



WP3 – ENABLE: Creating LSD Enabler Tools

D3.7: Matchmaking tool



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

Document Information

Grant Agreement Number	873468	Acronym	DigiCirc
Full Title	European cluster-led accelerator for digitization of the circular economy across key emerging sectors		
Start Date	1 st May 2020	Duration	32 months
Project URL	https://digicirc.eu/		
Deliverable	D 3.7 – Matchmaking tool		
Work Package	WP 3 - ENABLE: Creating LSD Enabler Tools		
Date of Delivery	Contractual	31 st January 2021	Actual
Nature	Report	Dissemination Level	Public
Lead Beneficiary	DRAXIS		
Responsible Author	Konstantinos Skianis CLMS		
Contributions from	Yiannis Zorgios CLMS, Vasilis Vasilopoulos CLMS, Theodoros Kalampoukis CLMS, Konstantinos Skianis CLMS		

Document History

Version	Issue Date	Stage	Description	Contributor
0.1	01/09/2020	Draft	TOC	Konstantinos Skianis CLMS
0.2	10/09/2020	Draft	First version of document	Konstantinos Skianis CLMS
0.3	25/10/2020	V1	Resources	Theodoros Kalampoukis CLMS
0.4	15/01/2021	V1	Final version	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This deliverable contains original unpublished work except where clearly indicated otherwise.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 Circular Economy & Related work	8
3.1 Circular Economy	8
3.1.1 Buy side ('Wants' or 'Requests') benefits	9
3.1.2 Sell side ('Haves' or 'Offers') benefits	9
3.2 Related work	9
4 Operational Characteristics / Requirements for the Matchmaking Tool	12
4.1 Operational characteristics	12
4.2 User Requirements	13
4.2.1 Login/Register of User	13
4.2.2 Notifications	13
4.2.3 Search	13
4.2.4 Management	14
4.2.5 Registration of actors	14
4.2.6 View actor/company page	14
4.2.7 Add new actor	14
4.3 Matchmaking requirements	14
4.3.1.1 Matchmaking leading to synergies	14
4.3.1.2 Synergy tracking	14
4.3.1.3 Matching process	15
5 The Matchmaking Tool	16
5.1 Development platform	16
5.1.1 Fast, modern & native apps	16
5.1.2 Rich & responsive UI	16
5.1.3 Quick integration with APIs	17
5.1.4 Built-in security	17
5.1.5 Events	18
5.1.6 Comprehensive cache management	18
5.1.7 Painless localization	18
5.1.8 Unlimited extensibility	19
5.2 System architecture	19
5.3 Components	19
5.3.1 User Interface	19



D3.7: Matchmaking tool

5.3.2 Data layer.....	20
5.3.3 Ontologies.....	21
5.3.4 Requests and Offers.....	21
5.3.5 Matchmaking engine	22
5.3.5.1 Matching with search query (text).....	24
5.3.5.2 Matching offers with requests.....	25
5.4 Users of the Matchmaking Tool.....	28
5.5 Workflow	28
6 Web platform functionalities.....	30
6.1 User registration and login.....	30
6.2 Navigation and Search tool	32
6.3 Add new entity-actor	34
6.4 Offers and requests	35
6.5 “Synergy” suggestion: Matching via paths in the knowledge graph.....	36
6.6 Matching and ranking synergies	38
7 Use Cases	40
7.1 Use case 1	40
7.2 Use case 2	40
7.3 Use case 3	40
7.4 Use case 4	40
7.5 Use case 5	40
8 Conclusion.....	41
9 Bibliography	42
10 Appendix – Development documentation	43



Table of Figures

Figure 1: Architecture of Collaboration Platform for Enabling Industrial Symbiosis by J. Low et al. (2018).	10
Figure 2: Visualization of possible pathways for converting apple peels into resources, as shown in J. Low et al. (2018).	11
Figure 3: The DigiCirc platform architecture, with regards to the Matchmaking tool. The Matchmaking tool is the last component.....	19
Figure 4: an illustration of an organizations' requests and offers.....	22
Figure 5: actor and service profile.	24
Figure 6: an illustration of the matching process between offers and requests.	25
Figure 7: an illustration of an organization's or company's requests, offers and matches.....	26
Figure 8: matching offers and requests via the knowledge graph.	26
Figure 9: old and new matches stored at the database.....	27
Figure 10: the full workflow of the Industrial Symbiosis and Matchmaking tools.	27
Figure 11: the starting page of the Matchmaking tool.	30
Figure 12: registration of user.	31
Figure 13: user login UI component.....	32
Figure 14: the starting page of the matchmaking tool, after logging in as a standard user.	32
Figure 15: matching with options on country, sector, resources or text.	33
Figure 16: entities of the platform shown in a map frame.	34
Figure 17: an example of an entity in the matchmaking tool.	34
Figure 18: adding new entities or actors via the platform.	35
Figure 19: accessing the offers and requests components by clicking on the Resources button.....	35
Figure 20: the UI component responsible for adding new offers or requests of resources.	36
Figure 21: adding an offer or request of resources.	36
Figure 22: a synergy proposal by the Matchmaking tool, visualized as a graph.	37
Figure 23: the matching results returned as a list of the involved actors.....	37
Figure 24: the matching results can also be shown as a table of suggestions.....	38
Figure 25: the proposed synergies can be exported to files.	38
Figure 26: multiple paths available for synergies, as shown in the development environment.	39



List of acronyms

Acronym	Designation
AI	Artificial Intelligence
API	Application Programming Interface
CE	Circular Economy
GIS	Geographic Information System
GUI	Graphical User Interface
KPI	Key Performance Indicator
LSD	Large scale demonstration
NLP	Natural Language Processing
OGC	Open Geospatial Consortium
RTO	Regional & Technology Organisation
SME	Small and Medium Enterprise
UI	User Interface
US	User Story
WMS	Web Map Service



1 Executive summary

The DigiCirc project aims to boost the circular economy using digital tools, by supporting innovative 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, RTOs, academia, civil society, etc.) will be selected through open calls. DigiCirc will offer the consortia four DigiCirc Digital Tools to support them in the development, demonstration and commercialization of their solutions. Specifically, this includes a digital tool to facilitate data use for Circular applications, a Matchmaking tool to assist in creating complementary consortia. The focus of this deliverable is the Matchmaking tool, presenting in detail all the developed components and functionalities.

The Matchmaking tool (link: <http://digicirc.clms.io/>) is a complete web platform for providing relevant information within an Industrial Symbiosis environment. The tool is complementary to the Industrial Symbiosis tool, but also independent, offering search and matching capabilities for the participating parties regarding offers and requests of resources.

Initially, the project builds upon existing components from existing matchmaking modules. As such, the consortium assessed the functions, components and standards of these tools, and compared them to the service description of the DigiCirc matchmaking tool. Existing components were selected and combined where possible, and new components were developed where needed.

The main goal of the Matchmaking tool development is to propose synergies between participating parties in an Industrial Symbiosis environment. The tool offers two mechanisms for matching involved parties: a "Semantic" and a "Synergy" proposal. The first matching functionality offers searching actors of interest for collaboration by filtering criteria, like country, sector or specific resource. Moreover, the functionality enables querying with text, which essentially searches for relevant terms in the actors' information. The second functionality offers "Synergy" proposal, by matching offers and requests of the actors. The matching is achieved via the material flow knowledge graph, also used in the Industrial Symbiosis tool. The proposed synergies are shown via graph representations, lists or tables. The implementation of the Matchmaking tool can be accessed via a public Github repository: <https://github.com/CLMSUK/DigicircPlatform>.

The Matchmaking tool is a user-friendly tool, which is live and accessible to all the DigiCirc users. A simple and intuitive interface is designed, using the visual identity of the project. The information gathered and presented is defined as a result of a consultation between implementation partners, to ensure needed data for the semantic algorithms to be valuable, yet maintaining a high degree of simplicity.

The tool is tested to ensure ease-of-use and robust performance. This is key to motivate quadruple-helix innovation actors to register, and will thus be tested with innovation actors in DigiCirc partner clusters, integrating feedback for improvement, before it is formally launched.

Finally, the Matchmaking Tool is using semantic models and knowledge graphs to identify complementary partners for beneficiaries, based on their specific needs. This will be leveraged to develop consortia which provide the expertise, equipment, etc. to conduct large scale demonstration.



2 Introduction

An Industrial symbiosis environment is effective when companies work collectively and collaboratively to establish resource and waste exchanges with one another. Towards this direction, the Industrial Symbiosis platform and the Matchmaking tools, developed by DigiCirc, form a collaboration platform that allows companies to simulate and analyze the economic viability of establishing such resource and waste exchanges.

In this deliverable, we describe in detail the developed Matchmaking tool (link: <http://digicirc.clms.io/>), powered by a database engine for waste-to-resource matching. The database engine aims to fill the gap in information and knowledge of what valuable resources can be recovered from wastes or by-products, and what wastes or by-products can be used as substitute raw materials, and what technology or processes are required to make the waste-to-resource conversion happen. The database engine is built on a graph database that models and stores data for resource matching, and comprises a query processor for determining the matches from the stored data. Finally, the database engine is implemented as a web application and demonstrated through use cases.

The main goal of the Matchmaking tool development is to propose synergies between participating parties in an Industrial Symbiosis environment. The tool offers two mechanisms for matching involved parties: a “Semantic” and a “Synergy” proposal. The first matching functionality offers searching actors of interest for collaboration by filtering criteria, like country, sector or specific resource. Moreover, the functionality enables querying with text, which essentially searches for relevant terms in the actors’ information. The second functionality offers “Synergy” proposal, by matching offers and requests of the actors. The matching is achieved via the material flow knowledge graph, also used in the Industrial Symbiosis tool. The proposed synergies are shown via graph representations, lists or tables. The implementation of the Matchmaking tool can be accessed via a public Github repository: <https://github.com/CLMSUK/DigicircPlatform>.

The Matchmaking tool is a user-friendly tool, which is live and accessible to all the DigiCirc users. A simple and intuitive interface is designed, using the visual identity of the project. The information gathered and presented is defined as a result of a consultation between implementation partners, to ensure needed data for the semantic algorithms to be valuable, yet maintaining a high degree of simplicity.

The tool is tested to ensure ease-of-use and robust performance. This is key to motivate quadruple-helix innovation actors to register, and will thus be tested with innovation actors in DigiCirc partner clusters, integrating feedback for improvement, before it is formally launched.

The tool facilitates highly valuable collaborations with complementary quadruple-helix actors, who will provide the expertise and context in which to develop test and fine-tune new approaches to addressing specific problems and challenges, in line with the EC’s demonstrating at a large-scale approach. This includes IT-based SMEs to support digitization where the beneficiary does not have these capacities, thematic SMEs to allow cross-border/-sector scaling, corporates to provide access to market and as a direct client, research groups to provide specialized knowledge and experimentation facilities, and other facilitators (local authorities, industry partners).

The platform will be populated during the Engagement Campaigns, wherein quadruple-helix actors from around the EU will be reached and encouraged to sign-up, with direct support from cluster “nodes”, especially to integrate their own members.



3 Circular Economy & Related work

3.1 Circular Economy

Today, Circular Economy and Industrial Symbiosis environments provide additional means to increase the competitive advantages and the sustainability of enterprises, letting them reduce turnaround times for identifying, qualifying, and contracting other members.

The DIGICIRC environment builds upon existing progress which has been made in Circular Economy and especially Industrial Symbiosis markets. DIGICIRC will not build a marketplace platform, as this is clearly out of the project scope. The project seeks to integrate exchange of materials and resources, to implement the community building, promotion, content management aspects and features, as found in contemporary marketplaces.

The main project differentiator is the assistance, support and guidance it offers to the IS supply chain members with the exploitation of knowledge based synergies identification, and supply – demand matching. From the business perspective, in the Industrial Symbiosis exchanges very important is the introduction of the role of the IS actors/practitioners.

The actors/practitioners are actively involved in the IS workflow, which engages all the partners in the exchange of relevant information. Each stage of the process will be approved according to the policy set by the practitioners and will require agreement from all the parties involved. Also, at each stage the benefits (economic and environmental) will be available to the users in metrics.

DIGICIRC aims to bring together businesses enabling them to perform and create IS relations between the ‘haves’ and the ‘wants’ which in commerce terms boils down to a Sell Side / Buy Side relationship, commonly known as exchanges.

Exchanges solve fundamental and pervasive inefficiencies that hinder trade, such as the fragmentation of buyers and sellers, high search and transaction costs, and limited market information or highly variable demand.

Traditional exchanges (e.g., stock or commodity exchanges) provide liquidity and a standardized process for trading commodity-type goods where long-term, highly integrated relationships are not necessary.

