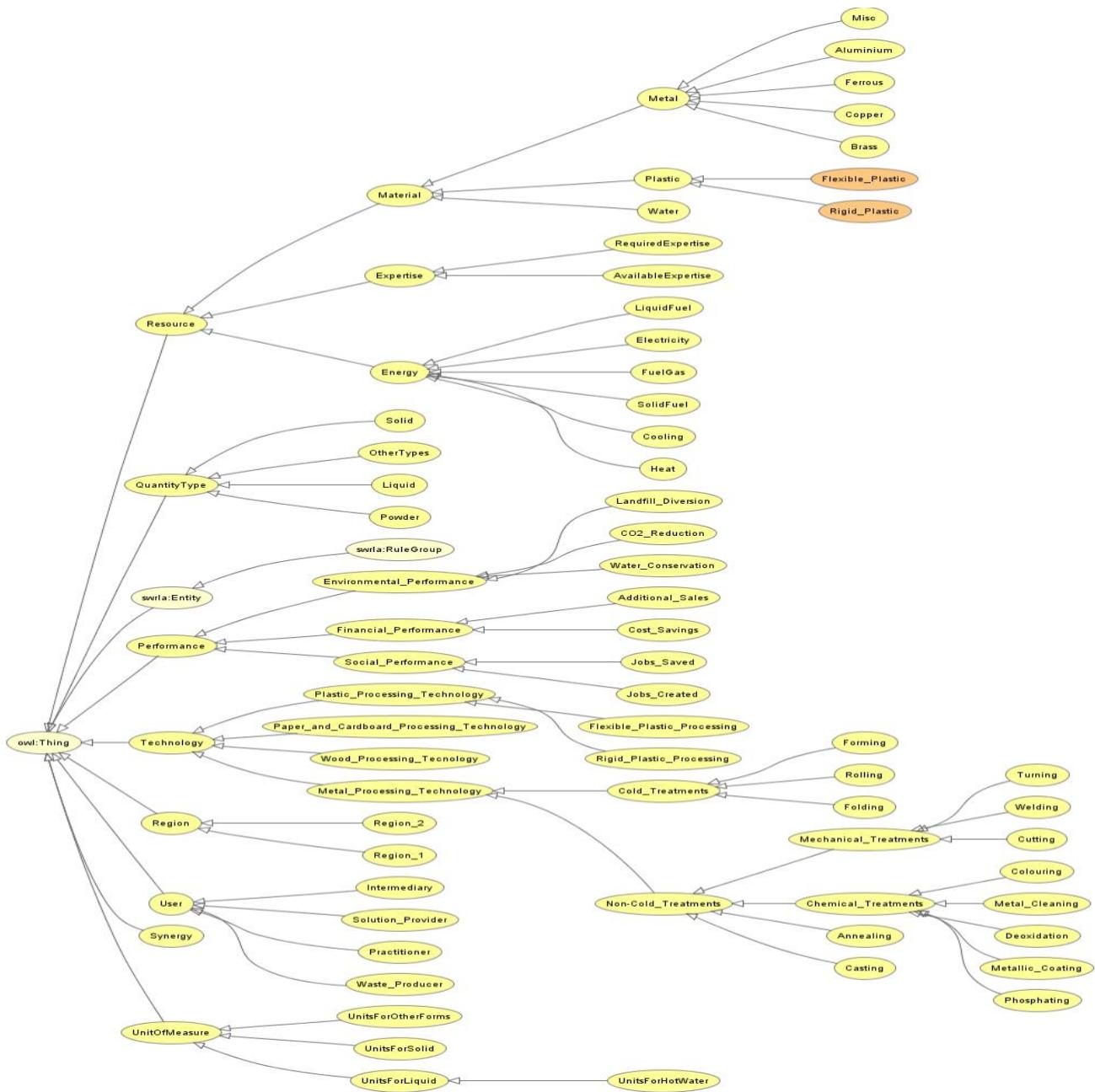


Figure 3: the Industrial Symbiosis concepts ontology, taken by the eSymbiosis platform.

The above diagram (

Figure 3) demonstrates the structure of a domain relevant ontology to IS, as used in the eSymbiosis<sup>1</sup> project. As described before, the ontology will be used as a guide for the navigation of the user. The diagram demonstrates the basic structure of the IS ontology includes some general concepts (user, resource, technology etc.) which are being further detailed as the user is navigated through the hierarchical structure. For simplicity, the diagram does not include the properties and other features that facilitate most of the functions that the ontology offers.

In the above paradigm, there are four types of user (Intermediary, Waste Producer, Technology Provider, Practitioner) and three different types of resources (Material, Expertise and Energy).

<sup>1</sup> <http://www.esymbiosis.gr/site/>

A very rough example describing a basic navigation scenario would be the following: The user enters the portal and chooses the type of user that is more appropriate for his case.

According to this choice the user will be guided to the resource or the technology concept where he will have to classify in detail the resource/technology he can offer. The resource/technology of the user could be classified in more than one class in order to facilitate the matchmaking. (i.e. a plastic water bottle can be classified as a plastic bottle, as PET plastic waste or as flexible plastic).

## 5.2 Databases

The data layer is where the data repositories with information of the participating companies, their inputs and outputs and knowledge database on feasible matches between output/wastes and input/resources are located.

The database subsystem is used to store all the data of the system. The database considered in order to fully meet the requirements is MariaDB<sup>2</sup>. MariaDB is a leading open-source document database, which is built on a distributed, scale-out architecture and handles transactional, operational, and analytical workloads at scale. This database allows for enhancements to the document model, more specifically supports the storage of data in almost any structure, and each field – even those deeply nested in subdocuments and arrays – can be indexed and efficiently searched. Elements can be added dynamically to the document model and the query engine, to handle all types of data tags. This expands the type of queries and analytics that can be performed by the users. These powerful features make MariaDB a good fit for the purposes of this tool that will handle a great amount of various data that will undergo further processing and be presented as a data catalogue.

## 5.3 Knowledge graphs

The knowledge graph (KG) represents a collection of interlinked descriptions of entities – real-world objects and events, or abstract concepts (e.g., documents) – where:

Descriptions have formal semantics that allow both people and computers to process them in an efficient and unambiguous manner;

Entity descriptions contribute to one another, forming a network (or graph), where each entity represents part of the description of the entities, related to it, and provides context for their interpretation.

Figure 4 illustrates an example of a knowledge graph, dealing with resources that are coming from paper. Circular nodes represent resources, while edges represent properties associated to a specific resource. For example, the node labeled “paper”, which is located in the center of the knowledge graph, comes with many available properties. One such property, which is tied to an edge, is “broader”. This edge points to the node “processed material”, telling us that the node “paper” belongs in the family of “processed material”. If we examine the knowledge graph further, we can find additional interesting properties for a variety of resources. This example shows us the power of a knowledge graph, enabling us to create and store multiple and diverse types of resources and associated properties, as well as processes and technologies.

---

<sup>2</sup> <https://mariadb.org/>



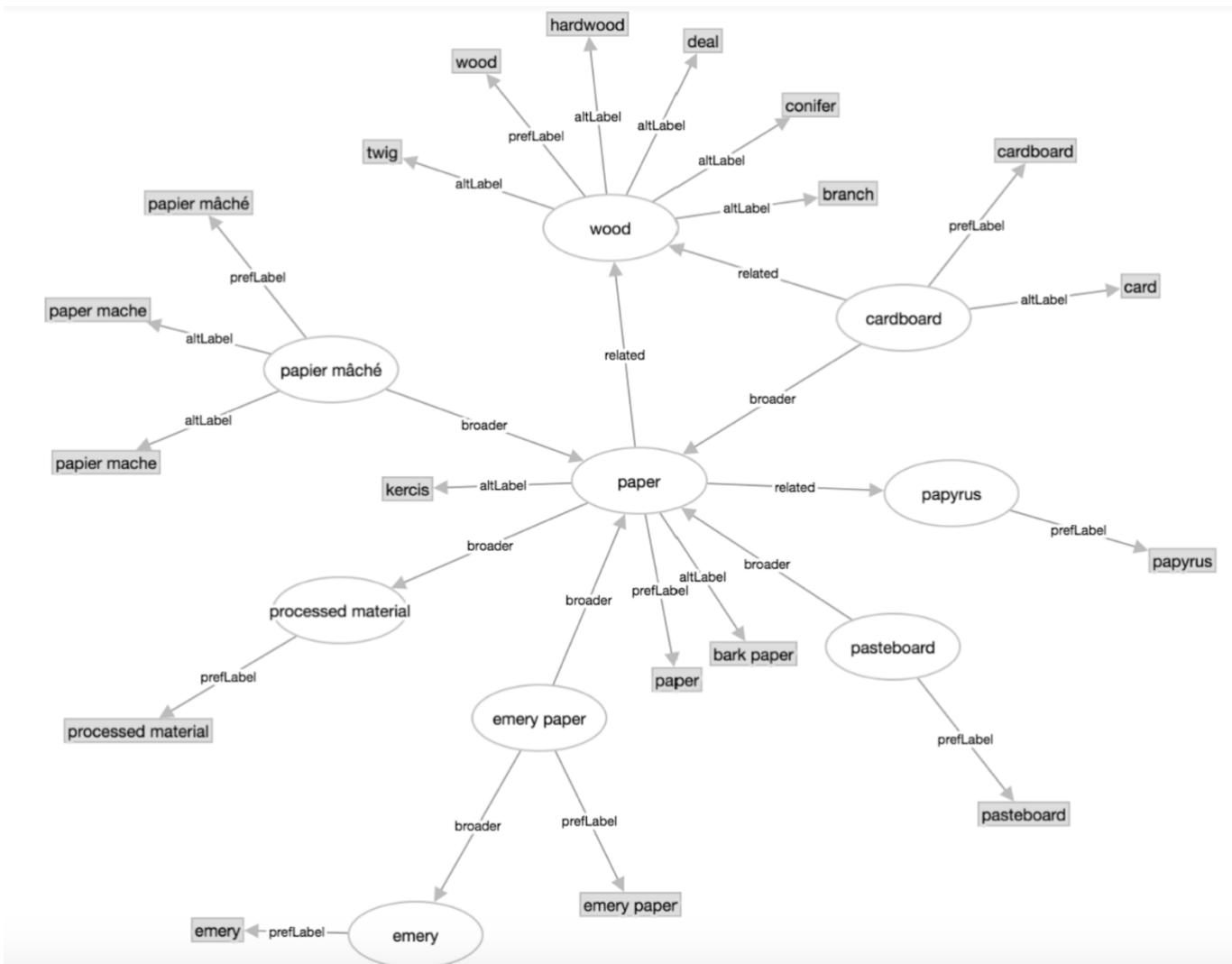


Figure 4: an example of a knowledge graph.

### 5.3.1 Key Characteristics

Knowledge graphs combine characteristics of several data management paradigms:

- **Database**, because the data can be explored via structured queries;
- **Graph**, because they can be analyzed as any other network data structure;
- **Knowledge base**, because they bear formal semantics, which can be used to interpret the data and infer new facts.

Knowledge graphs, represented in RDF, provide the best framework for data integration, unification, linking and reuse, because they combine:

- **Expressivity**: The standards in the Semantic Web stack – RDF(S) and OWL – allow for a fluent representation of various types of data and content: data schema, taxonomies and vocabularies, all sorts of metadata, reference and master data. The RDF\* extension makes it easy to model provenance and other structured metadata.
- **Performance**: All the specifications have been thought out, and proven in practice, to allow for efficient management of graphs of billions of facts and properties.

- **Interoperability:** There is a range of specifications for data serialization, access (SPARQL Protocol for end-points), management (SPARQL Graph Store) and federation. The use of globally unique identifiers facilitates data integration and publishing.
- **Standardization:** All the above is standardized through the W3C community process, to make sure that the requirements of different actors are satisfied – all the way from logicians to enterprise data management professionals and system operations teams.

### 5.3.2 Ontologies and Formal Semantics

Ontologies represent the backbone of the formal semantics of a knowledge graph. They can be seen as the data schema of the graph. They serve as a formal contract between the developers of the knowledge graph and its users regarding the meaning of the data in it. A user could be another human being or a software application that wants to interpret the data in a reliable and precise way. Ontologies ensure a shared understanding of the data and its meanings.

When formal semantics are used to express and interpret the data of a knowledge graph, there are a number of representation and modeling instruments:

- **Classes.** Most often an entity description contains a classification of the entity with respect to a class hierarchy. For instance, when dealing with business information there could be classes *Person*, *Organization* and *Location*. Persons and organizations can have a common superclass *Agent*. Location usually has numerous sub-classes, e.g., *Country*, *Populated place*, *City*, etc. The notion of class is borrowed by the object-oriented design, where each entity usually belongs to exactly one class.
- **Relationship types.** The relationships between entities are usually tagged with types, which provide information about the nature of the relationship, e.g., *friend*, *relative*, *competitor*, etc. Relationship types can also have formal definitions, e.g., that *parent-of* is inverse relation of *child-of*, they both are special cases of *relative-of*, which is a symmetric relationship. Or defining that *sub-region* and *subsidiary* are transitive relationships.
- **Categories.** An entity can be associated with categories, which describe some aspect of its semantics, e.g., “*Big four consultants*” or “*XIX century composers*”. A book can belong simultaneously to all these categories: “*Books about Africa*”, “*Bestseller*”, “*Books by Italian authors*”, “*Books for kids*”, etc. The categories are described and ordered into taxonomy.
- **Free text descriptions.** Often a ‘human-friendly text’ description is provided to further clarify design intentions for the entity and improve search.

### 5.3.3 Advantages of a knowledge graph

The backend of a knowledge graph offers multiple advantages:

- Scalable data processing
- Easy-to-use interface
- High-performance querying and analytics
- Built-in inferencing and custom services
- Standard connectors for a variety of data formats
- Single server, embedded mode, high availability, and scale out

Creating a knowledge graph allows also for:



- Curation and interlinking of data from heterogeneous sources
- Collaborative management and authoring
- Custom query and templates catalogs
- Data annotation
- Capturing of provenance information

A knowledge graph enables:

- Rapid development of end-user oriented applications
- Web components for end user friendly presentation and interaction
- Interactive visualization
- Rich semantic search with visual query construction and faceting
- Customizable semantic clipboard



# 6 Industrial Symbiosis tool

Having established the requirements and used technologies, we proceed with the presentation of the DigiCirc Industrial Symbiosis tool.

## 6.1 Development platform

The platform for designing and implementing all the functionalities of the Industrial Symbiosis tool is ZappDev<sup>3</sup>. zAppDev is a development platform created by CLMS. zAppDev offers powerful web application development in the cloud, by designing, building, running and deploying custom applications. Furthermore, it offers an easy development experience by speeding up application development, eliminate operational costs, while increasing efficiency and innovation. In this subsection, we briefly describe the key characteristics of zAppDev, which made the platform ideal for developing the Industrial Symbiosis tool.

### 6.1.1 Fast, modern & native apps

With a single-click, all business semantics can turn into high-quality, standardized, consistent code, with compilation templates that are always aligned with state-of-the-art technologies:

- Application Frameworks
- Industry Best Practices
- Design Standards and Patterns
- Open-Source and Public Frameworks and Libraries

The generated code and artifacts are then linked, compiled and built, producing a fully functional Application ready to be downloaded or Deployed on any server.

The generated Applications are ready to be used with zero ongoing commitment, running independently of zAppDev, providing full ownership and control over both the Solution and the Source Code for:

- Review
- Customization
- Rapid Prototyping

### 6.1.2 Rich & responsive UI

zAppDev comes with a comprehensive set of visual, data and model-aware Components that include anything required to create an Application, from every-day Form elements to all-inclusive, customizable components such as:

- Data Lists
- Picklists
- Charts

---

<sup>3</sup> <http://www.zappdev.com/>



- Maps
- Calendars

that can be brought with drag-and-drop onto the UI.

Additionally, a UI can be easily drafted with a set of pre-designed Form Templates, simply by dragging and dropping items defined by your Models and Data-Sources, or even create ready-to-go, fully operational Forms and sets of Pages with a Single Click, just by defining the Business Objects that are required to be represented.

Without ever leaving this integrated Visual Designer, a user may add any basic or advanced functionality to the used Forms by:

- Setting the Actions to be performed for every component and user interaction, either from scratch or by using a set of pre-defined processes
- Adding Rules, Restrictions, Validations, Conditional UI Formattings and Calculation at any level
- Handling Events and their subsequent representation in pages

### 6.1.3 Quick integration with APIs

Application development in zAppDev is service oriented by design. Connect to external systems and services in a matter of minutes and expose a modern REST API to the world with a few clicks.

EXTERNAL APIS:

- Connect with REST and SOAP Services
- Import structures defined in XSD format
- Write custom SQL Queries against the application's database
- Execute scripts against local or remote network entities
- Create automated transformations between heterogeneous data representations

EXPOSED APIS:

- Define secure REST APIs for your application
- Control their access with Permissions and Audit Trailing
- Increase their speed with fully customizable caching mechanisms
- Make instant use of the auto-generated Data Contracts
- Expose the fully implemented CRUD Operations of any Business Object with a single click

### 6.1.4 Built-in security

zAppDev offers you a wide range of out-of-the-box capabilities to ensure security, compliance and reliability. Create your bullet proof Application in a matter of a clicks by

- Restricting your Application Processes with Access Permissions
- Defining your Application's User Roles
- Easily assigning Permissions to Roles
- Monitoring the access to your Application, at any level, with Audit Trails and Logging



- Defining any Encryption Strategy to secure your Data
- Selecting your preferred Authentication Type and in a matter of seconds

## 6.1.5 Events

When developing a modern, dynamic Application enhanced with real-time events and procedures, the mixing of GUI and business logic can result to a cluttered, unmaintainable solution. zAppDev hides the underlying complexity and produces clean, modular, model-based solutions tackling the design and engineering of event handlers. So, whether you wish to allow your users to chat with each other, or pause an Order until your Sales Department issues a Confirmation, you can

- Easily create Application-wide Events
- Set their Operation representing your business processes and logic
- Assign their Target to specific Application Users or Groups
- Define them as synchronous (blocking) or asynchronous (parallel)
- Raise them anytime and anywhere you want in your Application

so that later, in any Form you wish you can

- Simply select the Event you want to Listen to
- Define its listening User or Group
- Set up your GUI to react accordingly

## 6.1.6 Comprehensive cache management

zAppDev lets you define caching strategy for your application's Services with a few clicks in an easy-to-use visual way. Every single operation defined in External or Exposed services can be thoroughly tweaked regarding its cache. It may follow the globally defined strategy, or apply to specific precise options, including separate cache for each individual end-user.

Cache Expiration Modes available:

- Sliding
- Absolute
- Custom Function

## 6.1.7 Painless localization

The translation of a zAppDev application is an easy task. The integrated localization tools hide technicalities and allow quick translation, to multiple languages, of every text resource used across the application:

- Global resources
- Domain-related resources
- Form-specific resources

End-users of your application can customize their profile by choosing their preferred language and locale settings.



## 6.1.8 Unlimited extensibility

zAppDev is an open development platform, compared to a proprietary development environment or runtime engine. The user has full ownership over the generated source code which can also be downloaded, modified and deployed without any dependency on zAppDev. The platform was developed with extensibility in mind. Native and JavaScript libraries can be imported and work flawlessly. Furthermore, any custom UI can be created using the HTML Form editor in combination with the zAppDev JS API which allows comprehensive client-side interaction with given applications models.

## 6.2 System architecture

The role of the Industrial Symbiosis tool is to bring together waste producers/owners with possible users of the materials that could be recovered, upon some treatment. In this subsection we shall present a web based industrial symbiosis platform of circular economy for classifying, use and exploiting the industrial residue in a specific sector of three major areas: Circular cities, Blue economy, and Bioeconomy.

The main functionality of the Industrial Symbiosis platform, especially as regards the Knowledge Based aspects of it, will be delivered in the form of web services fully accessible on the Internet which will allow the Business Users, i.e. the waste stream engaged members, together with the IS practitioners to register their interest and to describe their provisions. The Industrial Symbiosis platform is providing users the following functions:

- Registration: the resources' providers are guided by the waste knowledge model (ontology) to provide details necessary for the full description: waste composition, availability and anticipated availability, geographical location etc.
- Knowledge Model Maintenance: Industrial Symbiosis register respective datasets describing / characterizing waste streams.
- Technology Registration: providers are guided by the technology knowledge model to help users to provide necessary for the full description: technology description, technological, economic and environmental characteristics, geographical location etc.
- Communications: Industrial Symbiosis supports partners' information exchange. The process may be guided / monitored by IS actors.
- Operational Support: once the symbiotic chain is set and operating, all the participants are able to introduce: quantities and type processed, volume of outputs, timing and changes, specific benefits and shortfalls, which may be further processed by Industrial Symbiosis services to compute relevant metrics for reporting.

The above functions are Knowledge driven, and are supported through core technology components of Industrial Symbiosis, which are Ontology based. The Ontology Models involve:

- Technology and Intermediary Classification to build a deeper understanding of the IS trade patterns and to lead the efficiency improvements of the IS network and enable the processing of the waste streams.
- Waste Classification handling the complexities of the significant volumes of data collected enabling the systematization of the analysis.

The main advantage and differentiator of the Industrial Symbiosis platform will be the built-in capabilities for the intelligent processing of resources. This is based on the Ontology Models which are driving Industrial Symbiosis functions through the deployment of configurable web services. Therefore, the Semantically Enriched Web Services are building upon an SWS Framework that schematizes the service design as the specification of a set of layers that cover all relevant Industrial Symbiosis service features involving all Service Descriptions and provisioning all Service



Requests arriving during the User Registration processes as well as during all User interactions through the Industrial Symbiosis Portal.

First, we describe the designed architecture of the DIGICIRC platform, which includes both the Industrial Symbiosis and the Matchmaking tools. The architecture contains three main layers: the User Interface, the Data layer, and the Logic-processing layer. More specifically, the web-based application developed, contains the following components which are also presented in detail in the next parts of this report:

- the company data input view
- the users' management
- the materials management (ontology, knowledge graph building)
- the matching algorithm
- the reporting tools
- the matching tool

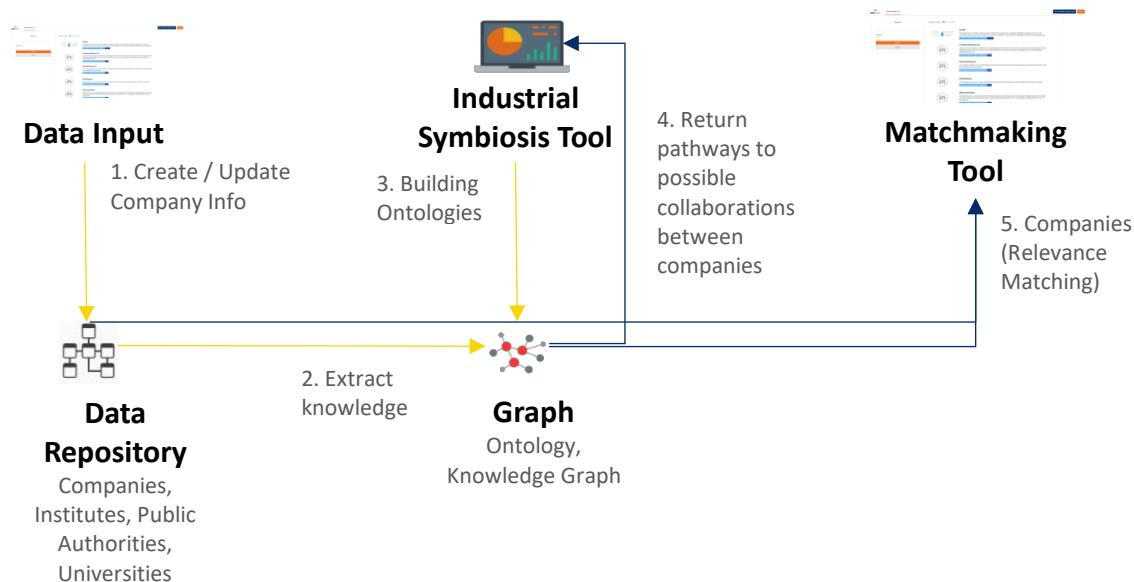


Figure 5: The Industrial Symbiosis Platform architecture.

In Figure 5, we present an abstractive illustration of the Industrial Symbiosis platform architecture.

## 6.3 Components

### 6.3.1 User Interface

A user-friendly interface should be easy and efficient to operate at the lowest effort on the part of the operator. The interface contains:

- User's registration and authentication
- User's profiles forms (fill-in forms using pop-down menus)
- Role of user
- Storing, visualizing, and searching resources, waste, by-products, processes and technologies
- Related materials



## 6.3.2 Data layer & resources

The data layer is where the data repositories with information of the participating companies, their inputs and outputs and knowledge database on resources and wastes are located.

The tool will be based on a database of the industrial residues resources of an area, to support experts and technicians of the companies to find a better use of materials through a material matching tool. The Industrial Symbiosis Tool will allow users to register and search for information, with the objective to establish collaboration between relevant parties. Potential partners will create a collaborative network of companies active in a particular domain. On such a network each company plays a role, such as:

- Supplier
- Aggregator
- Processing
- Technology provider
- Exploitation of the recovered material

Everything which is available in our environment and which can be used to satisfy our needs, it is provided in nature or it is technologically accessible and it is economically feasible and culturally acceptable can be termed as resource. Resources are classified according to several factors. For example, based on their origin, resources can be biotic (living resources, plants, animals) or abiotic (non-living, minerals, soils). Based on their exhaustibility, resources can be renewable (solar, wind, hydro, biomass) or non-renewable (oil, natural gas, coal, nuclear energy). Based on the ownership, resources can be individual (plots, fields, house, car) or community owned resources (pasture land, farming land, pond, wells, estate). Finally based on their development and use, resources are classified into actual resources, whose quantity is known (those that have been surveyed, trees, soil, mines) and potential resources, whose entire quantity may not be known (oil, gasoline, uranium).

It is evident that resources are inherently limited. As the population on earth continuously increase while the stocks of resources remain the same it is becoming more difficult to maintain the prosperity and a quality level of life. Measurable shortages of resources now exist in many regions of the world. The linear industrial model, defined as "take, make, dispose", of products with a finite lifespan, which end up in landfills leads to a dead end. In contrast, a circular industrial model aims at eliminating waste with the reuse, sharing, repair, refurbishment, remanufacturing and recycling of the resources. Thus a circular economy is currently a popular concept promoted by the several countries and in particular EU is funding several projects in the subject of industrial symbiosis (e-Symbiosis 2018) (Circ4Life 2018) (CircE 2020). A comprehensive analysis on the subject and portfolio review can be found in (F.3 2019).

### 6.3.2.1 The semantic web

In recent years, many applications have been developed in a pilot stage and few on commercial level, that support cooperation between companies with the aim to reuse materials that for some companies are considered waste while for others are useful. In general, such collaboration can be used also to discover materials nearby, which can reduce costs and thus increase the profits of the companies involved.

The heart in all these applications is a matchmaking tool between materials based on the supply and demand. It is evident that for such a collaboration a fundamental requirement is the existence of a common vocabulary between the partners. Of course, in addition to vocabulary, a language understood by computers as well as by humans is required.

The idea was proposed by Tim Berners-Lee in his paper "The Semantic Web" (T. Berners-Lee 2001), that computers would be able to process information on the Web, and take higher level decisions. The information that computers and programs can consume is presented in so-called ontologies. A definition of ontology that is well accepted in the Artificial Intelligence and the Semantic Web community was given by Tom Gruber in (Gruber 1993): An ontology defines a set of representational primitives with which to model a domain of knowledge or discourse. The



representational primitives are typically classes (or sets), attributes (or properties), and relationships (or relations among class members).

He makes the very important point that the main purpose of ontologies is that they enable sharing and reuse of knowledge.

Ontologies are expressed using the Resource Description Framework (RDF) (G. Klyne 2004), or in more recent years, by using the Web Ontology Language (OWL) (W3C 2009). RDF and OWL define subject–predicate–object triplets, used to describe a resource (the subject), its properties (the predicate) and the value of this property (the object). By the definition we note that the object can be itself a resource, used as subject in other statements. This implies that ontologies build complex graphs of connected knowledge.

### 6.3.2.2 Ontologies

Industrial Symbiosis supports the circular economy of transforming waste, and by-products of one industry to raw material of another. Companies usually lack-of the information and knowledge concerning the available resources. Industrial Symbiosis aims to bring together waste producers/owners with possible users of the materials that could be recovered, upon some treatment (Low 2018).

The use of ontologies is of critical importance for:

- terminology normalization (common vocabulary)
- Knowledge database
- Semantic search
- Classification of materials
- Uses of the material etc.

Circular Economy needs a Knowledge Management tool which helps to process, store, share, exchange data and information between stakeholders and products and materials involved in an efficient and effective manner. Ontologies will be used to achieve:

- 🔗 **Interoperability:** A specific-domain ontology-based approach to represent the knowledge of the Eco-Industrial Parks we shall follow within this project. A domain ontology represent data in a particular domain and provides vocabularies about concepts and their relationships. Such an ontology-based approach can increase the interoperability between the companies of an Eco-Industrial Park.
- 🔗 **Common terminology:** Interoperability is achieved by using a common terminology, derived from the ontology. Thus, we ensure information integration which facilitates the construction of an efficient indexing.
- 🔗 **Classification of materials:** Through ontologies, hierarchical structures of concepts related to materials can be defined using a common vocabulary of the knowledge area. Furthermore, adding some reasoning to this structure we can classify materials within the defined hierarchy.

The concept “Resources” acts as the point of reference for all semantic service descriptions and the synergy identification activities. The concept refers to materials, waste, energy, product and water that a user provides or requires. The Resource concept consists of four different classification streams (N. Trokanas 2014):

- (i) classification by Source,
- (ii) classification by Type,
- (iii) classification by Product and
- (iv) classification by Characteristic.

Classification by Source is implemented using both existing classifications, e.g. The European Waste Catalogue (EWC) (Commision 2000) and purpose-made classifications representing specific types of waste imposed by the application. These classifications are used for the registration of a item as a Resource or by-products of a processing technology. The nonstandard nature of several items (in particular wastes) makes it difficult to compare different types, hence imposing the use of a more standardized classification defining the composition of the item. Classification by Type itself includes three categorizations: (i) Materials, (ii) Energy and (iii) Water (See figure 1) Materials category is used to address the problem of standardization. This classification inherently invokes similarity



for concepts that have structural proximity in the ontology. For example, different types of biomass are interchangeable for bio-refinery processes.

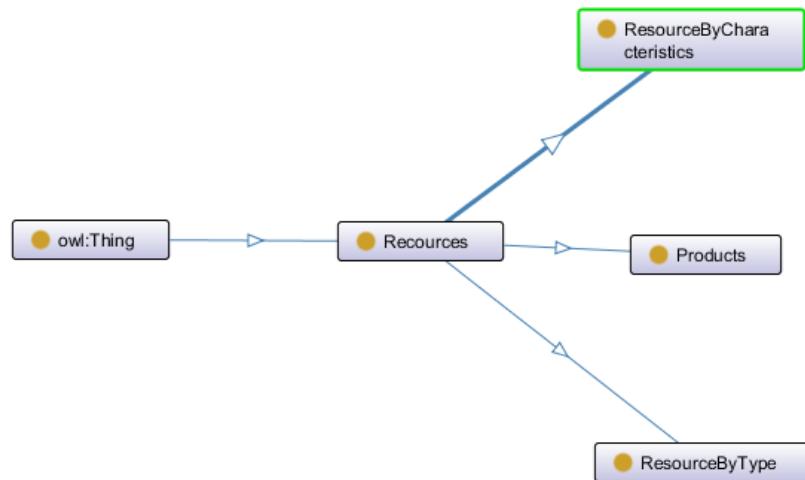


Figure 6: Resources classification.

The Materials classification is also used to define the composition of products and waste as well as to define inputs, by-products and in some cases, products and by-products of processing technologies (see **Error! Reference source not found.**, Figure 8).

The Energy classification is specified as a resource but is also used to define energy requirements for processing technologies, whereas Water classification is used to define input or pre-processing conditions for some processing technologies. The classification by Product, is used to define the outcome (product) from processing technologies. It is also used by resource producers allowing for resource classification as products when the composition is not known.

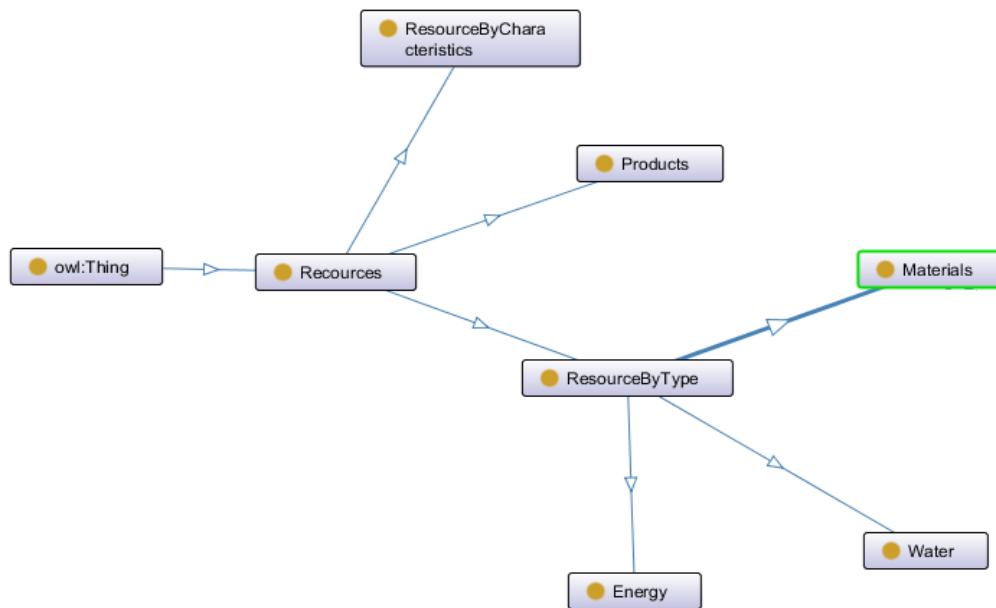


Figure 7: Classification of resources by Type.



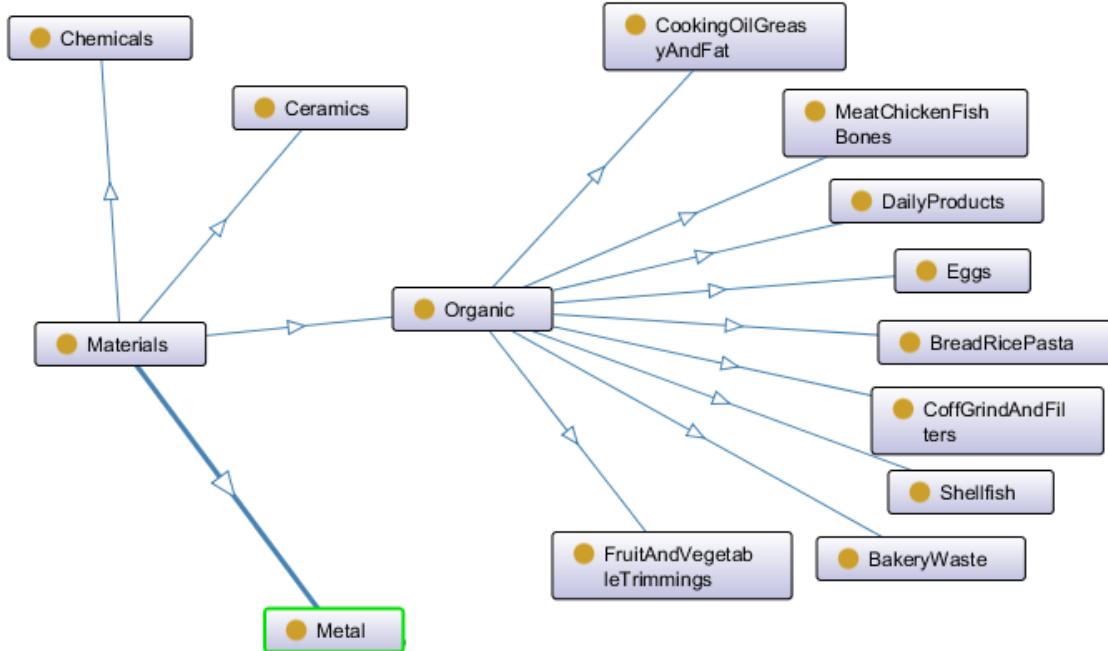


Figure 8: Classification of Organic Materials.

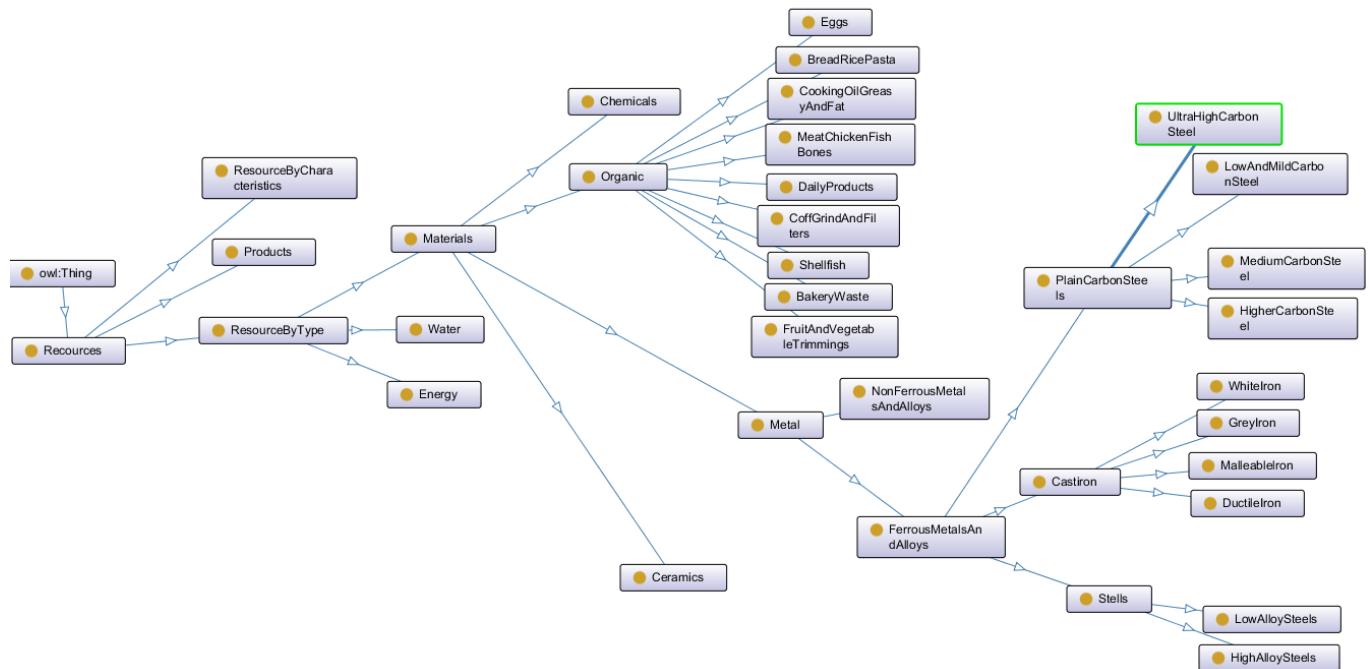


Figure 9: An integrated example of a resources ontology.

Digicirc project will investigate three strategic domains for digital-driven Circular innovation: Circular cities, Bio-economy and Blue Economy. This means that the program will potentially be able to support a wide range of



applications such as, waste management, mobility, renewable energy, logistics, urban agriculture, synergy and linkages across fisheries, logistics, tourism, aquaculture, blue biotechnology, tourism, ocean energy etc. This requires an ontology that can cover almost all possible collaborations that may arise in business and because there are no ready-made ontologies to cover all 3 domains, the ontology that will be used in the context of Digicirc will cover only the resources that will be used by the registered companies.

However a resource ontology alone does not provide the means required in the digital collaboration on the semantic web. Resources offerings on the web require more complex statements, as it is, for example the target group for an offer to be valid, in terms of the region, the distance, the eligible type of buyers.

### 6.3.3 Material Flow Knowledge Graph

The resources/materials are modelled and stored via a graph database. In a graph database, data is represented as nodes and relationships. This is different from relational database where the data is represented as tables and columns. Graph database, like relational database, can store data in the form of node properties. For example, we can represent each resource as a node, and the name and description of the resource as the properties of the node. In relational database, the same effect is achieved by representing each resource as a row in a table. In the table, it has the name and description of the resource as the columns. The graph database will play the role of our material flow knowledge graph.

A graph database offers equivalent abilities for storing properties as a relational database. However, a graph database treats relationship as a first class citizen. In a relational database, relationships are inferred during query execution by using foreign keys and table join operations. Therefore, during the query execution, if table X is to be joined with table Y, then the database engine needs to perform a lookup to find which tuples from table Y correspond to a certain tuple from table X. As such, even if table Y is indexed according to the relevant foreign key, the lookup runtime complexity will still be  $O(\log n)$ . Conversely, in a native graph database, the related tuples are physically linked, which means the lookup runtime complexity is only  $O(1)$ . Therefore, if the relevant query involves many relationships, the native graph database will perform faster. More information on the native graph database and its non-native variant can be found in the book by (I. Robinson 2015).

At the core of the platform stands the Industrial Symbiosis (IS) Knowledge Graph, which is based on a rich and inter-corporate ontology. The graph structures will be adopted by the database schemas to enhance flexibility:

- Creating schemas and adding information by inserting elements and relations.
- Dynamically updating existing schemas and adding info by creating new relations between existing elements.
- Discovering schemas and information (hidden connections) by examining elements and the relations about elements using graph traversal algorithms.

The IS Knowledge Graph is based on a materials, waste, products, by-products and processing technologies classification defined in line with the European Waste Catalogue standard classification and is substantially enriched with models from different domains and external resources, such as processing technologies, production workflows, and geographic knowledge for location-based searching. This ontological classification of the resources is what allows the Industrial Symbiosis tool to intelligently detect matches between seemingly incompatible resources.

Figure 10 illustrates the use of knowledge graph, used in the Industrial Symbiosis tool. First, experts like engineers, chemists etc., are able to add resources or processes via the web UI. These resources and processes are then stored in the knowledge graph, which are then available for simple users like public officials or entrepreneurs, to explore via again the web UI. The web UI also shows the material flow as a graph representation, where nodes represent resources and edges represent processes or technologies that are required to transform a resource to another.



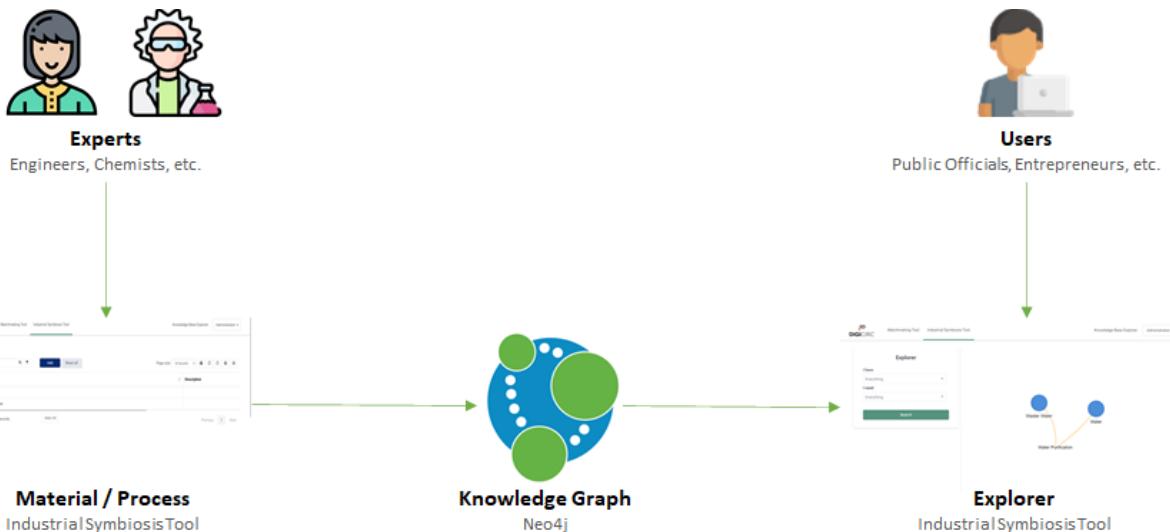


Figure 10: an illustration of the exploration process on the IS knowledge graph.

## 6.4 Users of the Industrial Symbiosis tool

The Users of the Industrial Symbiosis tool are distinguished into guests, registered users (SMEs, Universities, Research Institutes, Public Authorities), Partners, Experts, Knowledge Administrators and Administrators. In the following we present briefly the rights and the responsibilities of the users:

- **Guest users.** They are external users visiting the site and don't need to be registered in the platform. They have the ability to browse the companies registered in the platform and display the companies on a map based on geo-location restrictions, or by type of product. Locations and types of products are selected from pop-up menus.
- **Registered users.** They can describe and suggest new resources.
- **Partners.** They are users of the Industrial Symbiosis Tool. They are registered users with additional rights as soon as a synergetic activity is established. This is triggered by a request for collaboration between two actors. Depending on the domain of the companies the request is automatically broadcasting to an expert. When the recipient accept the request and the expert agree, a collaboration network is created. The participating users are becoming, Partners and they have additional rights. Such rights include:
  - the exchange of business documents concerning the quality, composition, quantity of a resource, means of delivery, availability, financial offer etc.
  - the submission of questions to their supervisor-expert for difficulties and problems they face on their collaborative network. These Q&A will be used for evaluating a synergy and will serve as a feedback on future development.
- **Experts:** They are technical users with expertise in circular economies which will act as supervisors to the synergetic networks; they will monitoring the progress of the collaboration and will report at the end on success stories or problems and difficulties which caused the failure of the collaboration. They will answer to queries of the partners in a network and will keep track each time a collaboration enters a higher level of synergy. Finally they inform experts when new resources enter into the system and suggest extensions of the resource ontologies.
- **Knowledge administrators:** They are responsible for the development, enrichment and update of the knowledge base.



- **Administrator:** He is responsible for the availability and good functionality of the platform. He gathers critical information from network users to determine problems and document troubleshooting efforts to help pinpoint solutions.

## 6.5 Workflow

The Industrial Symbiosis tool workflow will follow six key stages:

1. Registration or login (not necessary)
2. Visualization of the resources and material flow
3. Identification and search of useful materials and processes
4. Interacting with the material flow knowledge graph (user has the ability to click on the materials and processes to view additional information)
5. Finding interesting paths of transforming resources that enable novel processes or technologies, which could be valuable to an actor
6. Case Study Production



## 7 Web platform functionalities

The main functionality of the Industrial Symbiosis tool, especially with regards to the Knowledge Based aspects of it, is delivered in the form of web services fully accessible on the Internet which allows the Business Users, i.e. the resources and waste stream engaged members, together with the IS practitioners to register their interest and to describe their provisions. The Industrial Symbiosis tool is providing users with the following functionalities:

- 👉 **Registration of resources:** the waste stream providers are guided by the waste knowledge model (ontology) to provide details necessary for the full description: waste composition, availability and anticipated availability, geographical location etc.
- 👉 **Knowledge Model Maintenance:** register respective datasets describing / characterizing waste streams.
- 👉 **Technology Registration:** Providers are guided by the technology knowledge model to help users to provide necessary for the full description: technology description, technological, economic and environmental characteristics, geographical location etc.
- 👉 **Communications:** The Industrial Symbiosis platform supports partners' information exchange. The process may be guided / monitored by IS Practitioners.

The platform offers additional functionalities like:

- 👉 Advanced search options
- 👉 Reports
- 👉 Statistics
- 👉 Data export
- 👉 Monitoring after successful match etc.

The tool provides functionality for users to register their sites, materials as well as basic data handling instructions for the tools (as in what data to be used for the processes). Additionally, the platform displays information for said sites and materials, as well as potential synergies and matching capabilities. This data are available in various formats, such as exportable lists in .csv format, various graphs & maps (depending on the type of data). Finally, users are able to be notified for any updates stemming from the LSD tool processes and advance potential synergies.

The user (company or organization) is required first to register through the matchmaking tool, as described in deliverable D3.7. The functionalities have been collected into groups and they are listed below in detail.

Based on the user requirements identified earlier, the functionalities of the Industrial Symbiosis web platform have been identified. The user (company or organization) is required first to register through the matchmaking tool, as described in deliverable D3.7. They have been collected into groups and they are listed below.

### 7.1 User registration and login

As described previously, the users of the system are: guests, registered users, experts, knowledge base administrators and the system's administrator. Guests are not required to register and have limited capabilities such as, browsing the companies participating in the platform in a list or on a map in general and by country or to



browse the companies in one of the three domains (circular cities, bio-economy, blue-economy). All other users must register to use the platform. The registration is done in two levels:

The first level includes mandatory information and the user must complete the following fields:

1. First name
2. Last name
3. Email address
4. Password
5. Confirm password and agree with the Terms & Conditions Privacy Policy

In the second level the type of the system's actors is declared. This includes the following types:

1. Simple user
2. Expert
3. Sole trader
4. SME
5. University
6. Research Institute
7. Public Authority
8. Knowledge Base administrator
9. Administrator

The system will create the appropriate entry automatically and allocate the user with the relevant level of access to the site – ‘Simple user’, ‘Expert’ etc. will have different levels of access. The designation of the levels will be initially determined by the Administrator of the platform.

In the case of a simple user the registration process ends here. A simple user can use the tools such as, the Matchmaking tool or the information hub but cannot participate in an industrial symbiosis network.

An expert user has access to all the collaborative networks in the domain of his expertise is monitoring them during their life. He has access to all the correspondence he has exchange with the partners of the symbiosis networks. This information is used for reporting the benefits or the problems of the collaboration and will be used as feedback and to improve the symbiotic processes.

The sole trader should fill in the address of his headquarters, a brief description of his business activities and the URL of his website if he has one.

Similarly the actors, SMEs, Universities, Research Institutes, Public Authority must complete the information:

- Name of the organization
- Address
- URL
- Description of their business activities.

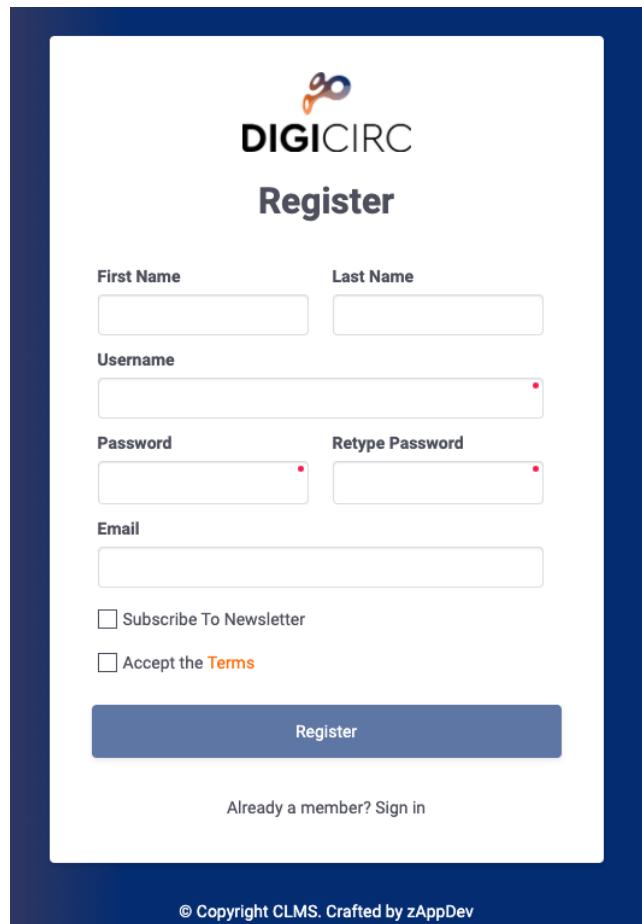
In this case, the user who created the registration is considered as the contact person of the organization. At any time a user can become actor by filling the information in the second level of the registration.

The knowledge base administrator is a user with additional rights to access and modify (extend/correct) the knowledge base.

Finally the system's administrator has the right to intervene to all the components of the system in order to ensure the proper functioning of the system.

Next, we present screenshots of the developed functionalities. Web forms are shown to the users so that they register themselves in the Industrial Symbiosis tool. The registration process requires a custom form to be developed that requests the appropriate information (name, email, etc.), as shown in Figure 11.





The image shows a registration form titled "Register" for the DIGICIRC platform. The form is contained within a white rectangular box with a dark blue border. At the top center is the DIGICIRC logo, which consists of a stylized orange and red swirl icon followed by the word "DIGICIRC" in a bold, sans-serif font. Below the logo, the word "Register" is centered in a large, bold, dark gray font. The form fields are arranged in two columns. The left column contains "First Name" and "Username" fields, each with a placeholder text box. The right column contains "Last Name" and "Retype Password" fields, also with placeholder text boxes. Both "Password" and "Retype Password" fields have red validation dots at their ends. Below these are "Email" and "Subscribe To Newsletter" fields. Underneath the "Email" field is a "Accept the Terms" checkbox. At the bottom of the form is a large, dark blue "Register" button. Below the button, the text "Already a member? Sign in" is displayed in a small, dark gray font. At the very bottom of the white box, the copyright notice "© Copyright CLMS. Crafted by zAppDev" is visible.

Figure 11: form for the registration process of a user.

If a user has already an account, he can login to the platform by using the login component, presented in Figure 12. The user can login to the tool by providing username and password. The login form also offers functionalities for users that forgot their username or password. The user can click on the “Forgot your username?” or “Forgot your password” links, and he/she will instantly receive an email with instructions on what to do.