One prevailing exchange format in commerce is the bid/ask exchange model. The bid/ask exchange is well known because this format has been used in the financial and commodity markets for decades. Bid/ask exchanges are simply “double-sided” auctions, which can be described as markets where multiple buyers post offers to purchase (commonly known as “bids”), and multiple sellers post offers to sell (often at a given price, known as the “ask” or “offer”) for identical goods. Essentially, each buyer is hosting a global reverse auction in which sellers compete, and each seller is hosting a forward auction in which buyers compete.

Regarding an Industrial Symbiosis environment, a similar model needs to be adopted. And this model needs to be in line with the market. In general the Industrial Symbiosis benefits for the market are the following :

- Time savings. The workflow and decision support phases of sourcing are reduced. Speeding the sourcing cycle means faster time to value and broader supply chain benefits such as lower inventory carrying costs.
- Cost savings. One of the most direct benefits is the possibility to create Values and win-win benefits from waste, in an environmental friendly collaborative manner. Further, a knowledge-based partnership assisting system adds value by making optimal matching recommendations that incorporate non-price factors and purchase policies in a matter of seconds.



- Smarter management. By monitoring key performance indicators, they can make better decisions regarding overall resources and focus on how purchasing can improve the company's competitive advantage and profitability.
- Improved buyer/supplier relations. With increased flexibility in the sourcing process, buying organizations can articulate needs more clearly, and selling organizations have more options to meet those needs.

When the above benefits are projected to the 'wants' and the 'haves' sides of the business transactions (in market terms buy and sell side), the adoption of this model to the Industrial Symbiosis is feasible.

Next, we present the main benefits of the offers-requests model regarding the market and the Industrial Symbiosis environment.

3.1.1 Buy side ('Wants' or 'Requests') benefits

- Lower Total Cost: Exchange will enable the purchasing organization to manage the total acquisition cost of IS 'waste' materials, creating a procurement price 'mix' of much lower cost.
- Improved Turnaround: Exchange will enhance the time consuming procurement process reducing the amount of time required to receive and analyze bids from a matter of months to a matter of days. Additionally, the negotiation process is being streamlined, saving time and resources.
- Better Information: Exchange will provide insight into true market prices and gives buyers leverage with incumbent, long-term suppliers.
- Increased Resources: Exchange will streamline the process for identifying and qualifying supply sources, promoting opportunities for business advancement.

3.1.2 Sell side ('Haves' or 'Offers') benefits

- Reduced Costs: Exchange will provide a low cost channel for selling goods quickly. Sales travel budgets and meeting costs are drastically reduced or eliminated.
- Lower Mark-ups: Exchange will lower the online operating costs, provide additional selling opportunities to attracting new customers and increasing sales volumes by selling and making profit of a loss.
- Economical Selling: Reduces the time and resources required to attract and retain buyers.
- Superior Market Intelligence: Information about the diverse needs of buying groups can be analyzed assisted by the Knowledge – Based matching process, to provide insights into forming partnerships and further exploiting the IS feedstock.
- Improved Working Capital: Using the Industrial Symbiosis platform to enhance supply chain operation makes the most of excess inventory, resulting in bottom line savings

The Industrial Symbiosis tool is providing the foundations for developing a solution on providing the resources and processes, organized in a knowledge database, that will be then used to populate the offers and requests of the participating actors or entities.

3.2 Related work

Population growth coupled with the rising middle-class and affluent consumers are accelerating the rate of global resource consumption per capita. Due to this, the rate at which the world is generating waste is faster than any other environmental pollutants including greenhouse gases, and at this rate, waste generation is expected to triple



and exceed 11 million tonnes per day by 2100 (D. Hoornweg 2013). This waste situation is especially pressing for densely populated and land-scarce cities. In less than 10 years from 2003 to 2012, global waste generation per capita has increased by more than 87% (P. B.-T. D. Hoornweg 2012). Because of this waste situation, Singapore and governments around the world are seeking better strategies to deal with the escalating cost of waste management.

Industrial symbiosis may be considered as a realization of the circular economy within the industrial landscape. From the technical standpoint, industrial symbiosis is already a viable option for most industries to adopt. In fact, for almost any waste, there are available technologies to convert (recycle) it into a resource. However, there are still barriers which are often of non-technical nature; such as lack of trust and cooperation between participating firms, missing information (knowledge gaps), lack of or rigid environmental regulations, and uneconomic waste-to-resource exchanges which includes end of life processes such as collection, sorting and recycling.

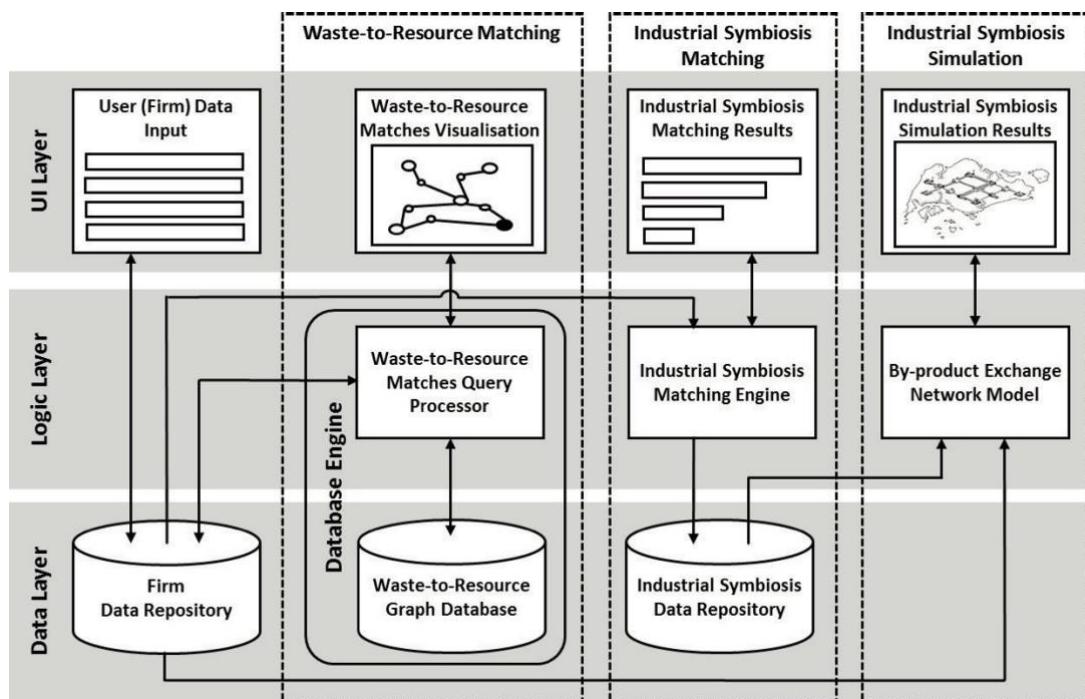


Figure 1: Architecture of Collaboration Platform for Enabling Industrial Symbiosis by J. Low et al. (2018).

To overcome some of these barriers, many different tools have been developed over the recent years, as shown in recent work by (J. Low 2018). In a recent literature survey of information systems facilitating the identification of industrial symbiosis, existing tools were categorized into six different types based on their facilitation approach. They are: 1) open online waste markets – digital environments for firms to find and establish partnerships for waste-to-resource exchanges; 2) facilitated synergy identification systems – intermediaries to actively identify and connect firms to establish waste-to-resource exchanges; 3) industry sector synergy identification – tools to determine potential waste-to- resource exchanges at the industry sector-level; 4) social network platforms and communities – platforms to foster the sharing and exchange of industrial symbiosis knowledge through formation of online common interest groups and communities; 5) industrial symbiosis repositories – central knowledge management systems that organizes and enables the sharing of industrial symbiosis knowledge; and 6) region identification systems for industrial symbiosis –tools for urban planners or policy makers to determine the potential of areas for industrial symbiosis development.

Similarly to the approach of 2) facilitated synergy identification systems, a recent paper (B. Raabe 2017) presented the industrial symbiosis simulation subsystem as part of the work on a collaboration platform for enabling industrial symbiosis. Following this paper, the authors extended the functionality of the collaboration platform by tying in the



D3.7: Matchmaking tool

approach of 5) industrial symbiosis repositories. More specifically, the focus was brought to the waste-to-resource matching subsystem, enabling by a database engine to work together with the rest of the components of the collaboration platform. Figure 1 illustrates the methodology proposed by the authors.

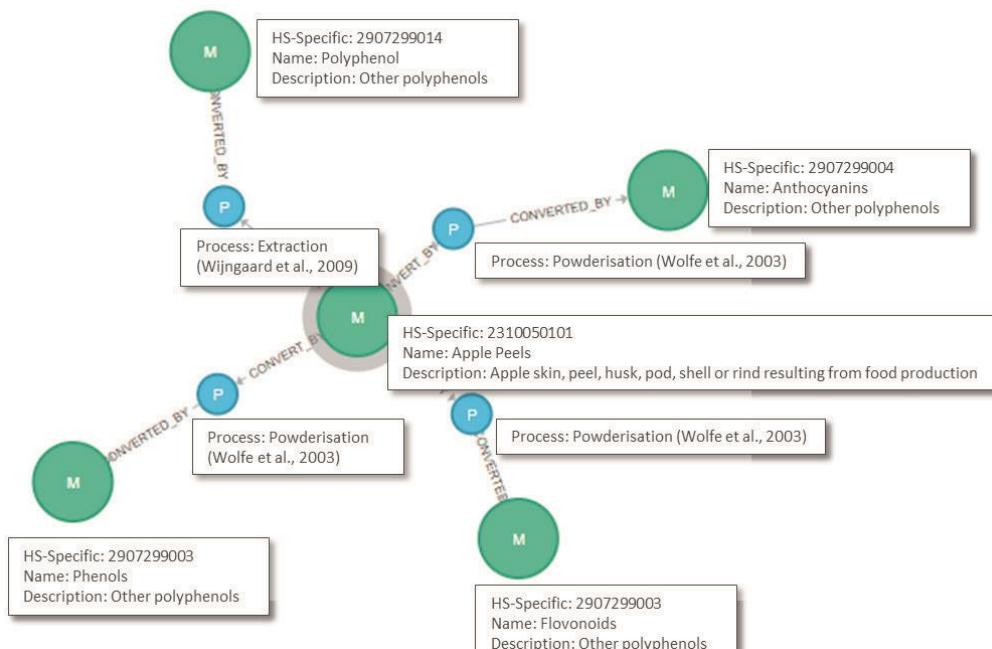


Figure 2: Visualization of possible pathways for converting apple peels into resources, as shown in J. Low et al. (2018).

The ultimate goal of the authors is to design the database engine for waste-to-resource matching for a wide range of waste-related applications. In Figure 2, an example using apple peels, which is a by-product from food production and the firm's waste of interest, is used to illustrate how the waste-to-resource conversion pathways are determined and visualized by the database engine. From the figure, we can see that there are four possible waste-to-resource conversion pathways for apple peels (waste). The resources that can be recovered through these pathways are from the cluster of 'other polyphenols' with specific codes. The visualization and information indicates that through a powderisation process that also involves acid dipping and oven-drying, organic chemical compounds including phenols (used in plastics production and drugs), anthocyanins (used as visual markers for live cell imaging) and flavonoids (used in dietary supplements) can be recovered from the apple peels.

Nevertheless, the last approach does not take under consideration more information regarding wastes and resources on properties like quantity, availability or distance. All this information is crucial for the companies to create a useful Industrial Symbiosis platform and thus a successful symbiotic chain.



4 Operational Characteristics / Requirements for the Matchmaking Tool

4.1 Operational characteristics

The Matchmaking tool integrates a database which keeps all information, a knowledge graph for storing the waste-to-resource relationships and a web tool to visualize it. The tool will suggest matches between companies with regards to circular economy.

More specifically, the matchmaking tool will be a platform that will facilitate collaboration with complementary stakeholders from other sectors, especially when IT SMEs are needed by circular economy vendors.

First, the tool shows all available companies in our circular economy environment. All resources and wastes coming in and going out of a company will be presented, as provided by the companies themselves.

The matching process will also offer possible resource-to-resource or waste-to-resource collaborations, with the following steps:

1. searching the knowledge graph for connecting paths
2. suggest collaborations with companies that have open slots for the searched connecting paths.

Feature Rich Functionality: As an effective solution, the Industrial Symbiosis tool has the ability to create a compelling end-user experience to keep participants returning to the website.

Specific features include innovative, ontology based product category definition, multiple parameters setting, featured items and sales, negotiations, effective product searching, and email alerts.

End-users should also be given extensive account preference and maintenance functions such as self - registration, actions lists, matching suppliers – customers, open deals, IS consulting assistance etc.

Collaboration Management: Rich content aggregated from the tool's participants. Once the content is loaded into the tool, it is immediately available to the entire trading community. Additionally, through the platform will be offered the capability for "real-time" multiple 'Haves'- 'Wants', multi-mode information exchange, planning, and collaboration between trading partners.

Personalized Trading Environment: the Industrial Symbiosis tool's participants may assume the role of buyer ('Wants'), seller ('Haves') or both, and indicate their preference for goods and services, price, delivery, quality, messaging, payment terms, and conditions. This is an opportunity for content personalization from all available sources, and also to tailor collaborative business process unique to its trading partner. When a new member is initiated, 'haves' or 'wants' are automatically indicated to participate using their business profiles.

Matching 'Wants' or 'Requests' (Buyers) and 'Haves' or 'Offers' (Sellers): The core of the Industrial Symbiosis platform functionality is the knowledge-graph based support in the formation and management of symbiotic relationships. Using sophisticated search algorithms, the matching function is enhanced with the platform's ontologies and knowledge graphs.

Symbiotic Channels Creation: Companies collaborating with others are provided with tools to publicize a comprehensive view of their business, and greater insights into their business needs and processes to create Symbiotic market synergies, make better decisions, and improve business operations as regards the procurement and sourcing functions.



4.2 User Requirements

Next, we present the user requirements, as declared in deliverable 3.1. The user needs helped the design of the Matchmaking tool in a significant way.

4.2.1 Login/Register of User

- 1) As a public user, I want to be able to access the platform without registering.
- 2) As a public user, I want to be able to register to the platform and provide my personal information.
- 3) During registration it will be mandatory for the user to complete the following information:
 - a) First name
 - b) Last name
 - c) Email address
 - d) Password
 - e) Confirm password and agree with the Terms & Conditions Privacy Policy
- 4) As a registered user, I want to be able to log in/out of the platform.
- 5) As a registered user, I want to be able to edit/delete my account.

4.2.2 Notifications

As a registered user, I want to receive email and in-app notifications.

Notifications about:

- product availability after a request has been made
 - newcomers on specific thematic areas interest in my product or service
- As a registered user, I want to be able to select what notifications I should receive.

As a registered user, I want to be able to view the status of the matching process.

4.2.3 Search

- 1) Display and search registered companies by their type, resource and location.
- 2) Display requests for services from universities and research institutes.
- 3) The user can send a search resource using a text term. The terms are constrained using common vocabulary (an ontology).
 - a) Fixed queries (a user may submit a fixed query for certain resources and been informed daily i.e. by mail
 - b) Temporary queries
- 4) The user can filter the top retrieved matches
 - a) by country
 - b) by region
 - c) by distance
- 5) Display the availability of the resources in the platform.



4.2.4 Management

1. The Administrator should be able to update the Parameters (Countries, Entity Types, Sectors, Expertise and Thematic Expertise, etc.).
2. The Administrator and Monitor User should be able to view reports.

4.2.5 Registration of actors

1. The Actor Manager should be able to register to the platform.
2. Actors/Companies can be added only from registered users.
3. Actor Manager can choose if he like to receive emails from the platform
4. The Actor Manager must be able to accept the usage of cookies.

4.2.6 View actor/company page

1. All Users should be able to view actor/company information.

4.2.7 Add new actor

1. The Actor Manager should be able to add a new actor
2. The Actor Manager should be able to assign an actor to a cluster.

4.3 Matchmaking requirements

Next, we provide requirements concerning the matchmaking process that may lead to synergies or not.

4.3.1.1 Matchmaking leading to synergies

1. Perform manual search for resources or technologies of other Members
2. Ability to specify search parameters such as resource type, technology type, region, timescale, quantity, and cost
3. The automatic synergy identification should provide relevant/accurate possible synergies
4. Synergies proposed to a Member should provide sufficient information for the Member to decide if the synergy is worth pursuing
5. The matchmaking algorithms for automatic synergy identification should be fast
6. The matchmaking algorithms should take into account parameters such as geographical location, availability of the resources, environmental factors, and economic issues

4.3.1.2 Synergy tracking

1. Ability to record finalization of a synergy stage
2. Ability to record status of a current synergy stage



D3.7: Matchmaking tool

3. Ability to record problem encountered at a current synergy stage
4. Ability to record reason for an unsuccessful synergy
5. Recorded problem should be flagged as “demands attention”
6. The system should issue reminders for updating the status of a synergy in progress
7. The system should present the time of inactivity for a synergy in progress
8. Ability to define the limit of time of inactivity for a synergy in progress, if time is exceeded the synergy should be flagged as “demands attention”
9. Easy to fill in information concerning the measured outputs of a successful synergy in the synergy reporting form
10. Clear explanation for every field of every synergy reporting form that needs to be filled in (online help or off-line document)

4.3.1.3 Matching process

- 1) As a registered user, I want to be able to receive a list of possible parties for creating a synergy.
- 2) The user should be presented with a list of the top k most relevant results.
- 3) As a registered user, I want to be able to select the parties from the produced list in order to receive offers by them.
- 4) The user should be able to select the users that they want to receive offers from.
- 5) As a registered user, I want to be able to receive an invitation to submit an offer.
- 6) As a registered user, I want to be able to select if I will manually choose the partners who submit offers, or if it will happen dynamically from the top k parties.
- 7) As a registered user, I want personal information to be shared with other parties only after a match is made.



5 The Matchmaking Tool

5.1 Development platform

The platform for designing and implementing all the functionalities of the Industrial Symbiosis tool is ZappDev¹. 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.

5.1.1 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

5.1.2 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
- Maps
- Calendars

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

¹ <http://www.zappdev.com/>



D3.7: Matchmaking tool

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

5.1.3 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

5.1.4 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



5.1.5 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

5.1.6 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

5.1.7 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.



5.1.8 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.

5.2 System architecture

Figure 3 shows the architectural diagram of the main subsystems of DigiCirc. First, the data input, where a user is requested to add company information. The data are then stored to data repositories, which are used for creating our ontologies and knowledge graphs used for our industrial symbiosis environment. Afterwards, the Industrial Symbiosis platform gives access to these ontologies and knowledge graphs via web services. Finally, the Matchmaking tool, with the help of the ontologies and the knowledge graphs, suggests collaborations between entities.

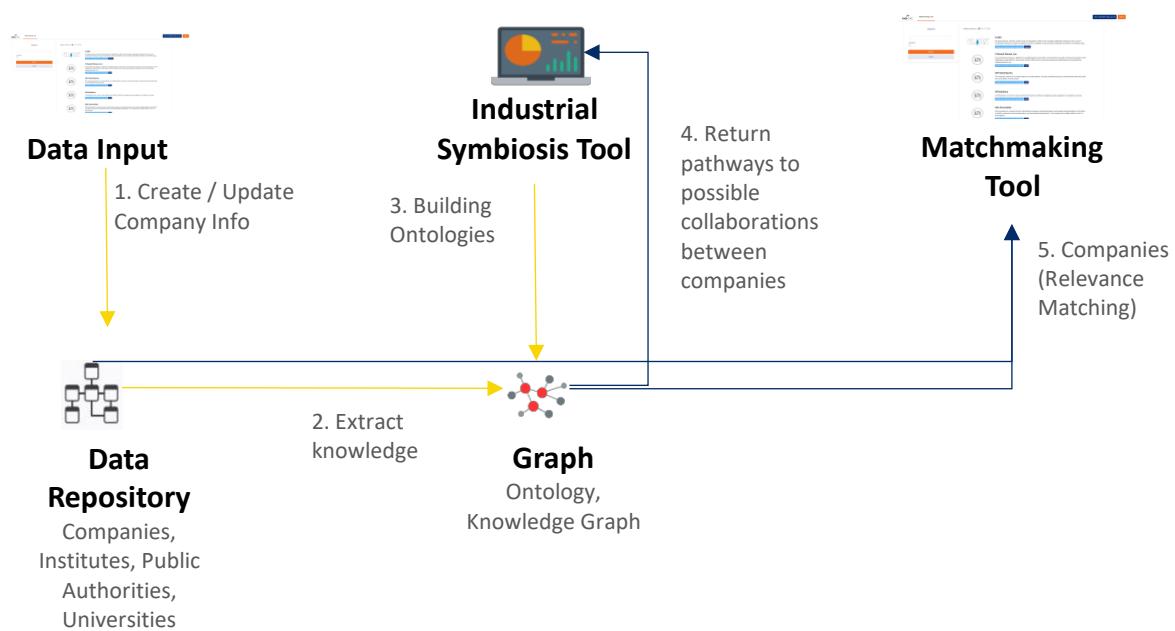


Figure 3: The DigiCirc platform architecture, with regards to the Matchmaking tool. The Matchmaking tool is the last component.

5.3 Components

5.3.1 User Interface



The User Interface subsystem is the human-computer point of interaction, where all information gathered will be displayed to users. A user-friendly interface should be easy and efficient to operate at the lowest effort on the part of the operator. The interface will contain:

- User's registration and login forms
- User's profiles forms (fill-in forms using pop-down menus)
- Role of user
- Required material
- By-products
- Supply of material
- Related materials

5.3.2 Data layer

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 data are provided by the users of the Industrial Symbiosis and the Matchmaking tools. All this data may include information about companies or organizations, waste, and resources availabilities by the entities, as well as processes and technologies offered.

The database subsystem is used to store all the data of the system. The database considered in order to fully meet the requirements is MariaDB². 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.

The Matchmaking tool will be based on a knowledge database of the industrial residues resources, to support experts and technicians of the companies to find a better use of materials through a material matching tool. The Matchmaking 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

The key factors of a partnership are based on:

² <https://mariadb.org/>



- The composition of resources (the same waste resources may have different composition)
- The demands of the end-users
- The data of the waste material
- The quantity of the supplied data
- The quality of the supplied waste
- The market specifications and regulations

5.3.3 Ontologies

A very important functionality of Matchmaking tool based on ontologies is the actual process of matching. The created user profiles (service description ontology) are being used by the match-maker in order to identify all the possible matches between users. The service description consists of properties and concepts that are important for the process of industrial symbiosis, such as the type of waste on offer (or on demand), the quantity, and the time availability, the physical form of the waste and the location of the user.

When a user requests a service, the user profile is matched against existing profiles of other users in an effort to find possible matches (synergies). Before that however, a preliminary elimination will take place based on the certain criteria (such as the type of the user) in order to speed up the matching process.

The matches are made against certain criteria that are considered important for the industrial symbiosis process. At this stage, the system establishes all the potential symbiotic synergies. The ontology driven conceptual model consists of two ontology levels:

- The first level is Domain Ontologies that include ontologies for User, Resource, and Technology.
- The second level linked to the domain level includes Application Specific Ontologies.

5.3.4 Requests and Offers

As presented in the Industrial Symbiosis deliverable D3.6, one of the most important components of the industrial symbiosis platform is enabling the companies or organizations to perform requests and offers of resources. A request or an offer of specific resources is accompanied with metadata which could include information about restrictions on the action of interest. These restrictions may regard quantities, availability, distances, and other additional information. The restrictions, which act as properties on offers and requests, are crucial for the following matching process. The key idea is that our matchmaking tool does not only match resources, but offers and requests on these resources with respect to the additional information like quantity, availability, distance and other.

Figure 4 presents a schematic illustration of an organization's offers and requests.



D3.7: Matchmaking tool

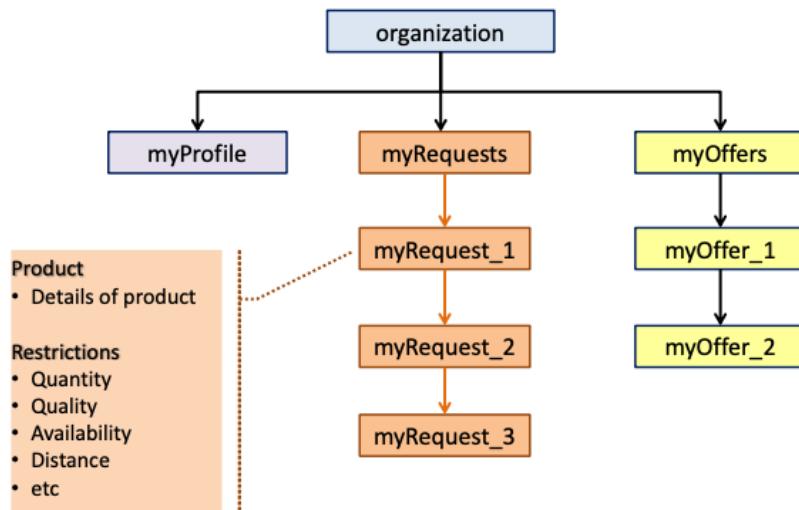


Figure 4: an illustration of an organizations' requests and offers.