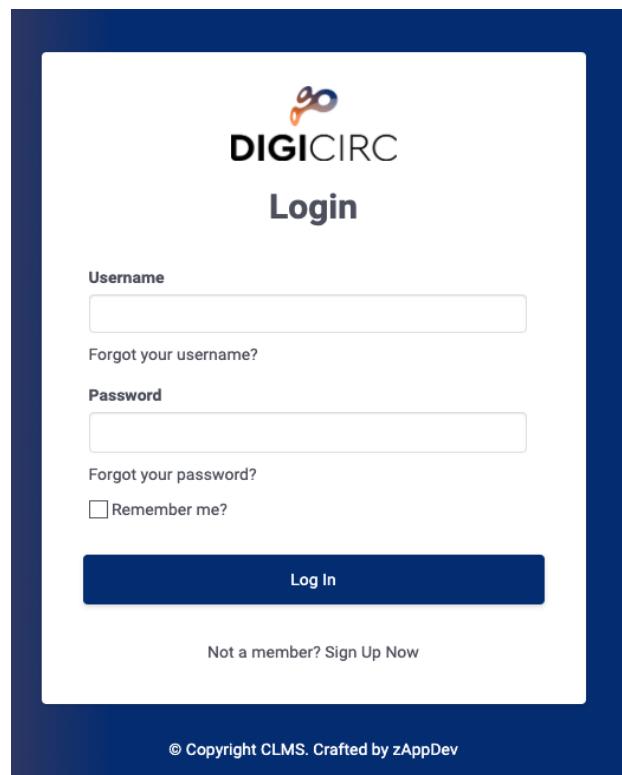


Figure 12: user login UI component.

## 7.2 Material flow explorer

The starting page of the industrial symbiosis tool after the user login, is showing the material flow explorer. The material flow explorer is visualizing the material-process knowledge graph with a graph representation (Figure 13).

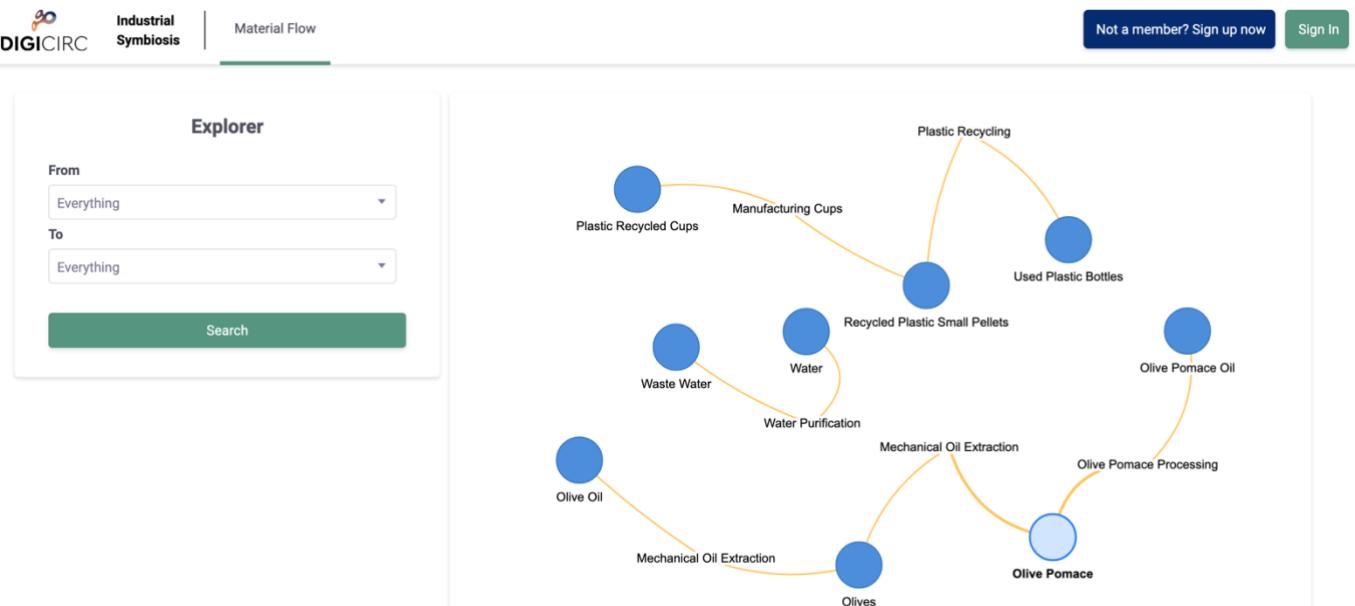


Figure 13: the starting page of the industrial symbiosis tool, with the material flow component.



### D3.6: Industrial Symbiosis tool

Figure 14 illustrates the starting page for the Administrator role. The Administrator is able to see also the materials and processes available on the knowledge database.

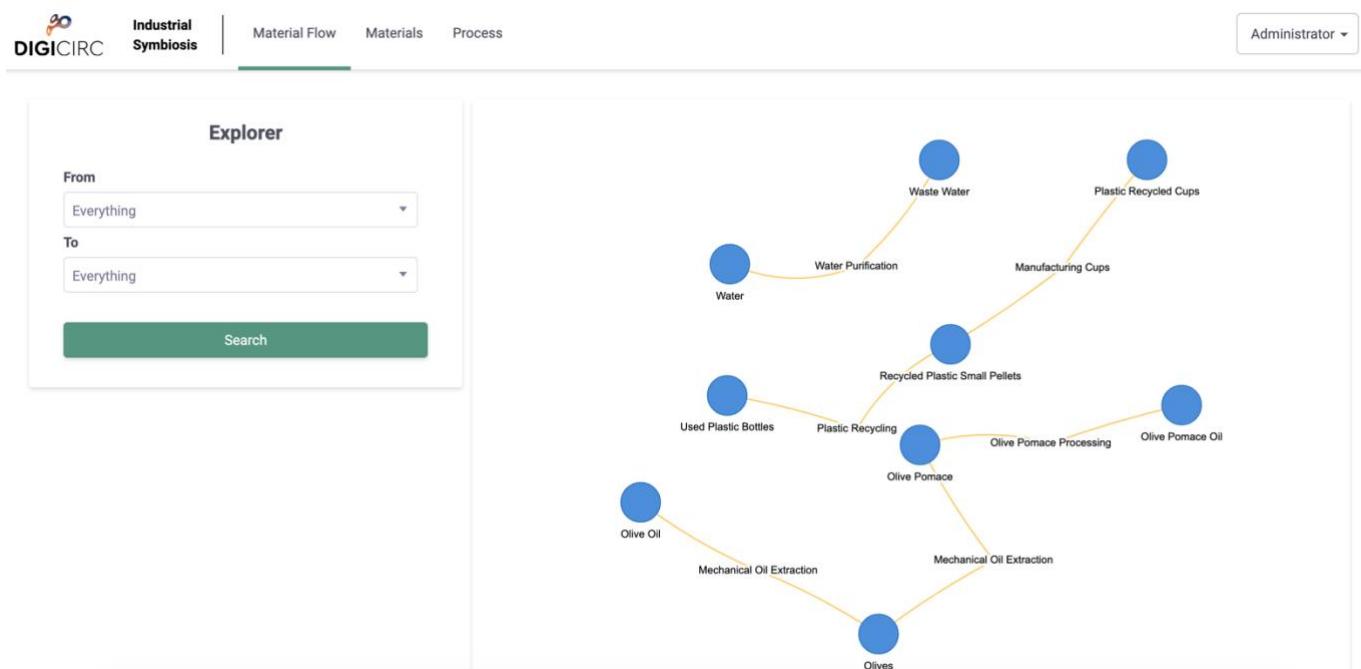


Figure 14: the starting page of the industrial symbiosis tool, viewed with an Administrator account.

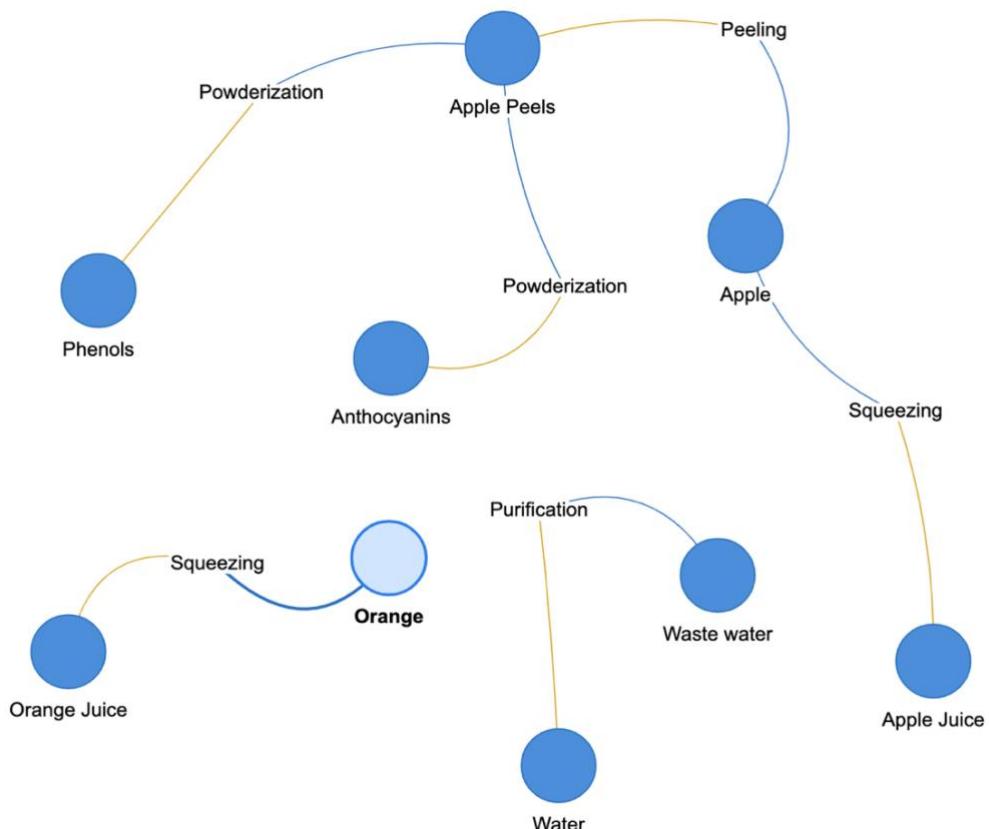
The material flow explorer can be accessed/updated by clicking on the “Industrial Symbiosis” link-button on the top left corner of the web platform. The user can then select their resources of interest, for example if they have or need a specific resource. Figure 15 shows the options offered to the user of the platform.

This screenshot shows the 'Explorer' interface for selecting resources. It features two dropdown menus: 'I have' and 'I need', both currently set to 'Everything'. Below these is a green 'Search' button. The interface is clean and minimalist, designed for easy resource selection.

Figure 15: selecting and visualizing resources via the knowledge graph.



After selecting the resources of interest, a graph representation of the knowledge database is shown. An example of this graph is shown in Figure 16. The nodes represent the resources and the edges represent the associated processes or technologies, required to transform the associated resources.



**Figure 16: another sample of the industrial symbiosis material flow knowledge graph.**

## 7.3 Resources and Processes

Through the Industrial Symbiosis tool, a user can add resources (materials or services) and processes (technologies) which are associated to resources. The resources and processes available on the tool can be shown individually by clicking on the “Materials” and “Process” links on the top bar of tool, next to “Material Flow”.

Materials	
Search	<input type="button" value=""/>
<input type="button" value="Add"/>	<input type="button" value="Reset all"/>
Page size: 20 records <input type="button" value=""/>	
Name	Description
1 Waste water	
2 Water	

Showing 1 to 2 of 2 records  Previous  Next

**Figure 17: the component which allows for showing the existing resources and adding new ones.**

In Figure 17 we can see the available resources in the Industrial Symbiosis platform, shown in a table format. By clicking on the “Add” button, a user can add more resources. The “Add” functionality is shown in Figure 18.



### D3.6: Industrial Symbiosis tool

The screenshot shows the 'Add Material' page of the DIGICIRC Industrial Symbiosis tool. At the top, there is a navigation bar with the DIGICIRC logo, 'Industrial Symbiosis' text, and links for 'Material Flow', 'Materials', and 'Process'. On the right, there is an 'Admin' dropdown menu. Below the navigation, the page title 'Add Material' is displayed, along with 'Back' and 'Save' buttons. The main form contains fields for 'Name', 'Type' (with a dropdown menu 'Please Select'), 'Physical Form' (with a dropdown menu 'Please Select'), 'Unit Of Measurement' (with a dropdown menu 'Please Select'), 'Is Hazardous' (with a checked checkbox), 'Description' (with a rich text editor toolbar), and two tables: 'Convert By' and 'Converted By'. The 'Convert By' table has columns 'Name' and 'Product' and displays 'No records.'. The 'Converted By' table has columns 'Name' and 'Source' and also displays 'No records.'

**Figure 18: adding a new material in the industrial symbiosis platform.**

Figure 19 shows the functionality responsible for presenting the available processes or technologies. A new process can be added here by clicking on the “Add” button. The “Add” functionality is shown in Figure 20. Furthermore, a process can be selected and then edited by the UI. Finally, adding a new process allows for adding references connected with the process, or even notes that could be useful for the industrial symbiosis project.

The screenshot shows the 'Processes' list page of the DIGICIRC Industrial Symbiosis tool. At the top, there is a search bar, an 'Add' button, and an 'IS Sync' button. To the right, there are buttons for changing the page size (20 records) and navigating through the pages. The main table lists one process: 'Purification' with a note '1'. The table has columns 'Name', 'Notes', and 'Ref'. At the bottom, it shows 'Showing 1 to 1 of 1 records' and page navigation buttons for 'Previous', '1', and 'Next'.

**Figure 19: the industrial symbiosis component responsible for showing existing processes and adding new ones.**

The screenshot shows the 'Add new Process' form. It has a header 'Add new Process' with a close button 'X'. The form includes fields for 'Name' (a text input field), 'Notes' (a text area), 'Ref' (a text input field), and a 'Products' section. The 'Products' section contains a table with columns 'Name' and 'Description', and a '+' button to add more rows. At the bottom, there is a 'Save' button.

**Figure 20: adding a new process for the newly added resource.**



# 8 Use Cases

## 8.1 Use case 1

A user (representing a company-actor) searches for specific material or waste information, and new by-products.

## 8.2 Use case 2

A user (representing a company-actor) inserts material or waste information, processes and new by-products coming from the processes. The user may also add links or references on wastes, processes, and the by-products.

## 8.3 Use case 3

A firm (user) is interested to know if a material or waste can be converted into another resource; In the case of waste, the user would like to investigate if there is a possibility of recovering value from the waste. And if it is possible, the firm is also interested to know how, or what technology is needed, to make it happen. Through the user interface (UI), the firm first inputs the name of the waste it is interested in to recover value from or selects the waste from a drop-down list. The knowledge graph is queried, showing possible pathways or options for converting the waste into resources. The waste-to-resource conversion pathways with reference to the waste (source) are returned and graphically visualized to the firm on the UI. Users can find out more information about a resource, waste and their corresponding conversion process (with references) by hovering the mouse pointer over their respective nodes in the graph.

## 8.4 Use case 3

A firm (user) is interested to know if it is possible to substitute its raw material with an alternative material originating from waste (by-product). And if possible, what is needed to make it happen. Again, through the same user interface (UI), the firm inputs the name of, or selects from a dropdown list, the product (resource) of which the raw materials it is interested to substitute. The knowledge graph is then searched, revealing possible pathways or options for substituting the raw materials of the product. Similar to the first use case but this time, with reference to the resource (target), the waste-to-resource conversion pathways are returned and graphically visualized to the firm on the UI.



## 9 Conclusion

This deliverable aims to describe the Industrial Symbiosis tool. The proposed system architecture of the industrial symbiosis platform takes into consideration all the aspects that were introduced by a modern Circular Economy environment. The Industrial symbiosis tool shows it incorporates features that are very relevant to the needs arising in the three domains of DigiCirc.

The intention of Industrial Symbiosis is to develop a web-based platform which enables users initially in the EU, to participate in industrial symbiosis (IS) activities which will improve resource efficiency across the economy.

The Industrial symbiosis tool serves as a centralized gateway of information for interested parties. The tool's main functionality is to allow users to investigate and interact with resources and associated processes or technologies, in an Industrial Symbiosis environment. All the registered resources and processes are stored following a state-of-the-art knowledge graph architecture, which is the key component of the Industrial Symbiosis tool. The knowledge graph acts as a resource/material knowledge database, connecting materials with processes via a graph representation. Inside the graph, nodes represent resources/materials and they can be connected via edges, which represent processes or technologies. Thus, the tool allows the users to view information about material flow; how a material can be transformed to another material via a specific process or technology. The material flow explorer also allows for searching both ends of the flow, meaning that the tool gives the ability to query either the available paths starting or ending from/to a specific material. Additionally, users are able to suggest new materials or processes for extending the tool, and afterwards a user with elevated rights (Administrator or Expert) may add the new knowledge to the system. Moreover, the newly added resources can carry additional information like type, physical form, unit of measurement, description and hazardness. Finally, the processes also allow for information like metadata, reference or any useful notes.

Moreover, the deliverable provides an overview of the business environment related to Industrial Symbiosis, defining the role of Industrial Symbiosis in the Circular Economy landscape. The deliverable presents the designed architecture, lists the user requirements and focuses on supported functionalities for the Industrial Symbiosis web platform.

Finally, we present use cases for the Industrial Symbiosis tool, which cover the majority of the needs of the participating parties.



# 10 Bibliography

- Beamon, Benita. 1999. *Designing the Green Supply Chain*. Logistics Information Management.
- Benedict, Martin, Kosmol, Linda, and Esswein, Werner. 2018. "Designing Industrial Symbiosis Platforms – from Platform Ecosystems to Industrial Ecosystems." *Twenty-Second Pacific Asia Conference on Information Systems*. PACIS 2018 Proceedings.
- Circ4Life. 2018. *A circular economy approach for lifecycles of products and services*. <https://www.circ4life.eu/project-overview>.
- CircE. 2020. *European regions toward Circular Economy*. <http://www.interregeurope.eu/circle/>.
- Commision, European. 2000. "Commission Decision 2000/532/EC replacing Decision 94/3/EC establishing a list of wastes pursuant to Article 1 (a) of Council Directive 75/442/EEC on waste and Council Decision 94/904/EC establishing a list of hazardous waste pursuant to Article 1 (4) of C." *Official Journal of the European Communities* 3-24.
- e-Symbiosis. 2018. *e-Symbiosis*. <http://www.esymbiosis.gr/site>.
- F.3, European Commission Directorate-General for Research and Innovation Directorate F – Prosperity Unit. 2019. 4. *Study and portfolio review of the cluster of projects on industrial symbiosis in the Directorate for Prosperity in DG Research and Innovation: Findings and recommendations*. Sustainable Industry Systems.
- G. Klyne, J. Carroll. 2004. "Resource description framework (RDF): concepts and abstract syntax." *W3C recommendation*.
- Gruber, T.R. 1993. "A translation approach to portable ontology specifications." *Knowledge acquisition* 5 (2) 199-220.
- I. Robinson, J. Webber, E. Eifrem. 2015. *Graph Databases: New Opportunities for Connected Data*. O'Reilly Media.
- Low, J.S.C., Tjandra, T.B., Yunus, F., Chung, S.Y., Tan, D.Z.L., Raabe, B., Ting, N.Y., Yeo, Z., Bressan, S., Ramakrishna, S. and Herrmann, C. 2018. "A collaboration platform for enabling industrial symbiosis: Application of the database engine for waste-to-resource matching." *Procedia CIRP* 849-854.
- N. Trokanas, F. Cecelja, T. Raafat. 2014. "Semantic input/output matching for waste processing in industrial symbiosis." *Computers and Chemical Engineering* 66 259-268.
- T. Berners-Lee, J. Hendler, O. Lassila. 2001. "The semantic web." *Scientific American* 284.
- W3C. 2009. "Owl 2 web ontology language document overview." *W3C OWL working group*.



# 11 Appendix – Development documentation

## Table Of Contents

- [Introduction](#)
- [Design approach](#)
- [Architecture Design](#)
  - [Logical View](#)
  - [Implementation Strategy Description](#)
  - [Key Characteristics](#)
- [Architectural Patterns](#)
- [More intro stuff that needs update goes here](#)
- [System Design](#)
  - [Business Objects](#)
    - [User Class Diagram](#)
    - [User Classes](#)
    - [User Associations](#)
    - [Actors Class Diagram](#)
    - [Actors Classes](#)
    - [Actors Associations](#)
    - [Business Class Diagram](#)
    - [Business Classes](#)
    - [SearchQuery Class Diagram](#)
    - [SearchQuery Classes](#)
    - [SearchQuery Associations](#)
    - [ValueList Class Diagram](#)
    - [ValueList Classes](#)
    - [ValueList Associations](#)
    - [DataImportHelper Class Diagram](#)
    - [DataImportHelper Classes](#)
    - [KnowledgeMaterialBase Class Diagram](#)
    - [KnowledgeMaterialBase Classes](#)
    - [KnowledgeMaterialBase Associations](#)
    - [KnowledgeProcessBase Class Diagram](#)
    - [KnowledgeProcessBase Classes](#)
    - [KnowledgeProcessBase Associations](#)
    - [MaterialsBase Class Diagram](#)
    - [MaterialsBase Classes](#)
    - [MaterialsBase Associations](#)
    - [TextSearch Class Diagram](#)
    - [TextSearch Classes](#)
    - [Product Class Diagram](#)
    - [Product Classes](#)
    - [Product Associations](#)
    - [ActorAPIHelper Class Diagram](#)
    - [ActorAPIHelper Classes](#)
    - [KnowledgeActorBase Class Diagram](#)
    - [KnowledgeActorBase Classes](#)
    - [KnowledgeActorBase Associations](#)
    - [ElasticHelpers Class Diagram](#)

- [ElasticHelpers Classes](#)
- [ElasticModel Class Diagram](#)
- [ElasticModel Classes](#)
- [ClmsGraphBackend Class Diagram](#)
- [ClmsGraphBackend Classes](#)
- [ClmsGraphBackend Associations](#)
- [Geolocation Class Diagram](#)
- [Geolocation Classes](#)
- [Geolocation Associations](#)
- [KnowledgeProducersMaterialBase Class Diagram](#)
- [KnowledgeProducersMaterialBase Classes](#)
- [KnowledgeProducersMaterialBase Associations](#)
- [GraphQuery Class Diagram](#)
- [GraphQuery Classes](#)
- [GraphQuery Associations](#)
- [Suggestions Class Diagram](#)
- [Suggestions Classes](#)
- [Suggestions Associations](#)
- [User Interface](#)
  - [ErrorPage Form](#)
  - [FirstAdminSetup Form](#)
  - [HomePage Form](#)
  - [NotFoundPage Form](#)
  - [SignInPage Form](#)
  - [SignOutPage Form](#)
  - [Unauthorized Form](#)
  - [UserPreferences Form](#)
  - [MasterPage Form](#)
  - [MasterPageForSlide Form](#)
  - [ApplicationSettingForm Form](#)
  - [ApplicationSettingsList Form](#)
  - [ChangePassword Form](#)
  - [ForgotPassword Form](#)
  - [ManageOperation Form](#)
  - [ManagePermission Form](#)
  - [ManageRole Form](#)
  - [ManageUser Form](#)
  - [OperationsList Form](#)
  - [PermissionsList Form](#)
  - [RolesList Form](#)
  - [UsersList Form](#)
  - [MasterPageSignIn Form](#)
  - [CreateAdmin Form](#)
  - [RegisterForm Form](#)
  - [SearchForm Form](#)
  - [ResultsForm Form](#)
  - [ActorForm Form](#)
  - [EntityTypeForm Form](#)
  - [CountryForm Form](#)
  - [CountryList Form](#)
  - [EntityTypeList Form](#)
  - [BusinessFunctionForm Form](#)

- [BusinessFunctionList Form](#)
- [BusinessTypeForm Form](#)
- [BusinessTypeList Form](#)
- [ActivitiesForm Form](#)
- [ActivitiesList Form](#)
- [Dashboard Form](#)
- [EmptyMasterPage Form](#)
- [GraphQueryDebug Form](#)
- [Bubble Form](#)
- [GraphCreateDebug Form](#)
- [GraphExportForm Form](#)
- [ExpertiseForm Form](#)
- [ExpertiseList Form](#)
- [SectorTypeForm Form](#)
- [SectorTypeList Form](#)
- [ServicesForm Form](#)
- [ServicesList Form](#)
- [ThematicExpertiseForm Form](#)
- [ThematicExpertiseList Form](#)
- [CompanyList Form](#)
- [ActorViewForm Form](#)
- [ClusterList Form](#)
- [ManageActors Form](#)
- [ActorsToAdministrators Form](#)
- [ClusterInitialization Form](#)
- [MaterialForm Form](#)
- [MaterialList Form](#)
- [ProcessForm Form](#)
- [ProcessList Form](#)
- [ProductTypeForm Form](#)
- [ProductTypeList Form](#)
- [ManageResources Form](#)
- [ResourceForm Form](#)
- [UnitOfMeasurementForm Form](#)
- [UnitOfMeasurementList Form](#)
- [PhysicalFormForm Form](#)
- [PhysicalFormList Form](#)
- [KnowledgeHub Form](#)
- [ForgotUsername Form](#)
- [OportunitiesExplorer Form](#)
- [UnderContructionPage Form](#)
- [MatchBaseExplorer Form](#)
- [Matching Form](#)
- [SymbiosisMasterPage Form](#)

## Introduction

The Application Design Specification document tracks the necessary information about the architecture and the system design in order to provide the development team with guidance on the underlying architecture of the application to be developed. Its intended audience is the project manager, project team, and development team.

Some portions of this document such as the user interface (UI) may on occasion be shared with the client/user, and other stakeholder whose input/approval into the UI is needed.

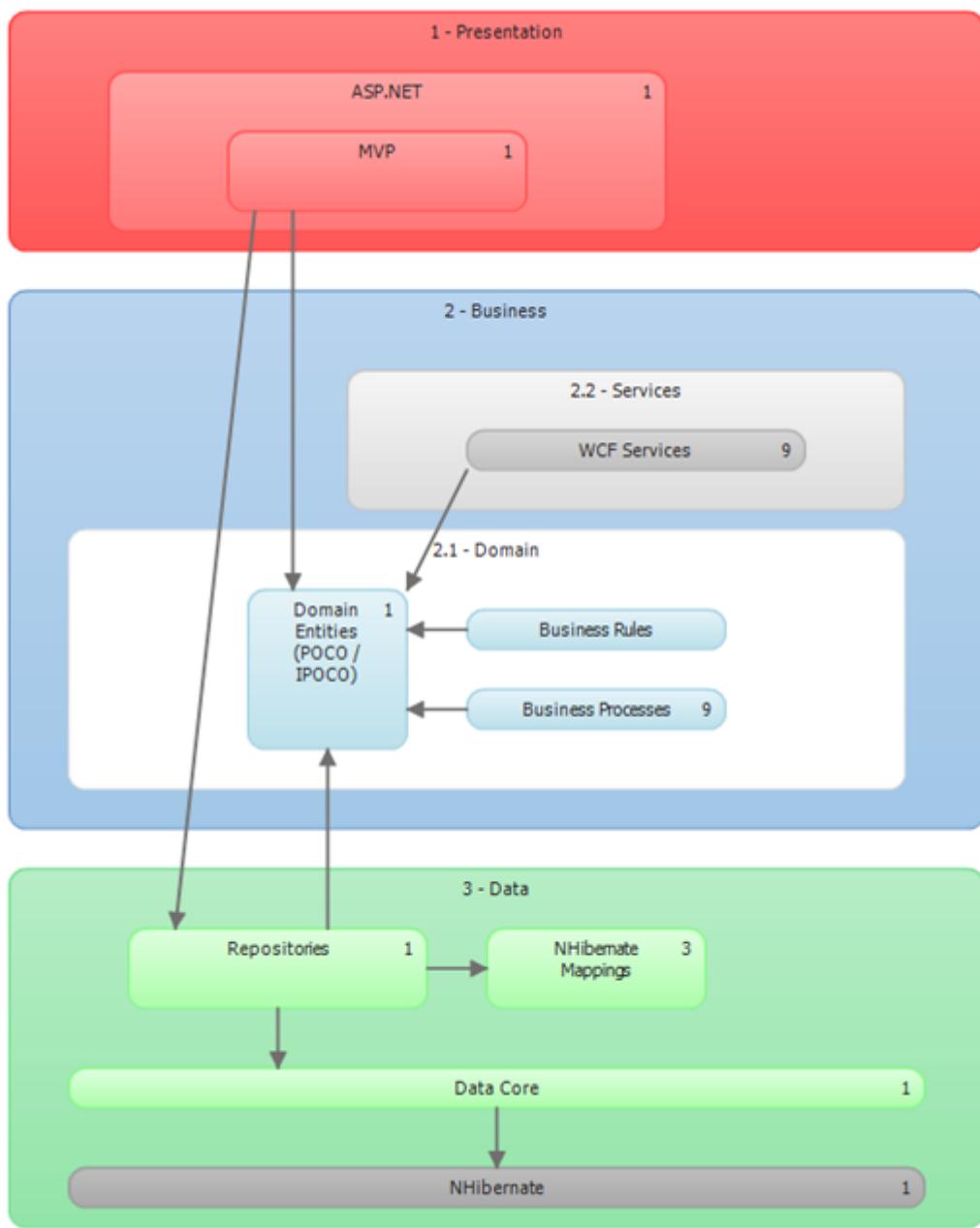
## **Design approach**

The Design Approach adopted for the application specification document is based on standards based modeling techniques including:

- Entity Relationship Diagrams
- IDEF0 Functional Models
- IDEF1X97 Object Oriented Models
- High Level Pseudo Code Specification Language

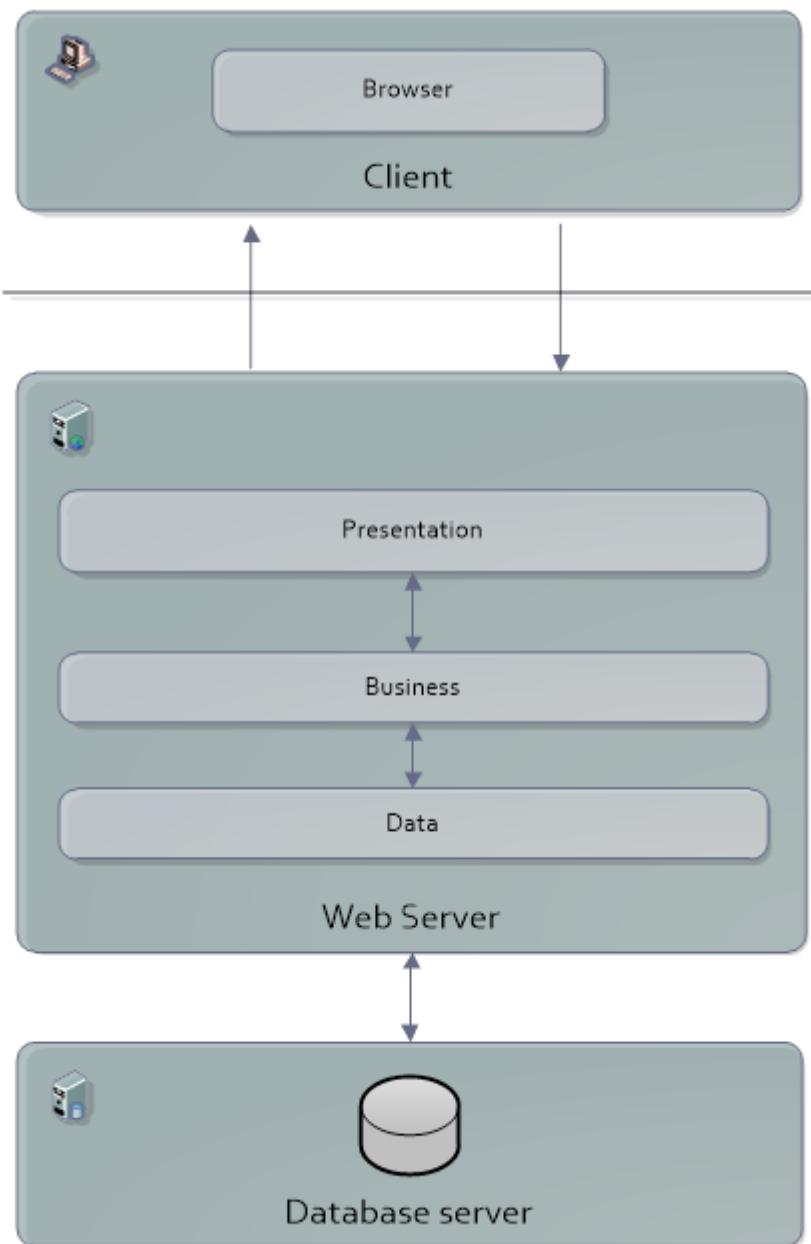
## **Architecture Design**

### **Logical View**



## Implementation Strategy Description

This is a three-tier design with the client workstation representing one tier, a web server representing the second tier, and a database server representing the third and final tier.

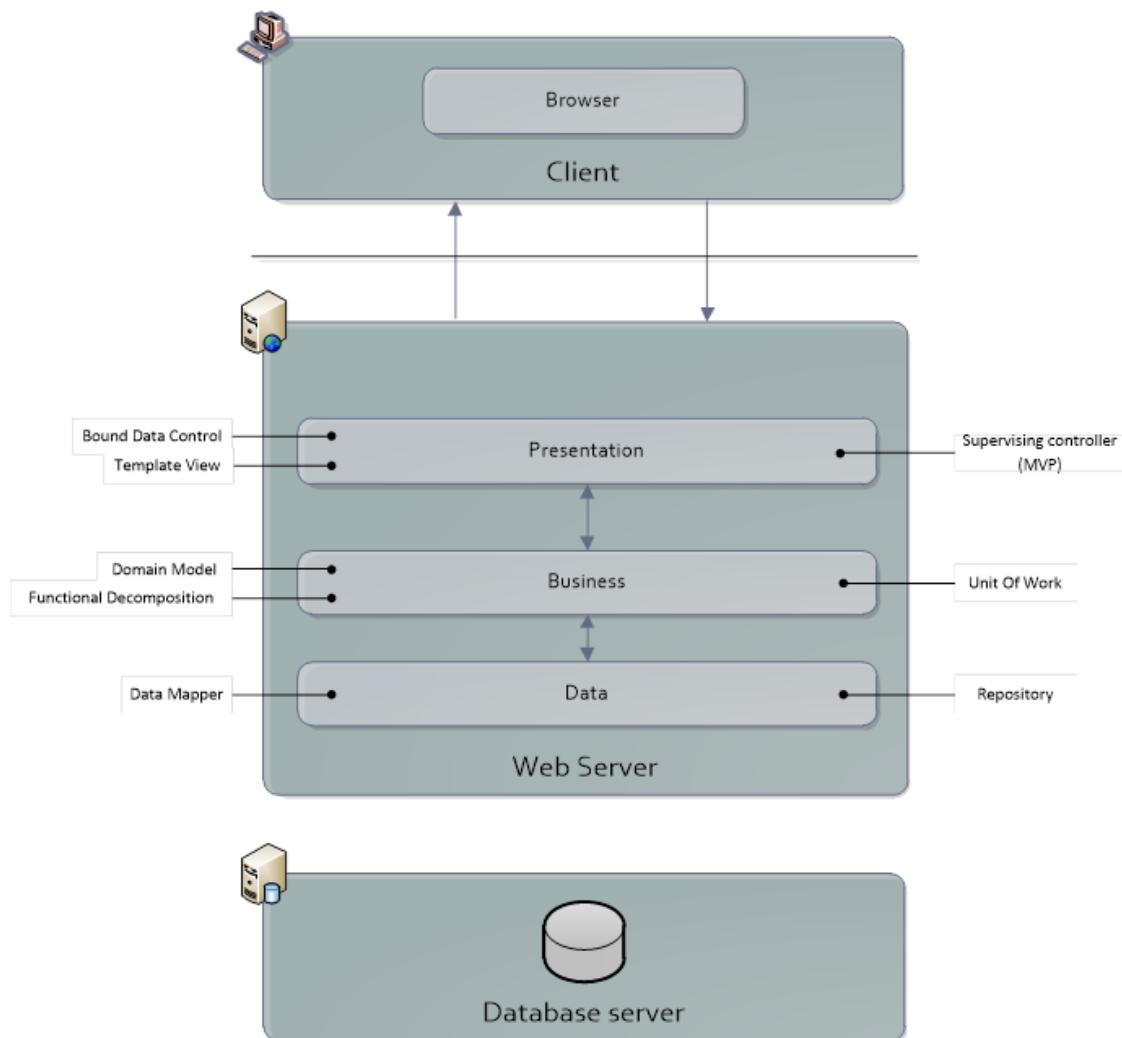


## Key Characteristics

- Stand-alone ASP.NET Web application that supports complex data models.
- Presentation and Business logic are located on the same physical machine.
- Browser interaction with the Web Server uses standard HTTP GET and POST requests.
- The application has full autonomy over the database schema.

## Architectural Patterns

The following diagram displays the major patterns used by this implementation strategy and the layers where those patterns are implemented.



The following is a summary of the patterns used by this scenario:

- User Interface processing is handled by a Supervising Controller pattern.
- The Template View pattern is used to define a common look and feel.
- Controls are bound to objects that contain data.
- The business layer uses a Façade pattern to implement a message-based interface between the presentation and business layer.
- The Domain model pattern is used to model the application domain.
- The Unit of Work pattern is used to keep track of everything you do during a business transaction that can affect the database. When you're done, it figures out everything that needs to be done to alter the database as a result of your work.
- A Repository pattern is used to access the Data Mapper entities.
- A Data Mapper pattern is used to map domain entities to the database schema and make the Domain Model persistence ignorant.

Pattern	Description
Bound Data Control	Control that can be bound to a data structure, which is used to provide data displayed by the control.

Template View	Combine individual views into a composite representation.
Data Mapper	Implement a mapping layer between objects and the database structure in order to move data from one structure to another while keeping them independent.
Domain Model	A Domain Model creates a web of interconnected objects, where each object represents some meaningful individual, which may be as large as a corporation, or as small as a single line on an order.
Supervising Controller	Separates the user interface code into three separate units; Model (data), View (user interface), and Presenter (processing logic), with a focus on the view.
Repository	A system with a complex domain model often benefits from a layer, such as the one provided by Data Mapper, that isolates domain objects from details of the database access code. In such systems it can be worthwhile to build another layer of abstraction over the mapping layer where query construction code is concentrated. This becomes more important when there are a large number of domain classes or heavy querying. In these cases particularly, adding this layer helps minimize duplicate query logic. A Repository mediates between the domain and data mapping layers, acting like an in-memory domain object collection. Client objects construct query specifications declaratively and submit them to Repository for satisfaction. Repository also supports the objective of achieving a clean separation and one-way dependency between the domain and data mapping layers.
Dependency Inversion	Use a base class or interface to define a shared abstraction that can be used by multiple layers in the design. The principal behind dependency inversion is that high level components should not be dependent on low level components. Instead, both should depend on an external abstraction.
Unit of Work	According to Martin Fowler, the Unit of Work pattern "maintains a list of objects affected by a business transaction and coordinates the writing out of changes and the resolution of concurrency problems."

## More intro stuff that needs update goes here

## System Design

### Business Objects

#### User Class Diagram

**DigicircUser**

↗ ApplicationSystemBO ApplicationUser

*UserName: string (255) [P, RQ]*  
*PasswordHash: string (2147483647) [P]*  
*SecurityStamp: string (2147483647) [P]*  
*EmailConfirmed: bool [P]*  
*LockoutEnabled: bool [P]*  
*PhoneNumberConfirmed: bool [P]*  
*TwoFactorEnabled: bool [P]*  
*AccessFailedCount: int [P]*  
*Name: string (256) [P]*  
*Email: string (255) [P]*  
*PhoneNumber: string (255) [P]*  
*LockoutEndDate: DateTime [P]*  
*FirstName: string (256) [P]*  
*LastName: string (256) [P]*  
*SubscribeToNewsLetter: bool [P]*  
*Permissions: Collection[ApplicationPermission]*  
*Roles: Collection[ApplicationRole]*  
*Clients: Collection[ApplicationClient]*  
*Logins: Collection[ApplicationUserLogin]*  
*Claims: Collection[ApplicationUserClaim]*  
*Profile: Profile*

## User Classes

### DigicircUser Class

Persisted:  Identity: *UserName*

#### Attributes

Name	Datatype	Persisted	Required	Encrypted
FirstName	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
LastName	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
SubscribeToNewsLetter	bool	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

#### Operations

Name: *IsInRole*

```
function bool IsInRole(string roleName)
{
    return this.Roles.Any(r => r.Name == roleName);
}
```

Name: HasPermission

```
function bool HasPermission(string permission)
{
    bool hasPermissionfromRoles = this.Roles.Any(rr => rr.Permissions.Any(pp => pp.Name
== permission));
    return hasPermissionfromRoles || this.Permissions.Any(pp => pp.Name ==
permission);
}
```

## User Associations

Actor - DigicircUser Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>Company</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>DigicircUser</b>	<i>AddedBy</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Actor - DigicircUser Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>ActorsCanManage</i>	<input checked="" type="checkbox"/>	*****
<b>DigicircUser</b>	<i>Administrators</i>	<input checked="" type="checkbox"/>	*****

Material - DigicircUser Accosiation

	Member	Navigable	Multiplicity
<b>Material</b>	<i>Material</i>	<input checked="" type="checkbox"/>	*****
<b>DigicircUser</b>	<i>RequestedBy</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

ApplicationUser - ApplicationPermission Accosiation

	Member	Navigable	Multiplicity
<b>ApplicationUser</b>	<i>Users</i>	<input checked="" type="checkbox"/>	*****
<b>ApplicationPermission</b>	<i>Permissions</i>	<input checked="" type="checkbox"/>	*****

ApplicationUser - ApplicationRole Accosiation

	Member	Navigable	Multiplicity
<b>ApplicationUser</b>	<i>Users</i>	<input checked="" type="checkbox"/>	*****
<b>ApplicationRole</b>	<i>Roles</i>	<input checked="" type="checkbox"/>	*****

ApplicationUser - ApplicationUser Accosiation

	Member	Navigable	Multiplicity

<b>ApplicationClient</b>	<i>Clients</i>	<input checked="" type="checkbox"/>	*****
<b> ApplicationUser</b>	<i>User</i>	<input checked="" type="checkbox"/>	<b>1</b>

ApplicationUser - ApplicationUserLogin Accosiation

	Member	Navigable	Multiplicity
<b> ApplicationUser</b>	<i>User</i>	<input checked="" type="checkbox"/>	<b>1</b>
<b> ApplicationUserLogin</b>	<i>Logins</i>	<input checked="" type="checkbox"/>	*****

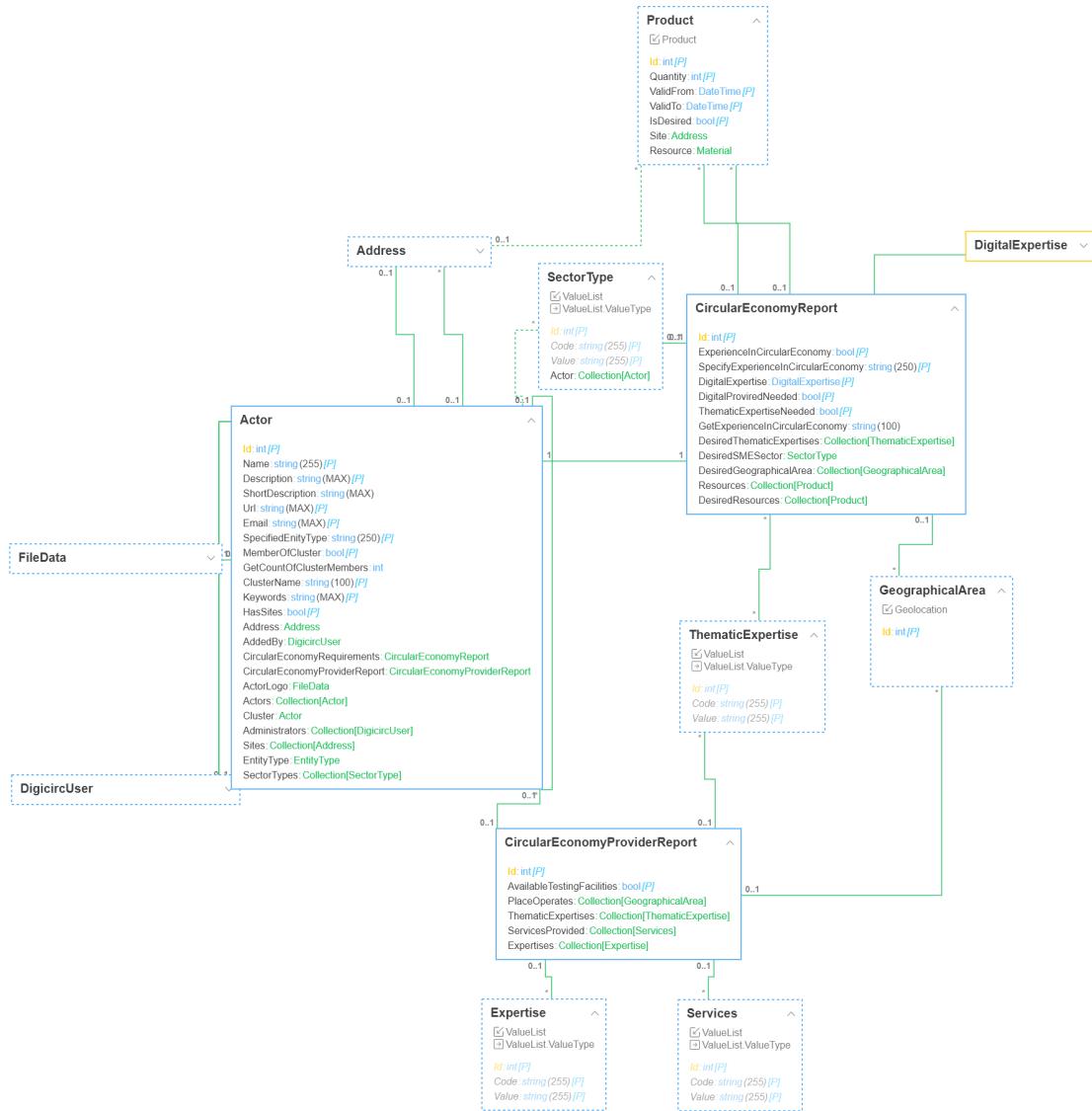
ApplicationUserClaim - ApplicationUser Accosiation

	Member	Navigable	Multiplicity
<b> ApplicationUserClaim</b>	<i>Claims</i>	<input checked="" type="checkbox"/>	*****
<b> ApplicationUser</b>	<i>User</i>	<input checked="" type="checkbox"/>	<b>1</b>

Profile - ApplicationUser Accosiation

	Member	Navigable	Multiplicity
<b> Profile</b>	<i>Profile</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b> ApplicationUser</b>	<i>ApplicationUser</i>	<input type="checkbox"/>	<b>0..1</b>

## Actors Class Diagram



## Actors Classes

### Actor Class

Persisted:  Identity: *Id*

### Attributes

Name	Datatype	Persisted	Required	Encrypted
Id	int	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Name	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Description	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
ShortDescription	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Url	string	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Email	string	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
SpecifiedEntityType	string	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
MemberOfCluster	bool	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
GetCountOfClusterMembers	int	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
ClusterName	string	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Keywords	string	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
HasSites	bool	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>

## Operations

Name: GetActorsDataset

```
static function Collection[Domain.Actor] GetActorsDataset(Collection[string] names)
{
    Collection[Domain.Actor] actors;
    if(names.Length == 0)
    {
        actors = Domain.Actor.GetAll();
    }
    else
    {
        //var onlyNames = names.Select(n => n.Name);
        actors = Domain.Actor.GetAll().Where(a => names.Contains(a.Name));
    }

    //    foreach Domain.ActorNames actorName in names
    //    {
    //        var actor = Domain.Actor.Find(a => a.Name == actorName.Name).First();
    //        if(actor == null)
    //        {
    //            continue;
    //        }
    //        actors.Add(actor);
    //    }
    return actors;
}
```

Name: GetActorsFromGraphResponse

```
static function Collection[Domain.Actor]
GetActorsFromGraphResponse(Domain.GraphBackendResponse response)
{
    Collection[Domain.Actor] actors;
    foreach Domain.Nodes node in response.Nodes.Where(n => n.Label == "Company")
    {
        var actor = Domain.Actor.Find(a => a.Name == node.Name).First();
        if(actor == null)
```

```

    {
        continue;
    }
    actors.Add(actor);
}
return actors;
}

```

#### Name: InitNewActor

```

static function Domain.Actor InitNewActor()
{
    Domain.Actor actor;

    actor.CircularEconomyRequirements = Domain.CircularEconomyReport.Create();
    actor.EntityType = Domain.EntityType.GetAll().First();

    return actor;
}

```

#### Name: GetShortDescription

```

function string GetShortDescription()
{

    var descLength =  this.Description.Length;
    if(descLength < 250)
    {
        return this.Description;
    }
    return this.Description.Substring(0, 250) + "...";
}

```

#### Name: GetActorNamesFromGraphResponse

```

static function Collection[Domain.ActorNames]
GetActorNamesFromGraphResponse(Domain.GraphBackendResponse response)
{
    Collection[Domain.ActorNames] actorNames;
    Collection[string] types = Domain.EntityType.GetAll().Select(a => a.Code);
    foreach Domain.Nodes node in response.Nodes.Where(n => types.Contains(n.Label))
    {
        Domain.ActorNames actorName = Domain.ActorNames.Create();
        actorName.Name = node.Name;
        actorNames.Add(actorName);
    }
    return actorNames;
}

```

#### Name: GetGetCountOfClusterMembers

```

function int GetGetCountOfClusterMembers()
{

```

```

    if(this.EntityType.IsCluster)
    {
        return this.Actors.Length;
    }
    return 0;
}

```

#### Name: AddProductFromAPI

```

function void AddProductFromAPI(Domain.Product product, bool desired)
{
    //product type
    if(product.Resource.Type != null && Domain.ProductType.Find(a => a.Name == product.Resource.Type.Name).Length > 0)
    {
        var productTypeDb = Domain.ProductType.Find(a => a.Name == product.Resource.Type.Name).First();
        product.Resource.Type = productTypeDb;
    }
    //unit of measurement
    if(product.Resource.UnitOfMeasurement != null && Domain.UnitOfMeasurement.Find(a => a.Code == product.Resource.UnitOfMeasurement.Code).Length > 0)
    {
        var unitOfMeasurementDb = Domain.UnitOfMeasurement.Find(a => a.Code == product.Resource.UnitOfMeasurement.Code).First();
        product.Resource.UnitOfMeasurement = unitOfMeasurementDb;
    }
    //material
    if(product.Resource != null && Domain.Material.Find(a => a.Name == product.Resource.Name).Length > 0)
    {
        var materialDb = Domain.Material.Find(a => a.Name == product.Resource.Name).First();
        product.Resource = materialDb;
    }
    //physical Form
    if(product.Resource.PhysicalForm != null && Domain.PhysicalForm.Find(a => a.Code == product.Resource.PhysicalForm.Code).Length > 0)
    {
        var physicalFormDb = Domain.PhysicalForm.Find(a => a.Code == product.Resource.PhysicalForm.Code).First();
        product.Resource.PhysicalForm = physicalFormDb;
    }
    if(product.Site != null && Domain.Address.Find(a => a.Alias == product.Site.Alias).Length > 0)
    {
        var siteDb = Domain.Address.Find(a => a.Alias == product.Site.Alias).First();
        product.Site = siteDb;
    }
    //site
    if(desired)

```

```

{
    this.CircularEconomyRequirements.DesiredResources.Add(product);
}
else
{
    this.CircularEconomyRequirements.Resources.Add(product);
}
this.Save();
}

```

#### Name: GetAddresses

```

static function Collection[Domain.Address] GetAddresses(int id)
{
    Collection[Domain.Address] addresses;
    Domain.Actor actor = Domain.Actor.GetByKey(id);
    if(actor.Address != null)
    {
        addresses.Add(actor.Address);
    }
    if(actor.HasSites)
    {
        addresses.AddRange(actor.Sites);
    }
    return addresses;
}

```

#### Name: GetActorNamesFromElasticResponse

```

static function Collection[Domain.ActorNames]
GetActorNamesFromElasticResponse(Interfaces.ElasticSearch.SearchResponse response)
{
    Collection[Domain.ActorNames] actorNames;
    var actors = response.Hits.Hits.ToCollection().Select(a => a.Source.Name);
    foreach string name in actors
    {
        Domain.ActorNames actorName = Domain.ActorNames.Create();
        actorName.Name = name;
        actorNames.Add(actorName);
    }
    return actorNames;
}

```

#### Name: Match

```

function void Match()
{
//    foreach Domain.Product product in
this.CircularEconomyRequirements.DesiredResources
//    {
//        Domain.ListProducersMaterialRequest req;
//        Domain.ListProducersMaterialStatements stat;
//
}

```

```

//      stat.Statement = "MATCH p= (producer:Actor)-[*]->(:Material {Id:
$props.MaterialId})-[*]->(consumer:Actor {Id: $props.ActorId}) WHERE NOT (consumer)-->
() RETURN producer";
//
//      stat.Parameters = Domain.ListProducersMaterialParameters.Create();
//      stat.Parameters.Properties = Domain.ListProducersMaterialProps.Create();
//
//      stat.Parameters.Properties.ActorId = this.Id;
//      stat.Parameters.Properties.MaterialId = product.Resource.Id;
//
//      req.Statements.Add(stat);
//
//      var reqParsed =
DataTransformations.KnowledgeBase.ListProducersMaterialRequest_To_ListProducersMaterialR
//
//      CommonLib.Serializer[Interfaces.KnowledgeBase.ListProducersMaterialRequest]
ser;
//      DebugLib.Logger.WriteLine("request " + serToJson(reqParsed));
//
//      Interfaces.KnowledgeBase.KnowledgeBaseResult result =
Interfaces.KnowledgeBase.API.ListPossibleProducersForMaterial(reqParsed);
//
//      CommonLib.Serializer[Interfaces.KnowledgeBase.KnowledgeBaseResult] serRes;
//      DebugLib.Logger.WriteLine("result " +
result.Results.Get(0).Data.Get(0).Row.Get(0).Name);
//    }
}

```

#### Name: ListPossibleMatches

```

function Collection[Domain.ActorNames] ListPossibleMatches(bool offers)
{
    Domain.ListProducersMaterialRequest req;
    Domain.ListProducersMaterialStatements stat;

    if(offers)
    {
        stat.Statement = "MATCH p=(:Actor{Id: $props.ActorId})-[*]->(:Material)-*
[*]->(a:Actor) RETURN a";
    }
    else
    {
        stat.Statement = "MATCH p=(a:Actor)-[*]->(:Material)-[*]->(:Actor {Id:
$props.ActorId}) RETURN a";
    }

    stat.Parameters = Domain.ListProducersMaterialParameters.Create();
    stat.Parameters.Properties = Domain.ListProducersMaterialProps.Create();

    stat.Parameters.Properties.ActorId = this.Id;
}