5.3.5 Matchmaking engine

In the logic layer reside the algorithms that processes and analyses the information from the data layer and the results of the analysis are presented to the user through the UI layer. Based on data of the participating companies the matching tool attempts to match a company with other participating companies on the platform. Finally using the matching results, a company is able to evaluate the economic sustainability in establishing collaboration with other participating members on the platform.

Regarding the first matchmaking functionality, which is searching by text, Elasticsearch³ has been utilized. Elasticsearch is a search engine based on the Lucene library. It provides a distributed, multitenant-capable full-text search engine with an HTTP web interface and schema-free JSON documents. The text search component integrates a Similarity module. The similarity module (scoring / ranking model) defines how matching documents are scored. Here, the default similarity is used, BM25. In information retrieval, **Okapi BM25** (*BM* is an abbreviation of *best matching*) is a ranking function used by search engines to estimate the relevance of documents to a given search query. It is based on the probabilistic retrieval framework (Robertson, et al. 1994). The fuller name, *Okapi BM25*, includes the name of the first system to use it, which was the Okapi information retrieval system, implemented at London's City University in the 1980s and 1990s. BM25 and its newer variants, e.g. BM25F (a version of BM25 that can take document structure and anchor text into account), represent state-of-the-art TF-IDF-like retrieval functions used in document retrieval.

The main issues in matching are how to represent residue supplies and requests, and how to calculate the similarity value between offers and request. The supplies or request can represent data or services by using a model of representation. The matching function will be based on a fusion between lexical and semantic similarity. The output of the matching tool will be a list of the players in the specific domain, i.e. suppliers of waste material, users of recovered materials sorted by their location (nearest first). The matching will be based on: the material needed or offered, the quantity, quality parameters and location.

We developed a web based application where actors like SMEs, Research Institutes, Universities, public authorities and sole traders are collaborated on the web by exchanging products and resources and generally services. The

³ <https://www.elastic.co/elasticsearch/service>



D3.7: Matchmaking tool

task of the matching engine is to use the semantic understanding of offer or request and to recognize their degree of mismatch and retrieve the offers of services that more closely match the request. Similarity of matching means that the offer is of some use for the requester. The outputs of the offer match the inputs of the request.

The main control loop of the algorithm in python notation is described by:

```
request:  
    list=[]  
    for offer in offers:  
        list.append([sim(req, offer), request])  
  
    sort(list, 'descend')  
return rank list of the top-k offers
```

In this section we present the structure of the profiles of the participated actors and the specification of a matching algorithm between service providers and service requests. This is a challenging task because there is no prior knowledge between suppliers and buyers. They do not have the same perception and knowledge about the same service and therefore can describe a request or an offer in such a way that doesn't meet the needs of the other. Therefore, in addition to a lexical similarity with a degree of morphological processing, a semantic matching is required. For Semantic Matching we need to know the relations between the concepts in order to support reasoning, something which can be achieved through the use of concept ontologies.

In Figure 5 we demonstrate the actors' profiles and the functionalities of their services. The part below the blue line of the figure presents the definition of an actor. It records the basic information about the provider or requester of a service. Above the blue line we have contains the type and description of the service. These information are used by the matchmaking engine. Other functional attributes and information include availability, quantity, quality, and radius of business.



D3.7: Matchmaking tool

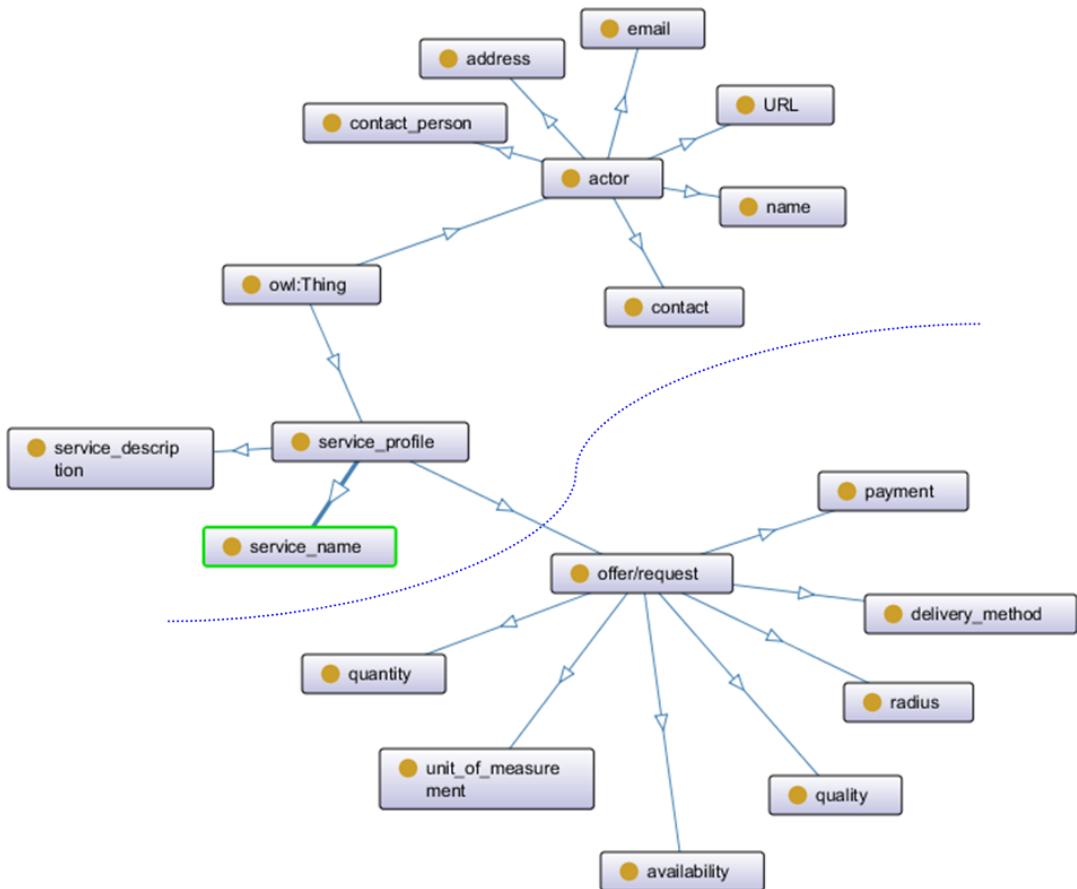


Figure 5: actor and service profile.

An offer matches a request when the advertisement describes a service that is sufficiently similar to the service requested. An offer is considered sufficiently similar when they describe the same service. The final decision depends on the restriction criteria. Similarity is a critical issue in the matching process. If we demand high precision we have more false positive and false negative results. In contrast if we increase the flexibility of the matching engine we get the opposite effect: the false negative are reduced at the expense of the increase of the false positives.

In our matching we support flexible semantic matching between offers and requests on the basis of our ontology available to the matching engine.

Semantic score is estimated by augmenting the request/offer with related terms on the resources ontology (broader, narrow terms or synonyms).

A request can be instant or fixed. In the former case we submit a request and get the results on-line. In the latter case the request is periodically run for new matches between in case there are new offers from the participating entities. The top-k matches of the request are stored in the database and they are compared with the new matches returned by the matchmaking tool. Only the new matches not found in the database, are sent to the user.

5.3.5.1 Matching with search query (text)

The main goal of this additional search component is to help the user query the database in an easier and effective way and find matches via smart text queries. Currently, the portal allows text search within a company's description. The user should type a word or sentences that is identical or similar with existing words in the database description. Thus, a user should match the exact words or sentences in the company's description.



D3.7: Matchmaking tool

We developed a smarter text similarity functionality, incorporating novel Artificial Intelligence (AI) and Natural Language Processing (NLP) tools, aiming to help the user search more effectively and more efficiently.

Keyword extraction

In many cases the companies' descriptions may consist of large textual data. The text may include stop-words, connecting verbs or adverbs that do not carry any valuable information about a company's identity. For this reason, we employ a keyword extraction method (Rose 2010). More specifically, we wish to extract only important keywords that carry useful semantic value. The keywords will then be added as an extra value in the respective column of the company database. We use a freely available Python library⁴.

5.3.5.2 Matching offers with requests

Based on data of the participating companies the matching tool attempts to match a company with other participating companies on the platform. Finally using the matching results, a company is able to evaluate the economic sustainability in establishing collaboration with other participating members on the platform.

Figure 4 presents a schematic illustration of an organization's offers and requests.

The main issues in matching are how to represent residue supplies and requests, and how to calculate the similarity value between offers and request. The supplies or request can represent data or services by using a model of representation. The matching function is based on matching these offers and requests by first using the industrial symbiosis knowledge graph to match the resources. Matching the resources is essentially exploring the paths in the knowledge graph, and see if there is an existing path between the two resources that connects them either directly or via other resources and processes. The output is visualized as graph of connected nodes that represent the actors and resources, while edges represent the connections between them. The output of the matching tool can also be shown as a list of the matching in the specific domain, i.e. suppliers of waste material, users of recovered materials sorted by their location (nearest first). To sum up, the matching process which is shown in Figure 6**Error! Reference source not found.**, is based on: the material needed or offered, the quantity, quality parameters and location.

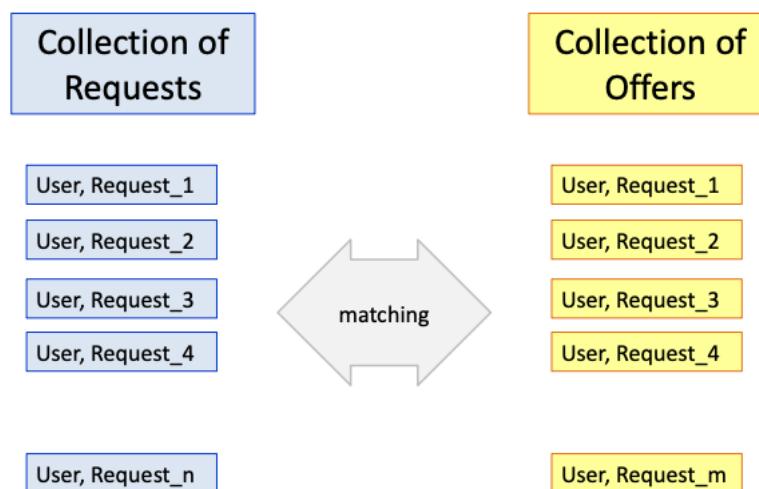


Figure 6: an illustration of the matching process between offers and requests.

⁴ <https://github.com/csurfer/rake-nltk> (Mikolov 2013)

D3.7: Matchmaking tool

Each entity, either a company or an organization, may offer or request specific resources. All this information is stored in the database. After the Matchmaking tool suggest matches between offers and requests, the matches can also be stored in the database for the interested entity. An illustration of an organization's or company's requests, offers and matches is shown in Figure 7.

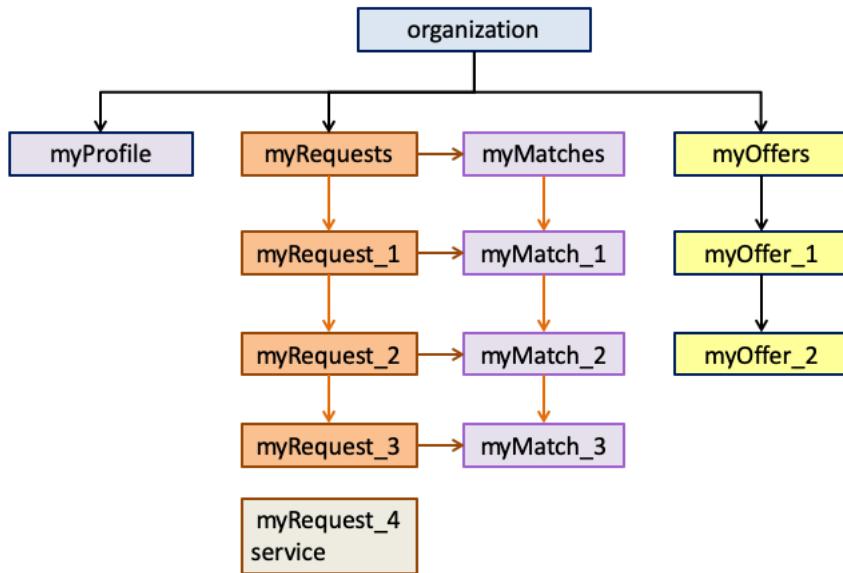


Figure 7: an illustration of an organization's or company's requests, offers and matches.

An illustration of the workflow of the matchmaking process is shown in Figure 8. User 1, a registered user with a connected actor (company or organization) may look for synergies, by either observing and searching the knowledge graph for specific resources and the connected actors or by creating a request for a specific material. With the first method, User 1 will see the actors connected to the resource and communicate directly with User 2, which offers the specific material. Otherwise, with the second method an automated matching will happen between the request of User 1 and the offer of User 2 in the same material.