```

```

        req.Statements.Add(stat);

        var reqParsed =
DataTransformations.KnowledgeBase.ListProducersMaterialRequest_To_ListProducersMaterialR

        CommonLib.Serializer[Interfaces.KnowledgeBase.ListProducersMaterialRequest]
ser;
        DebugLib.Logger.WriteLine("request " + ser.ToString(reqParsed));

        Interfaces.KnowledgeBase.KnowledgeBaseResult result =
Interfaces.KnowledgeBase.API.ListPossibleMatches(reqParsed);

        CommonLib.Serializer[Interfaces.KnowledgeBase.KnowledgeBaseResult] serRes;
        DebugLib.Logger.WriteLine("result " + serRes.ToString(result));

        Collection[Domain.ActorNames] names;
        if(result.Results.Length > 0 && result.Results.Get(0).Data.Length > 0)
        {
            for int i = 0; i < result.Results.Get(0).Data.Length; i + 1
            {
                Domain.ActorNames name;
                name.Name = result.Results.Get(0).Data.Get(i).Row.Get(0).Name;
                names.Add(name);
            }
        }
        return names;
    }
}

```

## DigitalExpertise Enumeration

### Values

- None
- Average
- Medium
- Sufficient
- High

## CircularEconomyReport Class

Persisted:  Identity: *Id*

### Attributes

Name	Datatype	Persisted	Required	Encrypted
Id	int	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
ExperiencelnCircularEconomy	bool	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
SpecifyExperiencelnCircularEconomy	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
DigitalExpertise	DigitalExpertise	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
DigitalProviredNeeded	bool	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

ThematicExpertiseNeeded	bool	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
GetExperiencelnCircularEconomy	string	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

## Operations

Name: GetGetExperienceInCircularEconomy

```
function string GetGetExperienceInCircularEconomy()
{
    return this.ExperienceInCircularEconomy? "yes" : "no";
}
```

## CircularEconomyProviderReport Class

Persisted:  Identity: *Id*

### Attributes

Name	Datatype	Persisted	Required	Encrypted
Id	int	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
AvailableTestingFacilities	bool	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

## Actors Associations

Actor - Address Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>Company</i>	<input type="checkbox"/>	<b>0..1</b>
<b>Address</b>	<i>Address</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Actor - DigicircUser Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>Company</i>	<input type="checkbox"/>	<b>0..1</b>
<b>DigicircUser</b>	<i>AddedBy</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

CircularEconomyReport - ThematicExpertise Accosiation

	Member	Navigable	Multiplicity
<b>CircularEconomyReport</b>	<i>CircularEconomyInformation</i>	<input type="checkbox"/>	*****
<b>ThematicExpertise</b>	<i>DesiredThematicExpertises</i>	<input checked="" type="checkbox"/>	*****

CircularEconomyReport - SectorType Accosiation

	Member	Navigable	Multiplicity
<b>CircularEconomyReport</b>	<i>CircularEconomyInformation</i>	<input type="checkbox"/>	<b>0..1</b>

<b>SectorType</b>	<i>DesiredSMESector</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
-------------------	-------------------------	-------------------------------------	-------------

CircularEconomyReport - Actor Accosiation

	Member	Navigable	Multiplicity
<b>CircularEconomyReport</b>	<i>CircularEconomyRequirements</i>	<input checked="" type="checkbox"/>	<b>1</b>
<b>Actor</b>	<i>Actor</i>	<input type="checkbox"/>	<b>1</b>

CircularEconomyReport - GeographicalArea Accosiation

	Member	Navigable	Multiplicity
<b>CircularEconomyReport</b>	<i>CircularEconomyInformation</i>	<input type="checkbox"/>	<b>0..1</b>
<b>GeographicalArea</b>	<i>DesiredGeographicalArea</i>	<input checked="" type="checkbox"/>	*****

GeographicalArea - CircularEconomyProviderReport Accosiation

	Member	Navigable	Multiplicity
<b>GeographicalArea</b>	<i>PlaceOperates</i>	<input checked="" type="checkbox"/>	*****
<b>CircularEconomyProviderReport</b>	<i>CircularEconomyProviderReport</i>	<input type="checkbox"/>	<b>0..1</b>

CircularEconomyProviderReport - ThematicExpertise Accosiation

	Member	Navigable	Multiplicity
<b>CircularEconomyProviderReport</b>	<i>CircularEconomyProviderReport</i>	<input type="checkbox"/>	<b>0..1</b>
<b>ThematicExpertise</b>	<i>ThematicExpertises</i>	<input checked="" type="checkbox"/>	*****

Services - CircularEconomyProviderReport Accosiation

	Member	Navigable	Multiplicity
<b>Services</b>	<i>ServicesProvided</i>	<input checked="" type="checkbox"/>	*****
<b>CircularEconomyProviderReport</b>	<i>CircularEconomyProviderReport</i>	<input type="checkbox"/>	<b>0..1</b>

CircularEconomyProviderReport - Expertise Accosiation

	Member	Navigable	Multiplicity
<b>CircularEconomyProviderReport</b>	<i>CircularEconomyProviderReport</i>	<input type="checkbox"/>	<b>0..1</b>
<b>Expertise</b>	<i>Expertises</i>	<input checked="" type="checkbox"/>	*****

Actor - CircularEconomyProviderReport Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>Actor</i>	<input type="checkbox"/>	<b>0..1</b>
<b>CircularEconomyProviderReport</b>	<i>CircularEconomyProviderReport</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Actor - FileData Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>Actor</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>FileData</b>	<i>ActorLogo</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Actor - Actor Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>Cluster</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Actor</b>	<i>Actors</i>	<input checked="" type="checkbox"/>	*****

Actor - DigicircUser Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>ActorsCanManage</i>	<input checked="" type="checkbox"/>	*****
<b>DigicircUser</b>	<i>Administrators</i>	<input checked="" type="checkbox"/>	*****

CircularEconomyReport - Product Accosiation

	Member	Navigable	Multiplicity
<b>CircularEconomyReport</b>	<i>CircularEconomyReport</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Product</b>	<i>Resources</i>	<input checked="" type="checkbox"/>	*****

CircularEconomyReport - Product Accosiation

	Member	Navigable	Multiplicity
<b>CircularEconomyReport</b>	<i>_CircularEconomyReport_1_</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Product</b>	<i>DesiredResources</i>	<input checked="" type="checkbox"/>	*****

Actor - Address Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>Actor</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Address</b>	<i>Sites</i>	<input checked="" type="checkbox"/>	*****

Actor - EntityType Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>Actor</i>	<input checked="" type="checkbox"/>	*****
<b>EntityType</b>	<i>EntityType</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

SectorType - Actor Accosiation

	Member	Navigable	Multiplicity
SectorType	<i>SectorTypes</i>	<input checked="" type="checkbox"/>	*****
Actor	<i>Actor</i>	<input checked="" type="checkbox"/>	*****

GraphQuery - Actor Accosiation

	Member	Navigable	Multiplicity
GraphQuery	<i>GraphQuery</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
Actor	<i>SelectedActor</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Match - Actor Accosiation

	Member	Navigable	Multiplicity
Match	<i>Match</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
Actor	<i>ActorOffer</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Match - Actor Accosiation

	Member	Navigable	Multiplicity
Match	<i>_Match_1_</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
Actor	<i>ActorRequest</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Material - DigicircUser Accosiation

	Member	Navigable	Multiplicity
Material	<i>Material</i>	<input checked="" type="checkbox"/>	*****
DigicircUser	<i>RequestedBy</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

ApplicationUser - ApplicationPermission Accosiation

	Member	Navigable	Multiplicity
ApplicationUser	<i>Users</i>	<input checked="" type="checkbox"/>	*****
ApplicationPermission	<i>Permissions</i>	<input checked="" type="checkbox"/>	*****

ApplicationUser - ApplicationRole Accosiation

	Member	Navigable	Multiplicity
ApplicationUser	<i>Users</i>	<input checked="" type="checkbox"/>	*****
ApplicationRole	<i>Roles</i>	<input checked="" type="checkbox"/>	*****

ApplicationClient - ApplicationUser Accosiation

	Member	Navigable	Multiplicity

<b>ApplicationClient</b>	<i>Clients</i>	<input checked="" type="checkbox"/>	*****
<b>ApplicationUser</b>	<i>User</i>	<input checked="" type="checkbox"/>	<b>1</b>

ApplicationUser - ApplicationUserLogin Accosiation

	Member	Navigable	Multiplicity
<b> ApplicationUser</b>	<i>User</i>	<input checked="" type="checkbox"/>	<b>1</b>
<b> ApplicationUserLogin</b>	<i>Logins</i>	<input checked="" type="checkbox"/>	*****

ApplicationUserClaim - ApplicationUser Accosiation

	Member	Navigable	Multiplicity
<b> ApplicationUserClaim</b>	<i>Claims</i>	<input checked="" type="checkbox"/>	*****
<b> ApplicationUser</b>	<i>User</i>	<input checked="" type="checkbox"/>	<b>1</b>

Profile - ApplicationUser Accosiation

	Member	Navigable	Multiplicity
<b> Profile</b>	<i>Profile</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b> ApplicationUser</b>	<i>ApplicationUser</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Address - Country Accosiation

	Member	Navigable	Multiplicity
<b> Address</b>	<i>Address</i>	<input checked="" type="checkbox"/>	*****
<b> Country</b>	<i>Country</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Product - Address Accosiation

	Member	Navigable	Multiplicity
<b> Product</b>	<i>Product</i>	<input checked="" type="checkbox"/>	*****
<b> Address</b>	<i>Site</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

SearchQuery - SectorType Accosiation

	Member	Navigable	Multiplicity
<b> SearchQuery</b>	<i>SearchQuery</i>	<input checked="" type="checkbox"/>	*****
<b> SectorType</b>	<i>SelectedSector</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Product - Material Accosiation

	Member	Navigable	Multiplicity
<b> Product</b>	<i>Product</i>	<input checked="" type="checkbox"/>	*****

<b>Material</b>	<i>Resource</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
-----------------	-----------------	-------------------------------------	-------------

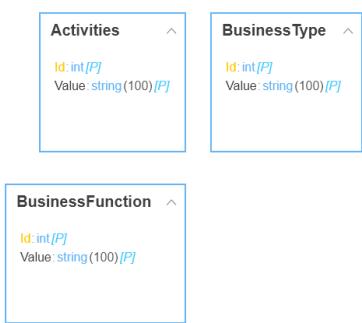
GraphQuery - Product Accosiation

	Member	Navigable	Multiplicity
<b>GraphQuery</b>	<i>GraphQuery</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Product</b>	<i>DesiredProduct</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

GraphQuery - Product Accosiation

	Member	Navigable	Multiplicity
<b>GraphQuery</b>	<i>_GraphQuery_1_</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Product</b>	<i>ResourceProduct</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

### **Business Class Diagram**



## Business Classes

### BusinessFunction Class

Persisted:  Identity: *Id*

#### Attributes

--	--	--	--	--	--

Name	Datatype	Persisted	Required	Encrypted
Id	int	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Value	string	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

### BusinessType Class

Persisted:  Identity: *Id*

#### Attributes

Name	Datatype	Persisted	Required	Encrypted
Id	int	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Value	string	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

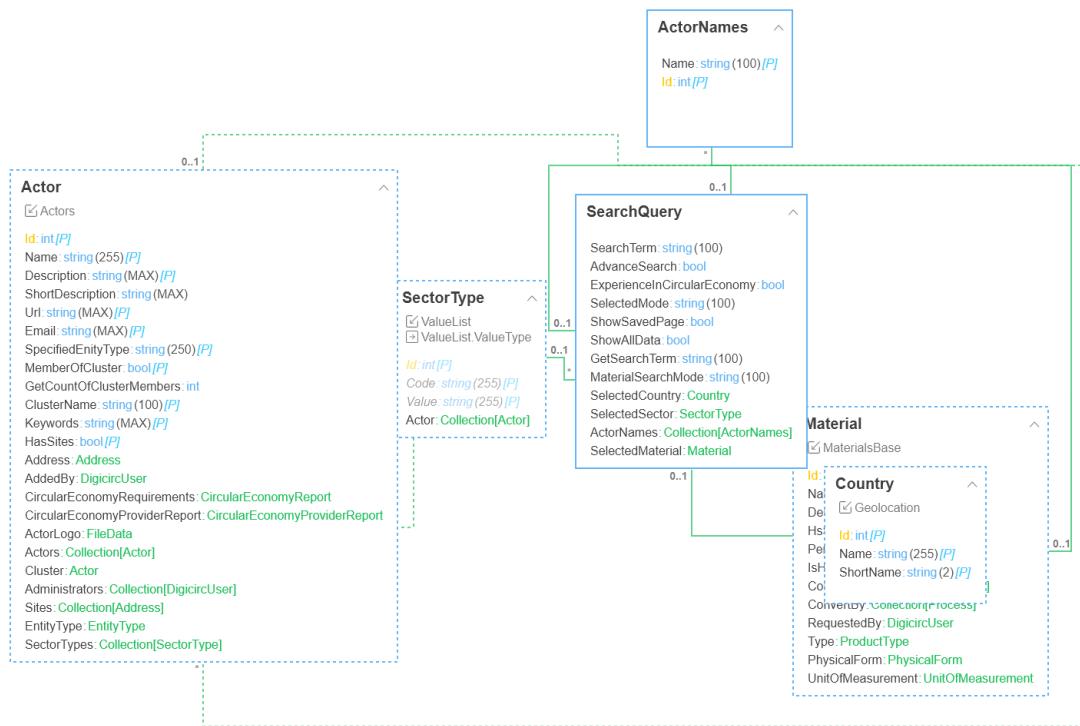
### Activities Class

Persisted:  Identity: *Id*

#### Attributes

Name	Datatype	Persisted	Required	Encrypted
Id	int	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Value	string	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

### SearchQuery Class Diagram



## SearchQuery Classes

### SearchQuery Class

Persisted:  Identity: \_\_

#### Attributes

Name	Datatype	Persisted	Required	Encrypted

SearchTerm	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
AdvanceSearch	bool	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
ExperiencelnCircularEconomy	bool	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
SelectedMode	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
ShowSavedPage	bool	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
ShowAllData	bool	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
GetSearchTerm	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
MaterialSearchMode	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

## Operations

Name: Reset

```
function void Reset()
{
    this.SearchTerm = "";
    this.SelectedCountry = Domain.Country.Create();
    this.SelectedSector = Domain.SectorType.Create();
    this.ActorNames = null;
}
```

Name: GetGetSearchTerm

```
function string GetGetSearchTerm()
{
    return "list \"\" + this.SearchTerm + "\"";
}
```

## ActorNames Class

Persisted:  Identity: *Id*

### Attributes

Name	Datatype	Persisted	Required	Encrypted
Name	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Id	int	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

## SearchQuery Associations

SearchQuery - Country Accosiation

	Member	Navigable	Multiplicity
<b>SearchQuery</b>	<i>SearchQuery</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Country</b>	<i>SelectedCountry</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

SearchQuery - SectorType Accosiation

	Member	Navigable	Multiplicity
SearchQuery	<i>SearchQuery</i>	<input checked="" type="checkbox"/>	*****
SectorType	<i>SelectedSector</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

SearchQuery - ActorNames Accosiation

	Member	Navigable	Multiplicity
SearchQuery	<i>SearchQuery</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
ActorNames	<i>ActorNames</i>	<input checked="" type="checkbox"/>	*****

SearchQuery - Material Accosiation

	Member	Navigable	Multiplicity
SearchQuery	<i>SearchQuery</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
Material	<i>SelectedMaterial</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Address - Country Accosiation

	Member	Navigable	Multiplicity
Address	<i>Address</i>	<input checked="" type="checkbox"/>	*****
Country	<i>Country</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Actor - Address Accosiation

	Member	Navigable	Multiplicity
Actor	<i>Company</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
Address	<i>Address</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Actor - DigicircUser Accosiation

	Member	Navigable	Multiplicity
Actor	<i>Company</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
DigicircUser	<i>AddedBy</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

CircularEconomyReport - Actor Accosiation

	Member	Navigable	Multiplicity
CircularEconomyReport	<i>CircularEconomyRequirements</i>	<input checked="" type="checkbox"/>	<b>1</b>
Actor	<i>Actor</i>	<input checked="" type="checkbox"/>	<b>1</b>

Actor - CircularEconomyProviderReport Accosiation

---

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>Actor</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>CircularEconomyProviderReport</b>	<i>CircularEconomyProviderReport</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Actor - FileData Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>Actor</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>FileData</b>	<i>ActorLogo</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Actor - Actor Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>Cluster</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Actor</b>	<i>Actors</i>	<input checked="" type="checkbox"/>	*****

Actor - DigicircUser Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>ActorsCanManage</i>	<input checked="" type="checkbox"/>	*****
<b>DigicircUser</b>	<i>Administrators</i>	<input checked="" type="checkbox"/>	*****

Actor - Address Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>Actor</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Address</b>	<i>Sites</i>	<input checked="" type="checkbox"/>	*****

Actor - EntityType Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>Actor</i>	<input checked="" type="checkbox"/>	*****
<b>EntityType</b>	<i>EntityType</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

SectorType - Actor Accosiation

	Member	Navigable	Multiplicity
<b>SectorType</b>	<i>SectorTypes</i>	<input checked="" type="checkbox"/>	*****
<b>Actor</b>	<i>Actor</i>	<input checked="" type="checkbox"/>	*****

GraphQuery - Actor Accosiation

	Member	Navigable	Multiplicity

<b>GraphQuery</b>	<i>GraphQuery</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Actor</b>	<i>SelectedActor</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Match - Actor Accosiation

	Member	Navigable	Multiplicity
<b>Match</b>	<i>Match</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Actor</b>	<i>ActorOffer</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Match - Actor Accosiation

	Member	Navigable	Multiplicity
<b>Match</b>	<i>_Match_1_</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Actor</b>	<i>ActorRequest</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

CircularEconomyReport - SectorType Accosiation

	Member	Navigable	Multiplicity
<b>CircularEconomyReport</b>	<i>CircularEconomyInformation</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>SectorType</b>	<i>DesiredSMESector</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

GraphQuery - ActorNames Accosiation

	Member	Navigable	Multiplicity
<b>GraphQuery</b>	<i>GraphQuery</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>ActorNames</b>	<i>ActorNames</i>	<input checked="" type="checkbox"/>	*****

Material - Process Accosiation

	Member	Navigable	Multiplicity
<b>Material</b>	<i>Product</i>	<input checked="" type="checkbox"/>	*****
<b>Process</b>	<i>ConvertedBy</i>	<input checked="" type="checkbox"/>	*****

Material - Process Accosiation

	Member	Navigable	Multiplicity
<b>Material</b>	<i>Source</i>	<input checked="" type="checkbox"/>	*****
<b>Process</b>	<i>ConvertBy</i>	<input checked="" type="checkbox"/>	*****

Material - DigiCircUser Accosiation

	Member	Navigable	Multiplicity
<b>Material</b>	<i>Material</i>	<input checked="" type="checkbox"/>	*****

DigicircUser	RequestedBy	<input checked="" type="checkbox"/>	0..1
--------------	-------------	-------------------------------------	------

Material - ProductType Accosiation

	Member	Navigable	Multiplicity
<b>Material</b>	<i>Material</i>	☒	*****
<b>ProductType</b>	<i>Type</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Material - PhysicalForm Accosiation

	Member	Navigable	Multiplicity
<b>Material</b>	<i>Material</i>	☒	*****
<b>PhysicalForm</b>	<i>PhysicalForm</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Material - UnitOfMeasurement Accosiation

	Member	Navigable	Multiplicity
<b>Material</b>	<i>Material</i>	☒	*****
<b>UnitOfMeasurement</b>	<i>UnitOfMeasurement</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

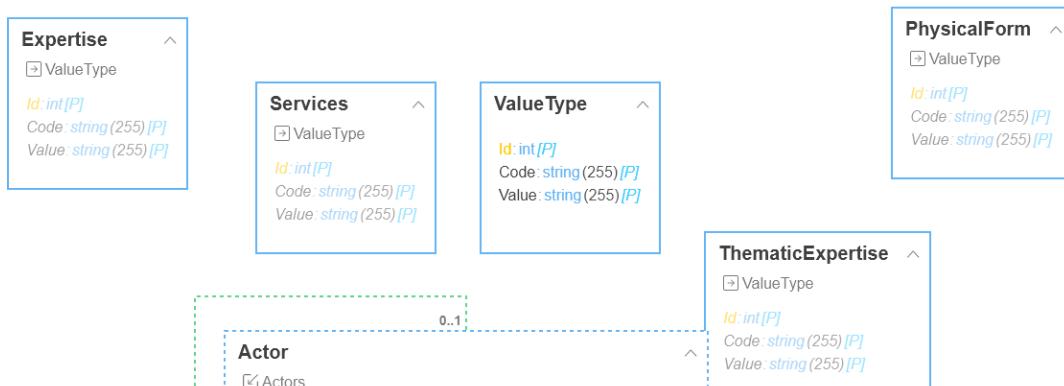
Product - Material Accosiation

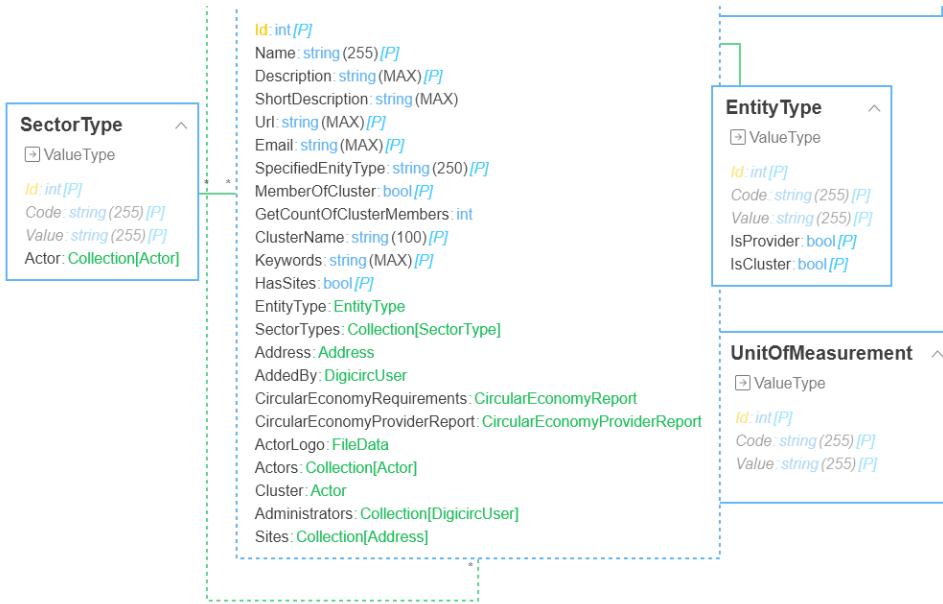
	Member	Navigable	Multiplicity
<b>Product</b>	<i>Product</i>	☒	*****
<b>Material</b>	<i>Resource</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Match - Material Accosiation

	Member	Navigable	Multiplicity
<b>Match</b>	<i>Match</i>	☒	<b>0..1</b>
<b>Material</b>	<i>Resource</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

## ValueList Class Diagram





## ValueList Classes

### SectorType Class

Persisted:  Identity: *Id*

#### Attributes

Name	Datatype	Persisted	Required	Encrypted

### EntityType Class

Persisted:  Identity: *Id*

#### Attributes

Name	Datatype	Persisted	Required	Encrypted
IsProvider	bool	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
IsCluster	bool	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

#### ValueType Class

Persisted:  Identity: *Id*

#### Attributes

Name	Datatype	Persisted	Required	Encrypted
Id	int	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Code	string	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Value	string	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

#### ThematicExpertise Class

Persisted:  Identity: *Id*

#### Attributes

Name	Datatype	Persisted	Required	Encrypted

#### Services Class

Persisted:  Identity: *Id*

#### Attributes

Name	Datatype	Persisted	Required	Encrypted

#### Expertise Class

Persisted:  Identity: *Id*

#### Attributes

Name	Datatype	Persisted	Required	Encrypted

#### PhysicalForm Class

Persisted:  Identity: *Id*

#### Attributes

Name	Datatype	Persisted	Required	Encrypted

#### UnitOfMeasurement Class

Persisted:  Identity: *Id*

#### Attributes

Name	Datatype	Persisted	Required	Encrypted

## ValueList Associations

Actor - EntityType Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>Actor</i>	<input checked="" type="checkbox"/>	*****
<b>EntityType</b>	<i>EntityType</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

SectorType - Actor Accosiation

	Member	Navigable	Multiplicity
<b>SectorType</b>	<i>SectorTypes</i>	<input checked="" type="checkbox"/>	*****
<b>Actor</b>	<i>Actor</i>	<input checked="" type="checkbox"/>	*****

CircularEconomyReport - SectorType Accosiation

	Member	Navigable	Multiplicity
<b>CircularEconomyReport</b>	<i>CircularEconomyInformation</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>SectorType</b>	<i>DesiredSMESector</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

SearchQuery - SectorType Accosiation

	Member	Navigable	Multiplicity
<b>SearchQuery</b>	<i>SearchQuery</i>	<input checked="" type="checkbox"/>	*****
<b>SectorType</b>	<i>SelectedSector</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Actor - Address Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>Company</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Address</b>	<i>Address</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Actor - DigicircUser Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>Company</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>DigicircUser</b>	<i>AddedBy</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

CircularEconomyReport - Actor Accosiation

	Member	Navigable	Multiplicity
<b>CircularEconomyReport</b>	<i>CircularEconomyRequirements</i>	<input checked="" type="checkbox"/>	<b>1</b>
<b>Actor</b>	<i>Actor</i>	<input checked="" type="checkbox"/>	<b>1</b>

Actor - CircularEconomyProviderReport Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>Actor</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>CircularEconomyProviderReport</b>	<i>CircularEconomyProviderReport</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Actor - FileData Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>Actor</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>FileData</b>	<i>ActorLogo</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Actor - Actor Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>Cluster</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Actor</b>	<i>Actors</i>	<input checked="" type="checkbox"/>	*****

Actor - DigicircUser Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>ActorsCanManage</i>	<input checked="" type="checkbox"/>	*****
<b>DigicircUser</b>	<i>Administrators</i>	<input checked="" type="checkbox"/>	*****

Actor - Address Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>Actor</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Address</b>	<i>Sites</i>	<input checked="" type="checkbox"/>	*****

GraphQuery - Actor Accosiation

	Member	Navigable	Multiplicity
<b>GraphQuery</b>	<i>GraphQuery</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Actor</b>	<i>SelectedActor</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Match - Actor Accosiation

	Member	Navigable	Multiplicity
<b>Match</b>	<i>Match</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Actor</b>	<i>ActorOffer</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Match - Actor Accosiation

	Member	Navigable	Multiplicity
<b>Match</b>	_Match_1_	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Actor</b>	<i>ActorRequest</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

CircularEconomyReport - ThematicExpertise Accosiation

	Member	Navigable	Multiplicity
<b>CircularEconomyReport</b>	<i>CircularEconomyInformation</i>	<input checked="" type="checkbox"/>	*****
<b>ThematicExpertise</b>	<i>DesiredThematicExpertises</i>	<input checked="" type="checkbox"/>	*****

CircularEconomyProviderReport - ThematicExpertise Accosiation

	Member	Navigable	Multiplicity
<b>CircularEconomyProviderReport</b>	<i>CircularEconomyProviderReport</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>ThematicExpertise</b>	<i>ThematicExpertises</i>	<input checked="" type="checkbox"/>	*****

Services - CircularEconomyProviderReport Accosiation

	Member	Navigable	Multiplicity
<b>Services</b>	<i>ServicesProvided</i>	<input checked="" type="checkbox"/>	*****
<b>CircularEconomyProviderReport</b>	<i>CircularEconomyProviderReport</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

CircularEconomyProviderReport - Expertise Accosiation

	Member	Navigable	Multiplicity
<b>CircularEconomyProviderReport</b>	<i>CircularEconomyProviderReport</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Expertise</b>	<i>Expertises</i>	<input checked="" type="checkbox"/>	*****

Material - PhysicalForm Accosiation

	Member	Navigable	Multiplicity
<b>Material</b>	<i>Material</i>	<input checked="" type="checkbox"/>	*****
<b>PhysicalForm</b>	<i>PhysicalForm</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Product - PhysicalForm Accosiation

	Member	Navigable	Multiplicity
<b>Product</b>	<i>Product</i>	<input checked="" type="checkbox"/>	*****
<b>PhysicalForm</b>	<i>PhysicalForm</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Material - UnitOfMeasurement Accosiation

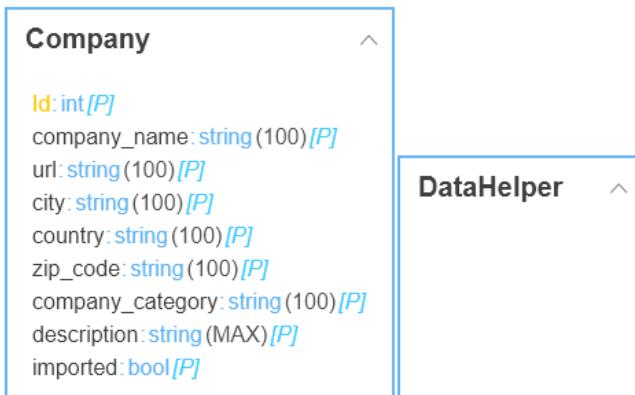
	Member	Navigable	Multiplicity

<b>Material</b>	<i>Material</i>	☒	*****
<b>UnitOfMeasurement</b>	<i>UnitOfMeasurement</i>	☒	<b>0..1</b>

Product - UnitOfMeasurement Accosiation

	Member	Navigable	Multiplicity
<b>Product</b>	<i>Product</i>	☒	*****
<b>UnitOfMeasurement</b>	<i>UnitOfMeasurement</i>	☒	<b>0..1</b>

### DataImportHelper Class Diagram



## DataImportHelper Classes

### Company Class

Persisted:  Identity: *Id*

#### Attributes

Name	Datatype	Persisted	Required	Encrypted
Id	int	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
company_name	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
url	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
city	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
country	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

zip_code	string	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
company_category	string	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
description	string	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
imported	bool	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

## Operations

Name: TransformToActor

```
static function void TransformToActor(Domain.EntityType entityType)
{
    foreach Domain.Company company in Domain.Company.GetAll()
    {
        Domain.Actor actor = Domain.Actor.Create();
        actor.EntityType = entityType;
        actor.Name = company.company_name;
        actor.Description = company.description;
        actor.Url = company.url;
        actor.Address = Domain.Address.Create();
        actor.Address.Town = company.city;
        actor.Address.Zip = company.zip_code;
        actor.Address.Country = Domain.Country.Find(a => a.ShortName ==
company.country).First();
        Domain.SectorType sector;
        var dbSector = Domain.SectorType.Find(s => s.Value ==
company.company_category).First();
        if(dbSector!= null)
        {
            sector = dbSector;
        }
        else
        {
            sector.Value = company.company_category;
        }
        Collection[Domain.SectorType] types;
        types.Add(sector);
        actor.SectorTypes = types;

        actor.CircularEconomyRequirements = Domain.CircularEconomyReport.Create();

        actor.Save();
    }
}
```

## DataHelper Class

Persisted:  Identity: \_\_

### Attributes

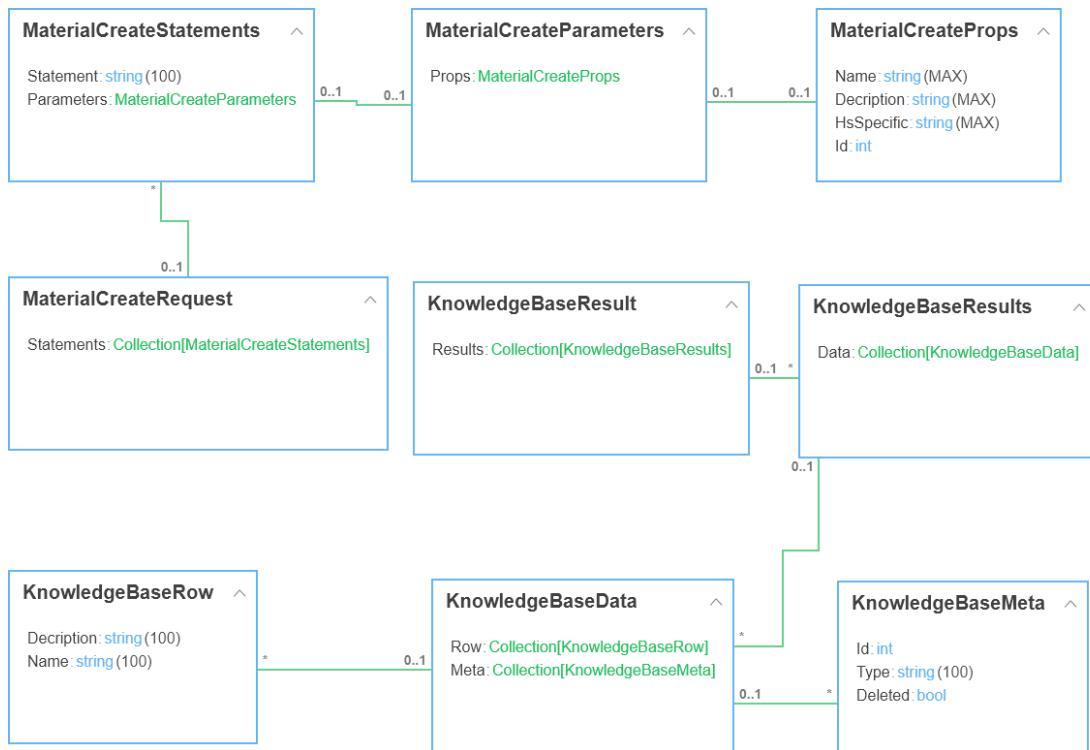
Name	Datatype	Persisted	Required	Encrypted

## Operations

Name: CleanDuplicateSectors

```
static function void CleanDuplicateSectors(Domain.SectorType sector)
{
    foreach Domain.Actor actor in Domain.Actor.GetAll()
    {
        if(actor.SectorTypes.Any(a => a.Value == sector.Value))
        {
            DebugLib.Logger.WriteLine("Found same sector value in actor");
            Domain.SectorType sectorToRemove = actor.SectorTypes.Where(a => a.Value == sector.Value).First();
            if(sectorToRemove.Id == sector.Id)
            {
                continue;
            }
            actor.SectorTypes.Remove(sectorToRemove);
            sectorToRemove.Actor = null;
            sectorToRemove.Delete();
            actor.SectorTypes.Add(sector);
            actor.Save();
        }
    }
}
```

## KnowledgeMaterialBase Class Diagram



## **KnowledgeMaterialBase Classes**

### **MaterialCreateRequest Class**

Persisted:  Identity: \_\_

#### **Attributes**

Name	Datatype	Persisted	Required	Encrypted
------	----------	-----------	----------	-----------

### **MaterialCreateStatements Class**

Persisted:  Identity: \_\_

#### **Attributes**

Name	Datatype	Persisted	Required	Encrypted
Statement	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

### **MaterialCreateParameters Class**

Persisted:  Identity: \_\_

#### **Attributes**

Name	Datatype	Persisted	Required	Encrypted
------	----------	-----------	----------	-----------

### **MaterialCreateProps Class**

Persisted:  Identity: \_\_

#### **Attributes**

Name	Datatype	Persisted	Required	Encrypted
Name	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Decription	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
HsSpecific	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Id	int	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

### **KnowledgeBaseResult Class**

Persisted:  Identity: \_\_

#### **Attributes**

Name	Datatype	Persisted	Required	Encrypted
------	----------	-----------	----------	-----------

### **KnowledgeBaseResults Class**

Persisted:  Identity: \_\_

#### **Attributes**

Name	Datatype	Persisted	Required	Encrypted
------	----------	-----------	----------	-----------

### **KnowledgeBaseData Class**

Persisted:  Identity: \_\_

**Attributes**

Name	Datatype	Persisted	Required	Encrypted
------	----------	-----------	----------	-----------

**KnowledgeBaseRow Class**Persisted:  Identity:   **Attributes**

Name	Datatype	Persisted	Required	Encrypted
Description	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Name	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

**KnowledgeBaseMeta Class**Persisted:  Identity:   **Attributes**

Name	Datatype	Persisted	Required	Encrypted
Id	int	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Type	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Deleted	bool	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

**KnowledgeMaterialBase Associations**

MaterialCreateRequest - MaterialCreateStatements Accosiation

	Member	Navigable	Multiplicity
<b>MaterialCreateRequest</b>	<i>MaterialCreateRequest</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>MaterialCreateStatements</b>	<i>Statements</i>	<input checked="" type="checkbox"/>	*****

MaterialCreateStatements - MaterialCreateParameters Accosiation

	Member	Navigable	Multiplicity
<b>MaterialCreateStatements</b>	<i>MaterialCreateStatements</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>MaterialCreateParameters</b>	<i>Parameters</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

MaterialCreateParameters - MaterialCreateProps Accosiation

	Member	Navigable	Multiplicity
<b>MaterialCreateParameters</b>	<i>MaterialCreateParameters</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>MaterialCreateProps</b>	<i>Props</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

KnowledgeBaseResult - KnowledgeBaseResults Accosiation

	Member	Navigable	Multiplicity

<b>KnowledgeBaseResult</b>	<i>KnowledgeBaseResult</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>KnowledgeBaseResults</b>	<i>Results</i>	<input checked="" type="checkbox"/>	*****

KnowledgeBaseResults - KnowledgeBaseData Accosiation

	Member	Navigable	Multiplicity
<b>KnowledgeBaseResults</b>	<i>KnowledgeBaseResults</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>KnowledgeBaseData</b>	<i>Data</i>	<input checked="" type="checkbox"/>	*****

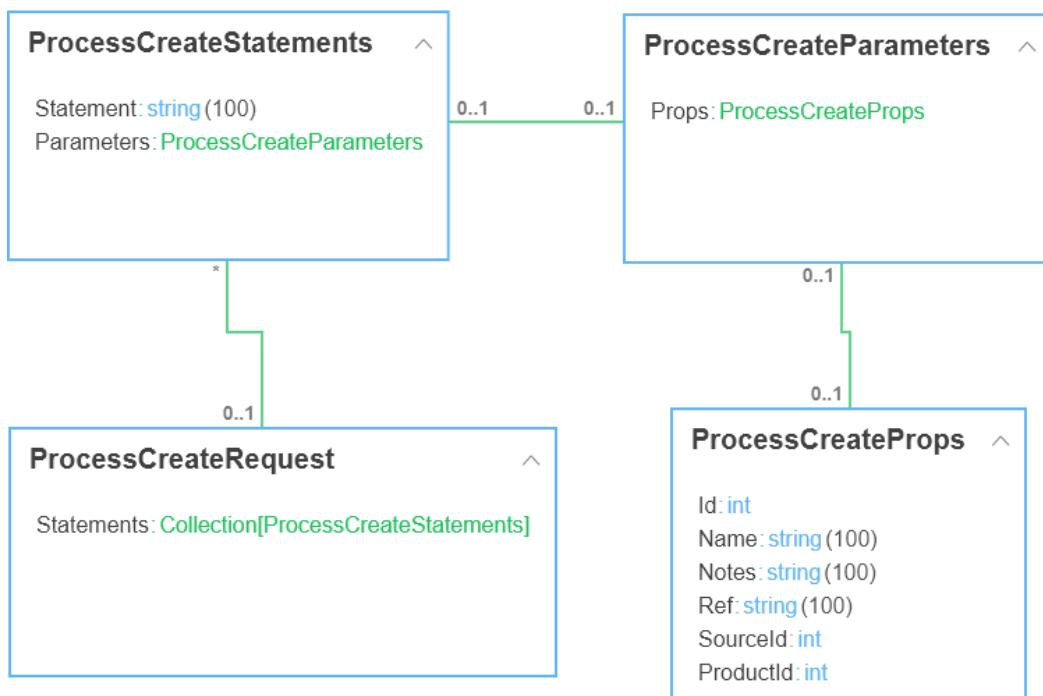
KnowledgeBaseData - KnowledgeBaseRow Accosiation

	Member	Navigable	Multiplicity
<b>KnowledgeBaseData</b>	<i>KnowledgeBaseData</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>KnowledgeBaseRow</b>	<i>Row</i>	<input checked="" type="checkbox"/>	*****

KnowledgeBaseData - KnowledgeBaseMeta Accosiation

	Member	Navigable	Multiplicity
<b>KnowledgeBaseData</b>	<i>KnowledgeBaseData</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>KnowledgeBaseMeta</b>	<i>Meta</i>	<input checked="" type="checkbox"/>	*****

## KnowledgeProcessBase Class Diagram



## **KnowledgeProcessBase Classes**

### **ProcessCreateRequest Class**

Persisted:  Identity: \_\_

#### Attributes

Name	Datatype	Persisted	Required	Encrypted
------	----------	-----------	----------	-----------

#### ProcessCreateStatements Class

Persisted:  Identity: \_\_

#### Attributes

Name	Datatype	Persisted	Required	Encrypted
Statement	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

#### ProcessCreateParameters Class

Persisted:  Identity: \_\_

#### Attributes

Name	Datatype	Persisted	Required	Encrypted
------	----------	-----------	----------	-----------

#### ProcessCreateProps Class

Persisted:  Identity: \_\_

#### Attributes

Name	Datatype	Persisted	Required	Encrypted
Id	int	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Name	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Notes	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Ref	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
SourceId	int	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
ProductId	int	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

#### KnowledgeProcessBase Associations

ProcessCreateRequest - ProcessCreateStatements Accosiation

	Member	Navigable	Multiplicity
<b>ProcessCreateRequest</b>	<i>ProcessCreateRequest</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>ProcessCreateStatements</b>	<i>Statements</i>	<input checked="" type="checkbox"/>	*****

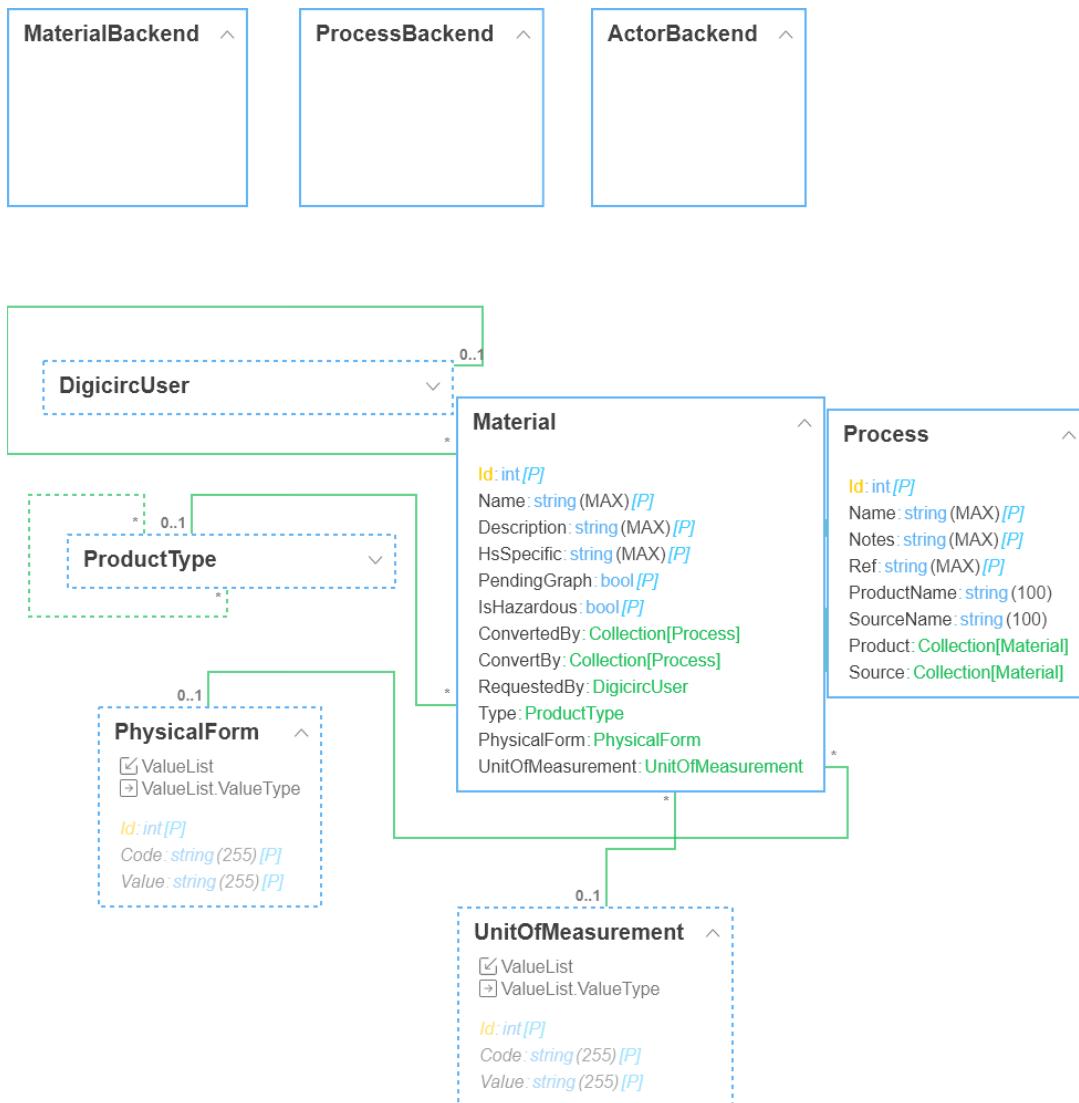
ProcessCreateStatements - ProcessCreateParameters Accosiation

	Member	Navigable	Multiplicity
<b>ProcessCreateStatements</b>	<i>ProcessCreateStatements</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>ProcessCreateParameters</b>	<i>Parameters</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

ProcessCreateParameters - ProcessCreateProps Association

	Member	Navigable	Multiplicity
<b>ProcessCreateParameters</b>	<i>ProcessCreateParameters</i>	☒	0..1
<b>ProcessCreateProps</b>	<i>Props</i>	☑	0..1

**MaterialsBase Class Diagram**



## MaterialsBase Classes

### Material Class

Persisted:  Identity: *Id*

#### Attributes

Name	Datatype	Persisted	Required	Encrypted
Id	int	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Name	string	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Description	string	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
HsSpecific	string	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
PendingGraph	bool	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
IsHazardous	bool	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

### Process Class

Persisted:  Identity: *Id*

#### Attributes

Name	Datatype	Persisted	Required	Encrypted
Id	int	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Name	string	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Notes	string	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Ref	string	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ProductName	string	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
SourceName	string	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

#### Operations

Name: GetProductName

```

function string GetProductName()
{
    Collection[string] productList;
    foreach Domain.Material product in this.Product
    {
        productList.Add(product.Name);
    }

    return string.Join(", ", productList);
}

```

Name: GetSourceName

```

function string GetSourceName()
{
    Collection[string] productList;
    foreach Domain.Material product in this.Source
    {
        productList.Add(product.Name);
    }

    return string.Join(", ", productList);
}

```

## **MaterialBackend Class**

Persisted:  Identity:   

### **Attributes**

Name	Datatype	Persisted	Required	Encrypted

### **Operations**

Name: CreateKnowledgeMaterial

```

static function void CreateKnowledgeMaterial(Domain.Material material)
{
    Domain.MaterialCreateRequest req;

    Domain.MaterialCreateStatements stat;

    stat.Statement = "CREATE (n:Material $props) RETURN n";

    stat.Parameters = Domain.MaterialCreateParameters.Create();
    stat.Parameters.Props = Domain.MaterialCreateProps.Create();

    stat.Parameters.Props.Id = material.Id;
    stat.Parameters.Props.Name = material.Name;
    stat.Parameters.Props.Description = material.Description;
    stat.Parameters.Props.HsSpecific = material.HsSpecific;

    req.Statements.Add(stat);

    var reqParsed =

```

```
DataTransformations.KnowledgeBase.MaterialCreateRequest_To_MaterialCreateRequestReversec  
    Interfaces.KnowledgeBase.API.CreateMaterial(reqParsed);  
}
```

#### Name: DeleteKnowledgeMaterial

```
static function void DeleteKnowledgeMaterial(Domain.Material material)  
{  
    Domain.MaterialCreateRequest req;  
  
    req.Statements.Add(Domain.MaterialBackend.DeleteMaterialStatement(material));  
  
    var reqParsed =  
DataTransformations.KnowledgeBase.MaterialCreateRequest_To_MaterialCreateRequestReversec  
    Interfaces.KnowledgeBase.API.CreateMaterial(reqParsed);  
}
```

#### Name: DeleteMaterialStatement

```
static function Domain.MaterialCreateStatements  
DeleteMaterialStatement(Domain.Material material)  
{  
    Domain.MaterialCreateStatements stat;  
  
    stat.Statement = "MATCH (n:Material {Id: $props.Id}) DETACH DELETE n";  
  
    stat.Parameters = Domain.MaterialCreateParameters.Create();  
    stat.Parameters.Props = Domain.MaterialCreateProps.Create();  
  
    stat.Parameters.Props.Id = material.Id;  
  
    return stat;  
}
```

#### Name: UpdateMaterialStatement

```
static function Domain.MaterialCreateStatements  
UpdateMaterialStatement(Domain.Material material)  
{  
    Domain.MaterialCreateStatements stat;  
  
    stat.Statement = "MATCH (m:Material {Id: $props.Id}) SET m = $props RETURN m";  
  
    stat.Parameters = Domain.MaterialCreateParameters.Create();  
    stat.Parameters.Props = Domain.MaterialCreateProps.Create();  
  
    stat.Parameters.Props.Id = material.Id;  
    stat.Parameters.Props.Name = material.Name;  
    stat.Parameters.Props.Description = material.Description;  
    stat.Parameters.Props.HsSpecific = material.HsSpecific;
```

```
        return stat;
    }
```

#### Name: UpdateKnowledgeMaterial

```
static function void UpdateKnowledgeMaterial(Domain.Material material)
{
    Domain.MaterialCreateRequest req;

    req.Statements.Add(Domain.MaterialBackend.UpdateMaterialStatement(material));

    var reqParsed =
DataTransformations.KnowledgeBase.MaterialCreateRequest_To_MaterialCreateRequestReversec

    Interfaces.KnowledgeBase.API.CreateMaterial(reqParsed);
}
```

#### ProcessBackend Class

Persisted:  Identity:   

#### Attributes

Name	Datatype	Persisted	Required	Encrypted

#### Operations

##### Name: CreateKnowledgeProcess

```
static function void CreateKnowledgeProcess(Domain.Process process)
{
    Domain.ProcessCreateRequest req;

    req.Statements.Add(Domain.ProcessBackend.CreateProcessStatement(process));

    var reqParsed =
DataTransformations.KnowledgeBase.ProcessCreateRequest_To_ProcessCreateRequest(req);
    Interfaces.KnowledgeBase.API.CreateProcess(reqParsed);

}
```

##### Name: CreateKnowledgeProcessPlus

```
static function void CreateKnowledgeProcessPlus(Domain.Process process, bool edit)
{
    Domain.ProcessCreateRequest req;

    req.Statements.Add(Domain.ProcessBackend.CreateProcessStatement(process));

    var source = process.Source.First();
    req.Statements.Add(Domain.ProcessBackend.CreateConvertByStatement(process,
source));

    var product = process.Product.First();
```

```

        req.Statements.Add(Domain.ProcessBackend.CreateConvertedByStatement(process,
product));

        var reqParsed =
DataTransformations.KnowledgeBase.ProcessCreateRequest_To_ProcessCreateRequest(req);
        Interfaces.KnowledgeBase.API.CreateProcess(reqParsed);
}

```

#### Name: CreateProcessStatement

```

static function Domain.ProcessCreateStatements CreateProcessStatement(Domain.Process
process)
{
    Domain.ProcessCreateStatements stat;
    stat.Statement = "CREATE (n:Process $props) RETURN n";
    stat.Parameters = Domain.ProcessCreateParameters.Create();
    stat.Parameters.Props = Domain.ProcessCreateProps.Create();
    stat.Parameters.Props.Id = process.Id;
    stat.Parameters.Props.Name = process.Name;
    stat.Parameters.Props.Notes = process.Notes;
    stat.Parameters.Props.Ref = process.Ref;
    return stat;
}

```

#### Name: CreateConvertedByStatement

```

static function Domain.ProcessCreateStatements
CreateConvertedByStatement(Domain.Process process, Domain.Material product)
{
    Domain.ProcessCreateStatements stat;
    stat.Statement = "MATCH (m:Material {Id: $props.ProductId}) MATCH (p:Process {Id:
$props.Id}) MERGE (p)-[rel:CONVERTED_BY]->(m) RETURN m";
    stat.Parameters = Domain.ProcessCreateParameters.Create();
    stat.Parameters.Props = Domain.ProcessCreateProps.Create();
    stat.Parameters.Props.Id = process.Id;
    stat.Parameters.Props.ProductId = product.Id;
    return stat;
}

```

#### Name: CreateConvertByStatement

```

static function Domain.ProcessCreateStatements CreateConvertByStatement(Domain.Process
process, Domain.Material product)
{
    Domain.ProcessCreateStatements stat;
    stat.Statement = "MATCH (m:Material {Id: $props.SourceId}) MATCH (p:Process {Id:
$props.Id}) MERGE (m)-[rel:CONVERT_BY]->(p) RETURN m";
    stat.Parameters = Domain.ProcessCreateParameters.Create();
    stat.Parameters.Props = Domain.ProcessCreateProps.Create();
    stat.Parameters.Props.Id = process.Id;
    stat.Parameters.Props.SourceId = product.Id;
    return stat;
}

```

#### Name: DeleteKnowledgeProcess

```
static function void DeleteKnowledgeProcess(Domain.Process process)
{
    Domain.ProcessCreateRequest req;

    req.Statements.Add(Domain.ProcessBackend.DeleteProcessStatement(process));
    var reqParsed =
DataTransformations.KnowledgeBase.ProcessCreateRequest_To_ProcessCreateRequest(req);
    Interfaces.KnowledgeBase.API.CreateProcess(reqParsed);
}
```

#### Name: DeleteProcessStatement

```
static function Domain.ProcessCreateStatements DeleteProcessStatement(Domain.Process
process)
{
    Domain.ProcessCreateStatements stat;
    stat.Statement = "MATCH (n:Process {Id: $props.Id}) DETACH DELETE n";
    stat.Parameters = Domain.ProcessCreateParameters.Create();
    stat.Parameters.Props = Domain.ProcessCreateProps.Create();
    stat.Parameters.Props.Id = process.Id;
    return stat;
}
```

#### Name: UpdateKnowledgeProcess

```
static function void UpdateKnowledgeProcess(Domain.Process process)
{
    Domain.ProcessCreateRequest req;

    req.Statements.Add(Domain.ProcessBackend.UpdateProcessStatement(process));
    var reqParsed =
DataTransformations.KnowledgeBase.ProcessCreateRequest_To_ProcessCreateRequest(req);
    Interfaces.KnowledgeBase.API.CreateProcess(reqParsed);
}
```

#### Name: UpdateProcessStatement

```
static function Domain.ProcessCreateStatements UpdateProcessStatement(Domain.Process
process)
{
    Domain.ProcessCreateStatements stat;
    stat.Statement = "MATCH (m:Process {Id: $props.Id}) SET m = $props RETURN m";
    stat.Parameters = Domain.ProcessCreateParameters.Create();
    stat.Parameters.Props = Domain.ProcessCreateProps.Create();
    stat.Parameters.Props.Id = process.Id;
    stat.Parameters.Props.Name = process.Name;
    stat.Parameters.Props.Notes = process.Notes;
    stat.Parameters.Props.Ref = process.Ref;
    return stat;
}
```

## **ActorBackend Class**

Persisted:  Identity:

### **Attributes**

Name	Datatype	Persisted	Required	Encrypted
------	----------	-----------	----------	-----------

### **Operations**

Name: CreateKnowledgeActor

```
static function void CreateKnowledgeActor(Domain.Actor actor)
{
    Domain.ActorCreateRequest req;

    Domain.ActorCreateStatements stat;

    stat.Statement = "MERGE (n:Actor {Id: $props.Id}) SET n = $props RETURN n";

    stat.Parameters = Domain.ActorCreateParameters.Create();
    stat.Parameters.Props = Domain.ActorCreateProps.Create();

    stat.Parameters.Props.Id = actor.Id;
    stat.Parameters.Props.Name = actor.Name;

    req.Statements.Add(stat);

    var reqParsed =
DataTransformations.KnowledgeBase.ActorCreateRequest_To_ActorCreateRequest(req);
    Interfaces.KnowledgeBase.API.CreateActor(reqParsed);
}
```

Name: ConnectActorOfferedBy

```
static function void ConnectActorOfferedBy(int actorId, Domain.Product product)
{
    Domain.ConnectActorMaterialRequest req;

    Domain.ConnectActorMaterialStatements stat;

    stat.Statement = "MATCH (a:Actor {Id: $props.ActorId}) MATCH (m:Material {Id:
$props.MaterialId}) MERGE (a)-[rel:OFFERED_BY]->(m)";

    stat.Parameters = Domain.ConnectActorMaterialParameters.Create();
    stat.Parameters.Props = Domain.ConnectActorMaterialProps.Create();

    stat.Parameters.Props.ActorId = actorId;
    stat.Parameters.Props.MaterialId = product.Resource.Id;

    req.Statements.Add(stat);

    var reqParsed =
DataTransformations.KnowledgeBase.ConnectActorMaterialRequest_To_ConnectActorMaterialReq
```

```

CommonLib.Serializer[Interfaces.KnowledgeBase.ConnectActorMaterialRequest] ser;
DebugLib.Logger.WriteLine("request " + ser.ToString(reqParsed));

Interfaces.KnowledgeBase.API.ConnectActorOfferedBy(reqParsed);
}

```

#### Name: ConnectActorRequests

```

static function void ConnectActorRequests(int actorId, Domain.Product product)
{
    Domain.ConnectActorMaterialRequest req;

    Domain.ConnectActorMaterialStatements stat;

    stat.Statement = "MATCH (a:Actor {Id: $props.ActorId}) MATCH (m:Material {Id: $props.MaterialId}) MERGE (m)-[rel:REQUESTED_BY]->(a)";

    stat.Parameters = Domain.ConnectActorMaterialParameters.Create();
    stat.Parameters.Props = Domain.ConnectActorMaterialProps.Create();

    stat.Parameters.Props.ActorId = actorId;
    stat.Parameters.Props.MaterialId = product.Resource.Id;

    req.Statements.Add(stat);

    var reqParsed =
        DataTransformations.KnowledgeBase.ConnectActorMaterialRequest_To_ConnectActorMaterialReq
    Interfaces.KnowledgeBase.API.ConnectActorRequests(reqParsed);
}

```

#### Name: DeleteKnowledgeActor

```

static function void DeleteKnowledgeActor(Domain.Actor actor)
{
    Domain.ActorCreateRequest req;

    req.Statements.Add(Domain.ActorBackend.PrepareDeleteKnowledgeActor(actor));

    var reqParsed =
        DataTransformations.KnowledgeBase.ActorCreateRequest_To_ActorCreateRequest(req);
    Interfaces.KnowledgeBase.API.CreateActor(reqParsed);
}

```

#### Name: PrepareDeleteKnowledgeActor

```

static function Domain.ActorCreateStatements PrepareDeleteKnowledgeActor(Domain.Actor
actor)
{
    Domain.ActorCreateStatements stat;

    stat.Statement = "MATCH (n:Actor {Id: $props.Id}) DETACH DELETE n";
}

```

```

stat.Parameters = Domain.ActorCreateParameters.Create();
stat.Parameters.Props = Domain.ActorCreateProps.Create();

stat.Parameters.Props.Id = actor.Id;
stat.Parameters.Props.Name = actor.Name;

return stat;
}

```

#### Name: UpdateKnowledgeActor

```

static function void UpdateKnowledgeActor(Domain.Actor actor)
{
    Domain.ActorCreateRequest req;

    req.Statements.Add(Domain.ActorBackend.PrepareUpdateKnowledgeActor(actor));

    var reqParsed =
DataTransformations.KnowledgeBase.ActorCreateRequest_To_ActorCreateRequest(req);
    Interfaces.KnowledgeBase.API.CreateActor(reqParsed);
}

```

#### Name: PrepareUpdateKnowledgeActor

```

static function Domain.ActorCreateStatements PrepareUpdateKnowledgeActor(Domain.Actor
actor)
{
    Domain.ActorCreateStatements stat;

    stat.Statement = "MATCH (n:Actor {Id: $props.Id}) SET m = $props RETURN m";

    stat.Parameters = Domain.ActorCreateParameters.Create();
    stat.Parameters.Props = Domain.ActorCreateProps.Create();

    stat.Parameters.Props.Id = actor.Id;
    stat.Parameters.Props.Name = actor.Name;

    return stat;
}

```

#### Name: DeleteRelationShips

```

static function void DeleteRelationShips(int actorId, int resourceId)
{
    Domain.ConnectActorMaterialRequest req;

    Domain.ConnectActorMaterialStatements stat;

    stat.Statement = "Match (a:Actor {Id: $props.ActorId})-[r]->(m:Material {Id:
$props.MaterialId}) DELETE r";

    stat.Parameters = Domain.ConnectActorMaterialParameters.Create();
}

```

```

stat.Parameters.Props = Domain.ConnectActorMaterialProps.Create();

stat.Parameters.Props.ActorId = actorId;
stat.Parameters.Props.MaterialId = resourceId;

req.Statements.Add(stat);

var reqParsed =
DataTransformations.KnowledgeBase.ConnectActorMaterialRequest_To_ConnectActorMaterialReq

```

CommonLib.Serializer[Interfaces.KnowledgeBase.ConnectActorMaterialRequest] ser;
DebugLib.Logger.WriteLine("request " + ser.ToString());

Interfaces.KnowledgeBase.API.DeleteRelationships(reqParsed);

}

## MaterialsBase Associations

Material - Process Accosiation

	Member	Navigable	Multiplicity
Material	<i>Product</i>	<input checked="" type="checkbox"/>	*****
Process	<i>ConvertedBy</i>	<input checked="" type="checkbox"/>	*****

Material - Process Accosiation

	Member	Navigable	Multiplicity
Material	<i>Source</i>	<input checked="" type="checkbox"/>	*****
Process	<i>ConvertBy</i>	<input checked="" type="checkbox"/>	*****

Material - DigicircUser Accosiation

	Member	Navigable	Multiplicity
Material	<i>Material</i>	<input checked="" type="checkbox"/>	*****
DigicircUser	<i>RequestedBy</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Material - ProductType Accosiation

	Member	Navigable	Multiplicity
Material	<i>Material</i>	<input checked="" type="checkbox"/>	*****
ProductType	<i>Type</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Material - PhysicalForm Accosiation

	Member	Navigable	Multiplicity
Material	<i>Material</i>	<input checked="" type="checkbox"/>	*****

<b>PhysicalForm</b>	<i>PhysicalForm</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
---------------------	---------------------	-------------------------------------	-------------

Material - UnitOfMeasurement Accosiation

	Member	Navigable	Multiplicity
<b>Material</b>	<i>Material</i>	<input checked="" type="checkbox"/>	*****
<b>UnitOfMeasurement</b>	<i>UnitOfMeasurement</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Product - Material Accosiation

	Member	Navigable	Multiplicity
<b>Product</b>	<i>Product</i>	<input checked="" type="checkbox"/>	*****
<b>Material</b>	<i>Resource</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Match - Material Accosiation

	Member	Navigable	Multiplicity
<b>Match</b>	<i>Match</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Material</b>	<i>Resource</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Actor - DigicircUser Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>Company</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>DigicircUser</b>	<i>AddedBy</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Actor - DigicircUser Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>ActorsCanManage</i>	<input checked="" type="checkbox"/>	*****
<b>DigicircUser</b>	<i>Administrators</i>	<input checked="" type="checkbox"/>	*****

ApplicationUser - ApplicationPermission Accosiation

	Member	Navigable	Multiplicity
<b>ApplicationUser</b>	<i>Users</i>	<input checked="" type="checkbox"/>	*****
<b>ApplicationPermission</b>	<i>Permissions</i>	<input checked="" type="checkbox"/>	*****

ApplicationUser - ApplicationRole Accosiation

	Member	Navigable	Multiplicity
<b>ApplicationUser</b>	<i>Users</i>	<input checked="" type="checkbox"/>	*****
<b>ApplicationRole</b>	<i>Roles</i>	<input checked="" type="checkbox"/>	*****

ApplicationClient - ApplicationUser Accosiation

	Member	Navigable	Multiplicity
<b>ApplicationClient</b>	<i>Clients</i>	<input checked="" type="checkbox"/>	*****
<b>ApplicationUser</b>	<i>User</i>	<input checked="" type="checkbox"/>	<b>1</b>

ApplicationUser - ApplicationUserLogin Accosiation

	Member	Navigable	Multiplicity
<b> ApplicationUser</b>	<i>User</i>	<input checked="" type="checkbox"/>	<b>1</b>
<b> ApplicationUserLogin</b>	<i>Logins</i>	<input checked="" type="checkbox"/>	*****

ApplicationUserClaim - ApplicationUser Accosiation

	Member	Navigable	Multiplicity
<b> ApplicationUserClaim</b>	<i>Claims</i>	<input checked="" type="checkbox"/>	*****
<b> ApplicationUser</b>	<i>User</i>	<input checked="" type="checkbox"/>	<b>1</b>

Profile - ApplicationUser Accosiation

	Member	Navigable	Multiplicity
<b> Profile</b>	<i>Profile</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b> ApplicationUser</b>	<i>ApplicationUser</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

ProductType - ProductType Accosiation

	Member	Navigable	Multiplicity
<b> ProductType</b>	<i>ParentType</i>	<input checked="" type="checkbox"/>	*****
<b> ProductType</b>	<i>SybTypes</i>	<input checked="" type="checkbox"/>	*****

Product - ProductType Accosiation

	Member	Navigable	Multiplicity
<b> Product</b>	<i>Product</i>	<input checked="" type="checkbox"/>	*****
<b> ProductType</b>	<i>Type</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

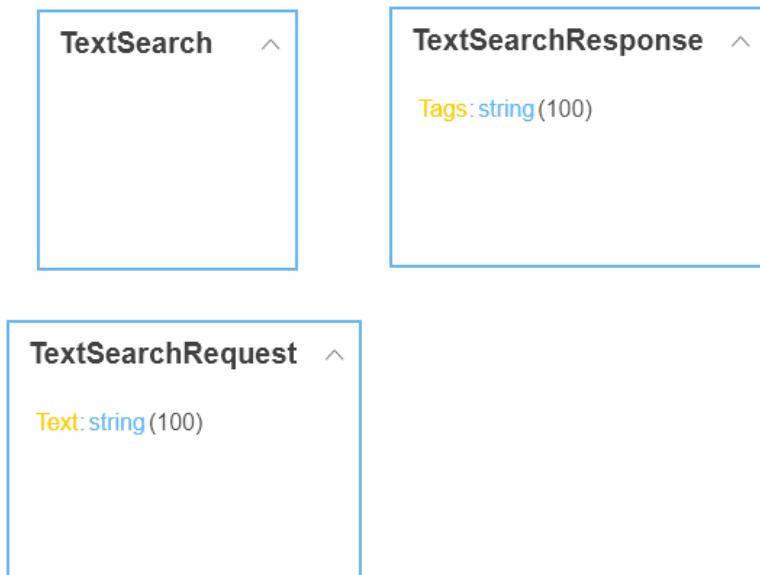
Product - PhysicalForm Accosiation

	Member	Navigable	Multiplicity
<b> Product</b>	<i>Product</i>	<input checked="" type="checkbox"/>	*****
<b> PhysicalForm</b>	<i>PhysicalForm</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Product - UnitOfMeasurement Accosiation

	Member	Navigable	Multiplicity
<b>Product</b>	<i>Product</i>	☒	*****
<b>UnitOfMeasurement</b>	<i>UnitOfMeasurement</i>	☑	<b>0..1</b>

**TextSearch Class Diagram**



## **TextSearch Classes**

### **TextSearch Class**

Persisted:  Identity:   

#### **Attributes**

Name	Datatype	Persisted	Required	Encrypted

#### **Operations**

Name: GetTags

```
static function string GetTags(string description)
{
    Domain.TextSearchRequest req;
    req.Text = description;

    var result = Interfaces.TextSearch.API
        .GetKeywords(DataTransformations.TextSearch.TextSearchRequest_To_TextSearchRequest(req))
```

```

        return result.Tags;
    }
}

```

### TextSearchRequest Class

Persisted:  Identity: *Text*

#### Attributes

Name	Datatype	Persisted	Required	Encrypted
Text	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

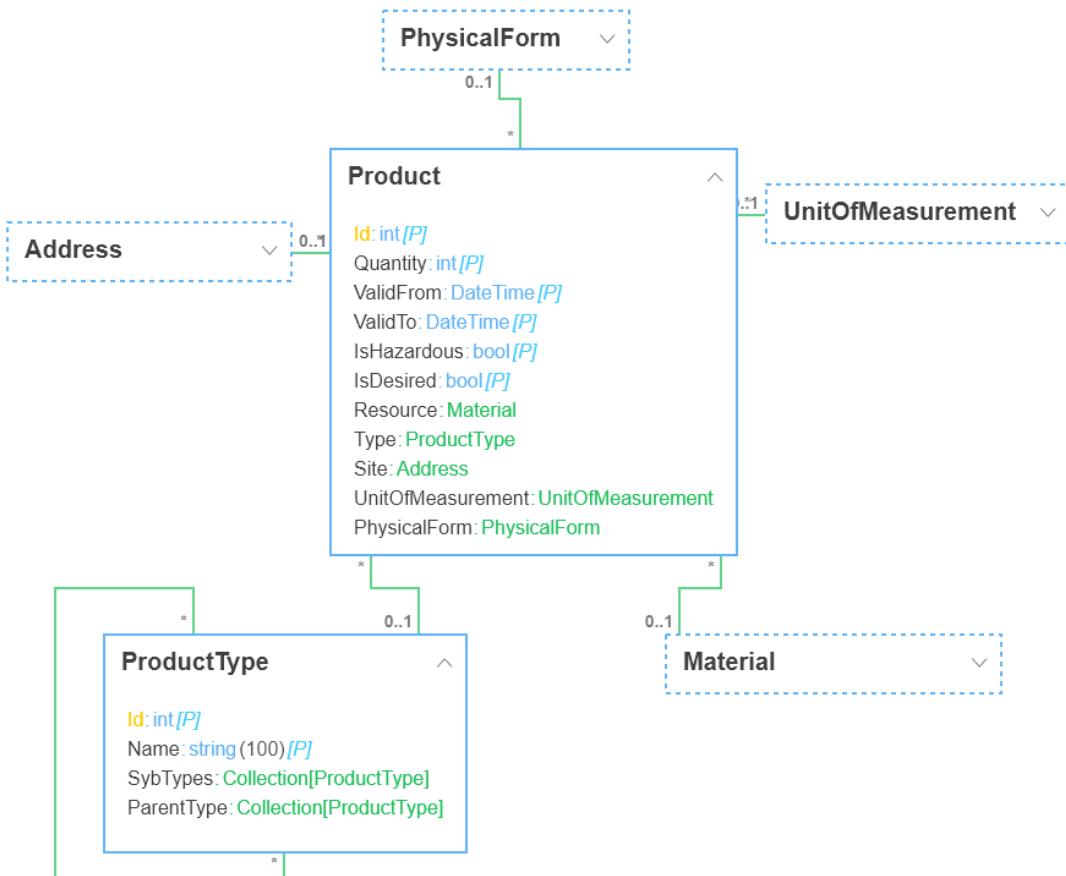
### TextSearchResponse Class

Persisted:  Identity: *Tags*

#### Attributes

Name	Datatype	Persisted	Required	Encrypted
Tags	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

### Product Class Diagram



## Product Classes

### Product Class

Persisted:  Identity: *Id*

#### Attributes

Name	Datatype	Persisted	Required	Encrypted
Id	int	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Quantity	int	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
ValidFrom	DateTime	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
ValidTo	DateTime	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
IsHazardous	bool	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
IsDesired	bool	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

## ProductType Class

Persisted:  Identity: *Id*

### Attributes

Name	Datatype	Persisted	Required	Encrypted
Id	int	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Name	string	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

## Product Associations

Product - Material Accosiation

	Member	Navigable	Multiplicity
<b>Product</b>	<i>Product</i>	<input type="checkbox"/>	*****
<b>Material</b>	<i>Resource</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

ProductType - ProductType Accosiation

	Member	Navigable	Multiplicity
<b>ProductType</b>	<i>ParentType</i>	<input checked="" type="checkbox"/>	*****
<b>ProductType</b>	<i>SybTypes</i>	<input checked="" type="checkbox"/>	*****

Product - ProductType Accosiation

	Member	Navigable	Multiplicity
<b>Product</b>	<i>Product</i>	<input type="checkbox"/>	*****
<b>ProductType</b>	<i>Type</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Product - Address Accosiation

	Member	Navigable	Multiplicity
<b>Product</b>	<i>Product</i>	<input type="checkbox"/>	*****
<b>Address</b>	<i>Site</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Product - UnitOfMeasurement Accosiation

	Member	Navigable	Multiplicity
<b>Product</b>	<i>Product</i>	<input type="checkbox"/>	*****
<b>UnitOfMeasurement</b>	<i>UnitOfMeasurement</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Product - PhysicalForm Accosiation

	Member	Navigable	Multiplicity

<b>Product</b>	<i>Product</i>	<input checked="" type="checkbox"/>	*****
<b>PhysicalForm</b>	<i>PhysicalForm</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

CircularEconomyReport - Product Accosiation

	Member	Navigable	Multiplicity
<b>CircularEconomyReport</b>	<i>CircularEconomyReport</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Product</b>	<i>Resources</i>	<input checked="" type="checkbox"/>	*****

CircularEconomyReport - Product Accosiation

	Member	Navigable	Multiplicity
<b>CircularEconomyReport</b>	<i>_CircularEconomyReport_1_</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Product</b>	<i>DesiredResources</i>	<input checked="" type="checkbox"/>	*****

GraphQuery - Product Accosiation

	Member	Navigable	Multiplicity
<b>GraphQuery</b>	<i>GraphQuery</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Product</b>	<i>DesiredProduct</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

GraphQuery - Product Accosiation

	Member	Navigable	Multiplicity
<b>GraphQuery</b>	<i>_GraphQuery_1_</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Product</b>	<i>ResourceProduct</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Material - Process Accosiation

	Member	Navigable	Multiplicity
<b>Material</b>	<i>Product</i>	<input checked="" type="checkbox"/>	*****
<b>Process</b>	<i>ConvertedBy</i>	<input checked="" type="checkbox"/>	*****

Material - Process Accosiation

	Member	Navigable	Multiplicity
<b>Material</b>	<i>Source</i>	<input checked="" type="checkbox"/>	*****
<b>Process</b>	<i>ConvertBy</i>	<input checked="" type="checkbox"/>	*****

Material - DigicircUser Accosiation

	Member	Navigable	Multiplicity
<b>Material</b>	<i>Material</i>	<input checked="" type="checkbox"/>	*****

DigicircUser	RequestedBy	<input checked="" type="checkbox"/>	0..1
--------------	-------------	-------------------------------------	------

Match - Material Accosiation

	Member	Navigable	Multiplicity
<b>Match</b>	<i>Match</i>	☒	<b>0..1</b>
<b>Material</b>	<i>Resource</i>	☑	<b>0..1</b>

Actor - Address Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>Company</i>	☒	<b>0..1</b>
<b>Address</b>	<i>Address</i>	☑	<b>0..1</b>

Actor - Address Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>Actor</i>	☒	<b>0..1</b>
<b>Address</b>	<i>Sites</i>	☑	*****

Address - Country Accosiation

	Member	Navigable	Multiplicity
<b>Address</b>	<i>Address</i>	☒	*****
<b>Country</b>	<i>Country</i>	☑	<b>0..1</b>

### ActorAPIHelper Class Diagram





## ActorAPIHelper Classes

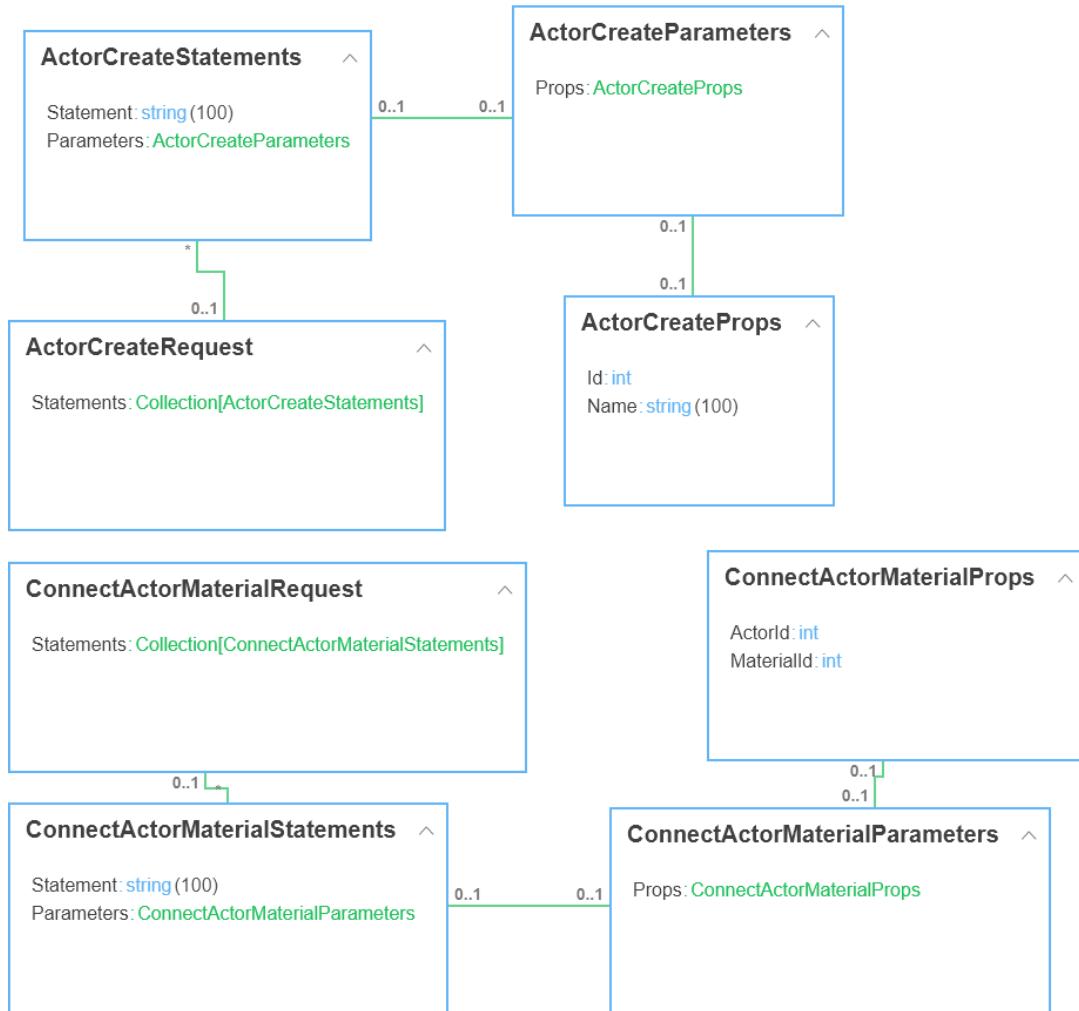
### Response Class

Persisted:  Identity:

#### Attributes

Name	Datatype	Persisted	Required	Encrypted
Code	int	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Status	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

## KnowledgeActorBase Class Diagram



## **KnowledgeActorBase Classes**

### **ActorCreateProps Class**

Persisted:  Identity: \_\_

#### **Attributes**

Name	Datatype	Persisted	Required	Encrypted
Id	int	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Name	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

### **ActorCreateParameters Class**

Persisted:  Identity: \_\_

#### **Attributes**

Name	Datatype	Persisted	Required	Encrypted

### **ActorCreateStatements Class**

Persisted:  Identity: \_\_

#### **Attributes**

Name	Datatype	Persisted	Required	Encrypted
Statement	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

### **ActorCreateRequest Class**

Persisted:  Identity: \_\_

**Attributes**

Name	Datatype	Persisted	Required	Encrypted
------	----------	-----------	----------	-----------

**ConnectActorMaterialRequest Class**Persisted:  Identity:   **Attributes**

Name	Datatype	Persisted	Required	Encrypted
------	----------	-----------	----------	-----------

**ConnectActorMaterialStatements Class**Persisted:  Identity:   **Attributes**

Name	Datatype	Persisted	Required	Encrypted
Statement	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

**ConnectActorMaterialParameters Class**Persisted:  Identity:   **Attributes**

Name	Datatype	Persisted	Required	Encrypted
------	----------	-----------	----------	-----------

**ConnectActorMaterialProps Class**Persisted:  Identity:   **Attributes**

Name	Datatype	Persisted	Required	Encrypted
ActorId	int	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
MaterialId	int	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

**KnowledgeActorBase Associations**

ActorCreateRequest - ActorCreateStatements Accosiation

	Member	Navigable	Multiplicity
<b>ActorCreateRequest</b>	<i>ActorCreateRequest</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>ActorCreateStatements</b>	<i>Statements</i>	<input checked="" type="checkbox"/>	*****

ActorCreateStatements - ActorCreateParameters Accosiation

	Member	Navigable	Multiplicity
<b>ActorCreateStatements</b>	<i>ActorCreateStatements</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>ActorCreateParameters</b>	<i>Parameters</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

ActorCreateParameters - ActorCreateProps Accosiation

	Member	Navigable	Multiplicity
<b>ActorCreateParameters</b>	<i>ActorCreateParameters</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>ActorCreateProps</b>	<i>Props</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

ConnectActorMaterialRequest - ConnectActorMaterialStatements Accosiation

	Member	Navigable	Multiplicity
<b>ConnectActorMaterialRequest</b>	<i>ConnectActorMaterialRequest</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>ConnectActorMaterialStatements</b>	<i>Statements</i>	<input checked="" type="checkbox"/>	*****

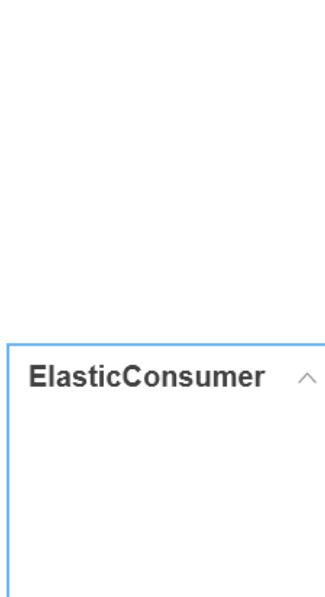
ConnectActorMaterialStatements - ConnectActorMaterialParameters Accosiation

	Member	Navigable	Multiplicity
<b>ConnectActorMaterialStatements</b>	<i>ConnectActorMaterialStatements</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>ConnectActorMaterialParameters</b>	<i>Parameters</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

ConnectActorMaterialParameters - ConnectActorMaterialProps Accosiation

	Member	Navigable	Multiplicity
<b>ConnectActorMaterialParameters</b>	<i>ConnectActorMaterialParameters</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>ConnectActorMaterialProps</b>	<i>Props</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

## ElasticHelpers Class Diagram



## **ElasticHelpers Classes**

## ElasticDoc Class

Persisted:  Identity:

### Attributes

Name	Datatype	Persisted	Required	Encrypted
------	----------	-----------	----------	-----------

### Operations

Name: CreateActorDoc

```
static function Interfaces.ElasticSearch.ActorDoc CreateActorDoc(Domain.Actor actor)
{
    Interfaces.ElasticSearch.ActorDoc actorDoc;

    actorDoc.ID = actor.Id;
    actorDoc.Name = actor.Name;
    actorDoc.Description = actor.Description;
    actorDoc.Tags = actor.Keywords;
    if(actor.Address != null && actor.Address.Country != null)
    {
        actorDoc.Country = actor.Address.Country.Name;
    }
    if(actor.SectorTypes.Length != 0)
    {
        actorDoc.Sector = actor.SectorTypes.Get(0).Value;
    }

    Collection[string] material;
//    if (actor.CircularEconomyRequirements.DesiredResources.Length > 0)
//    {
//    //
material.AddRange(actor.CircularEconomyRequirements.DesiredResources.Select(m =>
m.Resource.Name));
//    }

    if (actor.CircularEconomyRequirements.Resources.Length > 0)
    {
        material.AddRange(actor.CircularEconomyRequirements.Resources.Select(m =>
m.Resource.Name));
    }

    actorDoc.Resources = material;

    return actorDoc;
}
```

Name: SendActorDoc

```
static function string SendActorDoc(Domain.Actor actor)
{
    var doc = Domain.ElasticDoc.CreateActorDoc(actor);

    CommonLib.Serializer[Interfaces.ElasticSearch.ActorDoc] ser;
```

```

        DebugLib.Logger.WriteLine("Elastic request " + ser.ToString(doc));

        return Interfaces.ElasticSearch.API.CreateDoc(actor.Id, doc);
    }
}

```

### ElasticConsumer Class

Persisted:  Identity:  

#### Attributes

Name	Datatype	Persisted	Required	Encrypted
------	----------	-----------	----------	-----------

#### Operations

Name: InitElasticFromDb

```

static function void InitElasticFromDb()
{
    foreach Domain.Actor actor in Domain.Actor.GetAll()
    {
        Domain.ElasticDoc.SendActorDoc(actor);
    }
}

```

Name: Search

```

static function Interfaces.ElasticSearch.SearchResponse Search(Domain.SearchQuery
modelQuery)
{
    Interfaces.ElasticSearch.SearchRequest request;
    Interfaces.ElasticSearch.Query query;
    if(!modelQuery.AdvanceSearch)
    {
        Collection[Interfaces.ElasticSearch.Must] mustCollection =
            Domain.ElasticConsumer.Filters(modelQuery, true);

        Interfaces.ElasticSearch.BoolStatement boolElastic;
        query.Bool = boolElastic;
        query.Bool.Must = mustCollection.ToArray();
    }
    else
    {
        Collection[Interfaces.ElasticSearch.Must] mustCollection =
            Domain.ElasticConsumer.Filters(modelQuery, false);

        if(!string.IsNullOrEmpty(modelQuery.SearchTerm))
        {
            Interfaces.ElasticSearch.Must mustSearchTerm;
            Interfaces.ElasticSearch.MultiMatch matchSearchTerm;
            matchSearchTerm.Query = modelQuery.SearchTerm;
            matchSearchTerm.Fields = {"Name", "Description", "Resources"};
            mustSearchTerm.MultiMatch = matchSearchTerm;
            mustCollection.Add(mustSearchTerm);
        }
    }
}

```

```

        Interfaces.ElasticSearch.BoolStatement boolElastic;
        query.Bool = boolElastic;
        query.Bool.Must = mustCollection.ToArray();
    }

    request.Query = query;

    CommonLib.Serializer[Interfaces.ElasticSearch.SearchRequest] serQ;
    DebugLib.Logger.WriteLine("search query " + serQ.ToString());
}

var response = Interfaces.ElasticSearch.API.Search(request);

CommonLib.Serializer[Interfaces.ElasticSearch.SearchResponse] serR;
DebugLib.Logger.WriteLine("response elastic " + serR.ToString());
}

return response;
}

```

#### Name: Filters

```

static function Collection[Interfaces.ElasticSearch.Must] Filters(Domain.SearchQuery
modelQuery, bool filterResource)
{
    Collection[Interfaces.ElasticSearch.Must] mustCollection;

    if(modelQuery.SelectedCountry.Name != null)
    {
        Interfaces.ElasticSearch.Must must;
        Interfaces.ElasticSearch.MultiMatch match;
        match.Query = modelQuery.SelectedCountry.Name;
        match.Fields = {"Country"};
        must.MultiMatch = match;
        mustCollection.Add(must);
    }

    if(modelQuery.SelectedSector.Value != null)
    {
        Interfaces.ElasticSearch.Must must;
        Interfaces.ElasticSearch.MultiMatch match;
        match.Query = modelQuery.SelectedSector.Value;
        match.Fields = {"Sector"};
        must.MultiMatch = match;
        mustCollection.Add(must);
    }

    if(filterResource && modelQuery.SelectedMaterial.Name != null)
    {
        Interfaces.ElasticSearch.Must mustSearchTerm;
        Interfaces.ElasticSearch.MultiMatch matchSearchTerm;
        matchSearchTerm.Query = modelQuery.SelectedMaterial.Name;
    }
}

```

```
        matchSearchTerm.Fields = {"Resources"};
        mustSearchTerm.MultiMatch = matchSearchTerm;
        mustCollection.Add(mustSearchTerm);
    }

    return mustCollection;
}
```

## ElasticModel Class Diagram



## ElasticModel Classes

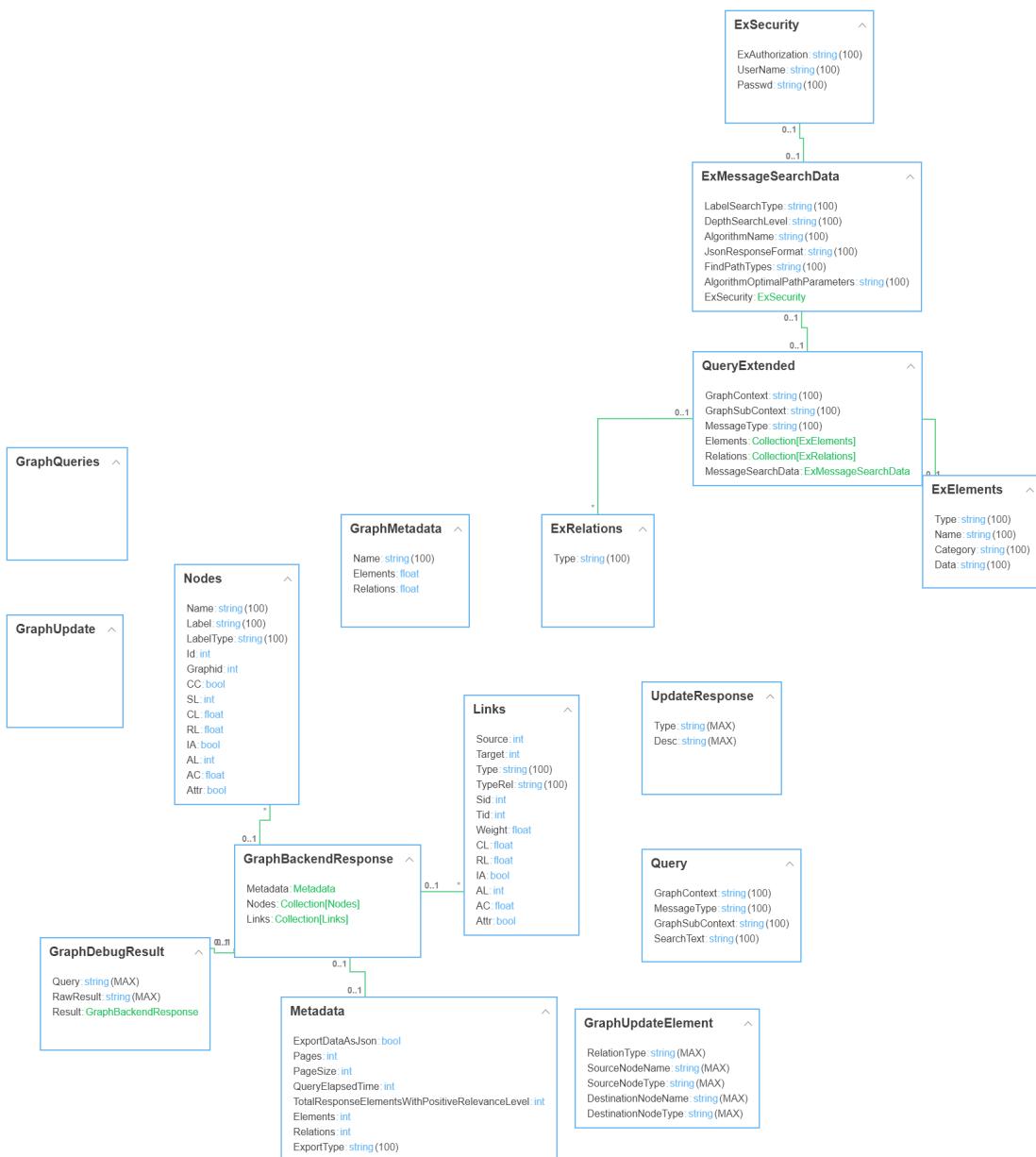
### SearchResponse Class

Persisted:  Identity: *Id*

#### Attributes

Name	Datatype	Persisted	Required	Encrypted
Id	int	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>

## ClmsGraphBackend Class Diagram



## ClmsGraphBackend Classes

### GraphDebugResult Class

Persisted:  Identity: \_\_\_\_\_

#### Attributes

Name	Datatype	Persisted	Required	Encrypted
Query	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
RawResult	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

### GraphBackendResponse Class

Persisted:  Identity: \_\_

#### Attributes

Name	Datatype	Persisted	Required	Encrypted

#### Metadata Class

Persisted:  Identity: \_\_

#### Attributes

Name	Datatype	Persisted	Required	Encrypted
ExportDataAsJson	bool	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Pages	int	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
PageSize	int	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
QueryElapsed Time	int	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
TotalResponseElementsWithPositiveRelevanceLevel	int	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Elements	int	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Relations	int	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
ExportType	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

#### Nodes Class

Persisted:  Identity: \_\_

#### Attributes

Name	Datatype	Persisted	Required	Encrypted
Name	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Label	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
LabelText	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Id	int	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Graphid	int	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
CC	bool	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
SL	int	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
CL	float	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
RL	float	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
IA	bool	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
AL	int	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
AC	float	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Attr	bool	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

### **Links Class**

Persisted:  Identity:   

#### **Attributes**

Name	Datatype	Persisted	Required	Encrypted
Source	int	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Target	int	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Type	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
TypeRel	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Sid	int	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Tid	int	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Weight	float	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
CL	float	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
RL	float	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
IA	bool	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
AL	int	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
AC	float	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Attr	bool	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

### **GraphMetadata Class**

Persisted:  Identity:   

#### **Attributes**

Name	Datatype	Persisted	Required	Encrypted
Name	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Elements	float	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Relations	float	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

### **Query Class**

Persisted:  Identity:   

#### **Attributes**

Name	Datatype	Persisted	Required	Encrypted
GraphContext	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
MessageType	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
GraphSubContext	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
SearchText	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

## Operations

Name: GetDefault

```
static function Interfaces.GraphBackend.Query GetDefault()
{
    Interfaces.GraphBackend.Query q;

    q.GraphContext = Application.Settings.GraphBackendGraphContext;
    q.GraphSubContext = Application.Settings.GraphBackendGraphSubContext;
    q.MessageType = "Find";

    return q;
}
```

## QueryExtended Class

Persisted:  Identity:  

### Attributes

Name	Datatype	Persisted	Required	Encrypted
GraphContext	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
GraphSubContext	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
MessageType	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

## Operations

Name: GetDefault

```
static function Interfaces.GraphBackend.QueryExtended GetDefault()
{
    Interfaces.GraphBackend.QueryExtended q;

    q.GraphContext = Application.Settings.GraphBackendGraphContext;
    q.GraphSubContext = Application.Settings.GraphBackendGraphSubContext;
    q.MessageType = "Find";

    return q;
}
```

Name: GetDeleteDefault

```
static function Interfaces.GraphBackend.QueryExtended GetDeleteDefault()
{
    Interfaces.GraphBackend.QueryExtended q;

    q.GraphContext = Application.Settings.GraphBackendGraphContext;
    q.GraphSubContext = Application.Settings.GraphBackendGraphSubContext;
    q.MessageType = "Delete";

    return q;
}
```

## ExElements Class

Persisted:  Identity:

### Attributes

Name	Datatype	Persisted	Required	Encrypted
Type	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Name	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Category	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Data	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

### Operations

Name: Transform

```
static function Collection[Interfaces.GraphBackend.ExElements]
Transform(Collection[Domain.ExElements] exElements)
{
    Collection[Interfaces.GraphBackend.ExElements] elements;
    foreach Domain.ExElements exElement in exElements
    {

elements.Add(DataTransformations.GraphBackend.ExElements_To_ExElements(exElement));
    }
    return elements;
}
```

Name: PrepareRelationNodes

```
static function Collection[Interfaces.GraphBackend.ExElements]
PrepareRelationNodes(Domain.GraphUpdateElement element)
{
    Interfaces.GraphBackend.ExElements sourceElement;

    sourceElement.Name = element.SourceNodeName;
    sourceElement.Type = element.SourceNodeType;

    Interfaces.GraphBackend.ExElements destinationElement;

    destinationElement.Name = element.DestinationNodeName;
    destinationElement.Type = element.DestinationNodeType;

    return {
        sourceElement,
        destinationElement
    };
}
```

Name: PrepareRalationNodesTypeText

```

static function Collection[Interfaces.GraphBackend.ExElements]
PrepareRalationNodesTypeText(Domain.GraphUpdateElement element, bool textIsInSource)
{
    Interfaces.GraphBackend.ExElements sourceElement;

    sourceElement.Name = element.SourceNodeName;
    sourceElement.Type = element.SourceNodeType;
    if(textIsInSource)
    {
        sourceElement.Category = "Text";
    }

    Interfaces.GraphBackend.ExElements destinationElement;

    destinationElement.Name = element.DestinationNodeName;
    destinationElement.Type = element.DestinationNodeType;
    if(!textIsInSource)
    {
        destinationElement.Category = "Text";
    }

    return {
        sourceElement,
        destinationElement
    };
}

```

## ExRelations Class

Persisted:  Identity:   

### Attributes

Name	Datatype	Persisted	Required	Encrypted
Type	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

### Operations

Name: GetDefaults

```

static function Collection[Interfaces.GraphBackend.ExRelations] GetDefaults()
{
    Collection[Interfaces.GraphBackend.ExRelations] relations;

    Interfaces.GraphBackend.ExRelations relation;
    relation.Type = "*";

    relations.Add(relation);

    return relations;
}

```

Name: GetRelationType

```

static function Collection[Interfaces.GraphBackend.ExRelations] GetRelationType(string
name)
{
    Collection[Interfaces.GraphBackend.ExRelations] relations;

    Interfaces.GraphBackend.ExRelations relation;
    relation.Type = name;

    relations.Add(relation);

    return relations;
}

```

### **ExMessageSearchData Class**

Persisted:  Identity: \_\_

#### **Attributes**

Name	Datatype	Persisted	Required	Encrypted
LabelSearchType	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
DepthSearchLevel	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
AlgorithmName	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
JsonResponseFormat	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
FindPathTypes	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
AlgorithmOptimalPathParameters	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

### **ExSecurity Class**

Persisted:  Identity: \_\_

#### **Attributes**

Name	Datatype	Persisted	Required	Encrypted
ExAuthorization	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
UserName	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Passwd	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

### **GraphQueries Class**

Persisted:  Identity: \_\_

#### **Attributes**

Name	Datatype	Persisted	Required	Encrypted

#### **Operations**

Name: Query

```

static function Domain.GraphBackendResponse Query(string queryText)
{
    var q = Domain.Query.GetDefault();
    q.SearchText = queryText;

    var response = Interfaces.GraphBackend.API.Query(q);

    return DataTransformations.GraphBackend
        .GraphBackendResponse_To_GraphBackendResponseReversed(response);
}

```

#### Name: RawQuery

```

static function string RawQuery(string queryText)
{
    var q = Domain.Query.GetDefault();
    q.SearchText = queryText;

    return Interfaces.GraphBackend.API.RawQuery(q);
}

```

#### Name: ExtentedQuery

```

static function Domain.GraphBackendResponse ExtentedQuery(Collection[Domain.ExElements] exElements)
{
    var q = Domain.QueryExtended.GetDefault();

    q.MessageSearchData = Domain.GraphQueries.GetExMessageSearchData();
    q.Elements = Domain.ExElements.Transform(exElements);
    q.Relations = Domain.ExRelations.GetDefaults();

    var response = Interfaces.GraphBackend.API.ExtendedQuery(q);
    return DataTransformations.GraphBackend
        .GraphBackendResponse_To_GraphBackendResponseReversed(response);
}

```

#### Name: RawExtentedQuery

```

static function string RawExtentedQuery(Collection[Domain.ExElements] exElements)
{
    var q = Domain.QueryExtended.GetDefault();

    q.MessageSearchData = Domain.GraphQueries.GetExMessageSearchData();
    q.Elements = Domain.ExElements.Transform(exElements);
    q.Relations = Domain.ExRelations.GetDefaults();

    return Interfaces.GraphBackend.API.RawExtentedQuery(q);
}

```

#### Name: GetExMessageSearchData

```

static function Interfaces.GraphBackend.ExMessageSearchData GetExMessageSearchData()
{
    Interfaces.GraphBackend.ExMessageSearchData exMessage;

    exMessage.LabelSearchType = "Whole";
    exMessage.DepthSearchLevel = "1";
    exMessage.JsonResponseFormat = "NotIndented";
    exMessage.AlgorithmName = "";

    Interfaces.GraphBackend.ExSecurity exSecurity;
    exSecurity.ExAuthorization = "NoAuth";
    exSecurity.UserName = Application.Settings.GraphBackendUserName;
    exSecurity.Passwd = Application.Settings.GraphBackendPasswd;

    exMessage.ExSecurity = exSecurity;

    return exMessage;
}

```

## GraphUpdate Class

Persisted:  Identity:   

### Attributes

Name	Datatype	Persisted	Required	Encrypted
------	----------	-----------	----------	-----------

### Operations

Name: AddNewRelation

```

static function Domain.UpdateResponse AddNewRelation(Domain.GraphUpdateElement
element)
{
    var q = Domain.QueryExtended.GetDefault();
    q.MessageType = "Update";
    q.MessageSearchData = Domain.GraphQueries.GetExMessageSearchData();

    q.Elements = Domain.ExElements.PrepareRelationNodes(element);

    q.Relations = Domain.ExRelations.GetRelationType(element.RelationType);

    var response = Interfaces.GraphBackend.API.Update(q);

    DebugLib.Logger.WriteLine(response.Type);

    return
DataTransformations.GraphBackend.UpdateResponse_To_UpdateResponseReversed(response);
}

```

Name: SendActorToGraph

```

static function void SendActorToGraph(Domain.Actor actor)
{
    //name to country
}

```

```

if(actor.Address != null && actor.Address.Country != null)
{
    Domain.GraphUpdateElement nameCountryElement =
Domain.GraphUpdateElement.Create();
    nameCountryElement.RelationType = "hasCompany";
    nameCountryElement.SourceNodeName = actor.Address.Country.Name;
    nameCountryElement.SourceNodeType = "Country";
    nameCountryElement.DestinationNodeName = actor.Name;
    nameCountryElement.DestinationNodeType = actor.EntityType.Code;
    var resultNameCountry = Domain.GraphUpdate.AddNewRelation(nameCountryElement);

    //reverse
    Domain.GraphUpdateElement nameCountryElementReverse =
Domain.GraphUpdateElement.Create();
    nameCountryElementReverse.RelationType = "isCompanyOf";
    nameCountryElementReverse.SourceNodeName = actor.Name;
    nameCountryElementReverse.SourceNodeType = actor.EntityType.Code;
    nameCountryElementReverse.DestinationNodeName = actor.Address.Country.Name;
    nameCountryElementReverse.DestinationNodeType = "Country";
    var resultNameCountryReverse =
Domain.GraphUpdate.AddNewRelation(nameCountryElementReverse);
}

//name to city
if(actor.Address != null && !string.IsNullOrEmpty(actor.Address.Town))
{
    Domain.GraphUpdateElement nameTownElement =
Domain.GraphUpdateElement.Create();
    nameTownElement.RelationType = "hasCompany";
    nameTownElement.SourceNodeName = actor.Address.Town;
    nameTownElement.SourceNodeType = "Town";
    nameTownElement.DestinationNodeName = actor.Name;
    nameTownElement.DestinationNodeType = actor.EntityType.Code;
    var resultNameTown = Domain.GraphUpdate.AddNewRelation(nameTownElement);
    //reverse
    Domain.GraphUpdateElement nameTownElementReverse =
Domain.GraphUpdateElement.Create();
    nameTownElementReverse.RelationType = "isCompanyOf";
    nameTownElementReverse.SourceNodeName = actor.Name;
    nameTownElementReverse.SourceNodeType = actor.EntityType.Code;
    nameTownElementReverse.DestinationNodeName = actor.Address.Town;
    nameTownElementReverse.DestinationNodeType = "Town";
    var resultNameTownReverse =
Domain.GraphUpdate.AddNewRelation(nameTownElementReverse);
}

//cityToCountry
if(actor.Address != null && !string.IsNullOrEmpty(actor.Address.Town) &&
actor.Address.Country != null)
{
    Domain.GraphUpdateElement nameTownElement =
Domain.GraphUpdateElement.Create();

```

```

nameTownElement.RelationType = "isLocationOf";
nameTownElement.SourceNodeName = actor.Address.Town;
nameTownElement.SourceNodeType = "Town";
nameTownElement.DestinationNodeName = actor.Address.Country.Name;
nameTownElement.DestinationNodeType = "Country";
var resultNameTown = Domain.GraphUpdate.AddNewRelation(nameTownElement);

//reverse
Domain.GraphUpdateElement nameTownElementReverse =
Domain.GraphUpdateElement.Create();
nameTownElementReverse.RelationType = "hasLocation";
nameTownElementReverse.SourceNodeName = actor.Address.Country.Name;
nameTownElementReverse.SourceNodeType = "Country";
nameTownElementReverse.DestinationNodeName = actor.Address.Town;
nameTownElementReverse.DestinationNodeType = "Town";
var resultNameTownReverse =
Domain.GraphUpdate.AddNewRelation(nameTownElementReverse);
}

//description
if(!string.IsNullOrEmpty(actor.Description))
{
    Domain.GraphUpdateElement nameDescriptionElement =
Domain.GraphUpdateElement.Create();
    nameDescriptionElement.RelationType = "isDescriptionOf";
    nameDescriptionElement.SourceNodeName = actor.Description;
    nameDescriptionElement.SourceNodeType = "Description";
    nameDescriptionElement.DestinationNodeName = actor.Name;
    nameDescriptionElement.DestinationNodeType = actor.EntityType.Value;
    var resultNameDesc =
Domain.GraphUpdate.AddNewRelationTypeText(nameDescriptionElement, true);

    //reverse
    Domain.GraphUpdateElement nameDescriptionElementReverse =
Domain.GraphUpdateElement.Create();
    nameDescriptionElementReverse.RelationType = "hasDescription";
    nameDescriptionElementReverse.SourceNodeName = actor.Name;
    nameDescriptionElementReverse.SourceNodeType = actor.EntityType.Value;
    nameDescriptionElementReverse.DestinationNodeName = actor.Description;
    nameDescriptionElementReverse.DestinationNodeType = "Description";
    var resultNameDescReverse =
Domain.GraphUpdate.AddNewRelationTypeText(nameDescriptionElementReverse, false);
}

//sector
if(actor.SectorTypes.Length > 0)
{
    foreach(Domain.SectorType sector in actor.SectorTypes)
    {
        Domain.GraphUpdateElement sectorElement =
Domain.GraphUpdateElement.Create();
        sectorElement.RelationType = "hasSectorType";
}

```

```

sectorElement.SourceNodeName = actor.Name;
sectorElement.SourceNodeType = actor.EntityType.Code;
sectorElement.DestinationNodeName = sector.Value;
sectorElement.DestinationNodeType = "SectorType";
var resultSector = Domain.GraphUpdate.AddNewRelation(sectorElement);

//reverse
Domain.GraphUpdateElement sectorElementReverse =
Domain.GraphUpdateElement.Create();
sectorElementReverse.RelationType = "isSectorTypeOf";
sectorElementReverse.SourceNodeName = sector.Value;
sectorElementReverse.SourceNodeType = "SectorType";
sectorElementReverse.DestinationNodeName = actor.Name;
sectorElementReverse.DestinationNodeType = actor.EntityType.Code;
var resultSectorReverse =
Domain.GraphUpdate.AddNewRelation(sectorElementReverse);
}

}
}

```

#### Name: InitGraphFromDB

```

static function void InitGraphFromDB()
{
    foreach Domain.Actor actor in Domain.Actor.GetAll()
    {
        Domain.GraphUpdate.SendActorToGraph(actor);
    }
}