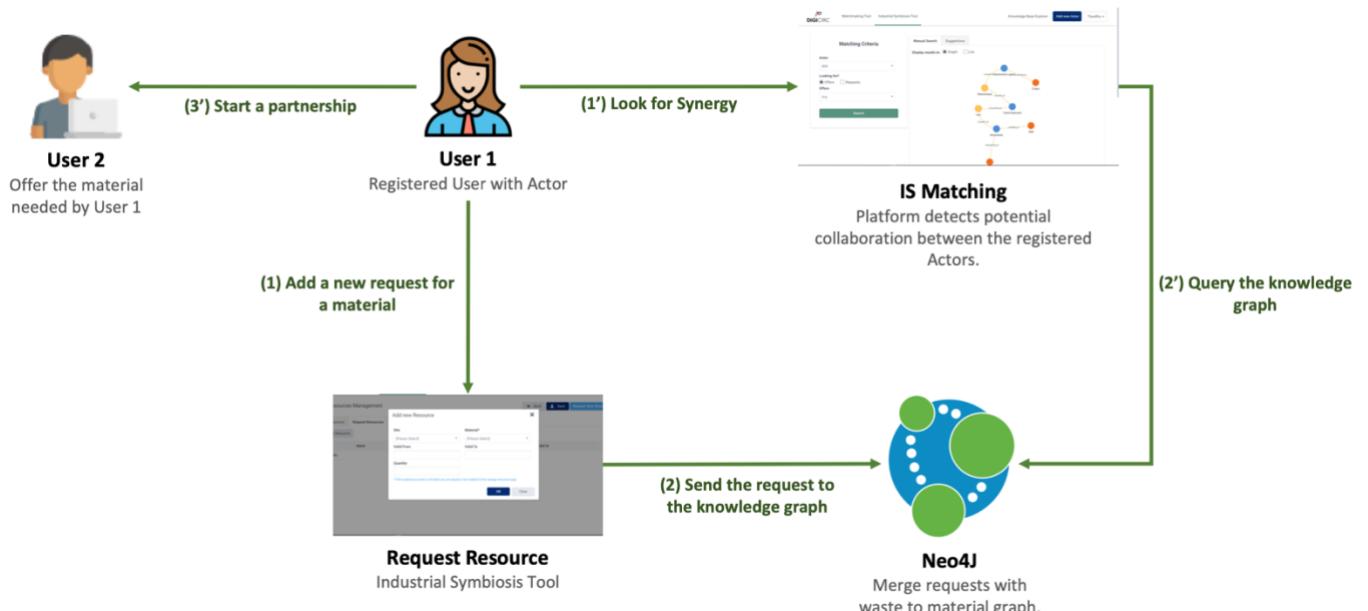


Figure 8: matching offers and requests via the knowledge graph.



D3.7: Matchmaking tool

The Matchmaking tool can periodically check for new matches between offers and requests of the participating entities. All matches are stored in the database, and are compared with the new matches returned by the matchmaking tool. Only the new matches not found in the database, will be shown to the user. The process of storing and comparing the new matches is illustrated briefly in Figure 9.

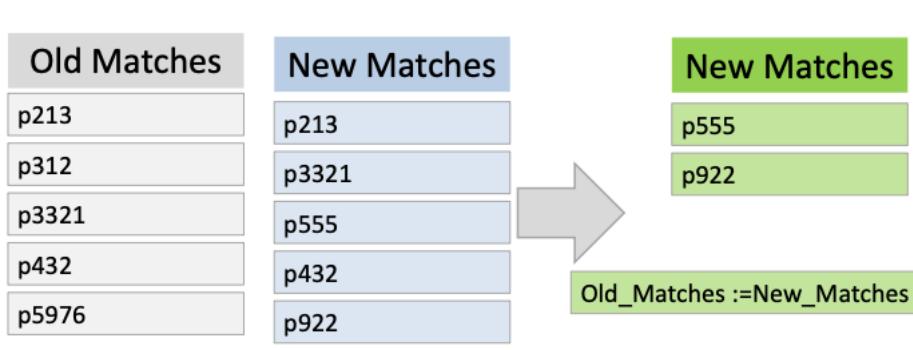


Figure 9: old and new matches stored at the database.

Having described all the necessary components, next we present the full workflow of the Industrial Symbiosis and the Matchmaking tools. A user first registers his/her company in the system as a new actor. The actor can first query the database for other actors of interest by text search. Alternatively, the actor creates offers or requests of specific resources, followed by specific restrictions on quantity, distance etc. The Matchmaking tool then receives this information and tries to match it with relevant offers or requests. The Matchmaking tool detects potential collaboration paths by exploiting the material flow knowledge graph and returns it to the interface. The complete workflow is illustrated in Figure 10.

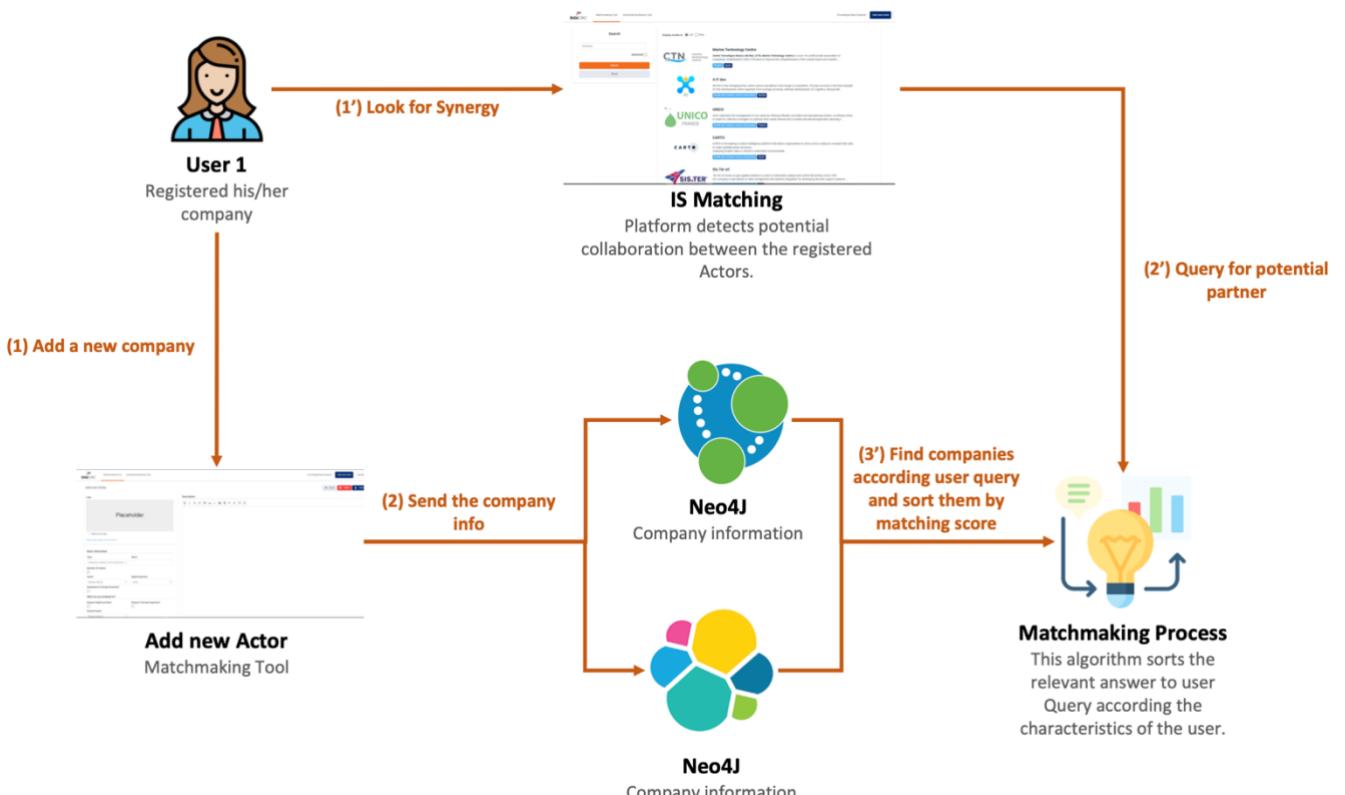


Figure 10: the full workflow of the Industrial Symbiosis and Matchmaking tools.



5.4 Users of the Matchmaking Tool

The Users of the Matchmaking 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 our site and don't need to register 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 a resource, and submit requests or offer. They can submit an instant request or a fixed request. A fixed request returns the list of the most relevant companies and then informs the user whenever there is a new company that has entered the system and matches its enquiring. It could initialize a proposal for collaboration by either selecting a company from the list returned from the Matchmaking Tool, of top most relevant matches, or they can select manually a company they desire from the list of companies.
- **Partners.** They are users of the Matchmaking Tool. They are registered users with additional rights as soon as a synergistic 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 synergistic networks; they will monitor 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.

5.5 Workflow

The Matchmaking workflow will follow the six key stages:

1. Registration of users and actors
2. Searching for synergies via text matching
3. Synergy Identification via automated ontology based semantic web service and matchmaking algorithms



D3.7: Matchmaking tool

4. Ranking and verifying suggested synergies
5. Synergy Reporting (success or failures)
6. Case Study Production



6 Web platform functionalities

The starting page of the Matchmaking tool is illustrated in Figure 11. The illustrated figure is showing the public page, without any logged in users. In the top bar, you can see the links for the Industrial Symbiosis tool, Sign up and Sign In functionalities. The Sign up and Sign In functionalities are described in detail below.

Figure 11: the starting page of the Matchmaking tool.

Main Functionalities:

-  **Registration of users and actors:** the tool offers registration of the users and the associated actors (companies or organizations).
-  **Search:** the matchmaking tool offers searching the DigiCirc's databases via matching the queries performed by a user.
-  **Matchmaking:** the Matchmaking tool performs automatic matching of the resources and technology providers and match ranking according to the semantic relevance. Matching also uses ranking based on economic and environmental relevance with priorities set by the IS practitioners.

6.1 User registration and login

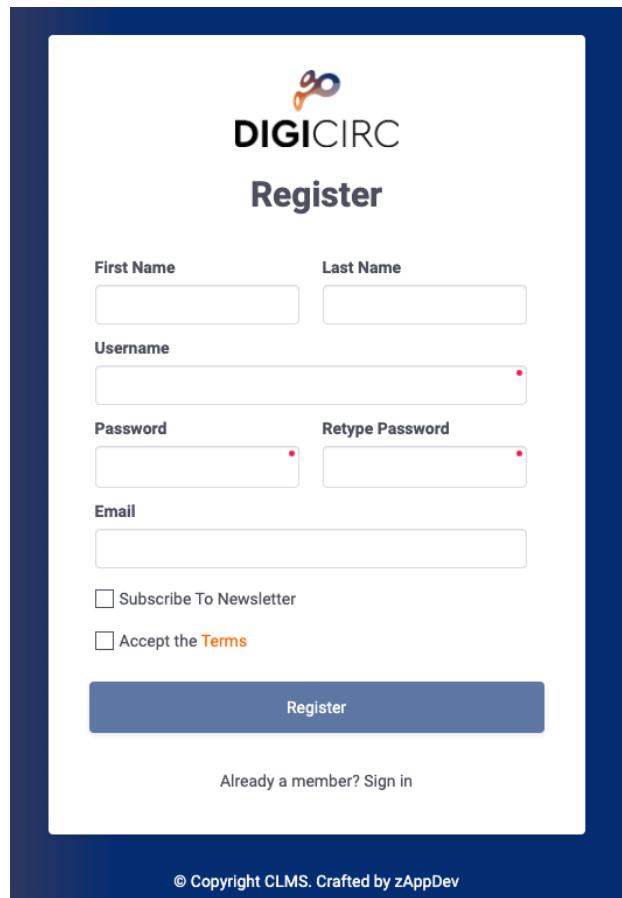
Users will register themselves with the system. This will require a custom form to be developed that requests appropriate information (name, email, etc.), as shown in Figure 12.

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



D3.7: Matchmaking tool

If a user has already an account, he/she can login to the platform by using the login component, presented in Figure 13.



The image shows the registration page for the DIGICIRC platform. The page has a white background with a dark blue header and footer. At the top center is the DIGICIRC logo, which consists of a stylized orange and red 'G' shape above the word "DIGICIRC". Below the logo, the word "Register" is centered in a large, bold, dark gray font. The form fields are arranged in two columns: "First Name" and "Last Name" (each with a single input field), "Username" (with a single input field), "Password" and "Retype Password" (each with a single input field), and "Email" (with a single input field). Below these fields are two checkboxes: "Subscribe To Newsletter" and "Accept the Terms". At the bottom of the form is a large, dark blue "Register" button. In the footer, there is a link "Already a member? Sign in" and a copyright notice: "© Copyright CLMS. Crafted by zAppDev".