```

#### Name: DeleteRelation

```

static function Domain.UpdateResponse DeleteRelation(Domain.GraphUpdateElement
element)
{
    var q = Domain.QueryExtended.GetDeleteDefault();
    q.MessageSearchData = Domain.GraphQueries.GetExMessageSearchData();

    q.Elements = Domain.ExElements.PrepareRelationNodes(element);

    q.Relations = Domain.ExRelations.GetRelationType(element.RelationType);

    var response = Interfaces.GraphBackend.API.Update(q);

    DebugLib.Logger.WriteLine(response.Type);

    return
DataTransformations.GraphBackend.UpdateResponse_To_UpdateResponseReversed(response);
}

```

#### Name: DeleteOldRelations

```

static function void DeleteOldRelations(Domain.Actor oldInstance, Domain.Actor
newInstance)
{
    DebugLib.Logger.WriteLine("Inside delete");
    //check country
    if(oldInstance.Address.Country != newInstance.Address.Country)
    {
        DebugLib.Logger.WriteLine("Delete country");
        Domain.GraphUpdateElement nameCountryElement =
Domain.GraphUpdateElement.Create();
        nameCountryElement.RelationType = "hasCompany";
        nameCountryElement.SourceNodeName = oldInstance.Address.Country.Name;
        nameCountryElement.SourceNodeType = "Country";
        nameCountryElement.DestinationNodeName = oldInstance.Name;
        nameCountryElement.DestinationNodeType = oldInstance.EntityType.Code;
        var resultNameCountry = Domain.GraphUpdate.DeleteRelation(nameCountryElement);

        //reverse
        Domain.GraphUpdateElement nameCountryElementReversed =
Domain.GraphUpdateElement.Create();
        nameCountryElementReversed.RelationType = "isCompanyOf";
        nameCountryElementReversed.SourceNodeName = oldInstance.Name;
        nameCountryElementReversed.SourceNodeType = oldInstance.EntityType.Code;
        nameCountryElementReversed.DestinationNodeName =
oldInstance.Address.Country.Name;
        nameCountryElementReversed.DestinationNodeType = "Country";
        var resultNameCountryReversed =
Domain.GraphUpdate.DeleteRelation(nameCountryElementReversed);
    }

    //description
    if(oldInstance.Description != newInstance.Description )
    {
        Domain.GraphUpdateElement nameDescriptionElement =
Domain.GraphUpdateElement.Create();
        nameDescriptionElement.RelationType = "isDescriptionOf";
        nameDescriptionElement.SourceNodeName = oldInstance.Description;
        nameDescriptionElement.SourceNodeType = "Description";
        nameDescriptionElement.DestinationNodeName = oldInstance.Name;
        nameDescriptionElement.DestinationNodeType = oldInstance.EntityType.Value;
        var resultNameDesc =
Domain.GraphUpdate.DeleteRelation(nameDescriptionElement);

        //reverse
        Domain.GraphUpdateElement nameDescriptionElementReverse =
Domain.GraphUpdateElement.Create();
        nameDescriptionElementReverse.RelationType = "hasDescription";
        nameDescriptionElementReverse.SourceNodeName = oldInstance.Name;
        nameDescriptionElementReverse.SourceNodeType = oldInstance.EntityType.Value;
        nameDescriptionElementReverse.DestinationNodeName = oldInstance.Description;
        nameDescriptionElementReverse.DestinationNodeType = "Description";
        var resultNameDescReverse =

```

```

Domain.GraphUpdate.DeleteRelation(nameDescriptionElementReverse);
}

//check sector type
if(oldInstance.SectorTypes.Length > 0)
{
    foreach Domain.SectorType sector in oldInstance.SectorTypes
    {
        if(newInstance.SectorTypes.Any(a => a.Code == sector.Code))
        {
            Domain.GraphUpdateElement sectorElement =
Domain.GraphUpdateElement.Create();
            sectorElement.RelationType = "hasSectorType";
            sectorElement.SourceNodeName = oldInstance.Name;
            sectorElement.SourceNodeType = oldInstance.EntityType.Code;
            sectorElement.DestinationNodeName = sector.Value;
            sectorElement.DestinationNodeType = "SectorType";
            var resultSector = Domain.GraphUpdate.DeleteRelation(sectorElement);

            //reverse
            Domain.GraphUpdateElement sectorElementReverse =
Domain.GraphUpdateElement.Create();
            sectorElementReverse.RelationType = "isSectorTypeOf";
            sectorElementReverse.SourceNodeName = sector.Value;
            sectorElementReverse.SourceNodeType = "SectorType";
            sectorElementReverse.DestinationNodeName = oldInstance.Name;
            sectorElementReverse.DestinationNodeType =
oldInstance.EntityType.Code;
            var resultSectorReverse =
Domain.GraphUpdate.DeleteRelation(sectorElementReverse);
        }
    }
}
}

```

Name: AddNewRelationTypeText

```

static function Domain.UpdateResponse AddNewRelationTypeText(Domain.GraphUpdateElement
element, bool isTextInSource)
{
    var q = Domain.QueryExtended.GetDefault();
    q.MessageType = "Update";
    q.MessageSearchData = Domain.GraphQueries.GetExMessageSearchData();

    q.Elements =
Domain.ExElements.PrepareRalationNodesTypeText(element,isTextInSource);

    q.Relations = Domain.ExRelations.GetRelationType(element.RelationType);

    var response = Interfaces.GraphBackend.API.Update(q);
    DebugLib.Logger.WriteLine(response.Type);
}

```

```

    return
DataTransformations.GraphBackend.UpdateResponse_To_UpdateResponseReversed(response);
}

```

### **UpdateResponse Class**

Persisted:  Identity:   

#### **Attributes**

Name	Datatype	Persisted	Required	Encrypted
Type	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Desc	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

### **GraphUpdateElement Class**

Persisted:  Identity:   

#### **Attributes**

Name	Datatype	Persisted	Required	Encrypted
RelationType	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
SourceNodeName	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
SourceNodeType	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
DestinationNodeName	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
DestinationNodeType	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

### **ClmsGraphBackend Associations**

GraphBackendResponse - Metadata Accosiation

	Member	Navigable	Multiplicity
<b>GraphBackendResponse</b>	<i>GraphBackendResponce</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Metadata</b>	<i>Metadata</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

GraphBackendResponse - Nodes Accosiation

	Member	Navigable	Multiplicity
<b>GraphBackendResponse</b>	<i>GraphBackendResponce</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Nodes</b>	<i>Nodes</i>	<input checked="" type="checkbox"/>	*****

GraphBackendResponse - Links Accosiation

	Member	Navigable	Multiplicity
<b>GraphBackendResponse</b>	<i>GraphBackendResponce</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Links</b>	<i>Links</i>	<input checked="" type="checkbox"/>	*****

GraphDebugResult - GraphBackendResponse Accosiation

	Member	Navigable	Multiplicity
<b>GraphDebugResult</b>	<i>GraphDebugResult</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>GraphBackendResponse</b>	<i>Result</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

QueryExtended - ExElements Accosiation

	Member	Navigable	Multiplicity
<b>QueryExtended</b>	<i>QueryExtended</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>ExElements</b>	<i>Elements</i>	<input checked="" type="checkbox"/>	*****

QueryExtended - ExRelations Accosiation

	Member	Navigable	Multiplicity
<b>QueryExtended</b>	<i>QueryExtended</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>ExRelations</b>	<i>Relations</i>	<input checked="" type="checkbox"/>	*****

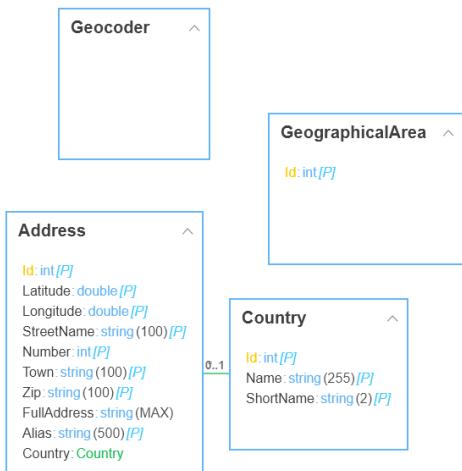
ExMessageSearchData - ExSecurity Accosiation

	Member	Navigable	Multiplicity
<b>ExMessageSearchData</b>	<i>ExMessageSearchData</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>ExSecurity</b>	<i>ExSecurity</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

QueryExtended - ExMessageSearchData Accosiation

	Member	Navigable	Multiplicity
<b>QueryExtended</b>	<i>QueryExtended</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>ExMessageSearchData</b>	<i>MessageSearchData</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

## Geolocation Class Diagram



## Geolocation Classes

### Address Class

Persisted:  Identity: *Id*

#### Attributes

Name	Datatype	Persisted	Required	Encrypted
Id	int	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Latitude	double	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Longitude	double	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
StreetName	string	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Number	int	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Town	string	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Zip	string	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
FullAddress	string	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Alias	string	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>

#### Operations

Name: GetFullAddress

```
function string GetFullAddress()
{
    return this.StreetName + " " + this.Number + ",<br />" + this.Town + " " +
this.Zip + "<br />" + this.Country.Name;
}
```

#### Country Class

Persisted:  Identity: *Id*

#### Attributes

Name	Datatype	Persisted	Required	Encrypted
Id	int	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Name	string	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
ShortName	string	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>

#### Geocoder Class

Persisted:  Identity: *\_*

#### Attributes

Name	Datatype	Persisted	Required	Encrypted

#### Operations

Name: Query

```
static function Domain.Address Query(Domain.Address address)
{
```

```

        DebugLib.Logger.WriteLine(address.FullAddress);

        var result = Interfaces.Geocoder.API
            .Query(address.FullAddress, Application.Settings.OpenCageApiKey)
            .Results
            .First();

        address.Latitude = result.Geometry.Latitude;
        address.Longitude = result.Geometry.Longitude;

        return address;
    }
}

```

### GeographicalArea Class

Persisted:  Identity: *Id*

#### Attributes

Name	Datatype	Persisted	Required	Encrypted
Id	int	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>

### Geolocation Associations

Address - Country Accosiation

	Member	Navigable	Multiplicity
<b>Address</b>	<i>Address</i>	<input checked="" type="checkbox"/>	*****
<b>Country</b>	<i>Country</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Actor - Address Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>Company</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Address</b>	<i>Address</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Actor - Address Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>Actor</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Address</b>	<i>Sites</i>	<input checked="" type="checkbox"/>	*****

Product - Address Accosiation

	Member	Navigable	Multiplicity
<b>Product</b>	<i>Product</i>	<input checked="" type="checkbox"/>	*****
<b>Address</b>	<i>Site</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

SearchQuery - Country Accosiation

	Member	Navigable	Multiplicity
<b>SearchQuery</b>	<i>SearchQuery</i>	☒	<b>0..1</b>
<b>Country</b>	<i>SelectedCountry</i>	☑	<b>0..1</b>

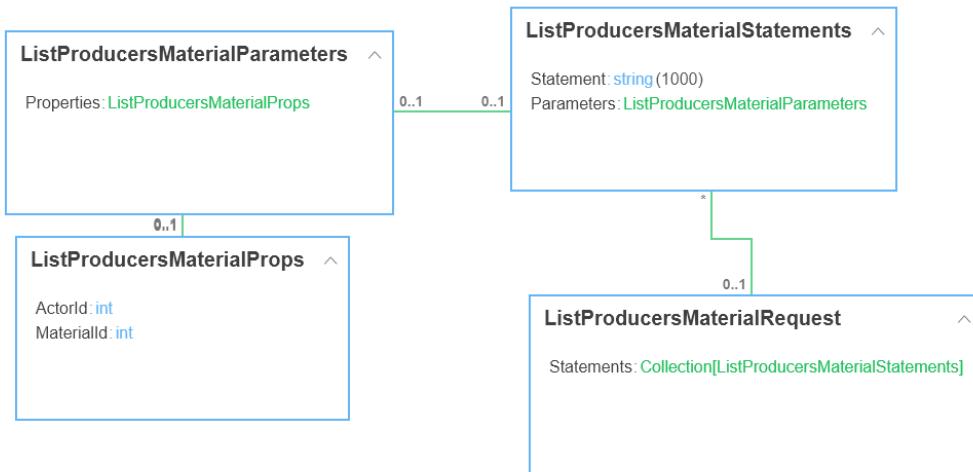
CircularEconomyReport - GeographicalArea Accosiation

	Member	Navigable	Multiplicity
<b>CircularEconomyReport</b>	<i>CircularEconomyInformation</i>	☒	<b>0..1</b>
<b>GeographicalArea</b>	<i>DesiredGeographicalArea</i>	☑	*****

GeographicalArea - CircularEconomyProviderReport Accosiation

	Member	Navigable	Multiplicity
<b>GeographicalArea</b>	<i>PlaceOperates</i>	☑	*****
<b>CircularEconomyProviderReport</b>	<i>CircularEconomyProviderReport</i>	☒	<b>0..1</b>

## KnowledgeProducersMaterialBase Class Diagram



## **KnowledgeProducersMaterialBase Classes**

### **ListProducersMaterialParameters Class**

Persisted:  Identity: \_\_

#### **Attributes**

Name	Datatype	Persisted	Required	Encrypted

### **ListProducersMaterialStatements Class**

Persisted:  Identity: \_\_

#### **Attributes**

Name	Datatype	Persisted	Required	Encrypted
Statement	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

### **ListProducersMaterialProps Class**

Persisted:  Identity: \_\_

#### **Attributes**

Name	Datatype	Persisted	Required	Encrypted

ActorId	int	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
MaterialId	int	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

### ListProducersMaterialRequest Class

Persisted:  Identity: \_\_\_\_\_

#### Attributes

Name	Datatype	Persisted	Required	Encrypted

### KnowledgeProducersMaterialBase Associations

ListProducersMaterialProps - ListProducersMaterialParameters Accosiation

	Member	Navigable	Multiplicity
ListProducersMaterialProps	Properties	<input checked="" type="checkbox"/>	<b>0..1</b>
ListProducersMaterialParameters	ListProducersMaterialParameters	<input checked="" type="checkbox"/>	<b>0..1</b>

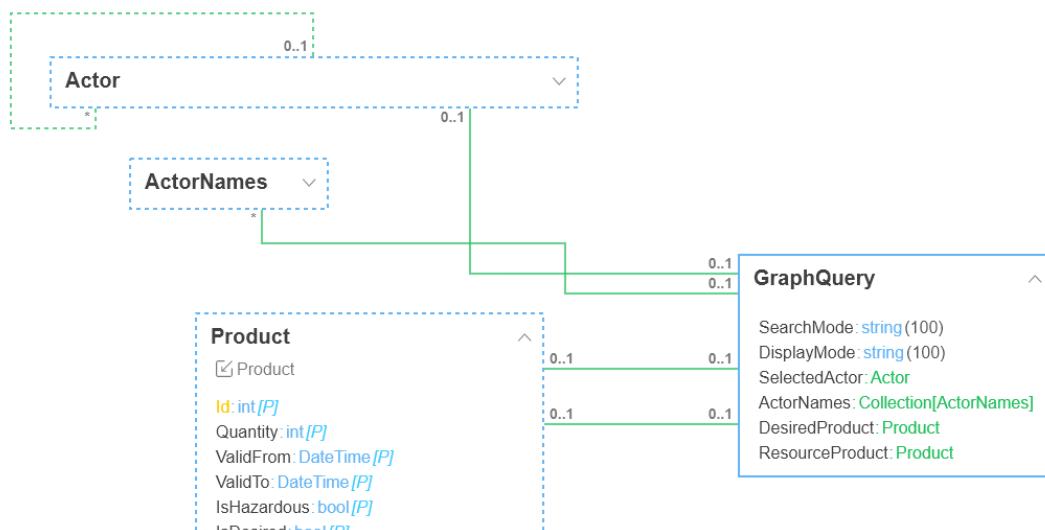
ListProducersMaterialStatements - ListProducersMaterialParameters Accosiation

	Member	Navigable	Multiplicity
ListProducersMaterialStatements	ListProducersMaterialStatements	<input checked="" type="checkbox"/>	<b>0..1</b>
ListProducersMaterialParameters	Parameters	<input checked="" type="checkbox"/>	<b>0..1</b>

ListProducersMaterialRequest - ListProducersMaterialStatements Accosiation

	Member	Navigable	Multiplicity
ListProducersMaterialRequest	ListProducersMaterialRequest	<input checked="" type="checkbox"/>	<b>0..1</b>
ListProducersMaterialStatements	Statements	<input checked="" type="checkbox"/>	*****

### GraphQuery Class Diagram



isDesired: [UoM/P](#)  
Resource: [Material](#)  
Type: [ProductType](#)  
Site: [Address](#)  
UnitOfMeasurement: [UnitOfMeasurement](#)  
PhysicalForm: [PhysicalForm](#)

## GraphQL Classes

### GraphQL Class

Persisted:  Identity: \_\_

#### Attributes

Name	Datatype	Persisted	Required	Encrypted
SearchMode	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
DisplayMode	string	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

## GraphQuery Associations

GraphQuery - Actor Accosiation

	Member	Navigable	Multiplicity
GraphQuery	<i>GraphQuery</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
Actor	<i>SelectedActor</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

GraphQuery - ActorNames Accosiation

	Member	Navigable	Multiplicity
GraphQuery	<i>GraphQuery</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
ActorNames	<i>ActorNames</i>	<input checked="" type="checkbox"/>	*****

Actor - Address Accosiation

	Member	Navigable	Multiplicity
Actor	<i>Company</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
Address	<i>Address</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Actor - DigicircUser Accosiation

	Member	Navigable	Multiplicity
Actor	<i>Company</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
DigicircUser	<i>AddedBy</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

CircularEconomyReport - Actor Accosiation

	Member	Navigable	Multiplicity
CircularEconomyReport	<i>CircularEconomyRequirements</i>	<input checked="" type="checkbox"/>	<b>1</b>
Actor	<i>Actor</i>	<input checked="" type="checkbox"/>	<b>1</b>

Actor - CircularEconomyProviderReport Accosiation

	Member	Navigable	Multiplicity
Actor	<i>Actor</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
CircularEconomyProviderReport	<i>CircularEconomyProviderReport</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Actor - FileData Accosiation

	Member	Navigable	Multiplicity
Actor	<i>Actor</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
FileData	<i>ActorLogo</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Actor - Actor Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>Cluster</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Actor</b>	<i>Actors</i>	<input checked="" type="checkbox"/>	*****

Actor - DigicircUser Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>ActorsCanManage</i>	<input checked="" type="checkbox"/>	*****
<b>DigicircUser</b>	<i>Administrators</i>	<input checked="" type="checkbox"/>	*****

Actor - Address Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>Actor</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Address</b>	<i>Sites</i>	<input checked="" type="checkbox"/>	*****

Actor - EntityType Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>Actor</i>	<input checked="" type="checkbox"/>	*****
<b>EntityType</b>	<i>EntityType</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

SectorType - Actor Accosiation

	Member	Navigable	Multiplicity
<b>SectorType</b>	<i>SectorTypes</i>	<input checked="" type="checkbox"/>	*****
<b>Actor</b>	<i>Actor</i>	<input checked="" type="checkbox"/>	*****

SearchQuery - ActorNames Accosiation

	Member	Navigable	Multiplicity
<b>SearchQuery</b>	<i>SearchQuery</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>ActorNames</b>	<i>ActorNames</i>	<input checked="" type="checkbox"/>	*****

CircularEconomyReport - Product Accosiation

	Member	Navigable	Multiplicity
<b>CircularEconomyReport</b>	<i>CircularEconomyReport</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Product</b>	<i>Resources</i>	<input checked="" type="checkbox"/>	*****

CircularEconomyReport - Product Accosiation

---

	Member	Navigable	Multiplicity
<b>CircularEconomyReport</b>	_CircularEconomyReport_1_	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Product</b>	<i>DesiredResources</i>	<input checked="" type="checkbox"/>	*****

Product - Material Accosiation

	Member	Navigable	Multiplicity
<b>Product</b>	<i>Product</i>	<input checked="" type="checkbox"/>	*****
<b>Material</b>	<i>Resource</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Product - ProductType Accosiation

	Member	Navigable	Multiplicity
<b>Product</b>	<i>Product</i>	<input checked="" type="checkbox"/>	*****
<b>ProductType</b>	<i>Type</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Product - Address Accosiation

	Member	Navigable	Multiplicity
<b>Product</b>	<i>Product</i>	<input checked="" type="checkbox"/>	*****
<b>Address</b>	<i>Site</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Product - UnitOfMeasurement Accosiation

	Member	Navigable	Multiplicity
<b>Product</b>	<i>Product</i>	<input checked="" type="checkbox"/>	*****
<b>UnitOfMeasurement</b>	<i>UnitOfMeasurement</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Product - PhysicalForm Accosiation

	Member	Navigable	Multiplicity
<b>Product</b>	<i>Product</i>	<input checked="" type="checkbox"/>	*****
<b>PhysicalForm</b>	<i>PhysicalForm</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

GraphQuery - Product Accosiation

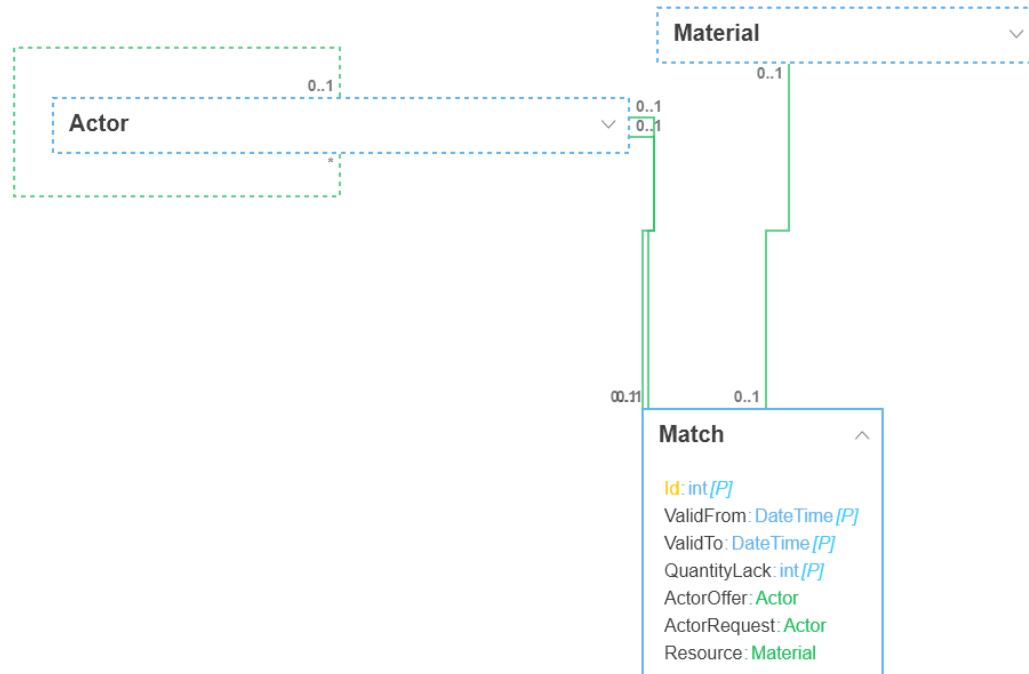
	Member	Navigable	Multiplicity
<b>GraphQuery</b>	<i>GraphQuery</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Product</b>	<i>DesiredProduct</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

GraphQuery - Product Accosiation

	Member	Navigable	Multiplicity

<b>GraphQuery</b>	_GraphQuery_1_	☒	<b>0..1</b>
<b>Product</b>	<i>ResourceProduct</i>	☒	<b>0..1</b>

### Suggestions Class Diagram



## Suggestions Classes

### Match Class

Persisted:  Identity: *Id*

#### Attributes

Name	Datatype	Persisted	Required	Encrypted
Id	int	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ValidFrom	DateTime	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ValidTo	DateTime	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
QuantityLack	int	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

#### Operations

Name: MatchProducts

```
static function void MatchProducts(Domain.Actor actor)
{
    foreach Domain.Product product in actor.CircularEconomyRequirements.Resources
    {
        var possibleMatches = Domain.Product.Find(p =>
            p.IsDesired
            && p.Resource.Id == product.Resource.Id
            && p.ValidTo >= product.ValidFrom
            && p.ValidFrom <= product.ValidTo);

        if(possibleMatches.Length == 0)
        {
            continue;
        }

        Domain.Match match = Domain.Match.Create();
        var matchedProduct = possibleMatches.First();
        var actorRequested = Domain.Actor.Find(a =>
            a.CircularEconomyRequirements.DesiredResources.Any(p => p.Id ==
            matchedProduct.Id)).First();
    }
}
```

```

        if(actorRequested == null)
        {
            continue;
        }
        if(Domain.Match.Find(x => x.ActorOffer.Id == actor.Id && x.ActorRequest.Id == actorRequested.Id && x.Resource == matchedProduct.Resource).Any())
        {
            continue;
        }
        match.Resource = matchedProduct.Resource;
        match.ValidFrom = DateTime.Compare(product.ValidFrom,
matchedProduct.ValidFrom) <= 0 ? matchedProduct.ValidFrom : product.ValidFrom;
        match.ValidTo = DateTime.Compare(product.ValidTo, matchedProduct.ValidTo) <= 0
? product.ValidTo : matchedProduct.ValidTo;
        match.ActorOffer = actor;
        match.ActorRequest = actorRequested;
        match.QuantityLack = product.Quantity - matchedProduct.Quantity;

        match.Save();
    }
}

```

## Suggestions Associations

Match - Actor Accosiation

	Member	Navigable	Multiplicity
<b>Match</b>	<i>Match</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Actor</b>	<i>ActorOffer</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Match - Actor Accosiation

	Member	Navigable	Multiplicity
<b>Match</b>	<i>_Match_1_</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Actor</b>	<i>ActorRequest</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Match - Material Accosiation

	Member	Navigable	Multiplicity
<b>Match</b>	<i>Match</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Material</b>	<i>Resource</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Actor - Address Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>Company</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Address</b>	<i>Address</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Actor - DigiCircUser Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>Company</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>DigiCircUser</b>	<i>AddedBy</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

CircularEconomyReport - Actor Accosiation

	Member	Navigable	Multiplicity
<b>CircularEconomyReport</b>	<i>CircularEconomyRequirements</i>	<input checked="" type="checkbox"/>	<b>1</b>
<b>Actor</b>	<i>Actor</i>	<input checked="" type="checkbox"/>	<b>1</b>

Actor - CircularEconomyProviderReport Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>Actor</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>CircularEconomyProviderReport</b>	<i>CircularEconomyProviderReport</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Actor - FileData Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>Actor</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>FileData</b>	<i>ActorLogo</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Actor - Actor Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>Cluster</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Actor</b>	<i>Actors</i>	<input checked="" type="checkbox"/>	*****

Actor - DigiCircUser Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>ActorsCanManage</i>	<input checked="" type="checkbox"/>	*****
<b>DigiCircUser</b>	<i>Administrators</i>	<input checked="" type="checkbox"/>	*****

Actor - Address Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>Actor</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Address</b>	<i>Sites</i>	<input checked="" type="checkbox"/>	*****

Actor - EntityType Accosiation

	Member	Navigable	Multiplicity
<b>Actor</b>	<i>Actor</i>	<input checked="" type="checkbox"/>	*****
<b>EntityType</b>	<i>EntityType</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

SectorType - Actor Accosiation

	Member	Navigable	Multiplicity
<b>SectorType</b>	<i>SectorTypes</i>	<input checked="" type="checkbox"/>	*****
<b>Actor</b>	<i>Actor</i>	<input checked="" type="checkbox"/>	*****

GraphQuery - Actor Accosiation

	Member	Navigable	Multiplicity
<b>GraphQuery</b>	<i>GraphQuery</i>	<input checked="" type="checkbox"/>	<b>0..1</b>
<b>Actor</b>	<i>SelectedActor</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Material - Process Accosiation

	Member	Navigable	Multiplicity
<b>Material</b>	<i>Product</i>	<input checked="" type="checkbox"/>	*****
<b>Process</b>	<i>ConvertedBy</i>	<input checked="" type="checkbox"/>	*****

Material - Process Accosiation

	Member	Navigable	Multiplicity
<b>Material</b>	<i>Source</i>	<input checked="" type="checkbox"/>	*****
<b>Process</b>	<i>ConvertBy</i>	<input checked="" type="checkbox"/>	*****

Material - DigicircUser Accosiation

	Member	Navigable	Multiplicity
<b>Material</b>	<i>Material</i>	<input checked="" type="checkbox"/>	*****
<b>DigicircUser</b>	<i>RequestedBy</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Material - ProductType Accosiation

	Member	Navigable	Multiplicity
<b>Material</b>	<i>Material</i>	<input checked="" type="checkbox"/>	*****
<b>ProductType</b>	<i>Type</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Material - PhysicalForm Accosiation

	Member	Navigable	Multiplicity

<b>Material</b>	<i>Material</i>	<input checked="" type="checkbox"/>	*****
<b>PhysicalForm</b>	<i>PhysicalForm</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Material - UnitOfMeasurement Accosiation

	Member	Navigable	Multiplicity
<b>Material</b>	<i>Material</i>	<input checked="" type="checkbox"/>	*****
<b>UnitOfMeasurement</b>	<i>UnitOfMeasurement</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

Product - Material Accosiation

	Member	Navigable	Multiplicity
<b>Product</b>	<i>Product</i>	<input checked="" type="checkbox"/>	*****
<b>Material</b>	<i>Resource</i>	<input checked="" type="checkbox"/>	<b>0..1</b>

## User Interface

### ErrorPage Form

#### Model

```
|-- ErrorMessage: string
|-- StackTrace: string
|-- FriendlyErrorMessage: string
|-- AdditionalErrorInformation: string
```

#### View

500 We are sorry, there was an error while loading the page.

Error Description

≡ FriendlyErrorMessage

Additional Information

≡ AdditionalErrorInformation

[🏠 Back to Home](#)
[↗ Send it to Administrator](#)

## Controller

### Render Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Render()
{
    Model.StackTrace = AppLib.Application.GetLastError().StackTrace;
    Model.ErrorMessage = AppLib.Application.GetLastError().Message;
    Model.FriendlyErrorMessage =
        AppLib.Application.GetLastError().GetFriendlyMessage();

}
```

### SendErrorToAdministrator Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```

function void SendErrorToAdministrator()
{
    string developerEmail =
AppLib.Application.GetConfigurationKey("AdministratorEmail");
    if(string.IsNullOrWhiteSpace(developerEmail))
    {
        ShowMessage(LocalResources.RES_CUSTOM_Invalid_AdminEmail_Message,
AppLib.MessageType.Error);
        return;
    }

    Model.StackTrace = AppLib.Application.GetLastError().StackTrace;
    Model.ErrorMessage = AppLib.Application.GetLastError().Message;
    Model.FriendlyErrorMessage =
AppLib.Application.GetLastError().GetFriendlyMessage();

    CommonLib.EmailMessage mail;

    Collection<string> recipients;
    recipients.Add(developerEmail);

    mail.To = recipients;
    mail.Subject = AppLib.Application.Name + " threw an Exception";
    mail.Body = "<b>Error Message:</b> " + Model.ErrorMessage + "<br /><br />";
    mail.Body = mail.Body + "<b>Friendly Error Message:</b> <span style='white-space: pre-line;'>" + Model.FriendlyErrorMessage + "</span><br /><br />";
    mail.Body = mail.Body + "<b>StackTrace:</b> " + Model.StackTrace + "<br />";
    if(!string.IsNullOrWhiteSpace(Model.AdditionalErrorInformation))
    {
        mail.Body = mail.Body + "<br /><b>Additional Information:</b> " +
Model.AdditionalErrorInformation;
    }

    CommonLib.Utilities.SendEmail(mail);
    ShowMessage(LocalResources.RES_CUSTOM_Successful_EMail_Message);
}

```

## FirstAdminSetup Form

### Model

```

|-- ApplicationUser: ApplicationUser
|-- AccessFailedCount: int
|-- Email: string
|-- EmailConfirmed: bool
|-- LockoutEnabled: bool
|-- LockoutEndDate: DateTime
|-- Name: string
|-- PasswordHash: string
|-- PhoneNumber: string
|-- PhoneNumberConfirmed: bool
|-- SecurityStamp: string

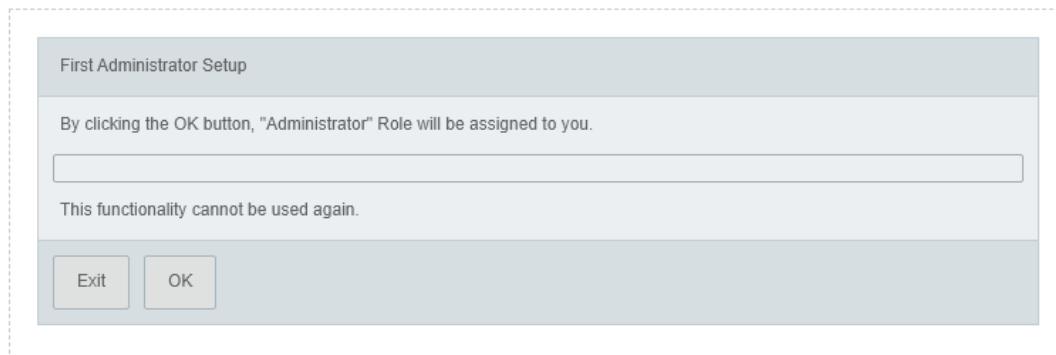
```

```

|--- TwoFactorEnabled: bool
|--- UserName: string
|--- Roles: ApplicationRole
|--- Description: string
|--- Id: int
|--- IsCustom: bool
|--- Name: string

```

## View



## Controller

### Render Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

### CODE

```

function void Execute()
{
    if Domain.ApplicationUser.GetAll().Any(x => x.Roles.Any(r => r.Name ==
"Administrator"))
    {
        ShowMessage(
            LocalResources.RES_CUSTOM_NoAccess,
            AppLib.MessageType.Error,
            FormModels.HomePage.Controller.Render.GetLink()
        );
        DebugLib.Logger.WriteLine("Admin user already exists");
        return;
    }

    Model ApplicationUser = AppLib.Session.GetCurrentUser();
}

```

```
    Model.Title = LocalResources.RES_PAGETITLE_Render;  
}
```

#### Save Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Execute() {  
    Domain.ApplicationRole role =  
        Domain.ApplicationRole.Find(r => r.Name == "Administrator").First();  
  
    if (role == null)  
    {  
        throw (LocalResources.RES_CUSTOM_NoAdminRoleFound);  
    }  
  
    if (Model ApplicationUser.IsInRole("Administrator"))  
    {  
        throw (LocalResources.RES_CUSTOM_AlreadyAdmin);  
    }  
  
    Model ApplicationUser.Roles.Add(role);  
    Model ApplicationUser.Save();  
    FormModels.HomePage.Controller.Render.Execute();  
}
```

### HomePage Form

#### Model

#### View

```
[REDACTED]
```

#### Controller

#### Render Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

**SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

**CODE**

```
function void Render()
{
    Model.Title = "";
}
```

**NotFoundPage Form****Model****View**

400

We are sorry, the page you requested cannot be found.

Possible Reasons:

1. An out-of-date bookmark/favourite

2. A search engine that has an out-of-date listing for us

3. A mis-typed address

4. A broken link

 Back to Home

**Controller****Render Controller Action**

Is Entrypoint:  Enabled Access Log:  Causes Validation:

**SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

**CODE**

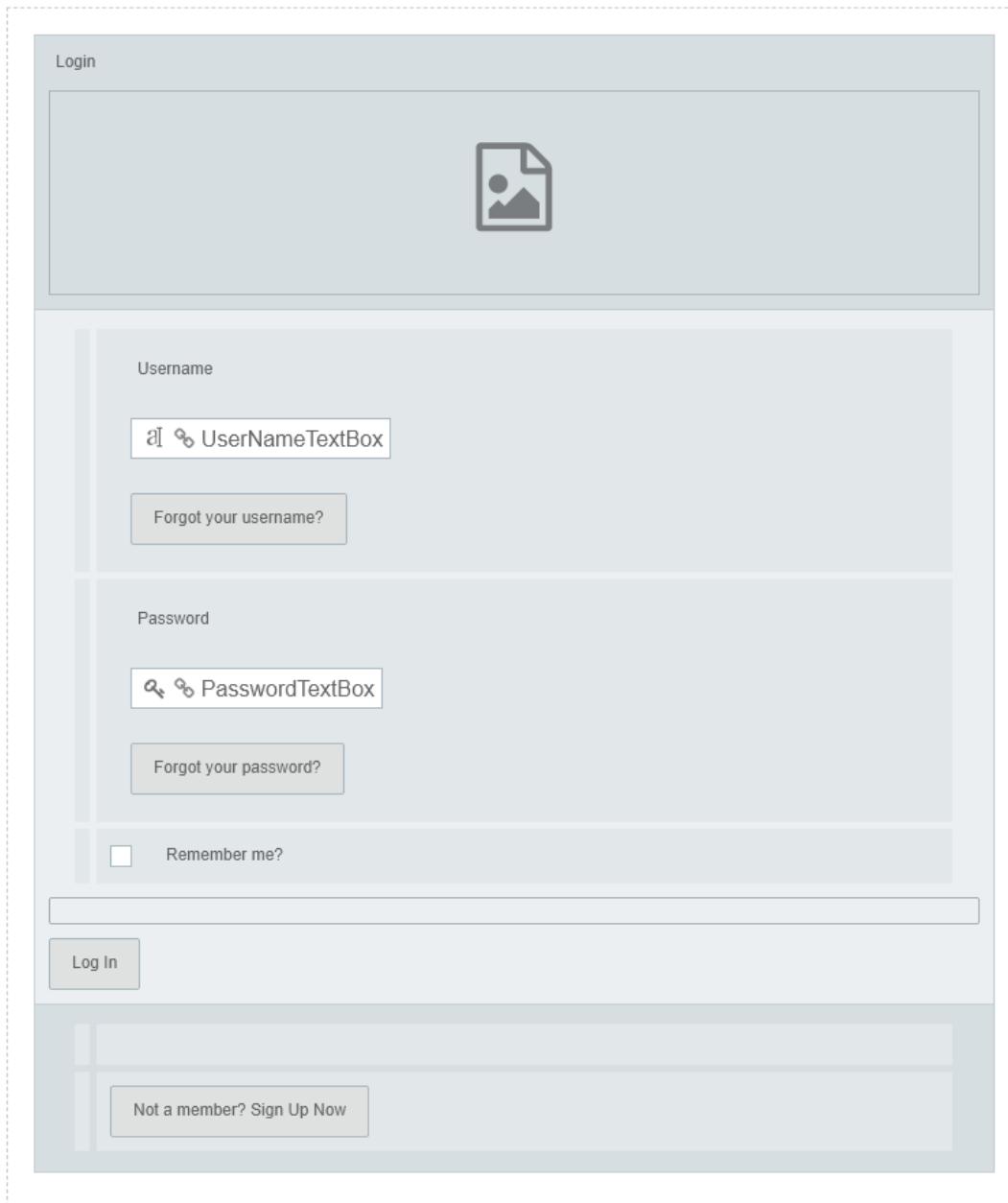
```
function void Execute() {  
}  
}
```

## SignInPage Form

### Model

```
|-- PasswordTextBox: string  
|-- UserNameTextBox: string  
|-- RememberMeCB: bool  
|-- FromMatching: bool
```

### View



## Controller

### Load Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		

### CODE

```

function void Load(bool fromMatching)
{
    Model.FromMatching = fromMatching;

    if (Domain ApplicationUser.Find(u => u.Roles.Any(r => r.Name ==
"Administrator")).Length == 0) {
        FormModels.CreateAdmin.Controller.Index.Execute();
    }
    elseif (AppLib.Session.GetCurrentUser() != null) {
        FormModels.HomePage.Controller.Render.Execute();
    }
}

```

### **LoadDefault Controller Action**

Is EntryPoint:  Enabled Access Log:  Causes Validation:

#### **SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		

#### **CODE**

```

function void LoadDefault()
{
    Model.FromMatching = true;
}

```

### **SignIn Controller Action**

Is EntryPoint:  Enabled Access Log:  Causes Validation:

#### **SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		

#### **CODE**

```

function void SignIn(bool fromMatching)
{
    bool success = AppLib.Security.SignInUser(Model.UserNameTextBox,
Model.PasswordTextBox, Model.RememberMeCB);

    if !success {
        ShowMessage(LocalResources.SignInFailed, AppLib.MessageType.Error);
        return;
    }

    string returnUrl = WebLib.Request.GetQueryStringParameter("returnUrl");

    if (!string.IsNullOrWhiteSpace(returnUrl)) {
        WebLib.Request.RedirectToUrl(CommonLib.Utilities.ResolveClientURL(returnUrl));
    }
}

```

```

        else {
            if (fromMatching) {
                FormModels.SearchForm.Controller.Index.Execute();
            } else {
                FormModels.KnowledgeHub.Controller.Index.Execute();
            }
        }
    }
}

```

## SignInPage Form

### Model

### View

Good Bye!

You have been signed out successfully.

### Controller

#### SignIn Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

### CODE

```

function void Execute(bool fromMatching) {
    AppLib.Security.SignInUser();
    if (fromMatching) {
        FormModels.SearchForm.Controller.Index.Execute();
    } else {
        FormModels.KnowledgeHub.Controller.Index.Execute();
    }
}

```

#### Render Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions

<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		
-------------------------------------	--------------------------	-------------------------------------	--	--

#### CODE

```
function void Execute() {
    Model.Title = LocalResources.RES_PAGETITLE_Render;
}
```

### Unauthorized Form

#### Model

#### View

403

You are not authorized to view this page.

Possible Reasons:

1. You don't have enough permissions

2. Permissions for this page have been changed

3. You are not signed in


[Back to Home](#)

#### Controller

##### Render Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Execute() {

}
```

### UserPreferences Form

## Model

```
|-- ApplicationUser: ApplicationUser
|-- AccessFailedCount: int
|-- Email: string
|-- EmailConfirmed: bool
|-- LockoutEnabled: bool
|-- LockoutEndDate: DateTime
|-- Name: string
|-- PasswordHash: string
|-- PhoneNumber: string
|-- PhoneNumberConfirmed: bool
|-- SecurityStamp: string
|-- TwoFactorEnabled: bool
|-- UserName: string
|-- Profile: Profile
  |-- Id: int
  |-- LanguageLCID: int
  |-- LocaleLCID: int
  |-- Theme: string
```

## View

The screenshot shows a user interface for 'User Preferences'. At the top, there is a header bar with the title 'User Preferences'. Below the header, there are three sections: 'Language', 'Locale', and 'Theme', each represented by a dropdown menu. The 'Language' dropdown is currently open, showing the path 'ApplicationUser.Profile.LanguageLCID'. The 'Locale' and 'Theme' dropdowns are also present but not currently open. At the bottom of the view, there are two buttons: 'Save' and 'Cancel'.

Section	Path
Language	ApplicationUser.Profile.LanguageLCID
Locale	ApplicationUser.Profile.LocaleLCID
Theme	ApplicationUser.Profile.Theme

## Controller

### Render Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

**SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

**CODE**

```
function void Render()
{
    Model.ApplicationUser = AppLib.Session.GetCurrentUser();
    Model.Title = null;
}
```

**Save Controller Action**

Is EntryPoint:  Enabled Access Log:  Causes Validation:

**SECURITY**

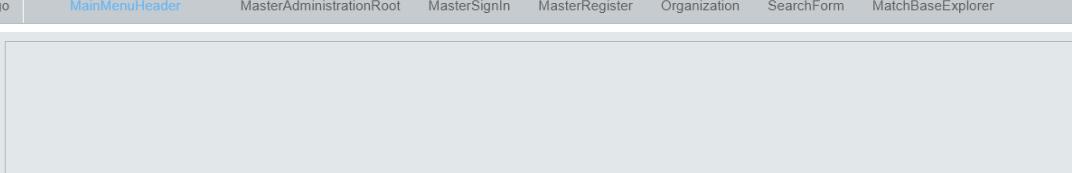
Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

**CODE**

```
function void Save()
{
    Model ApplicationUser.Save();
    CloseForm();
}
```

**MasterPage Form****Model**

```
|-- Title: string
```

**View**


The screenshot shows a web application interface. At the top, there is a navigation bar with the following items: Logo, MainMenuHeader (which is highlighted in blue), MasterAdministrationRoot, MasterSignIn, MasterRegister, Organization, SearchForm, and MatchBaseExplorer. Below the navigation bar is a large content area with a light gray background. In the bottom right corner of the content area, there is a block of JavaScript code:

```
<script src="https://cdnjs.cloudflare.com/ajax/libs/OwlCarousel2/2.3.4/owl.carousel.min.js" integrity="sha512-bPs7Ae6pVvhOSilcyUCIR7/q2OAsRiovw4vAkX+zJbw3ShAeeqezq50RIclURq7Oa20rW2n2q+fyXBNCU9lrw==" crossorigin="anonymous"></script> <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/OwlCarousel2/2.3.4/assets/owl.carousel.min.css" integrity="sha512-TS3S5qG0BlhnQRoYJXvNjeEM4UpMXHrQfTGmbQ1gKmElCxISeBUaxhRBj/EFTzpbP4RVsrEikbmdJobCvhE3g==" crossorigin="anonymous" /> <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/OwlCarousel2/2.3.4/assets/owl.theme.default.min.css" integrity="sha512-sMXIMNL1zRzoIHYKEujM2AqCLUR9F2C4/05cdbxjlSRvMQiciEPCQZo++nk7go3BtSuK9kfa/s+a4f4i5pLkw==" crossorigin="anonymous" /> <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/OwlCarousel2/2.3.4/assets/owl.theme.green.min.css" integrity="sha512-C8Movfk6DU/H5PzarG0+Dv9MA9IZzvrmQpO/3cIGflmtY3vlud07myMu4M/NTPJl8jmZtt/4mC9bAioMZBBdA==" crossorigin="anonymous" /> <div class="row_style"> <svg id="pattern" class="round"
```

xmins="http://www.w3.org/2000/svg" version="1.1" width="100%" height="100%" viewBox="0 0 100 100" preserveAspectRatio="none">> <path d="M0 100 C40 0 60 100 Z"></path> </div> <div class="container main-container"> <div class="row"> <div class="col-xs-12"> <div class="owl-carousel owl-theme" data-col\_lg="12" data-col\_md="6" data-col\_sm="4" data-col\_xs="1" data-item\_space="15" data-loop="true" data-autoplay="true" data-smartspeed="400" data-nav="false" data-dots="false"> <div><a href="https://www.capdigital.com/en/"></a></div> <div><a href="https://www.digipolis.fi/en/front-page"></a> </div> <div class="active"><a href="https://www.ctninnova.com/"></a></div> <div class="active"><a href="https://www.f6s.com/"></a></div> <div class="active"><a href="https://www2.deloitte.com/it/it/pages/about-deloitte/topics/officine-innovazione---deloitte-italy---about.html?icid=wn\_officine-innovazione---deloitte-italy---about"></a> </div> <div class="active"><a href="http://www.polito.it/"></a></div> <div href="https://fasttrack.vc/"></a></div> <div href="https://draxis.gr/"></a></div> <div href="https://www.clmsuk.com/"></a></div> <div><a href="https://www.arthurslegal.com/"></a></div> <div href="https://www.inspiringculture.org/"></a></div> <div class="row"> <div class="col-lg-6 col-md-6 col-xs-12 disclaimer-col"> <div class="disclaimer-container">  <p class="disclaimer-text">This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 873468.</p> </div> </div> <div class="col-lg-6 col-md-6 col-xs-12 zappdev-col"> <div class="zappdev-container"> <span>Crafted by</span> <a href="http://zappdev.com/"> <svg id="Layer\_1" data-name="Layer 1" xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink" viewBox="0 0 1201 317"><defs> <style>.cls-1{fill:none;}.cls-2{clip-path:url(#clip-path);}.cls-3{fill:#f4f4f4;}</style> <clipPath id="clip-path"><rect class="cls-1" x="14" y="10.28" width="1173" height="296.44"/></clipPath></defs> <title>zappdev-white</title> <g class="cls-2"> <polyline class="cls-3" points="152.36 198.03 203.89 71.19 34.3 71.33 17 117.53 122.81 117.45 152.36 198.03"/> <polygon class="cls-3" points="95.1 225.37 65.55 144.79 14 271.62 183.59 271.5 200.89 225.29 95.1 225.37"/> <path class="cls-3" d="M363.17,206.43v-.18h12.62L363.13,172.6v.14L326.39,75.32l-55.57,0L195.71,271.48i53.8,0,13.2-32.95,71.15,0,12.08,32.93,54.38,0-24.49-64.93Zm-86.38-6.73,21.43-63.49,22.08,63.47Z"/> <path class="cls-3" d="M586.56,128.19c-15-14.4-32.65-17.62-49.71-17.6l-31.82,0c6.07-11.63,7.09-24.7,0.97-35.11,0-16.46-3-32.93-18.26-47.6-15-14.4-32.66-17.63-49.7-17.61L363.10,35.11,132.67L387,206.4127,0L414,141.43i32,0c3.15,0,6.39-1.9,7.4-3.8l.12,165.7,50.85,0,0-65.32,0c15.29,0,33.51-2.09,48.78-17.08s17.32-33.23,17.32-48.84c0-16.45-3-32.92-18.26-47.59M452.46,94.06c-6.17,5.6-16.17,6.19-21.46,6.19H414l-48.78,17.35,0c6.45,0,15.86,88,21.76,7.5,29.5,28,5.87,12.34,5.87,17.64,0,4.7-28,12.63-6.45,17.92"/> <path class="cls-3" d="M761.14,102C735.83,77.65,702.9,75.683,2.75l-66.74.06,14,196.08,71.16,0c33.51,0,59.66-14.16,75.23-29.75,19.09-19.14,25.54-42.08,25.53-67.94,0-21.17-4.45-49.4-27.38-71.42m-40.5,111.74c-13.22,12.94-30.57,14.14-42.91,14.14-10.3,0-0.08-109.65,12.35,0c12.64,0,28.51,1.15,40.58,12.3,9.72,9.12,15.31,24.71,15.31,42.93,0,21.74-8.48,34.09-15.40,28"/> <path class="cls-3" d="M1000.71,178.43c0-20.47-3.32-52.39-29.11-76.92C949.9,81.06,922.48,76.6,901.2,76.6c-36.0-59.74,11.91-74.87,26.66-16,15.55-27.38,40.12-27.36,70.81,0,34.78,15.19,57.26,27.06,69.13,22.52,22.5,51.59,27.8,75.73,27.78,39.68,0,60.94-12.33,74.44-25.43a82.42,82.42,0,0,0,22.09-37.65l-62.63,0a30.32,30.32,0,0,1-11.44,12.28c-8.19,4.5-19.64,4.92-21.27,4.92-14.73,-0.22,92.4-8.87-27-9.77-7.77-11.47-20.85-11.49-30.66l136.27,-12ZM866.47,147.82a37.43,37.43,0,0,1,9.8-19.63c7-7,15.55-9.84,26.6-9.84,6.55,0,18.4,1.2,27,9.8a44,44,0,0,1,10.65,19.63Z"/> <polygon class="cls-3" points="1131.73,74.71,1084.18,194.1,1037.07,74.79,981.5,74.83,1063.97,270.86,1103.65,270.84,1186.98,74.67,1131.73,74.71"/> </g> </svg> <a href="https://digidirc.eu/privacy-policy/"><span style="color: #ff6600;">Privacy Policy</span></a> </strong></p> </div> </div> <div> <p class="copyright">Copyright &#169; 2020 CLMS. All Rights reserved. <strong><a href="https://digidirc.eu/privacy-policy/"><span style="color: #ff6600;">Privacy Policy</span></a> </strong></p> </div> </div> <script> Joove.Widgets.MenuControl.prototype.HandleOverflowMenuItems = function (menuQuery, name, leftItemsToShow) {};

## Controller

## Render Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

## SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
-----------------	-------------------------	------------	-----------------	-------------

<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
-------------------------------------	-------------------------------------	-------------------------------------	--	--

#### CODE

```
function void Render()
{
    Model.Title = LocalResources.RES_PAGETITLE_Render;
}
```

#### SignOut Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void SignOut()
{
    AppLib.Security.SignOutUser();
    FormModels.SignInPage.Controller.Load.Execute(true);
}
```

#### MasterPageForSlide Form

##### Model

```
|-- Title: string
```

##### View

% Title

#### Controller

##### Render Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions

<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
-------------------------------------	-------------------------------------	-------------------------------------	--	--

#### CODE

```
function void Execute() {
}
```

### ApplicationSettingForm Form

#### Model

```
|-- ApplicationSetting: ApplicationSetting
|-- Id: int
|-- IsCustom: bool
|-- Key: string
|-- Value: string
```

#### View

Details

Key

Value

Is Custom

Save
Delete
Exit

#### Controller

##### AddApplicationSetting Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ManageSettings

**CODE**

```
function void Execute() {
    Model.ApplicationSetting = Domain.ApplicationSetting.Create();
    Model.ApplicationSetting.IsCustom = true;
    Model.Title = LocalResources.RES_PAGETITLE_AddApplicationSetting;
}
```

**EditApplicationSetting Controller Action**Is Entrypoint:  Enabled Access Log:  Causes Validation: **SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ManageSettings

**CODE**

```
function void Execute(int Id) {
    Model.ApplicationSetting = Domain.ApplicationSetting.GetByKey(Id);
    Model.Title = LocalResources.RES_PAGETITLE_EditApplicationSetting;
}
```

**SaveApplicationSetting Controller Action**Is Entrypoint:  Enabled Access Log:  Causes Validation: **SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ManageSettings

**CODE**

```
function void Execute() {
    Model.ApplicationSetting.Save();
    CloseForm();
}
```

**DeleteApplicationSetting Controller Action**Is Entrypoint:  Enabled Access Log:  Causes Validation: **SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ManageSettings

**CODE**

```
function void Execute() {
    Model.ApplicationSetting.Delete();
    CloseForm();
}
```

**ApplicationSettingsList Form**

## Model

### View

%		Title	Unsaved Changes!			
		Create	Edit			
Page 1 of 10 pages		<<	<<	>>	>>	25 per page
		Key	Value	Is Custom		
1	abc	abc		<input checked="" type="checkbox"/>		
2	abc	abc		<input checked="" type="checkbox"/>		
3	...	...		<input type="checkbox"/>		

## Controller

### Retrieve Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ManageSettings

#### CODE

```
function void Retrieve()
{
    Model.Title = LocalResources.RES_PAGETITLE_Retrieve;
}
```

### Refresh Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ManageUsers

#### CODE

```

function void Refresh()
{
    FormControls.List.Refresh();
}

```

## ChangePassword Form

### Model

```

!-- txtCurrent: string
!-- txtNew: string
!-- txtNewRepeat: string

```

### View

The screenshot shows a user interface for a 'Change Password' form. The title bar is labeled 'Change Password'. The form contains three text input fields: 'Current password' (labeled 'txtCurrent'), 'New password' (labeled 'txtNew'), and 'Confirm password' (labeled 'txtNewRepeat'). Below each input field is a validation message: '%\_Validations.CurrentPasswordEmpty.Message' for the current password, '%\_Validations.NewPasswordEmpty.Message' for the new password, and '%\_Validations.RepeatPasswordEmpty.Message' for the confirm password. At the bottom of the form are two buttons: 'Change' and 'Cancel'.

### Controller

#### Render Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions

<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
--------------------------	-------------------------------------	-------------------------------------	--	--

#### CODE

```
function void Render()
{
    Model.Title = null;
}
```

#### ChangePassword Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void ChangePassword()
{
    if Model.txtNewRepeat != Model.txtNew
    {
        ShowMessage(LocalResources.RES_CUSTOM_PassError, AppLib.MessageType.Error);
        return;
    }

    string result =
AppLib.Security.ChangePasswordOfUser(AppLib.Session.GetCurrentUser(),
Model.txtCurrent, Model.txtNew);

    if !string.IsNullOrEmpty(result)
    {
        ShowMessage(result, AppLib.MessageType.Error);
        return;
    }

    CloseForm();
}
```

#### ForgotPassword Form

##### Model

```
|-- txtUsername: string
|-- FromMatching: bool
```

##### View

## Controller

### Render Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Render(bool fromMatching)
{
    Model.FromMatching = fromMatching;
}
```

### ResetPasswordRequest Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions



## CODE

```
function void ResetPasswordRequest()
{
    Domain ApplicationUser user = Domain ApplicationUser.GetKey(Model.txtUsername);

    if user == null
    {
        ShowMessage(LocalResources.RES_CUSTOM_NotFound, AppLib.MessageType.Error);
        return;
    }

    if string.IsNullOrWhiteSpace(user.Email) ||
    !RegExpLib.VerbalExpressions.IsEmail(user.Email)
    {
        ShowMessage(LocalResources.RES_CUSTOM_NoMail, AppLib.MessageType.Error);
        return;
    }

    string key = AppLib.Security.GeneratePasswordResetToken(user);
    var resetUrl = Controller.ResetPassword.GetLink(Model.FromMatching, user.UserName,
CommonLib.Utilities.EncodeUrl(key));

    CommonLib.EmailMessage mail;

    Collection<string> recipients;
    recipients.Add(user.Email);

    mail.To = recipients;
    mail.IsBodyHtml = true;
    mail.Subject = LocalResources.RES_CUSTOM_ResetPasswordLink + " " +
AppLib.Application.Name;
    mail.Body = "<h3>" + LocalResources.RES_CUSTOM_ClickToReset + "</h3>" +
        "<a href='!" + resetUrl + "'>" + LocalResources.RES_CUSTOM_ResetPassword +
        "</a>" +
        "<h3>" + LocalResources.RES_CUSTOM_CopyPaste + "</h3><p>" + resetUrl +
        "</p>";

    CommonLib.Utilities.SendEmail(mail);

    string signInUrl =
FormModels.SignInPage.Controller.Load.GetLink(Model.FromMatching);
    ShowMessage(LocalResources.RES_CUSTOM_MailSoon, AppLib.MessageType.Success,
signInUrl);
}
```

## ResetPassword Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

## SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
-----------------	-------------------------	------------	-----------------	-------------

<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
-------------------------------------	-------------------------------------	-------------------------------------	--	--

## CODE

```

function void ResetPassword(bool fromMatching, string username, string key)
{
    if (string.IsNullOrWhiteSpace(username)) {
        ShowMessage(LocalResources.RES_CUSTOM_InvalidLink, AppLib.MessageType.Error);
        return;
    }

    Domain ApplicationUser user = Domain ApplicationUser.GetByKey(username);

    if user == null
    {
        ShowMessage(LocalResources.RES_CUSTOM_NotFound, AppLib.MessageType.Error);
        return;
    }

    string newPassword = "Pa55!" + Guid.NewGuid().ToString().ToLower().Replace("-", "").Substring(0, 10);
    bool success = AppLib.Security.ResetPasswordOfUser(user, key, newPassword);

    if !success
    {
        ShowMessage(LocalResources.RES_CUSTOM_InvalidLink, AppLib.MessageType.Error);
        return;
    }

    string signInUrl =
FormModels.SignInPage.Controller.Load.GetLink(Model.FromMatching);

    ShowMessage(LocalResources.RES_CUSTOM_YourNewPass + " " + newPassword,
AppLib.MessageType.Success, signInUrl);
}

```

## ManageOperation Form

### Model

```

|-- ApplicationOperation: ApplicationOperation
|-- Id: int
|-- IsAvailableToAllAuthorizedUsers: bool
|-- IsAvailableToAnonymous: bool
|-- Name: string
|-- ParentControllerName: string
|-- Type: string
|-- Permissions: ApplicationPermission
|-- Description: string

```

```
|-- Id: int  
|-- IsCustom: bool  
|-- Name: string
```

## View

Details

Name	% ApplicationOperation.Name
Form	% ApplicationOperation.ParentControllerName
Type	% ApplicationOperation.Type
Available to anonymous users?	<input type="checkbox"/>
Available to all authorized users?	<input type="checkbox"/>

Permissions

	Name	Description
<input type="button" value="X"/>	% Name	% Description
<input type="button" value="Add"/>		

## Controller

### EditOperation Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ManageOperations

#### CODE

```
function void EditOperation(  
    int Id  
)  
{  
    Model.ApplicationOperation = Domain.ApplicationOperation.GetByKey(Id);  
    Model.Title = LocalResources.RES_PAGETITLE_EditOperation;  
}
```

#### SaveOperation Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ManageOperations

#### CODE

```
function void SaveOperation()  
{  
    Model.ApplicationOperation.Save();  
    CloseForm();  
}
```

## ManagePermission Form

#### Model

```
|-- ApplicationPermission: ApplicationPermission  
|-- Description: string  
|-- Id: int  
|-- IsCustom: bool  
|-- Name: string  
|-- Operations: ApplicationOperation  
    |-- Id: int  
    |-- IsAvailableToAllAuthorizedUsers: bool  
    |-- IsAvailableToAnonymous: bool  
    |-- Name: string  
    |-- ParentControllerName: string  
    |-- Type: string  
|-- Roles: ApplicationRole  
    |-- Description: string  
    |-- Id: int  
    |-- IsCustom: bool  
    |-- Name: string
```

```

|-- Users: ApplicationUser
|--- AccessFailedCount: int
|--- Email: string
|--- EmailConfirmed: bool
|--- LockoutEnabled: bool
|--- LockoutEndDate: DateTime
|--- Name: string
|--- PasswordHash: string
|--- PhoneNumber: string
|--- PhoneNumberConfirmed: bool
|--- SecurityStamp: string
|--- TwoFactorEnabled: bool
|--- UserName: string

```

## View

**Details**

Name	<input type="text" value="ApplicationPermission.Name"/> <small>% Validation.Required_Name.Message</small>				
Description	<input type="text" value="ApplicationPermission.Description"/>				

**Allowed Operations**

	Controller	Name	Anonymous Users?	All Authorized Users?
	% ParentControllerName	% Name	Yes   No	Yes   No

**Add Operations**

**Assigned to Roles**

	Name	Description
--	------	-------------

	Name	Description
<input type="button" value="X"/>	% Name	% Description
<input type="button" value="Add Roles"/>		
Assigned to Users		
	Username	Email
<input type="button" value="X"/>	% UserName	% Email
<input type="button" value="Add Users"/>		
<input type="button" value="Save"/>	<input type="button" value="Delete"/>	<input type="button" value="Exit"/>

## Controller

### NewPermission Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ManagePermissions

#### CODE

```
function void NewPermission()
{
    Domain.ApplicationPermission permission = Domain.ApplicationPermission.Create();
    permission.IsCustom = true;
    Model.ApplicationPermission = permission;
    Model.Title = LocalResources.RES_PAGETITLE_NewPermission;
}
```

### EditPermission Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ManagePermissions

**CODE**

```
function void EditPermission(
    int Id
)
{
    Model.ApplicationPermission = Domain.ApplicationPermission.GetByKey(Id);
    Model.Title = LocalResources.RES_PAGETITLE_EditPermission;
}
```

**SavePermission Controller Action**

Is Entrypoint:  Enabled Access Log:  Causes Validation:

**SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ManagePermissions

**CODE**

```
function void SavePermission()
{
    Model.ApplicationPermission.Save();
    CloseForm();
}
```

**DeletePermission Controller Action**

Is Entrypoint:  Enabled Access Log:  Causes Validation:

**SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ManagePermissions

**CODE**

```
function void DeletePermission()
{
    Model.ApplicationPermission.Delete();
    CloseForm();
}
```

**ManageRole Form****Model**

```
|-- ApplicationRole: ApplicationRole
|-- Description: string
|-- Id: int
|-- IsCustom: bool
|-- Name: string
|-- Permissions: ApplicationPermission
    |-- Description: string
```

```

| -- Id: int
| -- IsCustom: bool
| -- Name: string

```

## View

**Details**

Name

% ApplicationRole.Name

% Validations.Required\_Name.Message

Description

≡ % ApplicationRole.Description

**Permissions**

	Name	Description
<input type="button" value="X"/>	% Name	% Description

Add Permissions

## Controller

### NewRole Controller Action

Is Endpoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ManageRoles

**CODE**

```
function void NewRole()
{
    Domain.ApplicationRole role = Domain.ApplicationRole.Create();
    role.IsCustom = true;
    Model.ApplicationRole = role;
    Model.Title = LocalResources.RES_PAGETITLE_NewRole;
}
```

**EditRole Controller Action**Is Entrypoint:  Enabled Access Log:  Causes Validation: **SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ManageRoles

**CODE**

```
function void EditRole(
    int Id
)
{
    Model.ApplicationRole = Domain.ApplicationRole.GetByKey(Id);
    Model.Title = LocalResources.RES_PAGETITLE_EditRole;
}
```

**SaveRole Controller Action**Is Entrypoint:  Enabled Access Log:  Causes Validation: **SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ManageRoles

**CODE**

```
function void SaveRole()
{
    Model.ApplicationRole.Save();
    CloseForm();
}
```

**DeleteRole Controller Action**Is Entrypoint:  Enabled Access Log:  Causes Validation: **SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ManageRoles

**CODE**

```

function void DeleteRole()
{
    Model.ApplicationRole.Delete();
    CloseForm();
}

```

## ManageUser Form

### Model

```

|-- ApplicationUser: ApplicationUser
|   |-- AccessFailedCount: int
|   |-- Email: string
|   |-- EmailConfirmed: bool
|   |-- LockoutEnabled: bool
|   |-- LockoutEndDate: DateTime
|   |-- Name: string
|   |-- PasswordHash: string
|   |-- PhoneNumber: string
|   |-- PhoneNumberConfirmed: bool
|   |-- SecurityStamp: string
|   |-- TwoFactorEnabled: bool
|   |-- UserName: string
|   |-- Permissions: ApplicationPermission
|       |-- Description: string
|       |-- Id: int
|       |-- IsCustom: bool
|       |-- Name: string
|   |-- Roles: ApplicationRole
|       |-- Description: string
|       |-- Id: int
|       |-- IsCustom: bool
|       |-- Name: string
|   |-- Password: string
|   |-- PasswordRetype: string
|   |-- NewPassword: string

```

### View

Details	
Name	<input type="text" value=" ApplicationUser.Name"/>
Username	<input type="text" value=" ApplicationUser.UserName"/>

% Validations.Required\_UserName.Message

Password

Password

Retype password

PasswordRetype

Email

% ApplicationUser.Email

Failed sign in attempts

% ApplicationUser.AccessFailedCount

Locked out



#### Permissions

	Name	Description
	% Name	% Description

Add Permissions...

#### Roles

	Name	Description
	% Name	% Description

Add Roles...

Set Password

New Password

aI % NewPassword

Set new Password

Save    Delete    Exit

## Controller

### NewUser Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ManageUsers

#### CODE

```
function void NewUser()
{
    Model.ApplicationUser = Domain ApplicationUser.Create();
    Model.Title = LocalResources.RES_PAGETITLE_NewUser;
}
```

### EditUser Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ManageUsers

#### CODE

```
function void EditUser(
    string Id
)
{
    Model.ApplicationUser = Domain ApplicationUser.GetByKey(Id);
    Model.Title = LocalResources.RES_PAGETITLE_EditUser;
}
```

### SaveUser Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

--	--	--	--	--

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ManageUsers

#### CODE

```
function void SaveUser()
{
    if Controller.NewUser.IsActive()
    {
        if Model.Password != Model.PasswordRetype
        {
            ShowMessage(LocalResources.RES_CUSTOM_PasswordsNoMatch,
AppLib.MessageType.Error);
            return;
        }

        string error = AppLib.Security.RegisterUser(Model.ApplicationUser,
Model.Password);

        if !string.IsNullOrEmpty(error)
        {
            ShowMessage(error, AppLib.MessageType.Error);
            return;
        }
    }
    else
    {
        Model ApplicationUser.Save();
    }

    CloseForm();
}
```

#### DeleteUser Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ManageUsers

#### CODE

```
function void DeleteUser()
{
    Model ApplicationUser.Delete();
    CloseForm();
}
```

#### SetPassword Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

--	--	--	--	--

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ManageUsers

#### CODE

```

function void SetPassword()
{
    if !AppLib.Security.PasswordIsValid(Model.NewPassword)
    {
        ShowMessage(LocalResources.RES_CUSTOM_NotValidPassword + " " +
Model.NewPassword);
        return;
    }

    var error = AppLib.Security.ResetPasswordOfUserAsAdmin(Model.ApplicationUser,
Model.NewPassword);

    if (string.IsNullOrWhiteSpace(error))
    {
        ShowMessage("OK", AppLib.MessageType.Success);
    }
    else
    {
        ShowMessage(error, AppLib.MessageType.Error);
    }
}

```

### OperationsList Form

#### Model

#### View

The screenshot shows a web-based application interface for managing operations. At the top, there is a header bar with a title placeholder "% Title" and a note "Unsaved Changes!". Below the header is a toolbar with various icons for file operations like upload, search, and refresh.

The main area displays a grid of data. The grid has a header row with the following column titles:

- Parent Controller Name
- Name
- Is Available To All Authorized Users
- Is Available To Anonymous

The data grid contains three rows of data, each with the following values:

	Parent Controller Name	Name	Is Available To All Authorized Users	Is Available To Anonymous
1	abc	abc	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
2	abc	abc	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
3	...	...	...	...

At the bottom of the grid, there are navigation controls for pages and a per-page dropdown set to "25 per page".

## Controller

### Retrieve Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ManageOperations

#### CODE

```
function void Retrieve()
{
    Model.Title = LocalResources.RES_PAGETITLE_Retrieve;
}
```

### Refresh Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ManageUsers

#### CODE

```
function void Refresh()
{
    FormControls.List.Refresh();
}
```

## PermissionsList Form

### Model

### View

Page 1 of 10 pages    **25 per page**

	Name	Description	Is Custom
1	abc	abc	<input checked="" type="checkbox"/>
2	abc	abc	<input checked="" type="checkbox"/>
3	...	...	...

## Controller

### Retrieve Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		ManagePermissions

#### CODE

```
function void Retrieve()
{
    Model.Title = LocalResources.RES_PAGETITLE_Retrieve;
}
```

### Refresh Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		ManageUsers

#### CODE

```
function void Refresh()
{
    FormControls.List.Refresh();
}
```

## RolesList Form

## Model

```
|-- ApplicationRole: ApplicationRole
|  -- Description: string
|  -- Id: int
|  -- IsCustom: bool
|  -- Name: string
|  -- Permissions: ApplicationPermission
|    -- Description: string
|    -- Id: int
|    -- IsCustom: bool
|    -- Name: string
```

## View

The screenshot shows a web-based application interface. At the top, there is a header bar with a title 'Title' and a message 'Unsaved Changes!'. Below the header are two buttons: 'Create' and 'Edit'. To the right of these buttons is a toolbar with various icons. The main area displays a table with three columns: 'Name', 'Description', and 'Is Custom'. The table has 10 rows, numbered 1 to 10. Rows 1 and 2 have 'abc' in the 'Name' column and 'abc' in the 'Description' column. Row 3 has '...' in both columns. The 'Is Custom' column contains checked checkboxes for rows 1 and 2, and an ellipsis for row 3. Above the table, there is a page navigation bar showing 'Page 1 of 10 pages' and '25 per page'.

	Name	Description	Is Custom
1	abc	abc	<input checked="" type="checkbox"/>
2	abc	abc	<input checked="" type="checkbox"/>
3	...	...	...
4	...	...	...
5	...	...	...
6	...	...	...
7	...	...	...
8	...	...	...
9	...	...	...
10	...	...	...

## Controller

### Retrieve Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ManageRoles

### CODE

```
function void Retrieve()
{
    Model.Title = LocalResources.RES_PAGETITLE_Retrieve;
}
```

### Refresh Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

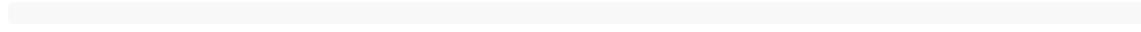
Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ManageUsers

#### CODE

```
function void Refresh()
{
    FormControls.List.Refresh();
}
```

### UsersList Form

#### Model



#### View

	User Name	Email	Name	Lockout Enabled
1	abc	abc	abc	<input checked="" type="checkbox"/>
2	abc	abc	abc	<input checked="" type="checkbox"/>
3	...	...	...	...

#### Controller

##### Retrieve Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ManageUsers

#### CODE

```

function void Retrieve()
{
    Model.Title = LocalResources.RES_PAGETITLE_Retrieve;
}

```

#### **Refresh Controller Action**

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### **SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ManageUsers

#### **CODE**

```

function void Refresh()
{
    FormControls.List.Refresh();
}

```

### **MasterPageSignIn Form**

#### **Model**

#### **View**



#### **Controller**

#### **Render Controller Action**

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### **SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### **CODE**

```
function void Render() {  
}  
}
```

## CreateAdmin Form

### Model

```
|-- username: string  
|-- password: string  
|-- repeatPassword: string  
|-- email: string
```

### View

Administrator User Setup

Username

Email

Password

Repeat Password

Create

### Controller

#### Index Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

**CODE**

```
function void Index() {
    Controller.AuthorizeAccess.Execute();
}
```

**Create Controller Action**

Is Entrypoint:  Enabled Access Log:  Causes Validation:

**SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

**CODE**

```
function void Create()
{
    Controller.AuthorizeAccess.Execute();

    // This is handled client side by a Data Validation
    if Model.password.Trim() != Model.repeatPassword.Trim() {
        throw "Passwords do not match!";
    }

    // This is handled client side by a Data Validation
    if Model.username.Trim() == "" {
        throw "No username provided!";
    }

    Domain.ApplicationRole adminRole =
        Domain.ApplicationRole.Find(r => r.Name == "Administrator").First();

    if (adminRole == null) {
        throw "No Administrator role found in Database!";
    }

    Domain ApplicationUser adminUser;
    adminUser.UserName = Model.username.Trim();
    adminUser.Email = Model.email.Trim();
    adminUser.Roles.Add(adminRole);

    string possibleError = AppLib.Security.RegisterUser(adminUser,
Model.password.Trim());

    if !string.IsNullOrEmpty(possibleError)
    {
        ShowMessage(possibleError, AppLib.MessageType.Error);
        return;
    }

    FormModels.SignInPage.Controller.Load.Execute(true);
}
```

### **AuthorizeAccess Controller Action**

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### **SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### **CODE**

```
function void AuthorizeAccess()
{
    if (Domain ApplicationUser.Find(u => u.Roles.Any(r => r.Name ==
"Administrator")).Length > 0) {
        throw "There is already a user with the Administrator role!";
    }
}
```

### **RegisterForm Form**

#### **Model**

```
|-- DigicircUser: DigicircUser
|-- AccessFailedCount: int
|-- Email: string
|-- EmailConfirmed: bool
|-- FirstName: string
|-- LastName: string
|-- LockoutEnabled: bool
|-- LockoutEndDate: DateTime
|-- Name: string
|-- PasswordHash: string
|-- PhoneNumber: string
|-- PhoneNumberConfirmed: bool
|-- SecurityStamp: string
|-- SubscribeToNewsLetter: bool
|-- TwoFactorEnabled: bool
|-- UserName: string
|-- Roles: ApplicationRole
|-- Description: string
|-- Id: int
|-- IsCustom: bool
|-- Name: string
|-- Permissions: ApplicationPermission
|-- Id: int
|-- Name: string
|-- Description: string
|-- IsCustom: bool
|-- Password: string
|-- RetypePassword: string
|-- UserName: string
|-- AcceptTerms: bool
|-- FromMatching: bool
```

**View**

Register



First Name	Last Name
<input type="text" value="aI % DigicircUser.FirstName"/>	<input type="text" value="aI % DigicircUser.LastName"/>

Username	
<input type="text" value="aI % UserName"/>	

Password	Retype Password
<input type="password" value="aI % Password"/>	<input type="password" value="aI % RetypePassword"/>

Email	
<input type="text" value="aI % DigicircUser.Email"/>	

<input type="checkbox"/>	Subscribe To Newsletter
--------------------------	-------------------------

<input type="checkbox"/>	Accept the <a href="#">Terms</a>
--------------------------	----------------------------------

Already a member? [Sign in](#)

**Controller**

### Index Controller Action

Is EntryPoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Index(bool fromMatching)
{
    Model.FromMatching = fromMatching;
}
```

### Register Controller Action

Is EntryPoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Register()
{
    if (Model.AcceptTerms == false)
    {
        ShowMessage("Terms must be accepted!");
        return;
    }

    if(Domain ApplicationUser.Find(a => a.UserName == Model.UserName).Length > 0)
    {
        ShowMessage("There is another user with the same username",
AppLib.MessageType.Error);
        return;
    }
    DebugLib.Logger.WriteLine("aaaaaa");
    Domain.ApplicationRole userRole =
        Domain.ApplicationRole.Find(r => r.Name == Application.Roles.User).First();

    if (userRole == null)      {
        throw "No Suitable role found in Database!";
    }

    Model.DigicircUser.Roles.Add(userRole);
    Model.DigicircUser.UserName = Model.UserName;
    Model.DigicircUser.Name = Model.DigicircUser.FirstName + " " +
Model.DigicircUser.LastName;

    string possibleError = AppLib.Security.RegisterUser(Model.DigicircUser,
Model.Password.Trim());
```

```

if !string.IsNullOrEmpty(possibleError)
{
    ShowMessage(possibleError, AppLib.MessageType.Error);
    return;
}

FormModels.SignInPage.Controller.Load.Execute(Model.FromMatching);
}

```

## SearchForm Form

### Model

```

|-- SelectedMode: string
|-- SelectedActor: Actor
|--- ClusterName: string
|--- Description: string
|--- Email: string
|--- GetCountOfClusterMembers: int
|--- HasSites: bool
|--- Id: int
|--- Keywords: string
|--- MemberOfCluster: bool
|--- Name: string
|--- ShortDescription: string
|--- SpecifiedEntityType: string
|--- Url: string
|-- Query: SearchQuery
|--- AdvanceSearch: bool
|--- ExperienceInCircularEconomy: bool
|--- GetSearchTerm: string
|--- MaterialSearchMode: string
|--- SearchTerm: string
|--- SelectedMode: string
|--- ShowAllData: bool
|--- ShowSavedPage: bool
|--- SelectedCountry: Country
|--- Id: int
|--- Name: string
|--- ShortName: string
|-- SelectedSector: SectorType
|--- Code: string
|--- Id: int
|--- Value: string
|-- ActorNames: ActorNames
|--- Id: int
|--- Name: string
|-- SelectedMaterial: Material
|--- Description: string
|--- HsSpecific: string
|--- Id: int

```

```

| -- IsHazardous: bool
| -- Name: string
| -- PendingGraph: bool

```

## View

The screenshot displays a user interface for filtering and searching data. On the left, a sidebar titled "Filtering Criteria" contains dropdown menus for "Country" (selected to "Query.SelectedCountry"), "Sector" (selected to "Query.SelectedSector"), "Material" (selected to "Query.MaterialSearchMode"), and "Matchmaking" (empty). Below these are search fields for "Query.SearchTerm" and buttons for "Search" and "Reset". On the right, a main panel titled "Display results in: Query.SelectedMode" shows a placeholder "Bubble(SelectedActor)" with a location pin icon. Below this, a foreach loop iterates over "Table\_CurrentItem" in "ActorDataSet1", displaying a thumbnail image and item details: "ActorDataSet1.Item.Name", "ActorDataSet1.Item.ShortDescription", "ActorDataSet1.Item.EntityType.Value", and "ActorDataSet1.Item.Address.Country.Name". A footer bar at the bottom indicates "ActorDataSet1.TotalItems Results Found".

## Controller

### Index Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

### CODE

```

function void Index() {
    Model.Title = "";
    Model.Query.ShowAllData = true;
    Model.Query.MaterialSearchMode = "offers";
//    Domain.SearchQuery cachedQuery = AppLib.Session.Storage.Get("results") as
Domain.SearchQuery;
//    if(cachedQuery != null)
//    {
//        Model.Query = cachedQuery;
//        if(cachedQuery.ShowSavedPage)
//        {
//        }
//    }
//    else
//    {
        Model.Query.SelectedMode = "list";
        Model.Query.AdvanceSearch = false;
    //}
//ExecuteJavascript("setTimeout(function(){
window._commander.gridGoToSavedPage(['Table']), 200);");
}

```

### FromBack Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		User

#### CODE

```

function void FromBack(int id)
{
    Model.Title = "";
    Model.Query.MaterialSearchMode = "offers";
    Domain.SearchQuery cachedQuery = AppLib.Session.Storage.Get("results") as
Domain.SearchQuery;
    if(cachedQuery != null)
    {
        Model.Query = cachedQuery;
    }
    else
    {
        Model.Query.SelectedMode = "list";
        Model.Query.AdvanceSearch = false;
        Model.Query.ShowAllData = true;
    }
    if(Model.Query.SelectedMode == "list")
    {
        ExecuteJavascript("setTimeout(function(){
window._commander.gridGoToSavedPage(['Table']), 200);");
        ExecuteJavascript("setTimeout(function(){ $(`[data-key='" + id + "']`)\n

```

```
[0].scrollIntoView({ behavior: \"smooth\", block: \"start\" })), 1500);";
    }
}
```

### **ChangeMode Controller Action**

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### **SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		User

#### **CODE**

```
function void ChangeMode()
{
    AppLib.Session.Storage.Add("results", Model.Query);
}
```

### **TestBubble Controller Action**

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### **SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		User

#### **CODE**

```
function void TestBubble(Domain.Actor actor)
{
    Model.SelectedActor = actor;
}
```

### **Search Controller Action**

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### **SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		User

#### **CODE**

```
function void Search()
{
    var countryId = Model.Query.SelectedCountry == null ? 0 :
Model.Query.SelectedCountry.Id;

    Collection[Domain.Actor] searchActors;
    if(countryId == 0 && string.IsNullOrEmpty(Model.Query.SearchTerm) )
    {
        searchActors = Domain.Actor.GetAll();
    }
}
```

```

else
{
    if(!string.IsNullOrEmpty(Model.Query.SearchTerm))
    {
        if(Model.Query.SearchTerm.Trim().StartsWith("\\")) &&
Model.Query.SearchTerm.EndsWith("\\"))
        {
            searchActors = Domain.Actor.GetAll();
        } else
        {
            Array[string] terms = Model.Query.SearchTerm.Split(' ');
            foreach string term in terms
            {
                DebugLib.Logger.WriteLine(term);
                searchActors.AddRange(Domain.Actor.Find(a =>
a.Description.Contains(term)));
            }
        }
    } else
    {
        searchActors = Domain.Actor.GetAll();
    }
    if(countryId != 0)
    {
        searchActors = searchActors.Where(a => a.Address.Country.Id == countryId);
    }

    //Model.Query.Results = searchActors;
}

AppLib.Session.Storage.Add("results", Model.Query);

FormControls.NewMap.Refresh();
FormModels.SearchForm.Controller.Refresh.Execute();
}

```

### SearchGraph Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		User

#### CODE

```

function void SearchGraph()
{
    Model.Query.ShowAllData = false;
    Collection[Domain.ExElements] q;

    if(!string.IsNullOrEmpty(Model.Query.SelectedCountry.Name))

```

```

{
    var countryClause = Domain.ExElements.Create();
    countryClause.Type = "Country";
    countryClause.Name = Model.Query.SelectedCountry.Name;
    q.Add(countryClause);
}

if(!string.IsNullOrEmpty(Model.Query.SelectedSector.Value))
{
    var sectorClause = Domain.ExElements.Create();
    sectorClause.Type = "Sector";
    sectorClause.Name = Model.Query.SelectedSector.Value;
    q.Add(sectorClause);
}

var clause = Domain.ExElements.Create();
clause.Type = "*";
clause.Name = Model.Query.GetSearchTerm;

q.Add(clause);

Domain.GraphBackendResponse response;
if(string.IsNullOrEmpty(Model.Query.SelectedCountry.Name) &&
string.IsNullOrEmpty(Model.Query.SelectedSector.Value))
{
    response = Domain.GraphQueries.Query(Model.Query.GetSearchTerm);
}
else
{
    response = Domain.GraphQueries.ExtenedQuery(q);
}

var responseElastic = Domain.ElasticConsumer.Search(Model.Query);

DebugLib.Logger.WriteLine(response);

//var actorNames = Domain.Actor.GetActorNamesFromGraphResponse(response);
var actorNames = Domain.Actor.GetActorNamesFromElasticResponse(responseElastic);
Model.Query.ActorNames = actorNames;
AppLib.Session.Storage.Add("results", Model.Query);
FormControls.NewMap.Refresh();
ExecuteJavascript("setTimeout(function()
{_commander.gridGotoFirstPage(['Table']);},50);");
}

```

### NewSearch Controller Action

Is Endpoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		User

**CODE**

```
function void NewSearch()
{
    if (string.IsNullOrEmpty(Model.Query.SearchTerm)
        && Model.Query.SelectedCountry == null
        && Model.Query.SelectedSector == null
        && Model.Query.SelectedMaterial == null)
    {
        Model.Query.ShowAllData = true;
        Model.Query.Reset();
        //FormControls.Table.Refresh();
        AppLib.Session.Storage.Remove("results");
        FormControls.NewMap.Refresh();
    }
    else
    {
        Model.Query.ShowAllData = false;
        var responseElastic = Domain.ElasticConsumer.Search(Model.Query);
        var actorNames =
Domain.Actor.GetActorNamesFromElasticResponse(responseElastic);
        Model.Query.ActorNames = actorNames;
        AppLib.Session.Storage.Add("results", Model.Query);
        FormControls.NewMap.Refresh();
        ExecuteJavascript("setTimeout(function()
{_commander.gridGotoFirstPage(['Table']);},50);");
    }
}
```

**Refresh Controller Action**Is Entrypoint:  Enabled Access Log:  Causes Validation: **SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		User

**CODE**

```
function void Refresh()
{
    FormControls.Table.Refresh();
    //FormControls.lblLogods.Refresh();
    ExecuteJavascript("setTimeout(function()
{_commander.gridGotoFirstPage(['Table']);},50);");
    ExecuteJavascript("setTimeout(function(){
window._commander.imageRefresh(['lblLogods']) }, 200);");

}
```

**Reset Controller Action**Is Entrypoint:  Enabled Access Log:  Causes Validation:

**SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		User

**CODE**

```
function void Reset()
{
    Model.Query.Reset();
    Model.Query.ShowAllData = true;
    //FormControls.Table.Refresh();
    AppLib.Session.Storage.Remove("results");
    FormControls.NewMap.Refresh();
    ExecuteJavascript("setTimeout(function()
{_commander.gridGotoFirstPage(['Table']);},50);");
    ExecuteJavascript("setTimeout(function()
{_commander.gridClearState(['Table']);},50);");
//    FormModels.SearchForm.Controller.Refresh.Execute();
}
```

**Action Controller Action**

Is Entrypoint:  Enabled Access Log:  Causes Validation:

**SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		User

**CODE**

```
function void FirstPageAndReset()
{
}
```

**GoToActorForm Controller Action**

Is Entrypoint:  Enabled Access Log:  Causes Validation:

**SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		User

**CODE**

```
function void GoToActorForm(int id)
{
    //Model.Query.ShowSavedPage = true;
    //AppLib.Session.Storage.Add("results",Model.Query);
    ExecuteJavascript("_commander.gridSaveState(['Table']);");
    FormModels.ActorViewForm.Controller.Show.Execute(id, false);
}
```

## ResultsForm Form

### Model

### View

The screenshot shows a web page with a header bar. Below it, a list of company data items is displayed in a table-like structure. Each item has fields for Name, Email, and Country, followed by an 'Edit' button. A summary at the bottom indicates there are 10 total items.

Table_CurrentItem	
Name:	% CompanyDataSet.Item.Name
Email:	% CompanyDataSet.Item.Email
Country:	% CompanyDataSet.Item.Address.Country.Name
Records:	10

### Controller

#### Index Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		User

#### CODE

```
function void Index() {  
}  
}
```

## ActorForm Form

### Model

```

|-- Actor: Actor
    |-- ClusterName: string
    |-- Description: string
    |-- Email: string
    |-- GetCountOfClusterMembers: int
    |-- HasSites: bool
    |-- Id: int
    |-- Keywords: string
    |-- MemberOfCluster: bool
    |-- Name: string
    |-- ShortDescription: string
    |-- SpecifiedEntityType: string
    |-- Url: string
    |-- Address: Address
        |-- Alias: string
        |-- FullAddress: string
        |-- Id: int
        |-- Latitude: double
        |-- Longitude: double
        |-- Number: int
        |-- StreetName: string
        |-- Town: string
        |-- Zip: string
    |-- Country: Country
        |-- Id: int
        |-- Name: string
        |-- ShortName: string
    |-- EntityType: EntityType
        |-- Code: string
        |-- Id: int
        |-- IsCluster: bool
        |-- IsProvider: bool
        |-- Value: string
    |-- CircularEconomyRequirements: CircularEconomyReport
        |-- DigitalExpertise: DigitalExpertise
        |-- DigitalProviredNeeded: bool
        |-- ExperienceInCircularEconomy: bool
        |-- GetExperienceInCircularEconomy: string
        |-- Id: int
        |-- SpecifyExperienceInCircularEconomy: string
        |-- ThematicExpertiseNeeded: bool
        |-- DesiredThematicExpertises: ThematicExpertise
            |-- Code: string
            |-- Id: int
            |-- Value: string
    |-- DesiredSMESector: SectorType
        |-- Code: string
        |-- Id: int
        |-- Value: string
    |-- DesiredGeographicalArea: GeographicalArea
        |-- Id: int
    |-- SectorTypes: SectorType

```

```
|-- Code: string
|-- Id: int
|-- Value: string
|-- CircularEconomyProviderReport: CircularEconomyProviderReport
    |-- AvailableTestingFacilities: bool
    |-- Id: int
    |-- PlaceOperates: GeographicalArea
        |-- Id: int
    |-- Expertises: Expertise
        |-- Code: string
        |-- Id: int
        |-- Value: string
    |-- ServicesProvided: Services
        |-- Code: string
        |-- Id: int
        |-- Value: string
    |-- ThematicExpertises: ThematicExpertise
        |-- Code: string
        |-- Id: int
        |-- Value: string
|-- ActorLogo: FileData
    |-- AllowedExtensions: string
    |-- Blob: Collection[byte]
    |-- FileName: string
    |-- FolderPath: string
    |-- Id: Guid
    |-- MaxFileSize: int
    |-- StorageMedium: StorageMedium
    |-- UploadDateTime: DateTime
    |-- UploadedBy: string
|-- AddedBy: DigicircUser
    |-- AccessFailedCount: int
    |-- Email: string
    |-- EmailConfirmed: bool
    |-- FirstName: string
    |-- LastName: string
    |-- LockoutEnabled: bool
    |-- LockoutEndDate: DateTime
    |-- Name: string
    |-- PasswordHash: string
    |-- PhoneNumber: string
    |-- PhoneNumberConfirmed: bool
    |-- SecurityStamp: string
    |-- SubscribeToNewsLetter: bool
    |-- TwoFactorEnabled: bool
    |-- UserName: string
|-- Cluster: Actor
    |-- ClusterName: string
    |-- Description: string
    |-- Email: string
    |-- GetCountOfClusterMembers: int
    |-- HasSites: bool
```

```
|-- Id: int
|-- Keywords: string
|-- MemberOfCluster: bool
|-- Name: string
|-- ShortDescription: string
|-- SpecifiedEntityType: string
|-- Url: string
|-- Administrators: DigicircUser
    |-- AccessFailedCount: int
    |-- Email: string
    |-- EmailConfirmed: bool
    |-- FirstName: string
    |-- LastName: string
    |-- LockoutEnabled: bool
    |-- LockoutEndDate: DateTime
    |-- Name: string
    |-- PasswordHash: string
    |-- PhoneNumber: string
    |-- PhoneNumberConfirmed: bool
    |-- SecurityStamp: string
    |-- SubscribeToNewsLetter: bool
    |-- TwoFactorEnabled: bool
    |-- UserName: string
|-- Sites: Address
    |-- Alias: string
    |-- FullAddress: string
    |-- Id: int
    |-- Latitude: double
    |-- Longitude: double
    |-- Number: int
    |-- StreetName: string
    |-- Town: string
    |-- Zip: string
    |-- Country: Country
        |-- Id: int
        |-- Name: string
        |-- ShortName: string
|-- Points: Actor
    |-- ClusterName: string
    |-- Description: string
    |-- Email: string
    |-- GetCountOfClusterMembers: int
    |-- HasSites: bool
    |-- Id: int
    |-- Keywords: string
    |-- MemberOfCluster: bool
    |-- Name: string
    |-- ShortDescription: string
    |-- SpecifiedEntityType: string
    |-- Url: string
|-- SelectedSector: SectorType
    |-- Code: string
```

```

|--- Id: int
|--- Value: string
|-- SignInUser: DigicircUser
    |-- AccessFailedCount: int
    |-- Email: string
    |-- EmailConfirmed: bool
    |-- FirstName: string
    |-- LastName: string
    |-- LockoutEnabled: bool
    |-- LockoutEndDate: DateTime
    |-- Name: string
    |-- PasswordHash: string
    |-- PhoneNumber: string
    |-- PhoneNumberConfirmed: bool
    |-- SecurityStamp: string
    |-- SubscribeToNewsLetter: bool
    |-- TwoFactorEnabled: bool
    |-- UserName: string
|-- NewClusterName: string
|-- ManuallySetGeolocation: bool

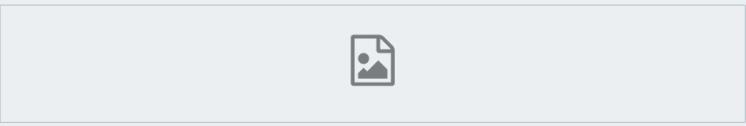
```

## View

% Title    Unsaved Changes!

[Back](#) [Delete](#) [Edit](#) [Save](#)

Logo



Please choose images with size 256x96

Description

Actor.Description

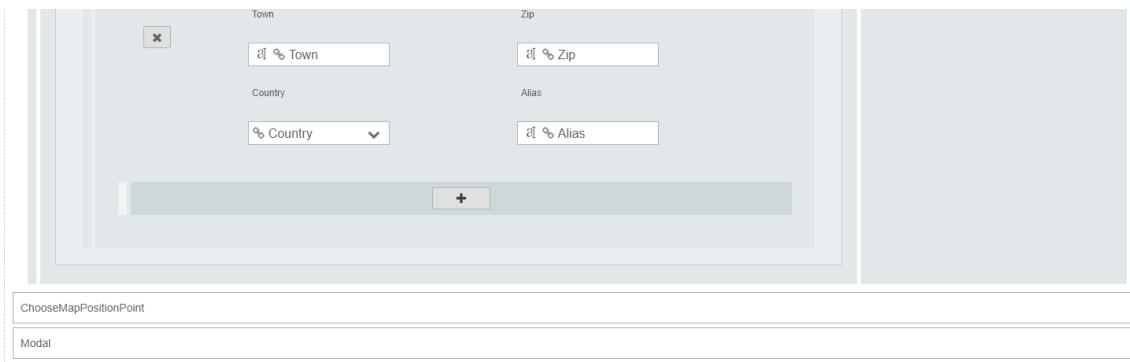
Description

Actor Description

Basic Information

Type	Name
<input style="width: 100%; height: 25px; border: none; background-color: #f0f0f0; padding: 2px;" type="button" value="Actor.EntityType"/> Please Specify Type	<input style="width: 100%; height: 25px; border: none; background-color: #f0f0f0; padding: 2px;" type="text" value="Actor Name"/>
<input style="width: 100%; height: 25px; border: none; background-color: #f0f0f0; padding: 2px;" type="button" value="Actor.SpecifiedEntityType"/>	
Member Of Cluster	<input style="width: 100%; height: 25px; border: none; background-color: #f0f0f0; padding: 2px;" type="button" value="Actor.Cluster"/> <a href="#">Add New</a>
Sector	Digital Expertise
<input style="width: 100%; height: 25px; border: none; background-color: #f0f0f0; padding: 2px;" type="button" value="SelectedSector"/>	<input style="width: 100%; height: 25px; border: none; background-color: #f0f0f0; padding: 2px;" type="button" value="Actor.CircularEconomyRequirements.DigitalExpertise"/>
Experience in Circular Economy?	Describe Experience
<input type="checkbox"/>	<input style="width: 100%; height: 25px; border: none; background-color: #f0f0f0; padding: 2px;" type="button" value="Actor.CircularEconomyRequirements.SpecifyExperienceInCircularEconomy"/>

What are you looking for?							
Request Digital provider?	<input type="checkbox"/> Request Thematic Expertise?						
<input type="checkbox"/>							
Desired Sector							
<input type="button" value="Actor.CircularEconomyRequirements.DesiredSMESector ▾"/>							
Available Testing Facilities							
<input type="checkbox"/>	<p style="background-color: #cccccc; padding: 5px;">Thematic Areas</p> <p style="font-size: small;">% Value</p> <p style="background-color: #cccccc; padding: 5px;">Thematic Areas</p> <p style="font-size: small;">% Value</p> <p style="text-align: center;"><input style="width: 20px; height: 20px; border: none; background-color: #cccccc; color: white; font-size: 10px; margin: 5px auto;" type="button" value="+"/></p>						
<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%; padding: 5px;">Expertise</td> <td style="width: 50%; padding: 5px;">Services</td> </tr> <tr> <td style="padding: 5px;">% Value</td> <td style="padding: 5px;">% Value</td> </tr> <tr> <td style="padding: 5px;"> <input style="width: 20px; height: 20px; border: none; background-color: #cccccc; color: white; font-size: 10px; margin: 5px auto;" type="button" value="+"/> </td> <td style="padding: 5px;"> <input style="width: 20px; height: 20px; border: none; background-color: #cccccc; color: white; font-size: 10px; margin: 5px auto;" type="button" value="+"/> </td> </tr> </table>		Expertise	Services	% Value	% Value	<input style="width: 20px; height: 20px; border: none; background-color: #cccccc; color: white; font-size: 10px; margin: 5px auto;" type="button" value="+"/>	<input style="width: 20px; height: 20px; border: none; background-color: #cccccc; color: white; font-size: 10px; margin: 5px auto;" type="button" value="+"/>
Expertise	Services						
% Value	% Value						
<input style="width: 20px; height: 20px; border: none; background-color: #cccccc; color: white; font-size: 10px; margin: 5px auto;" type="button" value="+"/>	<input style="width: 20px; height: 20px; border: none; background-color: #cccccc; color: white; font-size: 10px; margin: 5px auto;" type="button" value="+"/>						
Contact Details							
Email	Url						
<input type="button" value="Actor.Email"/>	<input type="button" value="Actor.Url"/> <input type="button" value="Actor.Url"/>						
Address							
Edit Geolocation							
Street Name	Number						
<input type="button" value="Actor.Address.StreetName"/>	<input type="button" value="Actor.Address.Number"/>						
Town	Zip						
<input type="button" value="Actor.Address.Town"/>	<input type="button" value="Actor.Address.Zip"/>						
Country	Has Sites						
<input type="button" value="Actor.Address.Country ▾"/>	<input type="checkbox"/>						
Sites							
<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%; padding: 5px;">Street Name</td> <td style="width: 50%; padding: 5px;">Number</td> </tr> <tr> <td style="padding: 5px;"><input type="button" value="StreetName"/></td> <td style="padding: 5px;"><input type="button" value="Number"/></td> </tr> </table>		Street Name	Number	<input type="button" value="StreetName"/>	<input type="button" value="Number"/>		
Street Name	Number						
<input type="button" value="StreetName"/>	<input type="button" value="Number"/>						



## Controller

### Add Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		AddActors

#### CODE

```
function void Add()
{
    Model.Title = LocalResources.RES_PAGETITLE_Add;
    Model.Actor = Domain.Actor.InitNewActor();
    Model.ManuallySetGeolocation = false;
}
```

### Show Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Show(int id)
{
    if(!string.IsNullOrEmpty(AppLib.Session.GetCurrentUserName()))
    {
        Model.SignInUser =
Domain.DigicircUser.GetByKey(AppLib.Session.GetCurrentUserName(), false);
    }

    Model.Actor = Domain.Actor.GetByKey(id);
    Model.Points.Add(Model.Actor);
    Model.Title = Model.Actor.Name;
```

```

    Model.SelectedSector = Model.Actor.SectorTypes.First();
}

```

### Edit Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		EditOrganisation

#### CODE

```

function void Edit(int id)
{
    Model.Actor = Domain.Actor.GetByKey(id);
    Model.Title = "Edit " + Model.Actor.Name;
    Model.ManuallySetGeolocation = false;

    var currentUserName = AppLib.Session.GetCurrentUserName();

    if(currentUserNames != Model.Actor.AddedBy.UserName &&
    Model.Actor.Administrators.Where(a => a.UserName == currentUserNames).Length == 0)
    {
        ShowMessage("You don't have permission to edit this actor.",
        AppLib.MessageType.Warning, FormModels.ActorForm.Controller.Show.GetLink(id));
        return;
    }

    Model.Points.Add(Model.Actor);

    Model.SelectedSector = Model.Actor.SectorTypes.First();
    AppLib.Session.Storage.Add("ActorOld", Model.Actor);
}

```

### SetNewGeolocation Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```

function void SetNewGeolocation()
{
    Model.ManuallySetGeolocation = true;

    FormControls.ChooseMapPositionPoint.Hide();
}

```

### UpdateGeolocation Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void UpdateGeolocation()
{
    Model.Points.Clear();
    Model.Points.Add(Model.Actor);

    FormControls.NewMap1.Refresh();
}
```

#### Save Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Save()
{
    Model.Actor.AddedBy = AppLib.Session.GetCurrentUser() as Domain.DigicircUser;

    if (Model.Actor.Address != null && !Model.ManuallySetGeolocation)
    {
        Model.Actor.Address.Alias = "Main";
        Model.Actor.Address = Domain.Geocoder.Query(Model.Actor.Address);
    }

    Model.Actor.Keywords = Domain.TextSearch.GetTags(Model.Actor.Description);

    // delete old relations if it's beign updated
    var oldInstance = AppLib.Session.Storage.Get("ActorOld") as Domain.Actor;
    // Domain.GraphUpdate.DeleteOldRelations(oldInstance, Model.Actor);
    // Domain.GraphUpdate.SendActorToGraph(Model.Actor);

    Model.Actor.Save();

    Domain.ActorBackend.CreateKnowledgeActor(Model.Actor);
    // if(Controller.Add.IsActive())
    // {
    //     Domain.ActorBackend.CreateKnowledgeActor(Model.Actor);
    // }
    // else
    // {
    //     Domain.ActorBackend.UpdateKnowledgeActor(Model.Actor);
    // }
}
```

```

//      }

try
{
    var response = Domain.ElasticDoc.SendActorDoc(Model.Actor);
    DebugLib.Logger.WriteLine("response elastic " + response);
}
catch Exception x
{
    DebugLib.Logger.WriteLine(x);
    ShowMessage("Something went wrong.", AppLib.MessageType.Error);
    return;
}
CloseForm();
}

```

### Delete Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```

function void Delete()
{
    Model.Actor.Delete();
    FormModels.SearchForm.Controller.Index.Execute();
}

```

### SetSector Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```

function void SetSector()
{
    Model.Actor.SectorTypes.Clear();
    Model.Actor.SectorTypes.Add(Model.SelectedSector);
}

```

### Back Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions

<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
-------------------------------------	-------------------------------------	-------------------------------------	--	--

#### CODE

```
function void Back()
{
    FormModels.SearchForm.Controller.FromBack.Execute(Model.Actor.Id);
}
```

#### SaveNewCluster Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void SaveNewCluster()
{
    Domain.Actor actor;
    actor.CircularEconomyRequirements = Domain.CircularEconomyReport.Create();
    actor.EntityType = Domain.EntityType.Find(a => a.IsCluster).First();
    actor.Name = Model.NewClusterName;
    try
    {
        actor.Save();
    }
    catch Exception x
    {
        DebugLib.Logger.WriteLine(x);
        ShowMessage("Something went wrong. " + x.Message, AppLib.MessageType.Error);
        return;
    }

    Model.Actor.Cluster = actor;
    ShowMessage("Successfully Saved");
    Model.NewClusterName = string.Empty;
    FormControls.dropdownBox3.Refresh();
    FormControls.modal.Hide();
    return;
}
```

#### AddNewCluster Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

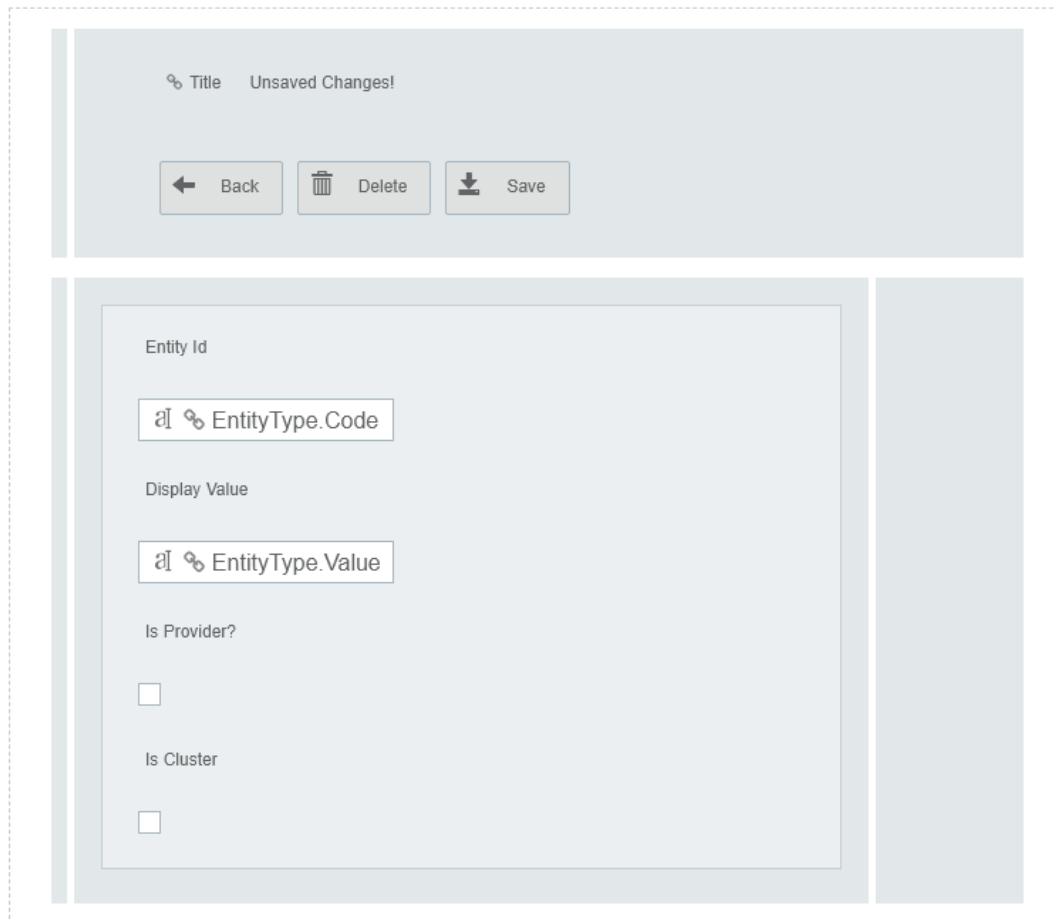
```
function void AddNewCluster()
{
    FormControls.Modal.Show();
}
```

## EntityTypeForm Form

### Model

```
|-- EntityType: EntityType
|-- Code: string
|-- Id: int
|-- IsCluster: bool
|-- IsProvider: bool
|-- Value: string
```

### View



### Controller

### Add Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Add() {
    Model.Title = LocalResources.RES_PAGETITLE_Add;
}
```

### Edit Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Edit(int id) {
    Model.EntityType = Domain.EntityType.GetByKey(id);
    Model.Title = Model.EntityType.Value;
}
```

### Save Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Save() {
    Model.EntityType.Save();
    CloseForm();
}
```

### Delete Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```

function void Delete() {
    Model.EntityType.Delete();
    CloseForm();
}

```

## CountryForm Form

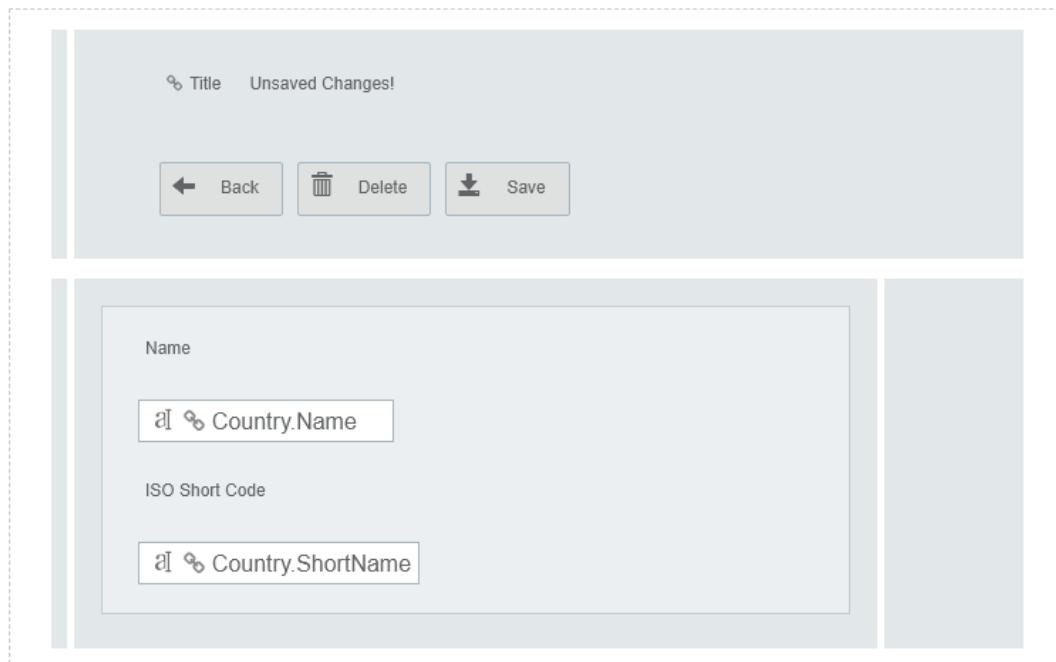
### Model

```

|-- Country: Country
|-- Id: int
|-- Name: string
|-- ShortName: string

```

### View



### Controller

#### Add Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```

function void Add() {
    Model.Title = LocalResources.RES_PAGETITLE_Add;
}

```

### Edit Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```

function void Edit(int id) {
    Model.Country = Domain.Country.GetByKey(id);
    Model.Title = LocalResources.RES_PAGETITLE_Edit;
}

```

### Save Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```

function void Save() {
    Model.Country.Save();
    CloseForm();
}

```

### Delete Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```

function void Delete() {
    Model.Country.Delete();
    CloseForm();
}

```

## CountryList Form

### Model

Model properties:

Model methods:

## View

	Name	ISO Short Code
1	abc	abc
2	abc	abc
3	...	...