Figure 12: registration of user.



D3.7: Matchmaking tool

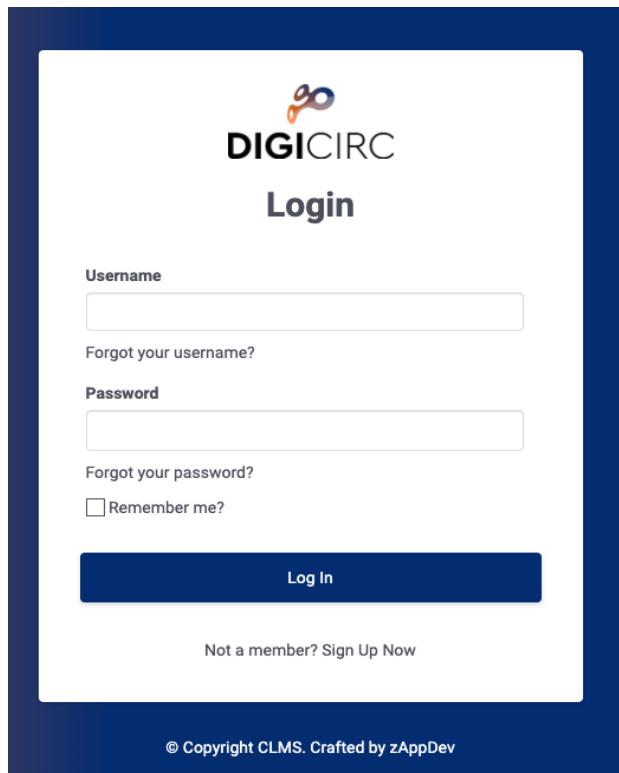


Figure 13: user login UI component.

6.2 Navigation and Search tool

The starting page of the Matchmaking tool after a standard user logs in, presents a list of all available actors-companies. The starting page is shown in Figure 14.

Figure 14: the starting page of the matchmaking tool, after logging in as a standard user.

The Matchmaking tool offers a navigation and search functionality for identifying and looking at companies or organizations of interest. The functionality is presented at the very first level of the tool, when a user accesses the platform. The starting web page of the Matchmaking tool is shown in Figure 14.



D3.7: Matchmaking tool

Furthermore, the user can search for companies of interest by utilizing the Search functionality, shown in the left part of Figure 15. The search is powered by AI and NLP tools, for searching relevant words with an actor's description or associated keywords.

The screenshot shows the DIGICIRC Matchmaking tool. At the top, there is a navigation bar with the DIGICIRC logo, followed by three tabs: 'Matchmaking' (which is active, indicated by an orange underline), 'Semantic', and 'Synergy'. Below the navigation bar is a section titled 'Matching Criteria' containing three dropdown menus: 'Country' (set to 'Any'), 'Sector' (set to 'Any'), and 'Resource' (set to 'Any'). To the right of these dropdowns is a 'Matchmaking' button with a magnifying glass icon. Below the dropdowns is a large, empty input field. At the bottom of the section are two buttons: an orange 'Search' button and a grey 'Reset' button.

Figure 15: matching with options on country, sector, resources or text.

The user has the option of navigating through all the companies and organizations participating in the DigiCirc project. The results can be either presented as list or as points in a real map frame (Figure 16), and each map point can be clicked for further information. Each presented entity comes with a company image (or logo), a small description, as well as other information like the nature of the company and the country of origin.



D3.7: Matchmaking tool



Figure 16: entities of the platform shown in a map frame.

The entity of interest can be selected by the user, and a new window is created, presenting all the information regarding the specific entity, as shown in Figure 17. All available information is offered to the user, like expertise, thematic areas, services and the official link.

About us

Created in 2006 as a non-profit organisation, Cap Digital is today recognized as the biggest cluster in Europe and one of the largest innovators' collective in the digital ecosystem. Since 2014, we have been certified as 'Cluster of Excellence' by the European Commission (Gold Label).

Figure 17: an example of an entity in the matchmaking tool.

6.3 Add new entity-actor

The user is able to add new entities (actors), basic information about them like name, type of entity, sector, digital expertise, logo and others. The functionality is shown in Figure 18.



D3.7: Matchmaking tool

The screenshot shows the DIGICIRC Matchmaking tool interface. At the top, there are navigation links: Matchmaking, Semantic, Synergy, Industrial Symbiosis, Add new Actor, and Theofilis. Below this, a header bar includes 'Add new Entity' on the left and 'Back', 'Delete', and 'Save' buttons on the right.

The main form area has two sections:

- Logo:** A placeholder image with a 'Placeholder' label. Below it is a button to 'Select your logo' and a note: 'Please choose images with size 256x96'.
- Description:** A rich text editor toolbar with various formatting options like bold (B), italic (I), underline (U), etc. Below the toolbar is a large text input field.

Basic Information: This section contains dropdown menus for Type (Cluster), Name, Sector ([Please Select]), Digital Expertise (None), and a checkbox for Experience in Circular Economy?

Figure 18: adding new entities or actors via the platform.

6.4 Offers and requests

The user can also create offers and requests of wanted resources via the DigiCirc Industrial Symbiosis platform. For a registered user, the functionality is accessed by clicking first on the profile of their company (actor) and afterwards clicking on the “Resources” button, shown in Figure 19.

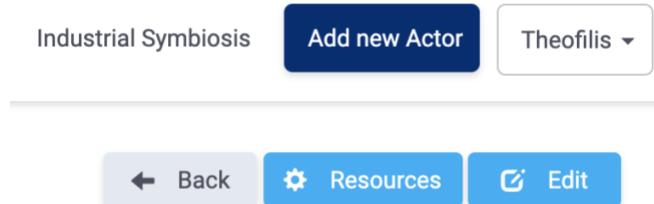


Figure 19: accessing the offers and requests components by clicking on the Resources button.

The user is able to add new offerings of resources (materials or services) and requests of resources, as shown in Figure 20.



D3.7: Matchmaking tool

Site	Name	Quantity	Unit Of Measurement	Valid From	Valid To
No records.					

Figure 20: the UI component responsible for adding new offers or requests of resources.

By clicking on the “Add New Resources”, a pop-up windows is created and the user is called to complete a form for the new offer or request of resources, as shown in Figure 21.

Add new Resource ×

Site	Material*
<input type="text" value="[Please Select]"/>	<input type="text" value="[Please Select]"/>
Valid From	Valid To
<input type="text"/>	<input type="text"/>
Quantity	
<input type="text"/>	

* If the material you want is not listed, you can request a new material in the manage resources page.

OK **Clear**

Figure 21: adding an offer or request of resources.

The offers and requests are then utilized by the Matchmaking tool to produce matches between them.

6.5 “Synergy” suggestion: Matching via paths in the knowledge graph

As mentioned previously, the Matchmaking tool offers two matching functionalities: “Semantic” and “Synergy”. The two options are shown in the top bar of the web tool. Here, we present the “Synergy” matching functionality.

Via the Matchmaking tool, the actors first register offers and requests of specific resources. The “Synergy” matching attempts to connect them by finding paths in the Industrial Symbiosis knowledge graph, as shown in Figure 22. The “Synergy” matching is essentially trying to identify paths that connected requested and offered resources by the participating actors. If this path exists, a synergy proposal is made.



D3.7: Matchmaking tool

The screenshot shows the DIGICIRC Matchmaking tool interface. On the left, there is a sidebar titled "Matching Criteria" with fields for "Actor" (set to "CLMS"), "Looking for?" (checkboxes for "Offers" and "Requests" are checked), and "Offers" (dropdown set to "Any"). An orange "Search" button is present. To the right, a knowledge graph is displayed with four nodes: "Used Monitor" (blue), "CLMS" (orange), "Used Laptop" (blue), and "GreenAI" (orange). Edges between them are labeled: "Used Monitor" has an "OFFERED_BY" edge to "CLMS" and a "REQUESTED_BY" edge to "Used Laptop"; "Used Laptop" has an "OFFERED_BY" edge to "CLMS" and a "REQUESTED_BY" edge to "GreenAI". The graph is visualized on a light gray background with a white header bar containing tabs for "Knowledge Matching" and "Suggestions", and a sub-header "Display results in: Graph" (selected) or "List".

Figure 22: a synergy proposal by the Matchmaking tool, visualized as a graph.

The matches, apart from being sent to the interested parties, can be visualized via the industrial symbiosis knowledge graph as paths. As presented in Figure 22, orange nodes represent entities, blue nodes represent resources and yellow nodes represent processes or technologies to deal with resources. The edges represent actions regarding either actors or resources. For example, some yellow edges which are labelled “OFFERED_BY” are connecting entities which offer specific resources. In addition, other yellow edges, labelled “CONVERT_BY” and “CONVERTED_BY”, connect resources that can be converted via specific processes or technologies.

The matching results can be shown additionally as a list of the involved parties, as shown Figure 23. The list presents the other parties, except our actor. In this case, only the actor “GreenAI” is shown. Our actor “CLMS” is not shown here, as this functionality is designed to only present the other ends of the proposed synergies, which are the actors of interest.

The screenshot shows the Matchmaking tool displaying a list of matching results. At the top, there are tabs for "Knowledge Matching" (selected) and "Suggestions", and a "Display results in: List" button. Below this, a section for "GreenAI" is shown, featuring a camera icon with a slash, indicating no offers or requests. A text block provides a placeholder text: "Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent ut bibendum nulla. Sed a convallis dui, sed fringilla velit. Sed porta, lorem in finibus vulputate, lorem justo eleifend turpis, ut cursus neque leo vitae odio. Quisqu...". Below this, two blue buttons show "Small and medium-sized enterprises" and "Greece". At the bottom, a message "1 Results Found" is displayed.

Figure 23: the matching results returned as a list of the involved actors.

Moreover, the “Synergy” functionality can output the matching results as a table of the proposed synergies in the following format, as shown in Figure 24.



D3.7: Matchmaking tool

The screenshot shows a user interface for a matchmaking tool. At the top, there are two tabs: "Knowledge Matching" and "Suggestions", with "Suggestions" being the active tab. Below the tabs is a search bar with a magnifying glass icon and a dropdown arrow. To the right of the search bar are several small icons for filtering or sorting. A "Page size" dropdown menu is set to "20 records". The main area displays a table with the following data:

	Material	Offered By	Requested By	Valid From	Valid To	Quantity Lack
1	Used Monitor	CLMS	GreenAI	01/02/2021	31/03/2021	0
2	Used Laptop	CLMS	GreenAI	02/02/2021	28/02/2021	1

Figure 24: the matching results can also be shown as a table of suggestions.

The results can be then exported to different formats like Excel, PDF or CSV, offering also many options for the resulted template, as shown in Figure 25.

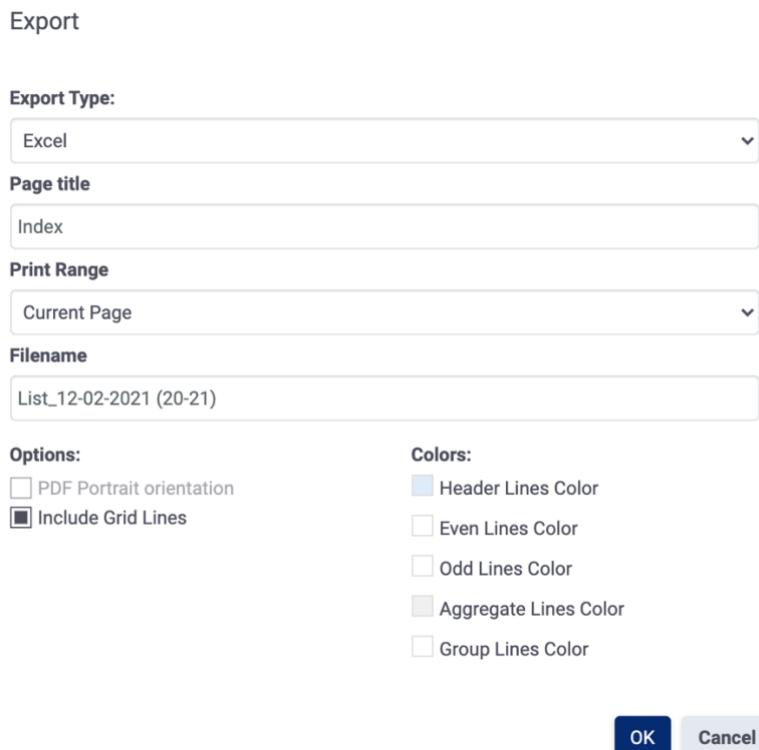


Figure 25: the proposed synergies can be exported to files.