## Controller

### Index Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		EditValueLists

### CODE

```
function void Index() { Model.Title = LocalResources.RES_PAGETITLE_Index; }
```

## EntityTypeList Form

### Model

## View

List of Entities				
	Entity Id	Display Value	Is Provider?	Is Cluster?
1	abc	abc	abc	abc
2	abc	abc	abc	abc
3	...	...	...	...

## Controller

### Index Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		EditValueLists

### CODE

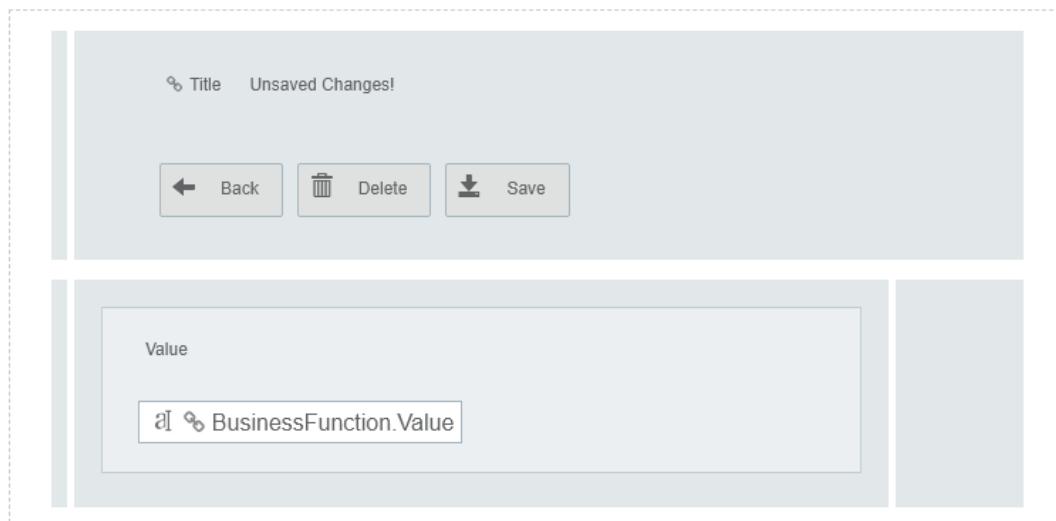
```
function void Index() { Model.Title = LocalResources.RES_PAGETITLE_Index; }
```

## BusinessFunctionForm Form

### Model

```
|-- BusinessFunction: BusinessFunction
|-- Id: int
|-- Value: string
```

### View



## Controller

### Add Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Add() {
    Model.Title = LocalResources.RES_PAGETITLE_Add;
}
```

### Edit Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Edit(int id) {
    Model.BusinessFunction = Domain.BusinessFunction.GetByKey(id);
    Model.Title = LocalResources.RES_PAGETITLE_Edit;
}
```

### Save Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

**SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

**CODE**

```
function void Save() {
    Model.BusinessFunction.Save();
    CloseForm();
}
```

**Delete Controller Action**

Is Entrypoint:  Enabled Access Log:  Causes Validation:

**SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

**CODE**

```
function void Delete() {
    Model.BusinessFunction.Delete();
    CloseForm();
}
```

**BusinessFunctionList Form****Model****View**

## Controller

### Index Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

## SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		EditValueLists

## CODE

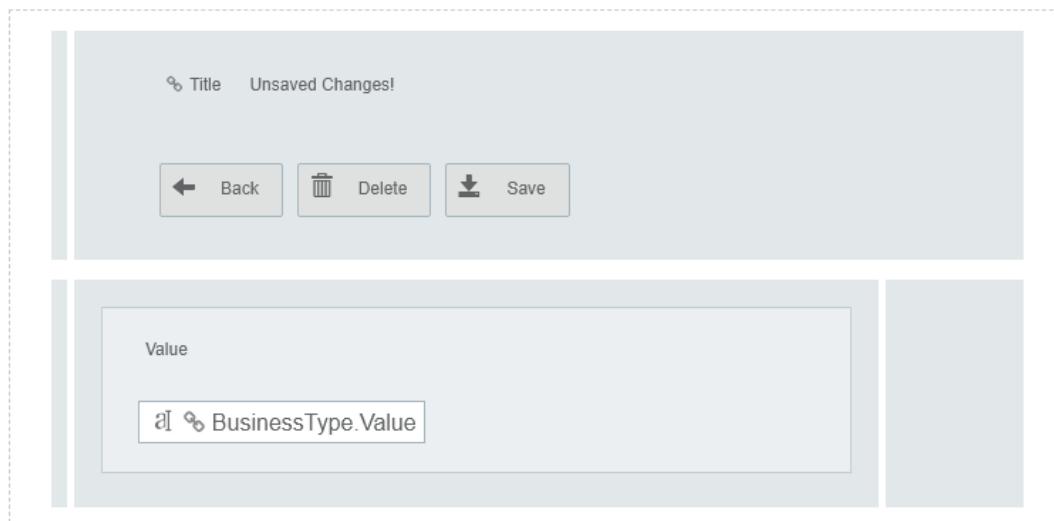
```
function void Index() { Model.Title = LocalResources.RES_PAGETITLE_Index; }
```

## BusinessTypeForm Form

### Model

```
|-- BusinessType: BusinessType
|-- Id: int
|-- Value: string
```

### View



## Controller

### Add Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Add() {
    Model.Title = LocalResources.RES_PAGETITLE_Add;
}
```

### Edit Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Edit(int id) {
    Model.BusinessType = Domain.BusinessType.GetByKey(id);
    Model.Title = LocalResources.RES_PAGETITLE_Edit;
}
```

### Save Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

**SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

**CODE**

```
function void Save() {
    Model.BusinessType.Save();
    CloseForm();
}
```

**Delete Controller Action**

Is Entrypoint:  Enabled Access Log:  Causes Validation:

**SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

**CODE**

```
function void Delete() {
    Model.BusinessType.Delete();
    CloseForm();
}
```

**BusinessTypeList Form****Model****View**

	Display Value	
1	abc	
2	abc	
3	...	

## Controller

### Index Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

## SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		EditValueLists

## CODE

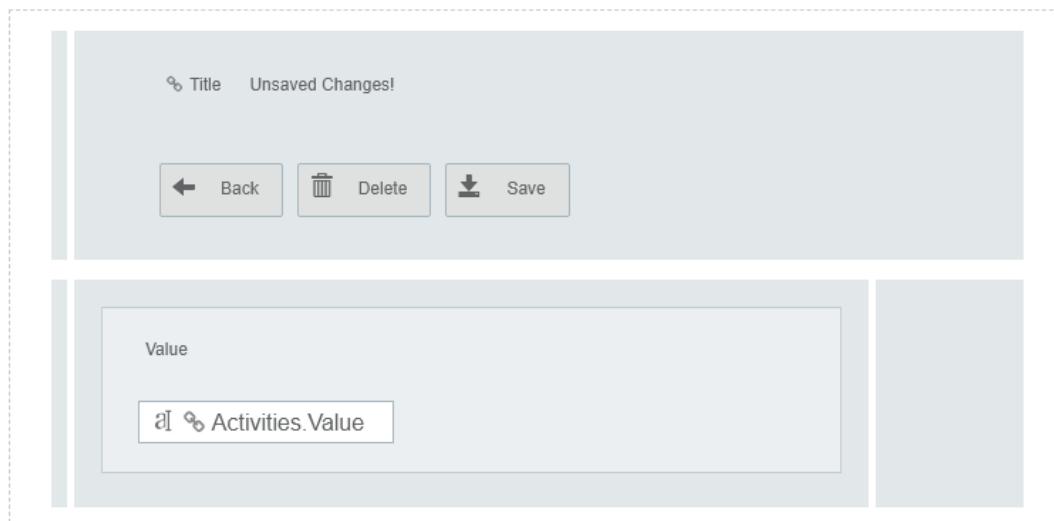
```
function void Index() { Model.Title = LocalResources.RES_PAGETITLE_Index; }
```

## ActivitiesForm Form

### Model

```
|-- Activities: Activities
|-- Id: int
|-- Value: string
```

### View



## Controller

### Add Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Add() {
    Model.Title = LocalResources.RES_PAGETITLE_Add;
}
```

### Edit Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Edit(int id) {
    Model.Activities = Domain.Activities.GetByKey(id);
    Model.Title = LocalResources.RES_PAGETITLE_Edit;
}
```

### Save Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

**SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

**CODE**

```
function void Save() {
    Model.Activities.Save();
    CloseForm();
}
```

**Delete Controller Action**

Is Entrypoint:  Enabled Access Log:  Causes Validation:

**SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

**CODE**

```
function void Delete() {
    Model.Activities.Delete();
    CloseForm();
}
```

**ActivitiesList Form****Model****View**

## Controller

### Index Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

## SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		EditValueLists

## CODE

```
function void Index() { Model.Title = LocalResources.RES_PAGETITLE_Index; }
```

## Dashboard Form

### Model

### View

## Controller

### **Index Controller Action**

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### **SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		User

#### **CODE**

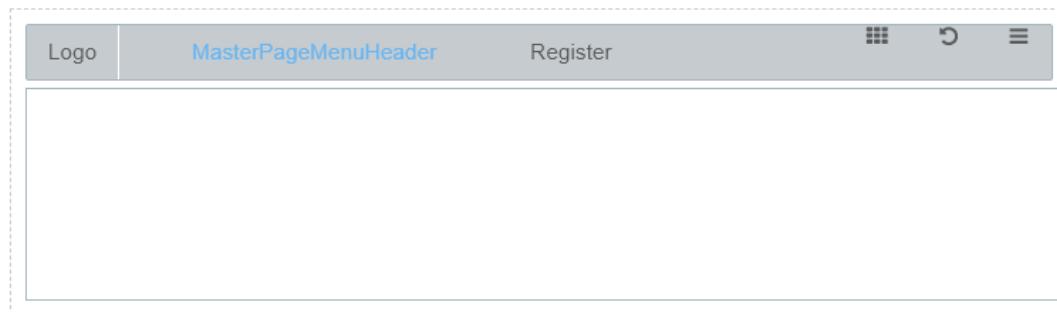
```
function void Index() {
    Model.Title = LocalResources.RES_PAGETITLE_Index;

}
```

### **EmptyMasterPage Form**

#### **Model**

#### **View**



#### **Controller**

### **Index Controller Action**

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### **SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### **CODE**

```
function void Index() {

}
```

## GraphQueryDebug Form

### Model

```
|-- Info: GraphDebugResult
|-- Query: string
|-- RawResult: string
|-- Result: GraphBackendResponse
|-- Metadata: Metadata
|-- ExportDataAsJson: bool
|-- Pages: int
|-- PageSize: int
|-- QueryElapsedTime: int
|-- TotalResponseElementsWithPositiveRelevanceLevel: int
|-- Elements: int
|-- Relations: int
|-- ExportType: string
|-- Nodes: Nodes
|-- Name: string
|-- Label: string
|-- LabelType: string
|-- Id: int
|-- Graphid: int
|-- CC: bool
|-- SL: int
|-- CL: float
|-- RL: float
|-- IA: bool
|-- AL: int
|-- AC: float
|-- Attr: bool
|-- Links: Links
|-- Source: int
|-- Target: int
|-- Type: string
|-- TypeRel: string
|-- Sid: int
|-- Tid: int
|-- Weight: float
|-- CL: float
|-- RL: float
|-- IA: bool
|-- AL: int
|-- AC: float
|-- Attr: bool
|-- CountryName: string
|-- IsExtended: bool
```

### View

Get Graph Info															
Search Text		Nodes													
Extended		Links													
Country		Results													
% CountryName	<input type="button" value="Search"/>	<input type="button" value="Extend Search"/>	Page 1 of 10 pages												
			Name	Label Type	Label	CC	IA	Attr	ID	GraphId	SL	AL	CL	RL	AC
1	abc	abc	abc	abc	abc	abc	abc	abc	abc	abc	abc	abc	abc	abc	abc
2	abc	abc	abc	abc	abc	abc	abc	abc	abc	abc	abc	abc	abc	abc	abc
3	abc	abc	abc	abc	abc	abc	abc	abc	abc	abc	abc	abc	abc	abc	abc

## Controller

### Index Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Developer

#### CODE

```
function void Index() {
    Model.Title = LocalResources.RES_PAGETITLE_Index;

    Model.Info.Query = "Germany";
    Model.Info.Result = Domain.GraphBackendResponse.Create();
    Model.Info.RawResult = "Click search...";

    FormModels.GraphQueryDebug.Controller.Search.Execute();
}
```

### Search Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Developer

#### CODE

```
function void Search()
{
    var response = Domain.GraphQueries.Query(Model.Info.Query);

    if (response != null) {
        Model.Info.Result = response;
    }

    Model.Info.RawResult = Domain.GraphQueries.RawQuery(Model.Info.Query);

    FormControls.ListLinks.Refresh();
    FormControls.ListNodes.Refresh();
}
```

### SearchExtend Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Developer

#### CODE

```
function void SearchExtend()
{
    Collection[Domain.ExElements] q;

    var countryClause = Domain.ExElements.Create();
    countryClause.Type = "Country";
    countryClause.Name = Model.CountryName;

    var clause = Domain.ExElements.Create();
    clause.Type = "*";
    clause.Name = Model.Info.Query;

    q.Add(countryClause);
    q.Add(clause);

    var response = Domain.GraphQueries.ExtenedQuery(q);

    if (response != null) {
        Model.Info.Result = response;
    }

    Model.Info.RawResult = Domain.GraphQueries.RawExtenedQuery(q);

    FormControls.ListLinks.Refresh();
    FormControls.ListNodes.Refresh();
}
```

### Bubble Form

#### Model

```
|-- Actor: Actor
|-- Description: string
|-- Email: string
|-- Id: int
|-- Name: string
|-- ShortDescription: string
|-- Url: string
```

#### View

**X**

Name  
Actor.Name

Description  
Actor.ShortDescription

**Show more**

## Controller

### RedirectToActorForm Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void RedirectToActorForm()
{
    // FormModels.ActorForm.Controller.Show.Execute(Model.Actor.Id);
    FormModels.ActorViewForm.Controller.Show.Execute(Model.Actor.Id, false);
}
```

## Close Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Close()
{
    ExecuteJavascript("$(\"[jb-type='PartialView'][jb-partial-
name='Bubble']\").hide()");
}
```

## GraphCreateDebug Form

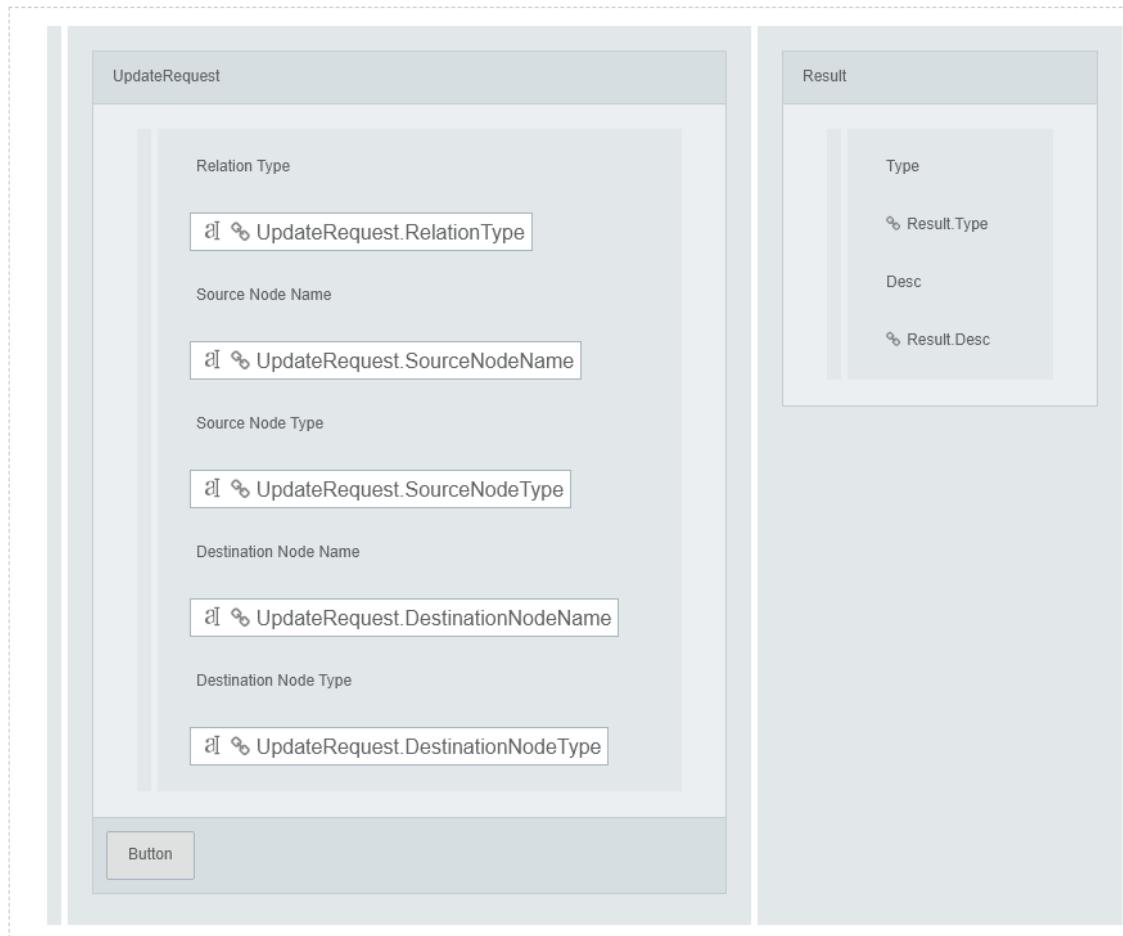
### Model

```

|-- UpdateRequest: GraphUpdateElement
|  -- RelationType: string
|  -- SourceNodeName: string
|  -- SourceNodeType: string
|  -- DestinationNodeName: string
|  -- DestinationNodeType: string
|-- Result: UpdateResponse
|  -- Type: string
|  -- Desc: string

```

## View



## Controller

### Index Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions

<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		Developer
--------------------------	--------------------------	-------------------------------------	--	-----------

#### CODE

```
function void Index() {
    Model.Title = LocalResources.RES_PAGETITLE_Index;

    Model.UpdateRequest.RelationType = "hasCompany";
    Model.UpdateRequest.SourceNodeType = "Country";
    Model.UpdateRequest.SourceNodeName = "Greece";
    Model.UpdateRequest.DestinationNodeType = "Company";
    Model.UpdateRequest.DestinationNodeName = "CLMS UK";
}
```

#### Save Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

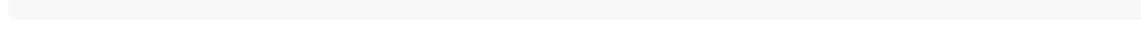
Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		Developer

#### CODE

```
function void Save()
{
    Model.Result = Domain.GraphUpdate.AddNewRelation(Model.UpdateRequest);
}
```

#### GraphExportForm Form

##### Model



##### View



#### Controller

#### Index Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Developer

#### CODE

```
function void Index() {
    Model.Title = LocalResources.RES_PAGETITLE_Index;
}
```

#### IntiGraph Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Developer

#### CODE

```
function void IntiGraph()
{
    Domain.GraphUpdate.InitGraphFromDB();
}
```

#### InitElastic Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Developer

#### CODE

```
function void InitElastic()
{
    Domain.ElasticConsumer.InitElasticFromDb();
}
```

#### ExpertiseForm Form

##### Model

```
| -- Expertise: Expertise
| -- Id: int
| -- Code: string
| -- Value: string
```

##### View

>Title    Unsaved Changes!

[Back](#) [Delete](#) [Save](#)

Expertise Id	
<a href="#">Expertise.Code</a>	
Display Value	
<a href="#">Expertise.Value</a>	

## Controller

### Add Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Add() {
    Model.Title = LocalResources.RES_PAGETITLE_Add;
}
```

### Edit Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Edit(int id) {
    Model.Expertise = Domain.Expertise.GetByKey(id);
```

```
    Model.Title = LocalResources.RES_PAGETITLE_Edit;  
}
```

#### Save Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Save() {  
    Model.Expertise.Save();  
    CloseForm();  
}
```

#### Delete Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Delete() {  
    Model.Expertise.Delete();  
    CloseForm();  
}
```

### ExpertiseList Form

#### Model

#### View

<span style="font-size: 2em;">%</span> Title		
Add	Edit	
Page 1 of 10 pages		
	25 per page	
	Expertise Id	Display Value
1	abc	abc
2	abc	abc
3	...	...

## Controller

### Index Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

## SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		EditValueLists

## CODE

```
function void Index() { Model.Title = LocalResources.RES_PAGETITLE_Index; }
```

## SectorTypeForm Form

### Model

```
|-- SectorType: SectorType
|-- Id: int
|-- Code: string
|-- Value: string
```

## View

>Title    Unsaved Changes!

Back     Delete     Save

Sector Id

SectorType.Code

Display Value

SectorType.Value

## Controller

### Add Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		EditValueLists

#### CODE

```
function void Add() {
    Model.Title = LocalResources.RES_PAGETITLE_Add;
}
```

### Edit Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		EditValueLists

#### CODE

```
function void Edit(int id) {
    Model.SectorType = Domain.SectorType.GetByKey(id);
```

```
    Model.Title = LocalResources.RES_PAGETITLE_Edit;  
}
```

#### Save Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		EditValueLists

#### CODE

```
function void Save() {  
    Model.SectorType.Save();  
    FormModels.SectorTypeList.Controller.Index.Execute();  
}
```

#### Delete Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		EditValueLists

#### CODE

```
function void Delete() {  
    Model.SectorType.Delete();  
    CloseForm();  
}
```

### SectorTypeList Form

#### Model

#### View

	Sector Id	Display Value
1	abc	abc
2	abc	abc
3	...	...

## Controller

### Index Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		EditValueLists

#### CODE

```
function void Index() { Model.Title = LocalResources.RES_PAGETITLE_Index; }
```

### Delete Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		EditValueLists

#### CODE

```
function void Delete(int id)
{
    Domain.SectorType sector = Domain.SectorType.GetByKey(id, false);
    if(sector != null)
    {
        sector.Delete();
    }
    FormControls.SectorTypeList.Refresh();
}
```

### CleanDuplicates Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Developer

#### CODE

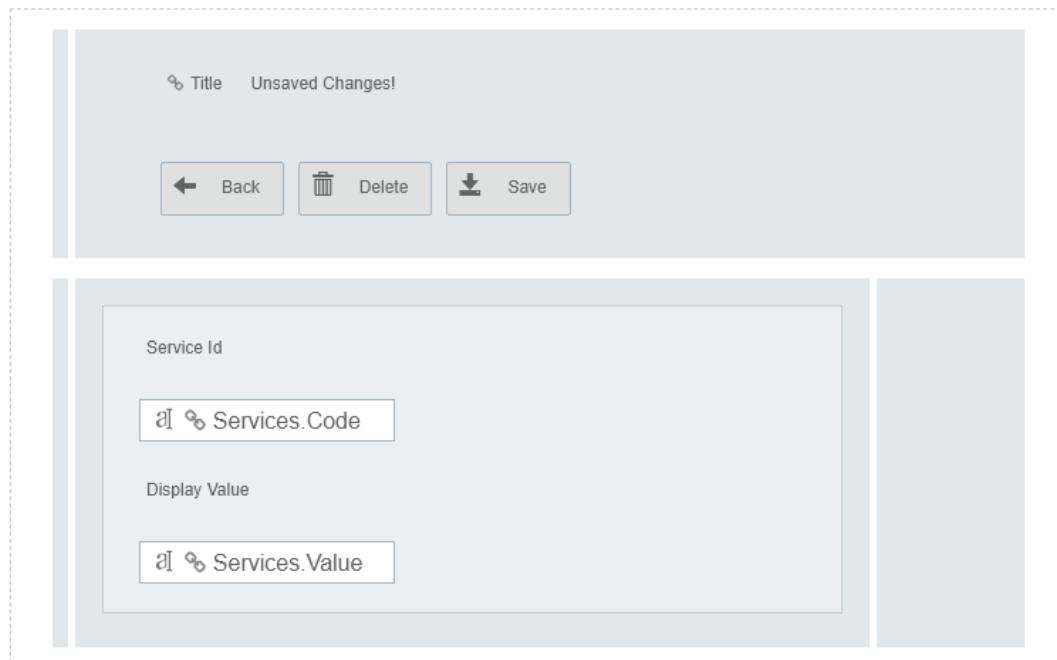
```
function void CleanDuplicates(Domain.SectorType sector)
{
    Domain.DataHelper.CleanDuplicateSectors(sector);
    FormControls.SectorTypeList.Refresh();
    ShowMessage("Done");
}
```

### ServicesForm Form

#### Model

```
|-- Services: Services
|-- Id: int
|-- Code: string
|-- Value: string
```

#### View



## Controller

### Add Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Add() {
    Model.Title = LocalResources.RES_PAGETITLE_Add;
}
```

### Edit Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Edit(int id) {
    Model.Services = Domain.Services.GetByKey(id);
    Model.Title = LocalResources.RES_PAGETITLE_Edit;
}
```

### Save Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Save() {
    Model.Services.Save();
    CloseForm();
}
```

### Delete Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```

function void Delete() {
    Model.Services.Delete();
    CloseForm();
}

```

## ServicesList Form

### Model

### View

		Title		
		Add	Edit	Upload Search Refresh Undo Redo Info *
		Page 1 of 10 pages < < > >>		25 per page
		Service Id		Display Value
1		abc		abc
2		abc		abc
3		...		...

### Controller

#### Index Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		EditValueLists

### CODE

```

function void Index() { Model.Title = LocalResources.RES_PAGETITLE_Index; }

```

## ThematicExpertiseForm Form

### Model

```

|-- ThematicExpertise: ThematicExpertise
|-- Id: int

```

```
|-- Code: string  
|-- Value: string
```

## View

The screenshot shows a user interface for managing a Thematic Expertise record. At the top, there's a header with a title and a message about unsaved changes. Below the header are three buttons: 'Back', 'Delete', and 'Save'. The main content area contains two input fields. The first field is labeled 'Thematic Expertise Id' and contains the placeholder text 'aI % ThematicExpertise.Code'. The second field is labeled 'Display Value' and also contains the placeholder text 'aI % ThematicExpertise.Value'.

## Controller

### Add Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Add() {  
    Model.Title = LocalResources.RES_PAGETITLE_Add;  
}
```

### Edit Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

**CODE**

```
function void Edit(int id) {  
    Model.ThematicExpertise = Domain.ThematicExpertise.GetByKey(id);  
    Model.Title = LocalResources.RES_PAGETITLE_Edit;  
}
```

**Save Controller Action**Is Entrypoint:  Enabled Access Log:  Causes Validation: **SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

**CODE**

```
function void Save() {  
    Model.ThematicExpertise.Save();  
    CloseForm();  
}
```

**Delete Controller Action**Is Entrypoint:  Enabled Access Log:  Causes Validation: **SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

**CODE**

```
function void Delete() {  
    Model.ThematicExpertise.Delete();  
    CloseForm();  
}
```

**ThematicExpertiseList Form****Model****View**

	Thematic Expertise Id	Display Value
1	abc	abc
2	abc	abc
3	...	...

## Controller

### Index Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

## SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		EditValueLists

## CODE

```
function void Index() { Model.Title = LocalResources.RES_PAGETITLE_Index; }
```

## CompanyList Form

### Model

```
|-- SelectedEntityType: EntityType
|-- Id: int
|-- Code: string
|-- Value: string
|-- IsProvider: bool
```

### View

The screenshot shows a user interface for managing entities. At the top, there are fields for 'Title' and 'Entity Type', with a dropdown menu labeled 'SelectedEntityType'. Below these are buttons for 'Delete' and 'Transform'. A navigation bar indicates 'Page 1 of 10 pages' and '25 per page'. The main area displays a grid of company data with columns: company\_name, url, city, country, zip\_code, company\_category, and description. The data in the grid consists of three rows with values 'abc' repeated across all columns.

## Controller

### Index Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		Developer

#### CODE

```
function void Index() { Model.Title = LocalResources.RES_PAGETITLE_Index; }
```

### Transform Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		Developer

#### CODE

```
function void Transform()
{
    Domain.Company.TransformToActor(Model.SelectedEntityType);
}
```

### Delete Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		Developer

#### CODE

```

function void Delete(Collection[Domain.Company] companies)
{
    foreach Domain.Company company in companies
    {
        company.Delete();
    }
    FormControls.CompanyList.Refresh();
}

```

## ActorViewForm Form

### Model

```

|-- Actor: Actor
|   |-- ClusterName: string
|   |-- Description: string
|   |-- Email: string
|   |-- GetCountOfClusterMembers: int
|   |-- HasSites: bool
|   |-- Id: int
|   |-- Keywords: string
|   |-- MemberOfCluster: bool
|   |-- Name: string
|   |-- ShortDescription: string
|   |-- SpecifiedEntityType: string
|   |-- Url: string
|   |-- Address: Address
|       |-- Alias: string
|       |-- FullAddress: string
|       |-- Id: int
|       |-- Latitude: double
|       |-- Longitude: double
|       |-- Number: int
|       |-- StreetName: string
|       |-- Town: string
|       |-- Zip: string
|       |-- Country: Country
|           |-- Id: int
|           |-- Name: string
|           |-- ShortName: string
|-- EntityType: EntityType
|   |-- Code: string
|   |-- Id: int
|   |-- IsCluster: bool
|   |-- IsProvider: bool
|   |-- Value: string
|-- CircularEconomyRequirements: CircularEconomyReport
|   |-- DigitalExpertise: DigitalExpertise
|   |-- DigitalProviredNeeded: bool
|   |-- ExperienceInCircularEconomy: bool
|   |-- GetExperienceInCircularEconomy: string
|   |-- Id: int

```

```
|-- SpecifyExperienceInCircularEconomy: string
|-- ThematicExpertiseNeeded: bool
|-- DesiredThematicExpertises: ThematicExpertise
    |-- Code: string
    |-- Id: int
    |-- Value: string
|-- DesiredSMESector: SectorType
    |-- Code: string
    |-- Id: int
    |-- Value: string
|-- DesiredGeographicalArea: GeographicalArea
    |-- Id: int
|-- SectorTypes: SectorType
    |-- Code: string
    |-- Id: int
    |-- Value: string
|-- CircularEconomyProviderReport: CircularEconomyProviderReport
    |-- AvailableTestingFacilities: bool
    |-- Id: int
    |-- PlaceOperates: GeographicalArea
        |-- Id: int
    |-- Expertises: Expertise
        |-- Code: string
        |-- Id: int
        |-- Value: string
    |-- ServicesProvided: Services
        |-- Code: string
        |-- Id: int
        |-- Value: string
    |-- ThematicExpertises: ThematicExpertise
        |-- Code: string
        |-- Id: int
        |-- Value: string
|-- ActorLogo: FileData
    |-- AllowedExtensions: string
    |-- Blob: Collection[byte]
    |-- FileName: string
    |-- FolderPath: string
    |-- Id: Guid
    |-- MaxFileSize: int
    |-- StorageMedium: StorageMedium
    |-- UploadDateTime: DateTime
    |-- UploadedBy: string
|-- AddedBy: DigicircUser
    |-- AccessFailedCount: int
    |-- Email: string
    |-- EmailConfirmed: bool
    |-- FirstName: string
    |-- LastName: string
    |-- LockoutEnabled: bool
    |-- LockoutEndDate: DateTime
    |-- Name: string
```

```
|-- PasswordHash: string
|-- PhoneNumber: string
|-- PhoneNumberConfirmed: bool
|-- SecurityStamp: string
|-- SubscribeToNewsLetter: bool
|-- TwoFactorEnabled: bool
|-- UserName: string
|-- Cluster: Actor
|--- ClusterName: string
|--- Description: string
|--- Email: string
|--- GetCountOfClusterMembers: int
|--- HasSites: bool
|--- Id: int
|--- Keywords: string
|--- MemberOfCluster: bool
|--- Name: string
|--- ShortDescription: string
|--- SpecifiedEntityType: string
|--- Url: string
|-- Administrators: DigicircUser
|--- AccessFailedCount: int
|--- Email: string
|--- EmailConfirmed: bool
|--- FirstName: string
|--- LastName: string
|--- LockoutEnabled: bool
|--- LockoutEndDate: DateTime
|--- Name: string
|--- PasswordHash: string
|--- PhoneNumber: string
|--- PhoneNumberConfirmed: bool
|--- SecurityStamp: string
|--- SubscribeToNewsLetter: bool
|--- TwoFactorEnabled: bool
|--- UserName: string
|-- Points: Actor
|--- ClusterName: string
|--- Description: string
|--- Email: string
|--- GetCountOfClusterMembers: int
|--- HasSites: bool
|--- Id: int
|--- Keywords: string
|--- MemberOfCluster: bool
|--- Name: string
|--- ShortDescription: string
|--- SpecifiedEntityType: string
|--- Url: string
|-- SelectedSector: SectorType
|--- Code: string
|--- Id: int
```

```

| -- Value: string
|-- SignInUser: DigicircUser
| -- AccessFailedCount: int
| -- Email: string
| -- EmailConfirmed: bool
| -- FirstName: string
| -- LastName: string
| -- LockoutEnabled: bool
| -- LockoutEndDate: DateTime
| -- Name: string
| -- PasswordHash: string
| -- PhoneNumber: string
| -- PhoneNumberConfirmed: bool
| -- SecurityStamp: string
| -- SubscribeToNewsLetter: bool
| -- TwoFactorEnabled: bool
| -- UserName: string
|-- FromGraph: bool

```

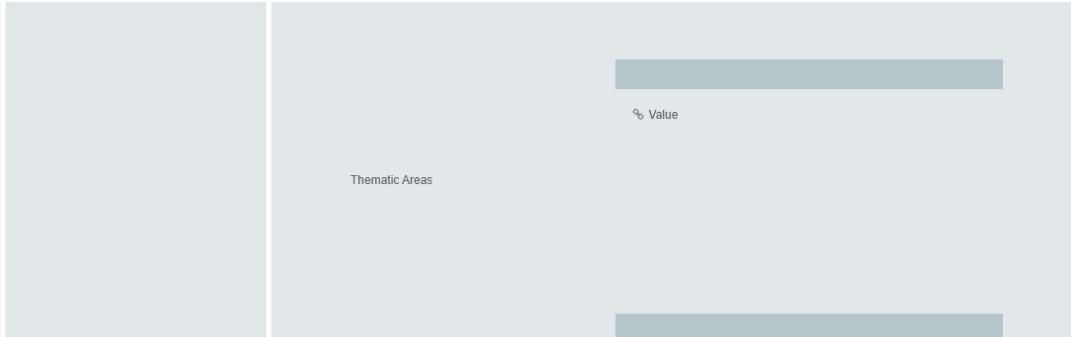
## View

Back
 Delete
 Resources
 Edit

 <span style="border: 1px solid #ccc; padding: 2px;">% Title</span> <span style="border: 1px solid #ccc; padding: 2px;">% Actor.EntityType.Value</span> <span style="border: 1px solid #ccc; padding: 2px;">Visit Website</span>	<div style="margin-bottom: 10px;"> <span>Please Specify Type</span> <span style="float: right;">% Actor.SpecifiedEntityType</span> </div> <div style="display: flex; justify-content: space-between;"> <div style="width: 45%;"> <span>Member Of Cluster</span> </div> <div style="width: 45%;"> <span>% Actor.Cluster.Name % Actor.ClusterName</span> </div> </div> <div style="display: flex; justify-content: space-between;"> <div style="width: 45%;"> <span>Sector</span> </div> <div style="width: 45%;"> <span>% SelectedSector.Value</span> </div> </div> <div style="display: flex; justify-content: space-between;"> <div style="width: 45%;"> <span>Digital Expertise</span> </div> <div style="width: 45%;"> <span>% Actor.CircularEconomyRequirements.DigitalExpertise</span> </div> </div> <div style="display: flex; justify-content: space-between;"> <div style="width: 45%;"> <span>Describe Experience</span> </div> <div style="width: 45%;"> <span>% Actor.CircularEconomyRequirements.SpecifyExperienceInCircularEconomy</span> </div> </div> <div style="display: flex; justify-content: space-between;"> <div style="width: 45%;"> <span>Experience in Circular Economy?</span> </div> <div style="width: 45%;"> <span>% Actor.CircularEconomyRequirements.GetExperienceInCircularEconomy</span> </div> </div> <div style="display: flex; justify-content: space-between;"> <div style="width: 45%;"> <span>Available Testing Facilities</span> </div> <div style="width: 45%;"> <span>% Actor.CircularEconomyProviderReport.AvailableTestingFacilities</span> </div> </div> <div style="text-align: center; margin-top: 20px;"> <span>% Value</span> </div> <div style="text-align: center; margin-top: 10px;"> <span>Expertise</span> </div>
---	---

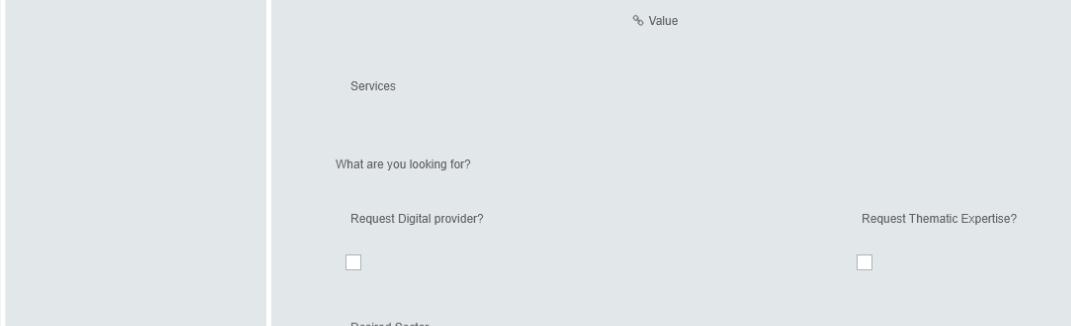
% Value

Thematic Areas



% Value

Services



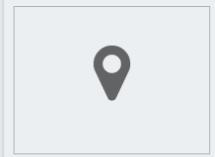
What are you looking for?

Request Digital provider?
Request Thematic Expertise?

Desired Sector

About us

Contact us



## Controller

### GoToWebsite Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void GoToWebsite()
{
```

```

    WebLib.Request.RedirectToUrl(Model.Actor.Url, "_blank");
}

```

### Show Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```

function void Show(int id, bool fromGraph)
{
    if(!string.IsNullOrEmpty(AppLib.Session.GetCurrentUserName()))
    {
        Model.SignInUser =
Domain.DigicircUser.GetByKey(AppLib.Session.GetCurrentUserName(), false);
    }
    Model.FromGraph = fromGraph;

    Model.Actor = Domain.Actor.GetByKey(id);
    Model.Points.Add(Model.Actor);
    Model.Title = Model.Actor.Name;

    Model.SelectedSector = Model.Actor.SectorTypes.First();
}

```

### Save Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```

function void Save()
{
    Model.Actor.AddedBy = AppLib.Session.GetCurrentUser() as Domain.DigicircUser;

    Model.Actor.Address = Domain.Geocoder.Query(Model.Actor.Address);

    //delete old relations if it's beign updated
    var oldInstance = AppLib.Session.Storage.Get("ActorOld") as Domain.Actor;
    Domain.GraphUpdate.DeleteOldRelations(oldInstance, Model.Actor);

    Domain.GraphUpdate.SendActorToGraph(Model.Actor);
    Model.Actor.Save();
    CloseForm();
}

```

### Delete Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Delete()
{
    Model.Actor.Delete();
    CloseForm();
}
```

### SetSector Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void SetSector()
{
    Model.Actor.SectorTypes.Clear();
    Model.Actor.SectorTypes.Add(Model.SelectedSector);
}
```

### Back Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Back()
{
    if(Model.FromGraph)
    {
        FormModels.MatchBaseExplorer.Controller.FromBack.Execute(Model.Actor.Id);
    }
    else
    {
        FormModels.SearchForm.Controller.FromBack.Execute(Model.Actor.Id);
    }
}
```

## ClusterList Form

### Model

### View

%		Title	Unsaved Changes!
Page 1 of 10 pages		◀◀ ▶▶	25 per page
	Cluster Name	Members	
1	abc	abc	
2	abc	abc	
3	...	...	

### Controller

#### Index Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ManageUsers, ManageRoles, ManagePermissions, ManageOperations

#### CODE

```
function void Index() { Model.Title = LocalResources.RES_PAGETITLE_Index; }
```

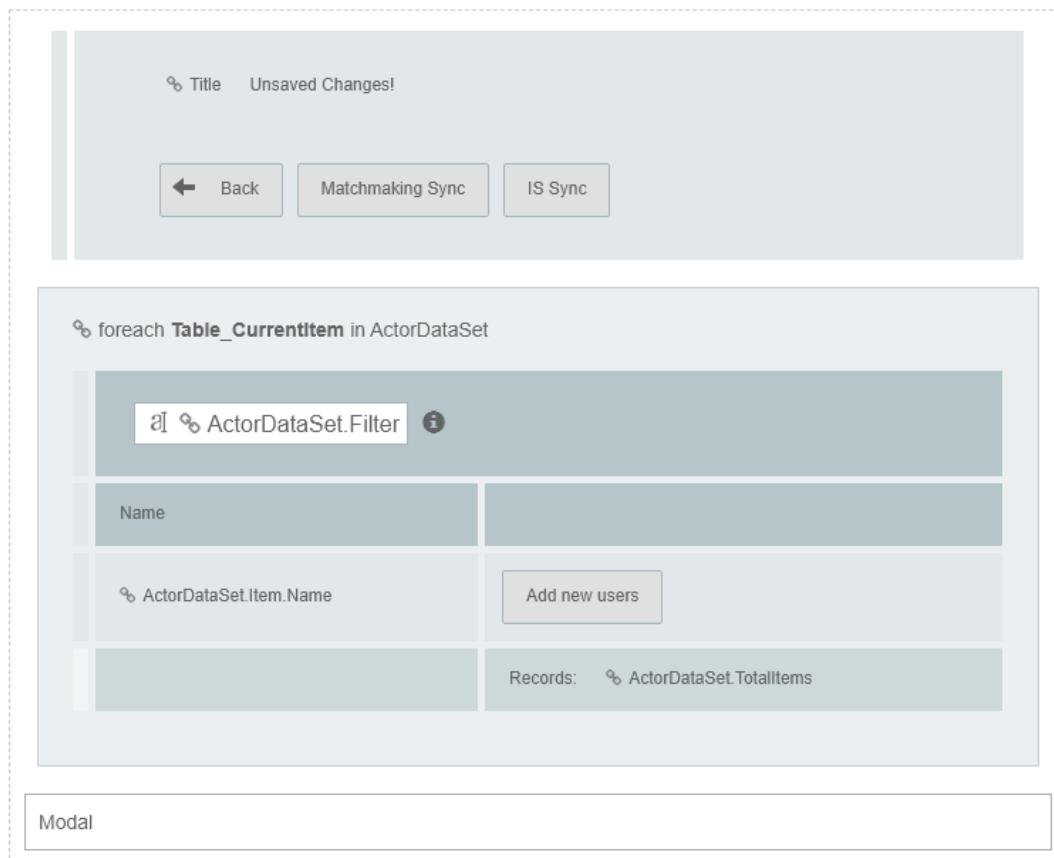
## ManageActors Form

### Model

```
|-- SelectedActor: Actor
|-- Id: int
|-- Name: string
|-- Description: string
|-- ShortDescription: string
|-- Url: string
|-- Email: string
```

```
|-- SpecifiedEntityType: string
|-- MemberOfCluster: bool
|-- GetCountOfClusterMembers: int
|-- ClusterName: string
|-- Administrators: DigicircUser
|--- UserName: string
|--- PasswordHash: string
|--- SecurityStamp: string
|--- EmailConfirmed: bool
|--- LockoutEnabled: bool
|--- PhoneNumberConfirmed: bool
|--- TwoFactorEnabled: bool
|--- AccessFailedCount: int
|--- Name: string
|--- Email: string
|--- PhoneNumber: string
|--- LockoutEndDate: DateTime
|--- FirstName: string
|--- LastName: string
|--- SubscribeToNewsLetter: bool
```

**View**



## Controller

### Index Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ManageActors

#### CODE

```
function void Index() {
    Model.Title = LocalResources.RES_PAGETITLE_Index;

}
```

### EsSync Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions



Developer

**CODE**

```
function void EsSync()
{
    var actors = Domain.Actor.GetAll();

    try
    {

        foreach (Domain.Actor actor in actors)
        {
            var response = Domain.ElasticDoc.SendActorDoc(actor);
            DebugLib.Logger.WriteLine("response elastic " + response);
        }

        ShowMessage("Successfully synchronized actors");
    }
    catch Exception x
    {
        DebugLib.Logger.WriteLine(x);
        ShowMessage("Something went wrong.", AppLib.MessageType.Error);
        return;
    }
}
```

**KnowledgeSync Controller Action**Is Endpoint:  Enabled Access Log:  Causes Validation: **SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		Developer

**CODE**

```
function void KnowledgeSync()
{
    var actors = Domain.Actor.GetAll();

    try
    {
        foreach (Domain.Actor actor in actors)
        {
            Domain.ActorBackend.CreateKnowledgeActor(actor);
        }

        ShowMessage("Successfully synchronized actors");
    }
    catch Exception x
    {
        DebugLib.Logger.WriteLine(x);
    }
}
```

```

        ShowMessage("Something went wrong.", AppLib.MessageType.Error);
        return;
    }
}

```

### SaveActor Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ManageActors

#### CODE

```

function void SaveActor()
{
    var adminlist = Model.SelectedActor.Administrators;
    Domain.Actor dbActor = Model.SelectedActor;
    dbActor.Refresh();
    dbActorAdministrators = adminlist;
    dbActor.Save();

    FormControls.Modal.Hide();
}

```

### SelectUsers Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ManageActors

#### CODE

```

function void SelectUsers(Domain.Actor actor)
{
    actor.Refresh();
    Model.SelectedActor = actor;
    FormControls.Modal.Show();
}

```

### ActorsToAdministrators Form

#### Model

```

|-- Actor: Actor
|   |-- Id: int
|   |-- Name: string
|   |-- Description: string
|   |-- ShortDescription: string
|   |-- Url: string
|   |-- Email: string

```

```

|--- SpecifiedEntityType: string
|--- MemberOfCluster: bool
|--- GetCountOfClusterMembers: int
|--- ClusterName: string
|--- Administrators: DigicircUser
    |--- UserName: string
    |--- PasswordHash: string
    |--- SecurityStamp: string
    |--- EmailConfirmed: bool
    |--- LockoutEnabled: bool
    |--- PhoneNumberConfirmed: bool
    |--- TwoFactorEnabled: bool
    |--- AccessFailedCount: int
    |--- Name: string
    |--- Email: string
    |--- PhoneNumber: string
    |--- LockoutEndDate: DateTime
    |--- FirstName: string
    |--- LastName: string
    |--- SubscribeToNewsLetter: bool

```

## View

The screenshot shows a user selection interface. At the top, there is a button labeled "Select new User". Below it, there are three input fields: "Name", "Username", and "Email". Each field has a placeholder below it: "% Name", "% UserName", and "% Email". The entire interface is enclosed in a dashed border.

## Controller

### Add Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

## CODE

```
function void Add(Domain.DigicircUser user)
{
}
```

## ClusterInitialization Form

### Model

```
| -- Name: string
```

### View

### Controller

#### Index Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		Developer

#### CODE

```
function void Index() {
    Model.Title = LocalResources.RES_PAGETITLE_Index;
}
```

#### Save Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

**SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

**CODE**

```

function void Save()
{
    Domain.Actor actor;
    actor.CircularEconomyRequirements = Domain.CircularEconomyReport.Create();
    actor.EntityType = Domain.EntityType.Find(a => a.IsCluster).First();
    actor.Name = Model.Name;
    try
    {
        actor.Save();
    }
    catch Exception x
    {
        DebugLib.Logger.WriteLine(x);
        ShowMessage("Something went wrong. " + x.Message,AppLib.MessageType.Error);
        return;
    }

    ShowMessage("Successfully Saved");
    Model.Name = string.GetEmpty();
    return;
}

```

**MaterialForm Form****Model**

```

|-- Material: Material
|-- Description: string
|-- HsSpecific: string
|-- Id: int
|-- IsHazardous: bool
|-- Name: string
|-- PendingGraph: bool
|-- ConvertedBy: Process
    |-- Id: int
    |-- Name: string
    |-- Notes: string
    |-- ProductName: string
    |-- Ref: string
    |-- SourceName: string
    |-- Product: Material
        |-- Description: string
        |-- HsSpecific: string
        |-- Id: int
        |-- IsHazardous: bool

```

```
| -- Name: string
| -- PendingGraph: bool
|-- Source: Material
| -- Description: string
| -- HsSpecific: string
| -- Id: int
| -- IsHazardous: bool
| -- Name: string
| -- PendingGraph: bool
|-- ConvertBy: Process
| -- Id: int
| -- Name: string
| -- Notes: string
| -- ProductName: string
| -- Ref: string
| -- SourceName: string
| -- Source: Material
|   -- Description: string
|   -- HsSpecific: string
|   -- Id: int
|   -- IsHazardous: bool
|   -- Name: string
|   -- PendingGraph: bool
|-- Product: Material
| -- Description: string
| -- HsSpecific: string
| -- Id: int
| -- IsHazardous: bool
| -- Name: string
| -- PendingGraph: bool
|-- Type: ProductType
| -- Id: int
| -- Name: string
|-- PhysicalForm: PhysicalForm
| -- Code: string
| -- Id: int
| -- Value: string
|-- UnitOfMeasurement: UnitOfMeasurement
| -- Code: string
| -- Id: int
| -- Value: string
|-- NewProcess: Process
| -- Id: int
| -- Name: string
| -- Notes: string
| -- ProductName: string
| -- Ref: string
| -- SourceName: string
| -- Product: Material
|   -- Description: string
|   -- HsSpecific: string
|   -- Id: int
```

```
|-- IsHazardous: bool
|-- Name: string
|-- PendingGraph: bool
|-- Source: Material
|-- Description: string
|-- HsSpecific: string
|-- Id: int
|-- IsHazardous: bool
|-- Name: string
|-- PendingGraph: bool
|-- ConvertBy: bool
|-- NewProcesses: Process
|-- Id: int
|-- Name: string
|-- Notes: string
|-- ProductName: string
|-- Ref: string
|-- SourceName: string
|-- DeleteProcesses: Process
|-- Id: int
|-- Name: string
|-- Notes: string
|-- ProductName: string
|-- Ref: string
|-- SourceName: string
|-- Edited: bool
|-- EditedProcesses: Process
|-- Id: int
|-- Name: string
|-- Notes: string
|-- ProductName: string
|-- Ref: string
|-- SourceName: string
```

**View**

% Title    Unsaved Changes!

[Back](#) [Delete](#) [Save](#)

Name	Convert By <a href="#">+</a>		
<a href="#">@ % Material.Name</a>	Name	Product	
Type	<a href="#">% Name</a>	<a href="#">% ProductName</a>	<a href="#">Delete</a> <a href="#">Edit</a>
Physical Form			
<a href="#">% Material.PhysicalForm</a>			
Unit Of Measurement	Converted By <a href="#">+</a>		
<a href="#">% Material.UnitOfMeasurement</a>	Name	Source	
Is Hazardous	<a href="#">% Name</a>	<a href="#">% SourceName</a>	<a href="#">Delete</a> <a href="#">Edit</a>
Description			
<a href="#">## % Material.Description</a>			

ProcessModal

## Controller

### Add Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions

<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
--------------------------	-------------------------------------	-------------------------------------	--	--

#### CODE

```
function void Add() {
    Model.Title = LocalResources.RES_PAGETITLE_Add;
}
```

#### Edit Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Edit(int id) {
    Model.Material = Domain.Material.GetByKey(id);
    Model.Title = LocalResources.RES_PAGETITLE_Edit;
}
```

#### Save Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Save() {
    Model.Material.Save();

    if (Controller.Add.IsActive())
    {
        Domain.MaterialBackend.CreateKnowledgeMaterial(Model.Material);
    }
    else
    {
        Domain.MaterialBackend.UpdateKnowledgeMaterial(Model.Material);
    }

    foreach Domain.Process process in Model.NewProcesses
    {
        process.Save();
        Domain.ProcessBackend.CreateKnowledgeProcessPlus(process, false);
    }

    foreach Domain.Process process in Model.EditedProcesses
    {
        Domain.ProcessBackend.UpdateKnowledgeProcess(process);
    }
}
```

```

}

foreach (Domain.Process process in Model.DeleteProcesses
{
    process.Delete();
    Domain.ProcessBackend.DeleteKnowledgeProcess(process);
}

CloseForm();
}

```

#### DeleteConvertedByProcess Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

##### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

##### CODE

```

function void DeleteConvertedByProcess(Domain.Process process)
{
    Model.Material.ConvertedBy.Remove(process);

    Model.DeleteProcesses.Add(process);
}

```

#### OpenEditConvertByProcess Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

##### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

##### CODE

```

function void OpenEditConvertByProcess(Domain.Process process)
{
    Model.Edited = true;
    Model.ConvertBy = true;

    Model.NewProcess = process;
    FormControls.ProcessModal.Show();
}

```

#### OpenEditConvertedByProcess Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

##### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions

<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
--------------------------	-------------------------------------	-------------------------------------	--	--

#### CODE

```
function void OpenEditConvertedByProcess(Domain.Process process)
{
    Model.Edited = true;
    Model.ConvertBy = false;

    Model.NewProcess = process;
    FormControls.ProcessModal.Show();
}
```

#### DeleteConvertByProcess Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void DeleteConvertByProcess(Domain.Process process)
{
    Model.Material.ConvertBy.Remove(process);
    Model.DeleteProcesses.Add(process);
}
```

#### Delete Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Delete() {
    Model.Material.Delete();
    Domain.MaterialBackend.DeleteKnowledgeMaterial(Model.Material);
    CloseForm();
}
```

#### OpenNewConvertedByProcess Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```

function void OpenNewConvertedByProcess()
{
    Model.ConvertBy = false;
    Model.Edited = false;

    Model.NewProcess = Domain.Process.Create();
    Model.NewProcess.Product.Add(Model.Material);

    FormControls.ProcessModal.Show();
}

```

### **OpenNewConvertByProcess Controller Action**

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### **SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### **CODE**

```

function void OpenNewConvertByProcess()
{
    Model.ConvertBy = true;
    Model.Edited = false;

    Model.NewProcess = Domain.Process.Create();
    Model.NewProcess.Source.Add(Model.Material);

    FormControls.ProcessModal.Show();
}

```

### **AddNewProcess Controller Action**

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### **SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### **CODE**

```

function void AddNewProcess()
{
    if (Model.Edited)
    {
        Model.EditedProcesses.Add(Model.NewProcess);
        if (Model.ConvertBy)
        {
            FormControls.Table.Refresh();
        }
        else
        {

```

```

        FormControls.Table1.Refresh();
    }

}

else
{
    if (Model.ConvertBy)
    {
        Model.Material.ConvertBy.Add(Model.NewProcess);
    }
    else
    {
        Model.Material.ConvertedBy.Add(Model.NewProcess);
    }

    Model.NewProcesses.Add(Model.NewProcess);
}

FormControls.ProcessModal.Hide();
}

```

## MaterialList Form

### Model

MaterialList Form																																																																			
MaterialList Form																																																																			
MaterialList Form																																																																			
<b>Model</b>																																																																			
<b>View</b>																																																																			
<table border="1"> <thead> <tr> <th colspan="8">MaterialList Form</th> </tr> <tr> <th colspan="8">MaterialList Form</th> </tr> <tr> <th colspan="8">MaterialList Form</th> </tr> </thead> <tbody> <tr> <td colspan="8"> <div style="border: 1px solid #ccc; padding: 5px;"> <div style="background-color: #f9f9f9; border-bottom: 1px solid #ccc; padding-bottom: 5px;"> <span>Add</span> <span>Edit</span> <span>Sync</span> <span>Reset all</span> </div> <div style="display: flex; justify-content: space-between;"> <span>Page 1 of 10 pages</span> <span></span> <span></span> <span></span> </div> <div style="text-align: right;">25 per page</div> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>Name</th> <th>Product Type</th> <th>Pending Graph</th> <th>Id</th> <th>Is Hazardous</th> <th>Description</th> <th>Hs Specific</th> </tr> </thead> <tbody> <tr> <td>1 abc</td> <td>abc</td> <td>abc</td> <td>abc</td> <td>abc</td> <td>abc</td> <td>abc</td> </tr> <tr> <td>2 abc</td> <td>abc</td> <td>abc</td> <td>abc</td> <td>abc</td> <td>abc</td> <td>abc</td> </tr> <tr> <td>3 ...</td> <td>...</td> <td>...</td> <td>...</td> <td>...</td> <td>...</td> <td>...</td> </tr> </tbody> </table> </div> </td></tr> </tbody> </table>								MaterialList Form								MaterialList Form								MaterialList Form								<div style="border: 1px solid #ccc; padding: 5px;"> <div style="background-color: #f9f9f9; border-bottom: 1px solid #ccc; padding-bottom: 5px;"> <span>Add</span> <span>Edit</span> <span>Sync</span> <span>Reset all</span> </div> <div style="display: flex; justify-content: space-between;"> <span>Page 1 of 10 pages</span> <span></span> <span></span> <span></span> </div> <div style="text-align: right;">25 per page</div> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>Name</th> <th>Product Type</th> <th>Pending Graph</th> <th>Id</th> <th>Is Hazardous</th> <th>Description</th> <th>Hs Specific</th> </tr> </thead> <tbody> <tr> <td>1 abc</td> <td>abc</td> <td>abc</td> <td>abc</td> <td>abc</td> <td>abc</td> <td>abc</td> </tr> <tr> <td>2 abc</td> <td>abc</td> <td>abc</td> <td>abc</td> <td>abc</td> <td>abc</td> <td>abc</td> </tr> <tr> <td>3 ...</td> <td>...</td> <td>...</td> <td>...</td> <td>...</td> <td>...</td> <td>...</td> </tr> </tbody> </table> </div>								Name	Product Type	Pending Graph	Id	Is Hazardous	Description	Hs Specific	1 abc	2 abc	3 ...	...	...	...	...	...	...												
MaterialList Form																																																																			
MaterialList Form																																																																			
MaterialList Form																																																																			
<div style="border: 1px solid #ccc; padding: 5px;"> <div style="background-color: #f9f9f9; border-bottom: 1px solid #ccc; padding-bottom: 5px;"> <span>Add</span> <span>Edit</span> <span>Sync</span> <span>Reset all</span> </div> <div style="display: flex; justify-content: space-between;"> <span>Page 1 of 10 pages</span> <span></span> <span></span> <span></span> </div> <div style="text-align: right;">25 per page</div> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>Name</th> <th>Product Type</th> <th>Pending Graph</th> <th>Id</th> <th>Is Hazardous</th> <th>Description</th> <th>Hs Specific</th> </tr> </thead> <tbody> <tr> <td>1 abc</td> <td>abc</td> <td>abc</td> <td>abc</td> <td>abc</td> <td>abc</td> <td>abc</td> </tr> <tr> <td>2 abc</td> <td>abc</td> <td>abc</td> <td>abc</td> <td>abc</td> <td>abc</td> <td>abc</td> </tr> <tr> <td>3 ...</td> <td>...</td> <td>...</td> <td>...</td> <td>...</td> <td>...</td> <td>...</td> </tr> </tbody> </table> </div>								Name	Product Type	Pending Graph	Id	Is Hazardous	Description	Hs Specific	1 abc	abc	abc	abc	abc	abc	abc	2 abc	abc	abc	abc	abc	abc	abc	3 ...	...	...	...	...	...	...																																
Name	Product Type	Pending Graph	Id	Is Hazardous	Description	Hs Specific																																																													
1 abc	abc	abc	abc	abc	abc	abc																																																													
2 abc	abc	abc	abc	abc	abc	abc																																																													
3 ...	...	...	...	...	...	...																																																													

### Controller

#### Index Controller Action

Is EntryPoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		ManageSymbiosis

#### CODE

```
function void Index() { Model.Title = LocalResources.RES_PAGETITLE_Index; }
```

#### SendToGraph Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ManageSymbiosis, Developer

#### CODE

```
function void SendToGraph(Collection[Domain.Material] materials)
{
    foreach Domain.Material material in materials
    {
        if(material.PendingGraph)
        {
            material.PendingGraph = false;
            material.Save();

            Domain.MaterialBackend.CreateKnowledgeMaterial(material);
        }
        else
        {
            Domain.MaterialBackend.UpdateKnowledgeMaterial(material);
        }
    }

    FormControls.MaterialList.Refresh();
    var message = materials.Length == 1 ? "Material" : "Materials";

    ShowMessage(message + " Successfully send to graph", AppLib.MessageType.Success);
}
```

#### ResetAll Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ManageSymbiosis, Developer

#### CODE

```
function void ResetAll()
{
    foreach Domain.Material material in Domain.Material.GetAll()
    {
        material.PendingGraph = true;
        material.Save();
    }
}
```

```
    FormControls.MaterialList.Refresh();  
}
```

## ProcessForm Form

### Model

```
|-- Process: Process  
  |-- Id: int  
  |-- Name: string  
  |-- Notes: string  
  |-- Ref: string  
  |-- Product: Material  
    |-- Id: int  
    |-- Name: string  
    |-- Description: string  
  |-- Source: Material  
    |-- Id: int  
    |-- Name: string  
    |-- Description: string
```

### View

% Title    Unsaved Changes!

Back
 Delete
 Save

Process

Name	<input type="text" value="Process.Name"/>
Ref	<input type="text" value="Process.Ref"/>
Notes	<input type="text" value="Process.Notes"/>

## Controller

### Add Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

### CODE

```
function void Add() {
    Model.Title = LocalResources.RES_PAGETITLE_Add;
}
```

### Edit Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

**SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

**CODE**

```
function void Edit(int id) {
    Model.Process = Domain.Process.GetByKey(id);
    Model.Title = LocalResources.RES_PAGETITLE_Edit;
}
```

**Save Controller Action**

Is Entrypoint:  Enabled Access Log:  Causes Validation:

**SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

**CODE**

```
function void Save() {
    Model.Process.Save();

    if (Controller.Add.IsActive())
    {
        Domain.ProcessBackend.CreateKnowledgeProcess(Model.Process);
    }
    else
    {
        Domain.ProcessBackend.UpdateKnowledgeProcess(Model.Process);
    }

    CloseForm();
}
```

**Delete Controller Action**

Is Entrypoint:  Enabled Access Log:  Causes Validation:

**SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

**CODE**

```
function void Delete() {
    Model.Process.Delete();
    Domain.ProcessBackend.DeleteKnowledgeProcess(Model.Process);
    CloseForm();
}
```

**ProcessList Form**

## Model

### View

%				Title	Unsaved Changes!
		Add	Edit	IS Sync	
Page 1 of 10 pages		<<	<<	>>	>>
		25 per page			
Source		Name		Product	
1	abc	abc		abc	
2	abc	abc		abc	
3	...	...		...	

## Controller

### Index Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ManageSymbiosis

#### CODE

```
function void Index() { Model.Title = LocalResources.RES_PAGETITLE_Index; }
```

### SendToGraph Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ManageSymbiosis, Developer

#### CODE

```
function void SendToGraph()
{
```

```

var processes = Domain.Process.GetAll();
foreach Domain.Process process in processes
{
    Domain.ProcessBackend.CreateKnowledgeProcessPlus(process, false);
}

ShowMessage("Successfully synchronized process");
}

```

## ProductTypeForm Form

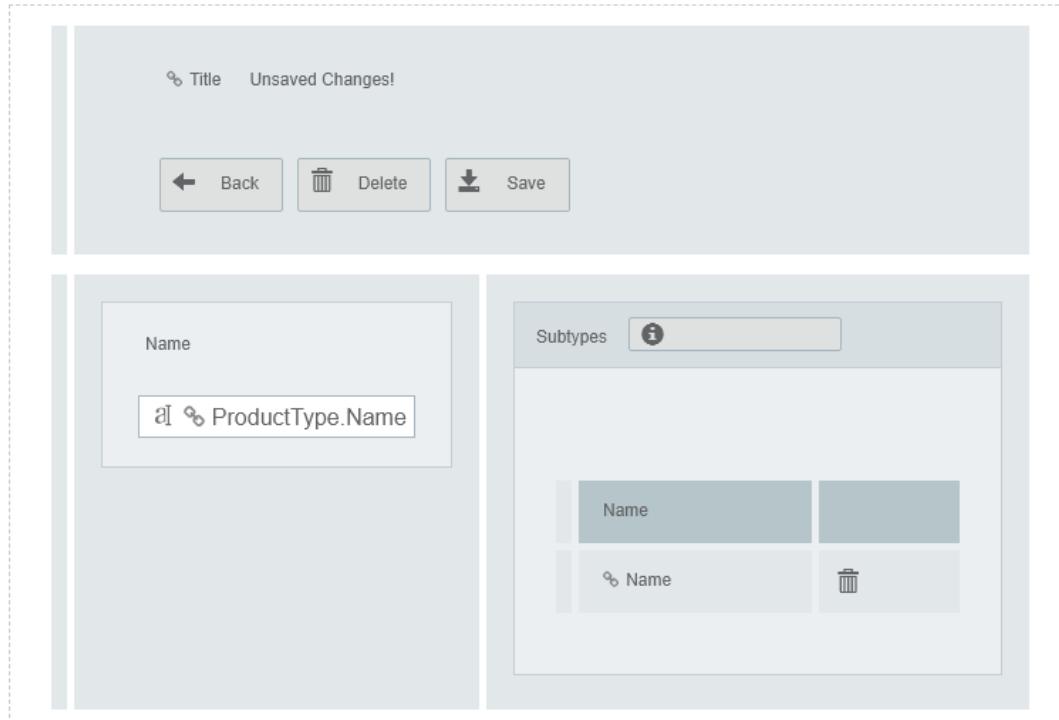
### Model

```

|-- ProductType: ProductType
|   |-- Id: int
|   |-- Name: string
|   |-- SybTypes: ProductType
|       |-- Id: int
|       |-- Name: string
|       |-- SybTypes: ProductType
|           |-- Id: int
|           |-- Name: string

```

### View



## Controller

### Add Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Add() {
    Model.Title = LocalResources.RES_PAGETITLE_Add;
}
```

### Edit Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Edit(int id) {
    Model.ProductType = Domain.ProductType.GetByKey(id);
    Model.Title = LocalResources.RES_PAGETITLE_Edit;
}
```

### Save Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Save() {
    Model.ProductType.Save();
    CloseForm();
}
```

### Delete Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```

function void Delete() {
    Model.ProductType.Delete();
    CloseForm();
}

```

#### DeleteSubType Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```

function void DeleteSubType(Domain.ProductType subType)
{
    Model.ProductType.SybTypes.Remove(subType);
    FormControls.Table.Refresh();
}

```

### ProductTypeList Form

#### Model

#### View

The screenshot shows a web-based application interface for managing product types. At the top, there is a header bar with a title placeholder 'Title'. Below the header, there are two buttons: 'Add' and 'Edit'. To the right of these buttons is a set of icons for file operations (upload, download, search, etc.). Below the buttons, there is a pagination control showing 'Page 1 of 10 pages' and a '25 per page' dropdown. The main content area is a table with three rows. The first row has a header column labeled 'Name'. The second row contains the value 'abc' under the 'Name' column. The third row contains the value '...' under the 'Name' column.

	Name
1	abc
2	abc
3	...

#### Controller

#### Index Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

**SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ManageSymbiosis

**CODE**

```
function void Index() { Model.Title = LocalResources.RES_PAGETITLE_Index; }
```

**ManageResources Form****Model**

```
|-- Actor: Actor
|-- ClusterName: string
|-- Description: string
|-- Email: string
|-- GetCountOfClusterMembers: int
|-- HasSites: bool
|-- Id: int
|-- Keywords: string
|-- MemberOfCluster: bool
|-- Name: string
|-- ShortDescription: string
|-- SpecifiedEntityType: string
|-- Url: string
|-- CircularEconomyRequirements: CircularEconomyReport
|-- DigitalExpertise: DigitalExpertise
|-- DigitalProviredNeeded: bool
|-- ExperienceInCircularEconomy: bool
|-- GetExperienceInCircularEconomy: string
|-- Id: int
|-- SpecifyExperienceInCircularEconomy: string
|-- ThematicExpertiseNeeded: bool
|-- Resources: Product
|-- Id: int
|-- IsDesired: bool
|-- Quantity: int
|-- ValidFrom: DateTime
|-- ValidTo: DateTime
|-- Resource: Material
|-- Description: string
|-- HsSpecific: string
|-- Id: int
|-- IsHazardous: bool
|-- Name: string
|-- PendingGraph: bool
|-- Site: Address
|-- Alias: string
|-- FullAddress: string
|-- Id: int
|-- Latitude: double
|-- Longitude: double
```

```
|-- Number: int
|-- StreetName: string
|-- Town: string
|-- Zip: string
|-- DesiredResources: Product
|-- Id: int
|-- IsDesired: bool
|-- Quantity: int
|-- ValidFrom: DateTime
|-- ValidTo: DateTime
|-- Resource: Material
|-- Description: string
|-- HsSpecific: string
|-- Id: int
|-- IsHazardous: bool
|-- Name: string
|-- PendingGraph: bool
|-- Site: Address
|-- Alias: string
|-- FullAddress: string
|-- Id: int
|-- Latitude: double
|-- Longitude: double
|-- Number: int
|-- StreetName: string
|-- Town: string
|-- Zip: string
|-- AddedBy: DigicircUser
|-- AccessFailedCount: int
|-- Email: string
|-- EmailConfirmed: bool
|-- FirstName: string
|-- LastName: string
|-- LockoutEnabled: bool
|-- LockoutEndDate: DateTime
|-- Name: string
|-- PasswordHash: string
|-- PhoneNumber: string
|-- PhoneNumberConfirmed: bool
|-- SecurityStamp: string
|-- SubscribeToNewsLetter: bool
|-- TwoFactorEnabled: bool
|-- UserName: string
|-- Administrators: DigicircUser
|-- AccessFailedCount: int
|-- Email: string
|-- EmailConfirmed: bool
|-- FirstName: string
|-- LastName: string
|-- LockoutEnabled: bool
|-- LockoutEndDate: DateTime
|-- Name: string
```

```
|-- PasswordHash: string
|-- PhoneNumber: string
|-- PhoneNumberConfirmed: bool
|-- SecurityStamp: string
|-- SubscribeToNewsLetter: bool
|-- TwoFactorEnabled: bool
|-- UserName: string
|-- SelectedProduct: Product
|-- Id: int
|-- IsDesired: bool
|-- Quantity: int
|-- ValidFrom: DateTime
|-- ValidTo: DateTime
|-- Resource: Material
|-- Description: string
|-- HsSpecific: string
|-- Id: int
|-- IsHazardous: bool
|-- Name: string
|-- PendingGraph: bool
|-- Site: Address
|-- Alias: string
|-- FullAddress: string
|-- Id: int
|-- Latitude: double
|-- Longitude: double
|-- Number: int
|-- StreetName: string
|-- Town: string
|-- Zip: string
|-- ModalTitle: string
|-- Material: Material
|-- Description: string
|-- HsSpecific: string
|-- Id: int
|-- IsHazardous: bool
|-- Name: string
|-- PendingGraph: bool
|-- RequestedBy: DigicircUser
|-- AccessFailedCount: int
|-- Email: string
|-- EmailConfirmed: bool
|-- FirstName: string
|-- LastName: string
|-- LockoutEnabled: bool
|-- LockoutEndDate: DateTime
|-- Name: string
|-- PasswordHash: string
|-- PhoneNumber: string
|-- PhoneNumberConfirmed: bool
|-- SecurityStamp: string
|-- SubscribeToNewsLetter: bool
```

```
|-- TwoFactorEnabled: bool
|-- UserName: string
|-- NewDesiredResources: Product
|-- Id: int
|-- IsDesired: bool
|-- Quantity: int
|-- ValidFrom: DateTime
|-- ValidTo: DateTime
|-- Resource: Material
|-- Description: string
|-- HsSpecific: string
|-- Id: int
|-- IsHazardous: bool
|-- Name: string
|-- PendingGraph: bool
|-- NewResources: Product
|-- Id: int
|-- IsDesired: bool
|-- Quantity: int
|-- ValidFrom: DateTime
|-- ValidTo: DateTime
|-- Resource: Material
|-- Description: string
|-- HsSpecific: string
|-- Id: int
|-- IsHazardous: bool
|-- Name: string
|-- PendingGraph: bool
|-- DeleteDesiredResources: Product
|-- Id: int
|-- IsDesired: bool
|-- Quantity: int
|-- ValidFrom: DateTime
|-- ValidTo: DateTime
|-- Resource: Material
|-- Description: string
|-- HsSpecific: string
|-- Id: int
|-- IsHazardous: bool
|-- Name: string
|-- PendingGraph: bool
|-- DeleteResources: Product
|-- Id: int
|-- IsDesired: bool
|-- Quantity: int
|-- ValidFrom: DateTime
|-- ValidTo: DateTime
|-- UpdateDesiredResources: Product
|-- Id: int
|-- IsDesired: bool
|-- Quantity: int
|-- ValidFrom: DateTime
```

```

|-- ValidTo: DateTime
|-- UpdateResource: Product
|-- Id: int
|-- IsDesired: bool
|-- Quantity: int
|-- ValidFrom: DateTime
|-- ValidTo: DateTime
|-- IsEdited: bool

```

## View

The screenshot shows a modal dialog box with the following components:

- Header:** "% Title Unsaved Changes!"
- Buttons:** Back, Save, Request New Material
- Section Headers:** Offer Resources, Request Resources, +
- Add new Resource:** A button labeled "Add new Resource".
- Form Fields:** Site, Name, Quantity, Valid From, Valid To.
- Labels:** "% Site.Town", "% Resource.Name", "% Quantity", "% ValidFrom", "% ValidTo".
- Actions:** A trash can icon and a pencil icon.

Below the modal, there are two tabs:

- Modal
- NewMaterialModal

## Controller

### Index Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		EditOrganisation

## CODE

```

function void Index(int id) {
    Model.Actor = Domain.Actor.GetByKey(id);
    Model.Title = "" + Model.Actor.Name + ' ' + LocalResources.RES_PAGETITLE_Index;
    Model.IsEdited = false;

    var currentUserName = AppLib.Session.GetCurrentUserName();

    if(currentUserName != Model.Actor.AddedBy.UserName &&
    Model.Actor.Administrators.Where(a => a.UserName == currentUserName).Length == 0)
    {
        ShowMessage("You don't have permission to edit this actor.",
        AppLib.MessageType.Warning, FormModels.ActorForm.Controller.Show.GetLink(id));
        return;
    }
}

```

### AddNewResource Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		EditOrganisation

#### CODE

```

function void AddNewResource(bool desired)
{
    Model.ModalTitle = "Add new Resource";
    Model.SelectedProduct = Domain.Product.Create();
    Model.SelectedProduct.IsDesired = desired;
    Model.IsEdited = false;
    FormControls.Modal.Show();
}

```

### EditResource Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		EditOrganisation

#### CODE

```

function void EditResource(Domain.Product product)
{
    Model.ModalTitle = "Edit " + product.Resource.Name;
//    if(product.IsDesired)
//    {
//        Model.Actor.CircularEconomyRequirements.DesiredResources.Remove(product);
//    }
//    else

```

```

//      {
//          Model.Actor.CircularEconomyRequirements.Resources.Remove(product);
//      }
//      Model.Edited = true;
//      Model.SelectedProduct = product;
//      FormControls.Modal.Show();
}

```

### **CloseModal Controller Action**

Is EntryPoint:  Enabled Access Log:  Causes Validation:

#### **SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		EditOrganisation

#### **CODE**

```

function void CloseModal()
{
    if(Model.SelectedProduct.IsDesired)
    {
        if (Model.Edited)
        {
            Model.UpdateDesiredResources.Add(Model.SelectedProduct);
        }
        else
        {

Model.Actor.CircularEconomyRequirements.DesiredResources.Add(Model.SelectedProduct);
            Model.NewDesiredResources.Add(Model.SelectedProduct);
        }
    }
    else
    {
        if (Model.Edited)
        {
            Model.UpdateResource.Add(Model.SelectedProduct);
        }
        else
        {

Model.Actor.CircularEconomyRequirements.Resources.Add(Model.SelectedProduct);
            Model.NewResources.Add(Model.SelectedProduct);
        }

    }
    Model.SelectedProduct = null;
    Model.Edited = false;
    FormControls.Modal.Hide();
}

```

### **DeleteResource Controller Action**

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		EditOrganisation

#### CODE

```
function void DeleteResource(Domain.Product product)
{
    if(product.IsDesired)
    {
        Model.Actor.CircularEconomyRequirements.DesiredResources.Remove(product);
    }
    else
    {
        Model.Actor.CircularEconomyRequirements.Resources.Remove(product);
    }
    Domain.ActorBackend.DeleteRelationships(Model.Actor.Id, product.Resource.Id);
    if(product.Id != 0)
    {
        product.Delete();
    }
}
```

#### Save Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		EditOrganisation

#### CODE

```
function void Save()
{
    //connect in graph
    foreach(Domain.Product product in Model.NewResources)
    {
        Domain.ActorBackend.ConnectActorOfferedBy(Model.Actor.Id, product);
    }

    foreach(Domain.Product product in Model.NewDesiredResources)
    {
        Domain.ActorBackend.ConnectActorRequests(Model.Actor.Id, product);
    }

    var response = Domain.ElasticDoc.SendActorDoc(Model.Actor);
    DebugLib.Logger.WriteLine("response elastic " + response);

    Model.Actor.Save();
}
```

```

        ShowMessage("Resources saved successfully.", AppLib.MessageType.Success);
    }
}

```

### **RequestNewMaterial Controller Action**

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### **SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		EditOrganisation

#### **CODE**

```

function void RequestNewMaterial()
{
    FormControls.NewMaterialModal.Show();
}

```

### **CloseMaterialModal Controller Action**

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### **SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		EditOrganisation

#### **CODE**

```

function void CloseMaterialModal(bool save)
{
    if(save)
    {
        Model.Material.RequestedBy =
Domain.DigicircUser.GetByKey(AppLib.Session.GetCurrentUserName());
        Model.Material.PendingGraph = true;
        Model.Material.Save();
    }
    Model.Material = null;
    FormControls.NewMaterialModal.Hide();
}

```

### **ResourceForm Form**

#### **Model**

```

|-- Product: Product
|-- Id: int
|-- IsDesired: bool
|-- Quantity: int
|-- ValidFrom: DateTime
|-- ValidTo: DateTime
|-- Resource: Material
|-- Description: string

```

```

|-- HsSpecific: string
|-- Id: int
|-- IsHazardous: bool
|-- Name: string
|-- PendingGraph: bool
|-- Site: Address
|-- Alias: string
|-- FullAddress: string
|-- Id: int
|-- Latitude: double
|-- Longitude: double
|-- Number: int
|-- StreetName: string
|-- Town: string
|-- Zip: string
|-- ActorId: int

```

## View

<p><b>Site</b></p> <div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 10px;"> <span style="font-size: 1.5em;">%</span> <span style="font-size: 1.5em;">%</span> Product.Site <span style="font-size: 1.5em;">▼</span> </div> <p>Valid From</p> <div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 10px;"> <span style="font-size: 1.5em;">📅</span> <span style="font-size: 1.5em;">%</span> Product.ValidFrom         </div> <p>Quantity</p> <div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 10px;"> <span style="font-size: 1.5em;">��</span> <span style="font-size: 1.5em;">%</span> Product.Quantity         </div>	<p><b>Material*</b></p> <div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 10px;"> <span style="font-size: 1.5em;">%</span> <span style="font-size: 1.5em;">%</span> Product.Resource <span style="font-size: 1.5em;">▼</span> </div> <p>Valid To</p> <div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 10px;"> <span style="font-size: 1.5em;">📅</span> <span style="font-size: 1.5em;">%</span> Product.ValidTo         </div>
<small>* If the material you want is not listed, you can request a new material in the manage resources page.</small>	

## Controller

### UnitOfWorkMeasurementForm Form

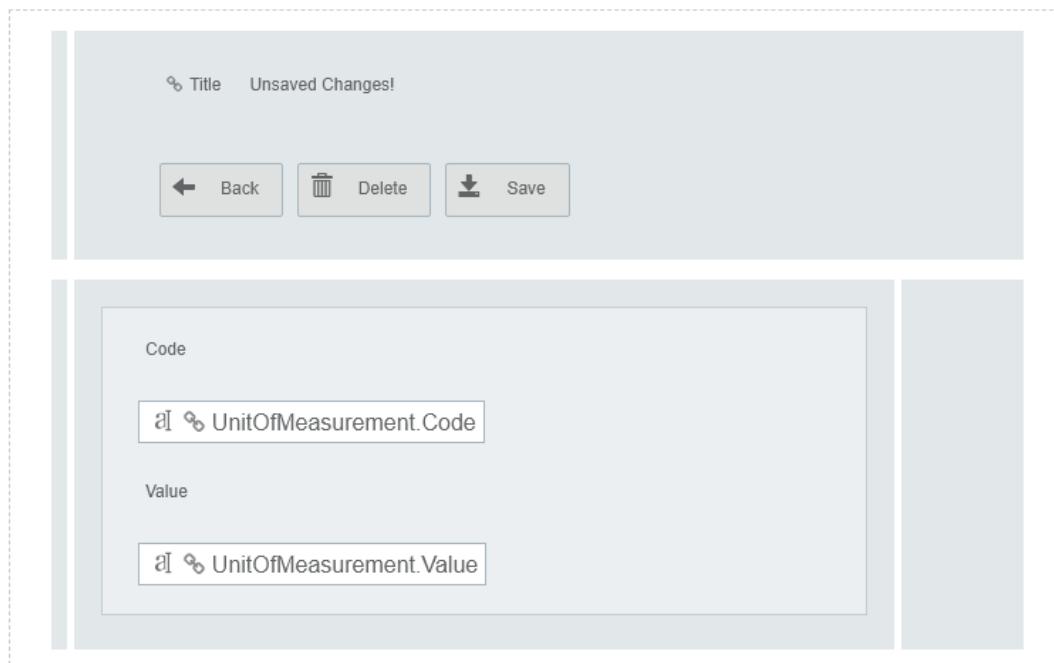
#### Model

```

|-- UnitOfMeasurement: UnitOfMeasurement
|-- Id: int
|-- Code: string
|-- Value: string

```

## View



## Controller

### Add Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Add() {
    Model.Title = LocalResources.RES_PAGETITLE_Add;
}
```

### Edit Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Edit(int id) {
    Model.UnitOfMeasurement = Domain.UnitOfMeasurement.GetByKey(id);
```

```
    Model.Title = LocalResources.RES_PAGETITLE_Edit;  
}
```

#### Save Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Save() {  
    Model.UnitOfMeasurement.Save();  
    CloseForm();  
}
```

#### Delete Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Delete() {  
    Model.UnitOfMeasurement.Delete();  
    CloseForm();  
}
```

### UnitOfMeasurementList Form

#### Model

#### View

	Code	Value
1	abc	abc
2	abc	abc
3	...	...

## Controller

### Index Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		ManageSymbiosis

### CODE

```
function void Index() { Model.Title = LocalResources.RES_PAGETITLE_Index; }
```

## PhysicalFormForm Form

### Model

```
|-- PhysicalForm: PhysicalForm
|-- Id: int
|-- Code: string
|-- Value: string
```

### View

>Title    Unsaved Changes!

[Back](#) [Delete](#) [Save](#)

Code	
<code>aI % PhysicalForm.Code</code>	
Value	
<code>aI % PhysicalForm.Value</code>	

## Controller

### Add Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Add() {
    Model.Title = LocalResources.RES_PAGETITLE_Add;
}
```

### Edit Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Edit(int id) {
    Model.PhysicalForm = Domain.PhysicalForm.GetByKey(id);
```

```
    Model.Title = LocalResources.RES_PAGETITLE_Edit;  
}
```

#### Save Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Save() {  
    Model.PhysicalForm.Save();  
    CloseForm();  
}
```

#### Delete Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Delete() {  
    Model.PhysicalForm.Delete();  
    CloseForm();  
}
```

### PhysicalFormList Form

#### Model

#### View

	Code	Value
1	abc	abc
2	abc	abc
3	...	...

## Controller

### Index Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		ManageSymbiosis

### CODE

```
function void Index() { Model.Title = LocalResources.RES_PAGETITLE_Index; }
```

## KnowledgeHub Form

### Model

```
|-- Waste: Material
|-- Description: string
|-- HsSpecific: string
|-- Id: int
|-- IsHazardous: bool
|-- Name: string
|-- PendingGraph: bool
|-- Product: Material
|-- Description: string
|-- HsSpecific: string
|-- Id: int
|-- IsHazardous: bool
|-- Name: string
|-- PendingGraph: bool
```

```
|-- Endpoint: string
|-- Material: Product
|  |-- Id: int
|  |-- IsDesired: bool
|  |-- Quantity: int
|  |-- ValidFrom: DateTime
|  |-- ValidTo: DateTime
|  |-- Resource: Material
|     |-- Description: string
|     |-- HsSpecific: string
|     |-- Id: int
|     |-- IsHazardous: bool
|     |-- Name: string
|     |-- PendingGraph: bool
|-- Actor: Actor
|  |-- ClusterName: string
|  |-- Description: string
|  |-- Email: string
|  |-- GetCountOfClusterMembers: int
|  |-- HasSites: bool
|  |-- Id: int
|  |-- Keywords: string
|  |-- MemberOfCluster: bool
|  |-- Name: string
|  |-- ShortDescription: string
|  |-- SpecifiedEntityType: string
|  |-- Url: string
|  |-- CircularEconomyRequirements: CircularEconomyReport
|     |-- DigitalExpertise: DigitalExpertise
|     |-- DigitalProviredNeeded: bool
|     |-- ExperienceInCircularEconomy: bool
|     |-- GetExperienceInCircularEconomy: string
|     |-- Id: int
|     |-- SpecifyExperienceInCircularEconomy: string
|     |-- ThematicExpertiseNeeded: bool
|     |-- DesiredResources: Product
|        |-- Id: int
|        |-- IsDesired: bool
|        |-- Quantity: int
|        |-- ValidFrom: DateTime
|        |-- ValidTo: DateTime
```

## View



## Controller

### Index Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Index() {
    Model.Title = LocalResources.RES_PAGETITLE_Index;
    Model.Endpoint = Application.Settings.GraphDBEndpoint;

    var currentUserName = AppLib.Session.GetCurrentUserName();
    if(!string.IsNullOrEmpty(currentUserName))
    {
        Model.Actor = Domain.Actor.Find(a => a.AddedBy.UserName ==
        currentUserName).Where(a => a.Id == 601).First();
    }
}
```

### Search Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

**CODE**

```
function void Search()
{
    ExecuteJavascript("update('" + Model.Waste.Id + "', '" + Model.Product.Id + "')");
}
```

**ForgotUsername Form****Model**

```
|-- txtEmail: string
|-- FromMatching: bool
```

**View**

The screenshot shows a user interface for a 'Forgot Username' form. At the top, there is a placeholder image of a user profile. Below it is an input field labeled 'Email' containing 'txtEmail'. A validation message '% Validations.EmailIsEmpty.Message' is displayed below the input field. To the right of the input field is a 'Search' button. At the bottom of the form is a link 'Back To Sign in'.

**Controller****Render Controller Action**

Is Entrypoint:  Enabled Access Log:  Causes Validation:

**SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions

<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		
-------------------------------------	--------------------------	-------------------------------------	--	--

#### CODE

```
function void Render(bool fromMatching)
{
    Model.FromMatching = fromMatching;
}
```

#### ResetPasswordRequest Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void ResetPasswordRequest()
{
    if (string.IsNullOrWhiteSpace(Model.txtEmail)
        || !RegExLib.VerbalExpressions.IsEmail(Model.txtEmail))
    {
        ShowMessage(LocalResources.RES_CUSTOM_NoMail, AppLib.MessageType.Error);
        return;
    }

    var user = Domain.ApplicationUser.Find(u => u.Email == Model.txtEmail).First();

    if (user == null)
    {
        ShowMessage(LocalResources.RES_CUSTOM_NotFound, AppLib.MessageType.Error);
        return;
    }

    CommonLib.EmailMessage mail;

    Collection<string> recipients;
    recipients.Add(user.Email);

    mail.To = recipients;
    mail.IsBodyHtml = true;
    mail.Subject = LocalResources.RES_CUSTOM_ResetPasswordLink + " " +
AppLib.Application.Name;
    mail.Body = "<h3>" + LocalResources.RES_CUSTOM_ClickToReset + "</h3>" +
        "<p>" + user.UserName + "</p>";

    CommonLib.Utilities.SendEmail(mail);

    string signInUrl =
FormModels.SignInPage.Controller.Load.GetLink(Model.FromMatching);
    ShowMessage(LocalResources.RES_CUSTOM_MailSoon, AppLib.MessageType.Success,
```

```
signInUrl);  
}
```

## OportunitiesExplorer Form

### Model

### View

```
[...]
```

### Controller

#### Index Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

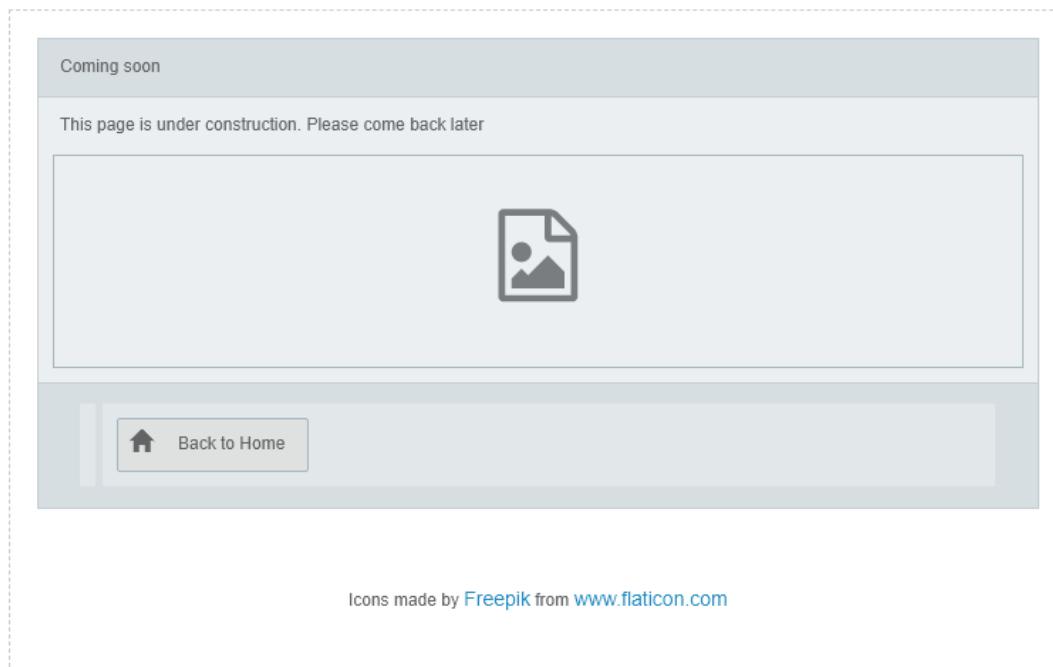
#### CODE

```
function void Index() {  
    Model.Title = LocalResources.RES_PAGETITLE_Index;  
}
```

## UnderConstructionPage Form

### Model

### View



Icons made by [Freepik](#) from [www.flaticon.com](#)

## Controller

### Index Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

### CODE

```
function void Index() {  
}  
}
```

## MatchBaseExplorer Form

### Model

```
|-- Endpoint: string  
|-- Query: GraphQuery  
  |-- DisplayMode: string  
  |-- SearchMode: string  
  |-- DesiredProduct: Product  
    |-- Id: int  
    |-- IsDesired: bool  
    |-- Quantity: int
```

```
|-- ValidFrom: DateTime
|-- ValidTo: DateTime
|-- Resource: Material
    |-- Description: string
    |-- HsSpecific: string
    |-- Id: int
    |-- IsHazardous: bool
    |-- Name: string
    |-- PendingGraph: bool
|-- ResourceProduct: Product
    |-- Id: int
    |-- IsDesired: bool
    |-- Quantity: int
    |-- ValidFrom: DateTime
    |-- ValidTo: DateTime
    |-- Resource: Material
        |-- Description: string
        |-- HsSpecific: string
        |-- Id: int
        |-- IsHazardous: bool
        |-- Name: string
        |-- PendingGraph: bool
|-- SelectedActor: Actor
    |-- ClusterName: string
    |-- Description: string
    |-- Email: string
    |-- GetCountOfClusterMembers: int
    |-- HasSites: bool
    |-- Id: int
    |-- Keywords: string
    |-- MemberOfCluster: bool
    |-- Name: string
    |-- ShortDescription: string
    |-- SpecifiedEntityType: string
    |-- Url: string
    |-- CircularEconomyRequirements: CircularEconomyReport
        |-- DigitalExpertise: DigitalExpertise
        |-- DigitalProviredNeeded: bool
        |-- ExperienceInCircularEconomy: bool
        |-- GetExperienceInCircularEconomy: string
        |-- Id: int
        |-- SpecifyExperienceInCircularEconomy: string
        |-- ThematicExpertiseNeeded: bool
        |-- Resources: Product
            |-- Id: int
            |-- IsDesired: bool
            |-- Quantity: int
            |-- ValidFrom: DateTime
            |-- ValidTo: DateTime
            |-- Resource: Material
                |-- Description: string
                |-- HsSpecific: string
```

```
|-- Id: int
|-- IsHazardous: bool
|-- Name: string
|-- PendingGraph: bool
|-- DesiredResources: Product
|-- Id: int
|-- IsDesired: bool
|-- Quantity: int
|-- ValidFrom: DateTime
|-- ValidTo: DateTime
|-- Resource: Material
|-- Description: string
|-- HsSpecific: string
|-- Id: int
|-- IsHazardous: bool
|-- Name: string
|-- PendingGraph: bool
|-- ActorNames: ActorNames
|-- Id: int
|-- Name: string
```

**View**

**Matching Criteria**

Actor

Looking for?

Requests

Offers

**Knowledge Matching**

Display results in:

```
<div id="viz"></div>
```

`% foreach Table_CurrentItem in ActorDataSet1`



`% ActorDataSet1.Item.Name`

`% ActorDataSet1.Item.ShortDescription`

`% ActorDataSet1.Item.EntityType.Value`

`% ActorDataSet1.Item.Address.Country.Name`

## Controller

### Index Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```
function void Index() {
    Model.Title = LocalResources.RES_PAGETITLE_Index;
    var currentUsername = AppLib.Session.GetCurrentUserName();
    Model.Query.SelectedActor = Domain.Actor.Find(a => a.AddedBy.UserName ==
    currentUsername || a.Administrators.Where(b => b.UserName == currentUsername).Length >
```

```

0).First();

    Model.Query.SearchMode = "offers";
    Model.Query.DisplayMode = "graph";
    Model.Endpoint = Application.Settings.GraphDBEndpoint;
    Model.Query.ActorNames = Model.Query.SelectedActor.ListPossibleMatches(true);
}

```

### FromBack Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		User

#### CODE

```

function void FromBack(int id)
{
    Domain.GraphQuery cachedQuery = AppLib.Session.Storage.Get("resultsGraph") as
Domain.GraphQuery;
    if(cachedQuery != null)
    {
        Model.Query = cachedQuery;
    }
    else
    {
        Model.Query.SelectedActor = Domain.Actor.Find(a => a.AddedBy.UserName ==
AppLib.Session.GetCurrentUserName()).First();
        Model.Query.SearchMode = "offers";
        Model.Query.DisplayMode = "graph";
    }
//    if(Model.Query.DisplayMode == "list")
//    {
//        ExecuteJavascript("setTimeout(function(){
window._commander.gridGoToSavedPage(['Table']), 200);");
//        ExecuteJavascript("setTimeout(function(){ $(\"[data-key='" + id + "']\")[0].scrollIntoView({ behavior: \"smooth\", block: \"start\" }), 1500);");
//    }
    Model.Endpoint = Application.Settings.GraphDBEndpoint;
}

```

### ChangeMode Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		User

#### CODE

```

function void ChangeMode()
{

```

```

        AppLib.Session.Storage.Add("resultsGraph", Model.Query);
        FormControls.List.Refresh();
    }
}

```

### **Update Controller Action**

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### **SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### **CODE**

```

function void Update()
{
    if(Model.Query.SearchMode == "offers")
    {
        if(Model.Query.ResourceProduct.Id > 0)
        {
            ExecuteJavascript("matchOffers('" +
Model.Query.ResourceProduct.Resource.Id + "', '" + Model.Query.SelectedActor.Id +
"')");
        }
        else
        {
            ExecuteJavascript("updateOffers('" + Model.Query.SelectedActor.Id + "')");
            Model.Query.ActorNames =
Model.Query.SelectedActor.ListPossibleMatches(true);
        }
    }
    else
    {
        if(Model.Query.DesiredProduct.Id > 0)
        {
            ExecuteJavascript("matchRequests('" +
Model.Query.DesiredProduct.Resource.Id + "', '" + Model.Query.SelectedActor.Id +
"')");
        }
        else
        {
            ExecuteJavascript("updateRequests('" + Model.Query.SelectedActor.Id +
"')");
            Model.Query.ActorNames =
Model.Query.SelectedActor.ListPossibleMatches(false);
        }
    }
    FormControls.List.Refresh();
    AppLib.Session.Storage.Add("resultsGraph", Model.Query);
}

```

### **GoToActorForm Controller Action**

Is Entrypoint:  Enabled Access Log:  Causes Validation:

**SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		User

**CODE**

```
function void GoToActorForm(int id)
{
    ExecuteJavascript("_commander.gridSaveState(['Table']);");
    FormModels.ActorViewForm.Controller.Show.Execute(id, true);
}
```

**Matching Form****Model****View**

Match

**Controller****Index Controller Action**Is Entrypoint:  Enabled Access Log:  Causes Validation: **SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Developer

**CODE**

```
function void Index() {
    Model.Title = LocalResources.RES_PAGETITLE_Index;
}
```

**Match Controller Action**Is Entrypoint:  Enabled Access Log:  Causes Validation: **SECURITY**

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Developer

## CODE

```
function void Match()
{
    foreach Domain.Actor actor in Domain.Actor.Find(a =>
a.CircularEconomyRequirements.Resources.Any())
    {
        Domain.Match.MatchProducts(actor);
    }

    ShowMessage("Finished");
}
```

## SymbiosisMasterPage Form

### Model

```
| -- Title: string
```

### View

Logo	MainMenuHeader	MasterAdministrationRoot	SymbiosisMasterSignIn	MasterRegister	MaterialFlow	SymbiosisMaterialsMenuItem
						

```
<script src="https://cdnjs.cloudflare.com/ajax/libs/owlCarousel2/2.3.4/owl.carousel.min.js" integrity="sha512-bPs7Ae6pVvhOSilcyUCIR7/q2OAsRiovw4vAkX+zJbw3ShAeeqezq50RIicURq7Oa20rW2n2q+fyXBNCu9lrw==" crossorigin="anonymous"></script> <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/owlCarousel2/2.3.4/assets/owl.carousel.min.css" integrity="sha512-ts3S5qG0BhnQRoyJxvNjeEM4UpMXHrQfTGmbQ1gKmeliCxISeBUaxhRBj/EFTzpbP4RVSpElkbmdJobCvhE3g==" crossorigin="anonymous" /> <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/owlCarousel2/2.3.4/assets/owl.theme.default.min.css" integrity="sha512-sMXTMNL1zRzoIHYKEujM2AqCLUR9F2C4/05cdbxjjLSRvMQiciEPCQZo++nk7go3BtSuK9kfa/s+a4f4i5pLkw==" crossorigin="anonymous" /> <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/owlCarousel2/2.3.4/assets/owl.theme.green.min.css" integrity="sha512-C8Movfk6DU/H5PzarG0+Dv9MA9lZzvmQpO/3clGImtY3vlud07myMu4M/NTPJl8jmZtt/4mC9bAioMZBBdA==" crossorigin="anonymous" /> <div class="row_style"> <svg id="pattern" class="round" xmlns="http://www.w3.org/2000/svg" version="1.1" width="100%" height="100% viewBox="0 0 100 100" preserveAspectRatio="none"> <path d="M0 100 C40 0 60 0 100 100 Z"></path> </svg> </div> <div class="container main-container"> <div class="row"> <div class="col-xs-12"> <div class="owl-carousel owl-theme" data-col_lg="12" data-col_md="6" data-col_sm="4" data-col_xs="1" data-item_space="15" data-loop="true" data-autoplay="true" data-smartspeed="400" data-nav="false" data-dots="false"> <div><a href="https://www.capdigital.com/en/"></a></div> <div><a href="https://www.digipolis.fi/en/front-page"></a></div> <div class="active"><a href="https://www.ctnnnova.com/"></a></div> <div class="active"><a href="https://www.f6s.com/"></a></div> <div class="active"><a href="https://www2.deloitte.com/it/it/pages/about-deloitte/topics/officine-innovazione---deloitte-italy---about.html?icid=wn_officine-innovazione---deloitte-italy---about"></a></div> <div class="active"><a href="http://www.polito.it/"></a></div> <div><a href="https://fasttrack.vc/"></a></div> <div><a href="https://draxis.gr/"></a></div> <div><a href="https://www.clmsuk.com/"></a></div> <div><a href="https://www.arthurslegal.com/"></a></div> <div><a href="https://www.inspiringculture.org/"></a></div> </div> </div> </div> </div> </div> </div>
```

```

6 col-md-6 col-xs-12 disclaimer-col"> <div class="disclaimer-container">  <p class="disclaimer-text">This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 873468.</p> </div> <div class="col-lg-6 col-md-6 col-xs-12 zappdev-col"> <div class="zappdev-container"> <span>Crafted by</span> <a href="http://zappdev.com/"> <svg id="Layer_1" data-name="Layer 1" xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink" viewBox="0 0 1201 317"><defs> <style>.cls-1{fill:none;}.cls-2{clip-path:url(#clip-path);}.cls-3{fill:#4f4f4f;}</style> <clipPath id="clip-path"><rect class="cls-1" x="14" y="10.28" width="1173" height="296.44"/> </clipPath> </defs> <title>zappdev-white</title> <g class="cls-2"> <polygon class="cls-3" points="152.36 198.03 203.89 71.19 34.3 71.33 17 117.53 122.81 117.45 152.36 198.03"/> <polygon class="cls-3" points="95.1 225.37 65.55 144.79 14 271.62 183.59 271.5 200.89 225.29 95.1 225.37"/> <path class="cls-3" d="M363.17,206.43v-.18h12.62L363.13,172.6v.14L326.39,75.32l-55.57,0L195.71,271.48l53.8,0,13.2-32.95,71.15,0,12.08,32.93,54.38,0-24.49-64.93Zm-86.38-6.73,21.43-63.49,22.08,63.47Z"/> <path class="cls-3" d="M586.56,128.19c-15-14.4-32.65-17.62-49.71-17.6l-31.82,0c6.07-11.63,7.09-24,7.09-35.11,0-16.46-3-32.93-18.26-47.6-15-14.4-32.66-17.63-49.7-17.61L363.10,35.1,1,132.67l387,206.4127,0L414,141.43l32,0c3.15,0,6.39-1.9,7.4-38l.12,165.7,50.85,0,0-65.32,0c15.29,0,33.51-2.09,48.78-17.08s17.32-33.23,17.32-48.84c0-16.45-3-32.92-18.26-47.59M452.46,94.06c-6.17,5.6-16.17,6.19-21.46,6.19H414l-48.78,17.35,0c6.45,0,15.86,88,21.76,7.5,29,5.28,5.87,12.34,5.87,17.64,0,4.7-28,12.62-6.43,17.92m92.69,100.29c-6.17,5.6-16.17,6.2-21.47,6.22h-17l-48.83H524c6.45,0,15.87,88,21.75,7.5,29,5.28,5.89,12.35,5.89,17.64,0,4.7-28,12.63-6.45,17.92"/> <path class="cls-3" d="M761.14,102C735.83,77.65,702.9,75.683,2.75l-66.74.06.14,196.08,71.16,0c33.51,0,59.66-14.16,75.23-29.75,19.09-19.14,25.54-42.08,25.53-67.94,0-21.17-4.45-49.4-27.38-71.42m-40.5,111.74c-13.22,12.94-30.57,14.14-42.91,14.14l-10.3,0-0.08-109.65,12.35,0c12.64,0,28.51,1.15,40.58,12.3,9.72,9.12,15.31,24.71,15.31,42.93,0,21.74-8.48,34.09-15.40,28"/> <path class="cls-3" d="M1000.71,178.43c0-20.47-3.32-52.39-29.11-76.92C949.9,81.06,922.48,76.6,901.2,76.6c-36,0-59.74,11.91-74.87,26.66-16,15.55-27,38.40.12-27.36,70.81,0,34.78,15.19,57.26,27.06,69.13,22.52,22.5,51.59,27.8,75.73,27.78,39.68,0,60.94-12.33,74.44-25.43a82.42,82.42,0,0,0,22.09-37.65l-62.63,0a30.32,30.32,0,0,1-11.44,12.28c-8.19,4.5-19.64,4.92-21.27,4.92-14.73,0-22.92-4.88-27-9.7-7.79-7.77-11.47-20.85-11.49-30.66l136.27-.12ZM866.47,147.82a37.43,37.43,0,0,1,9.8-19.63c7-7,15.55-9.84,26.6-9.84,6.55,0,18.4,1.2,27,9.8a44,44,0,0,1,10.65,19.63"/> <polyline class="cls-3" points="1131.73 74.71 1084.18 194.1 1037.07 74.79 981.5 74.83 1063.97 270.86 1103.65 270.84 1186.98 74.67 1131.73 74.71"/> </g> </svg> </a> </div> </div> <div> <p class="copyright">Copyright &#169; 2020 CLMS. All Rights reserved. <strong><a href="https://digicirc.eu/privacy-policy/">Privacy Policy</a></strong></p> </div> </div> <script> Joove.Widgets.MenuControl.prototype.HandleOverflowMenuItems = function (menuQuery, name, leftItemsToShow) {};
```

## Controller

### Render Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

#### CODE

```

function void Render()
{
    Model.Title = LocalResources.RES_PAGETITLE_Render;
}

```

### SignOut Controller Action

Is Entrypoint:  Enabled Access Log:  Causes Validation:

#### SECURITY

Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions

**CODE**

```
function void SignOut()
{
    AppLib.Security.SignOutUser();
    FormModels.SignInPage.Controller.Load.Execute(false);
}
```



## End of Document



This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 873468.