6.6 Matching and ranking synergies

For example, as shown in Figure 26, the actor “CLMS” is offering a specific resource: keyboards. This resource is either available directly by “Company 5”, or via longer paths in the knowledge graph.

The matching can be presented as a ranked list for example, with regards to the path length or the sum of intermediate steps, including resources and processes:



D3.7: Matchmaking tool

1. 3 Round Stones; Inc. (directly)
2. Draxis, which has requested "Refurbished laptops", but may be interested to use spare parts like "Used keyboards"

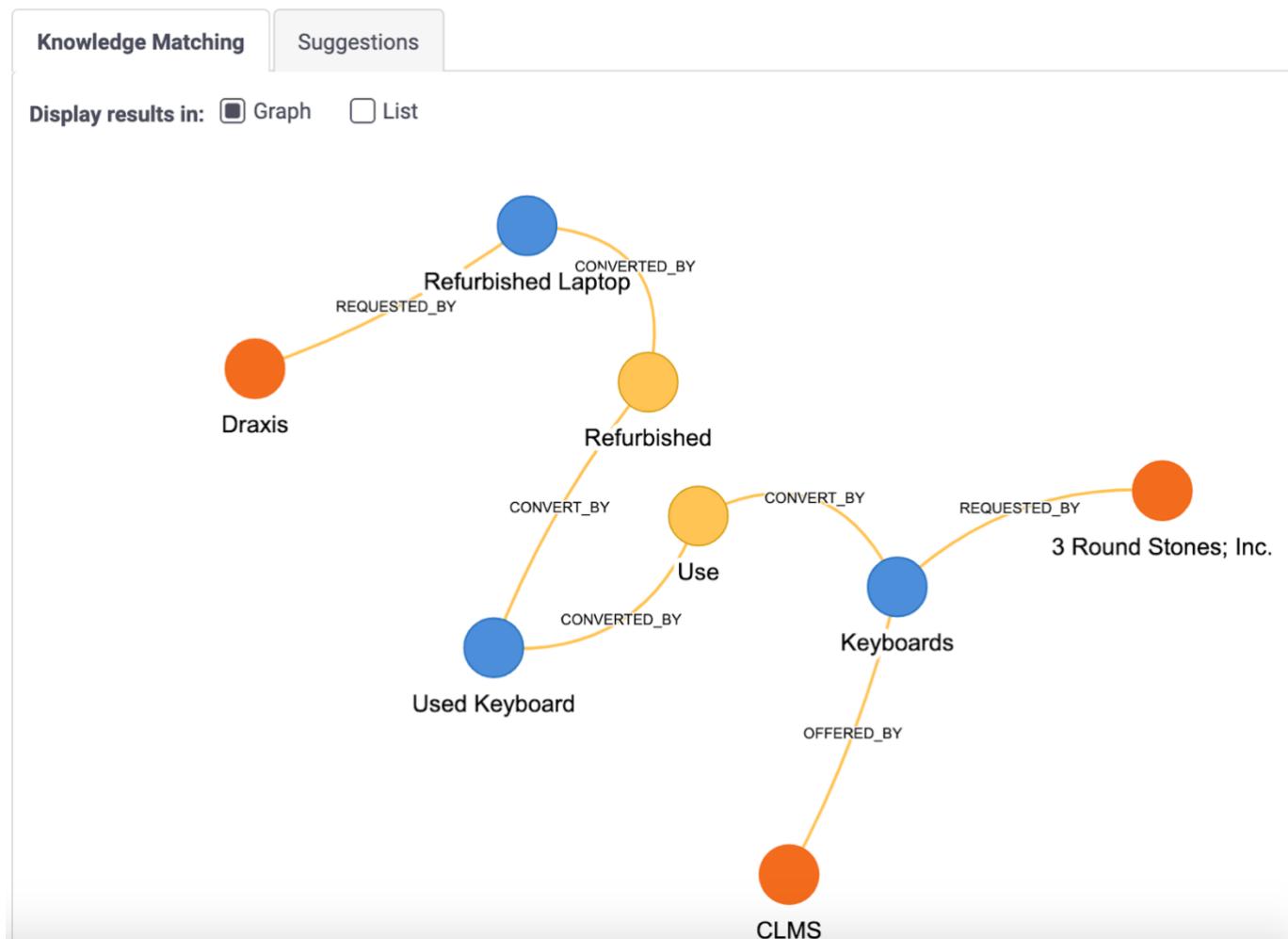


Figure 26: multiple paths available for synergies, as shown in the development environment.



7 Use Cases

7.1 Use case 1

In the first case, a firm (user) will be shown possible collaborations, automatically suggested (recommended) by the matchmaking tool. This process will run automatically, tracking wastes and resources that are not exploited currently by the company.

7.2 Use case 2

In another case, a firm (user) can search for specific wastes and resources, grouped by companies. The tool will then present companies that have open slots for collaboration.

7.3 Use case 3

A firm (user) can search a specific company in the company (cluster) database. The user can visualize the other company's wastes and resources, as well as visualize existing collaborations with third parties.

7.4 Use case 4

The user registers on the platform, provides information about their sites and their offerings and/or needs (requests). The matching tool will then try to find synergies with existing users and their respective offerings and needs (requests), after which the user will be notified of a potential synergy with a percentage representing the systems confidence in the match. After this process, the user can view the proposed synergy and chose if they want to pursue it further or not.

7.5 Use case 5

1. The user logs in (not mandatory)
2. They provide input in the form of selections from predefined drop-down lists, e.g., they select the domain of Bioeconomy.
3. They view the refined list of results
4. They select the dataset of interest, e.g., resources with locations and quantities of waste from food processing plants
5. They can read the description and decide if this dataset is what they are looking for
6. They can read the descriptions and technical specifications of all available resources of this dataset and select the ones that interest them
7. They can visualize a resource, e.g., in the form of a bar graph (higher to lower quantity) or on a map
8. They can choose whether to download the data as is, (excel format).



8 Conclusion

One of the biggest barriers in enabling industrial symbiosis is that companies' lack the understanding of the economic viability of participating in industrial symbiosis. Therefore, we proposed the concept of a collaboration platform for enabling industrial symbiosis and introduced the system framework undergirding the platform. This deliverable aims to describe the Matchmaking tool in detail.

The main goal of the Matchmaking tool development is to propose synergies between participating parties in an Industrial Symbiosis environment. The tool offers two mechanisms for matching involved parties: a "Semantic" and a "Synergy" proposal. The first matching functionality offers searching actors of interest for collaboration by filtering criteria, like country, sector or specific resource. Moreover, the functionality enables querying with text, which essentially searches for relevant terms in the actors' information. The second functionality offers "Synergy" proposal, by matching offers and requests of the actors. The matching is achieved via the material flow knowledge graph, also used in the Industrial Symbiosis tool. The proposed synergies are shown via graph representations, lists or tables.

Technically, almost everything we use today can be recycled – thus, making industrial symbiosis a viable option. Our work is focused on developing a collaboration platform for enabling industrial symbiosis, by introducing a matchmaking tool via waste-to-resource matching. We described how this subsystem is enabled by a database engine, which utilizes a classification method for organizing information on resources and wastes, and a graph database for modelling and storing the said information. The application of this database engine was demonstrated with use cases.

Since, existing tools for enabling industrial symbiosis lack the functionality to help companies evaluate the economic viability of participating in industrial symbiosis, a specific research gap is also being tackled by the work within the DIGICIRC project.

We demonstrated how through the visualization of resources conversion pathways generated by the database engine, the firm can obtain insights on which other parties can offer valuable resources can be recovered or converted from its resources or waste. Moreover, the visualization of these resource-to-resource or waste-to-resource conversion pathways allowed the firm to obtain insights on the possible substitute materials (wastes or by-products) that can be used to produce its product (resource) and which of the actors need to be communicated in order to start a synergy.



9 Bibliography

- B. Raabe, J.S.C. Low, M. Juraschek, C. Herrmann, T.B. Tjandra, Y.T. Ng, D. Kurle, F. Cerdas, J. Lueckenga, Z. Yeo, Y.S. Tan. 2017. *Collaboration Platform for Enabling Industrial Symbiosis : pplication of the By-product Exchange Network Model*. Procedia CIRP. 61.
- D. Hoornweg, P. Bhada-Tata. 2012. "What a waste : a global review of solid waste management." *World Bank*.
- D. Hoornweg, P. Bhada-tata, C. Kennedy. 2013. *Waste production must peak this century*. Nature. 502.
- I. Robinson, J. Webber, E. Eifrem. 2015. *Graph Databases: New Opportunities for Connected Data*. O'Reilly Media.
- J. Low, T. Tjandra, F. Yunus, S. Chung, D. Tan, B. Raabe, N. Ting, Z. Yeo, S. Bressan, S. Ramakrishna, C. Herrmann. 2018. "A Collaboration Platform for Enabling Industrial Symbiosis: Application of the Database Engine for Waste-to-Resource Matching." *25th CIRP Life Cycle Engineering (LCE) Conference*.
- Kusner, M., Sun, Y., Kolkin, N., & Weinberger, K. 2015. "From word embeddings to document distances." *International conference on machine learning*.
- Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. 2013. "Distributed representations of words and phrases and their compositionality." *In Advances in neural information processing systems*.
- Robertson, Stephen E., Steve Walker, Susan Jones, and Micheline Hancock-Beaulieu & Mike Gatford. 1994. "Okapi at TREC-3." *Proceedings of the Third Text REtrieval Conference*.
- Rose, S., Engel, D., Cramer, N., & Cowley, W. 2010. "Automatic keyword extraction from individual documents." In *Text mining: applications and theory*.



10 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.

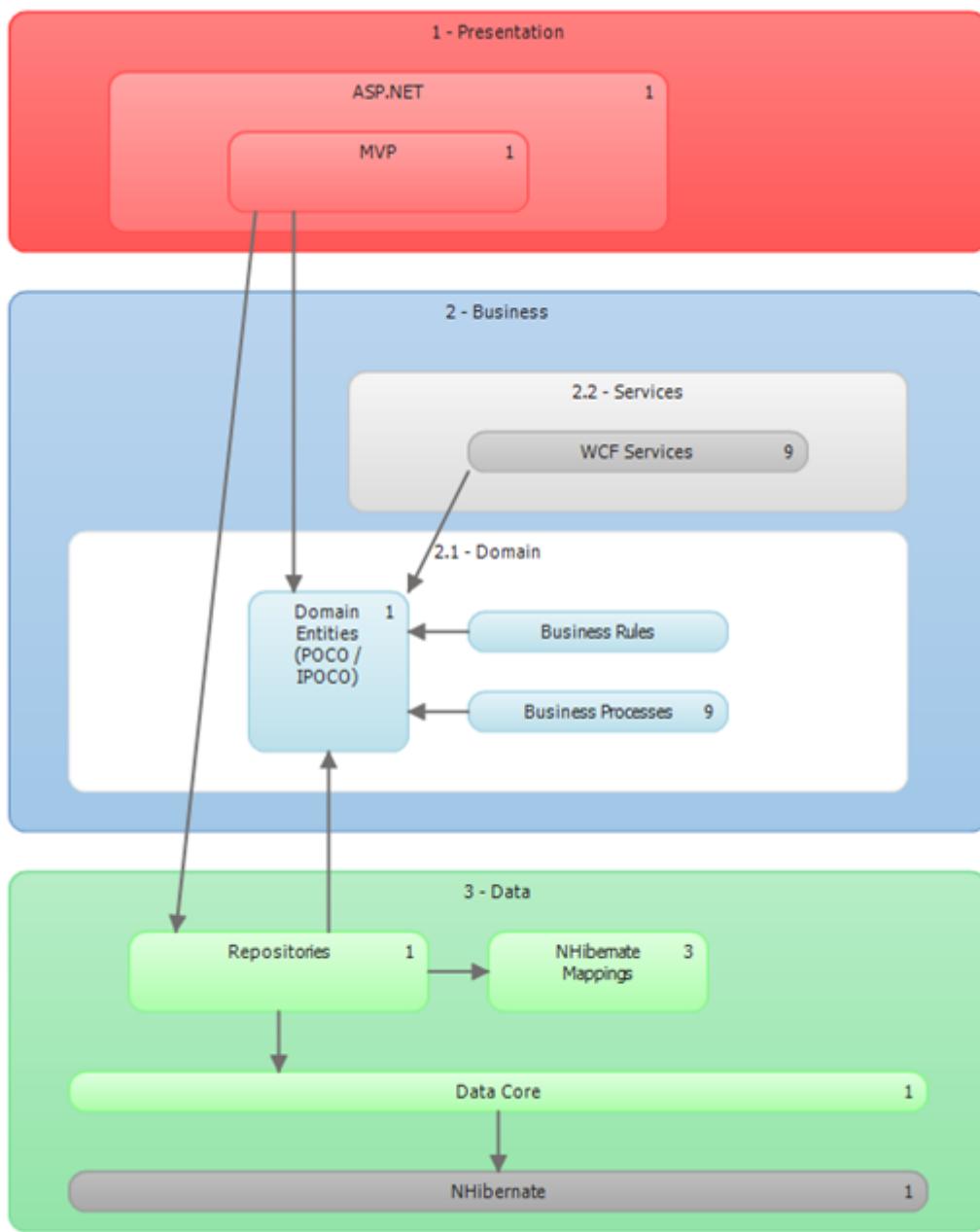
Design approach

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

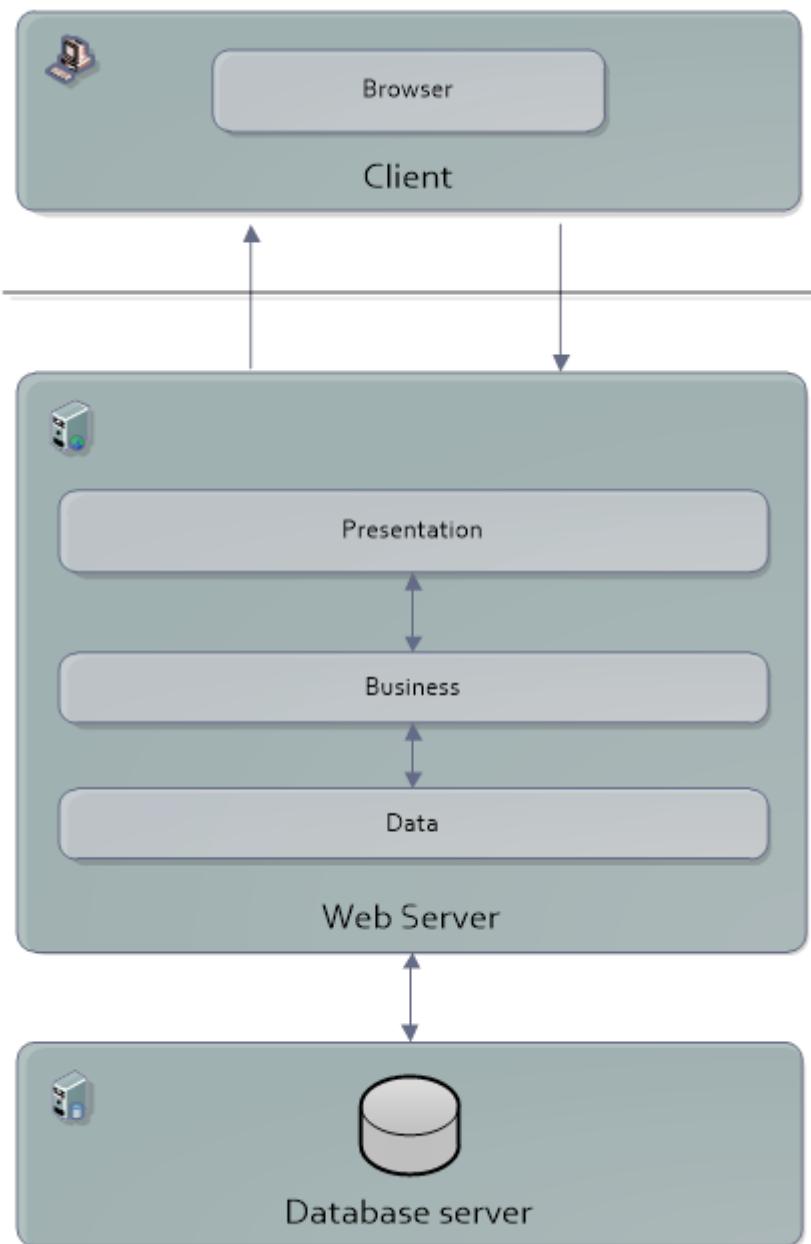
Architecture Design

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.

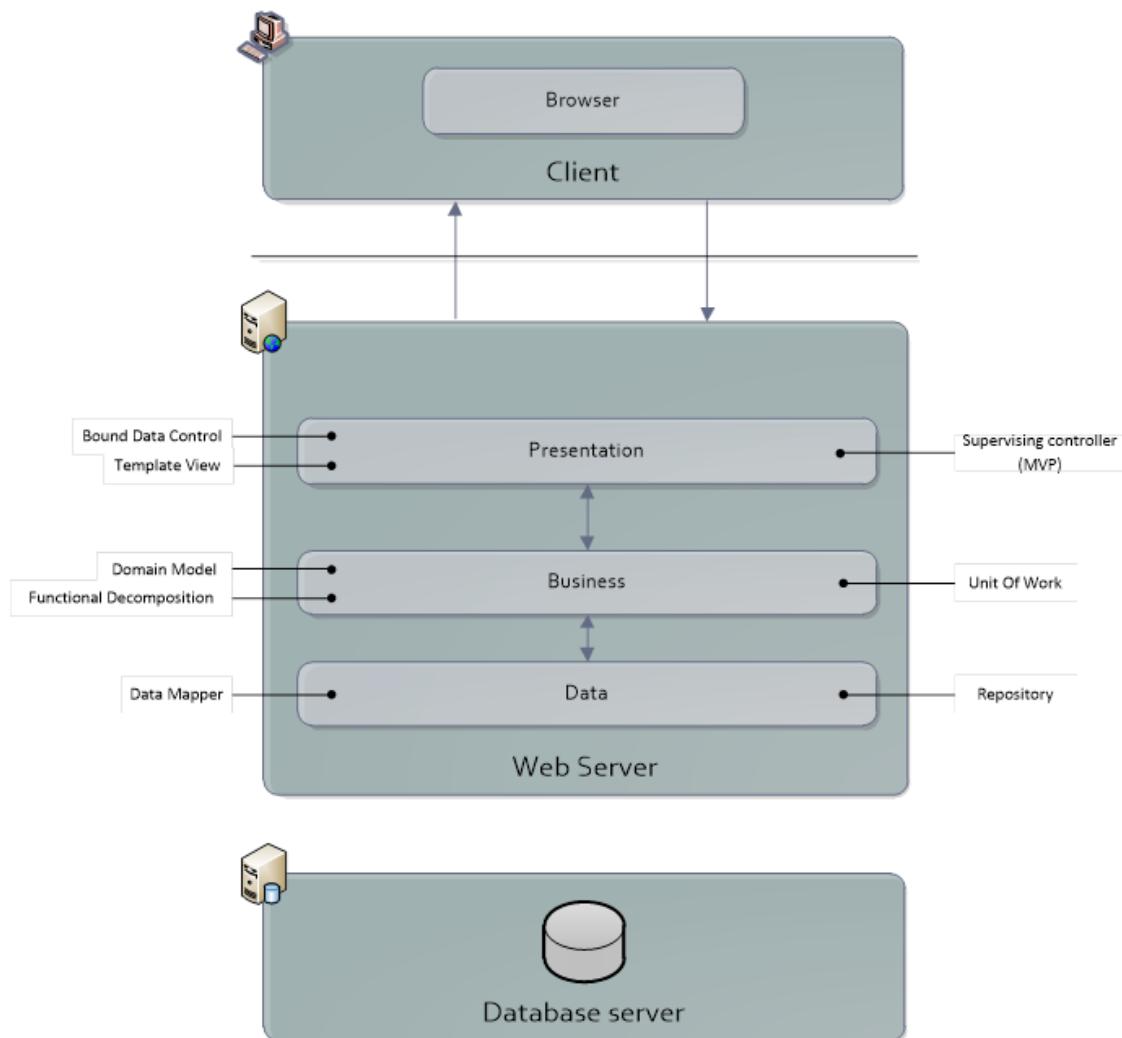


Key Characteristics

- 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
Actor	<i>Company</i>	<input checked="" type="checkbox"/>	0..1
DigicircUser	<i>AddedBy</i>	<input checked="" type="checkbox"/>	0..1

Actor - DigicircUser Accosiation

	Member	Navigable	Multiplicity
Actor	<i>ActorsCanManage</i>	<input checked="" type="checkbox"/>	*****
DigicircUser	<i>Administrators</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"/>	0..1

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"/>	*****

ApplicationUser - ApplicationUser Accosiation

	Member	Navigable	Multiplicity

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

ApplicationUser - ApplicationUserLogin Accosiation

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

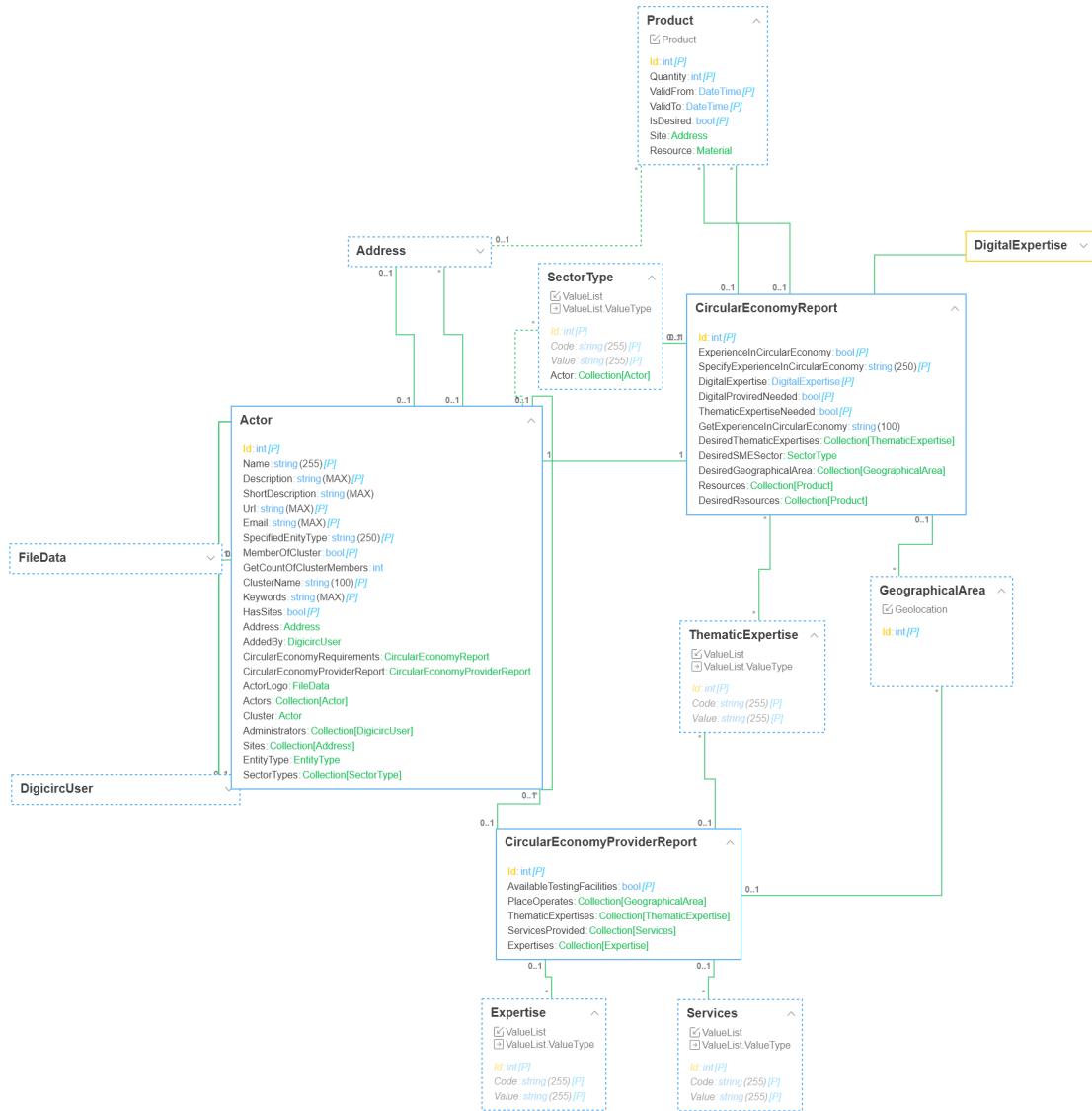
ApplicationUserClaim - ApplicationUser Accosiation

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

Profile - ApplicationUser Accosiation

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

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

Actor - DigicircUser Accosiation

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

CircularEconomyReport - ThematicExpertise Accosiation

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

CircularEconomyReport - SectorType Accosiation

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

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

CircularEconomyReport - Actor Accosiation

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

CircularEconomyReport - GeographicalArea Accosiation

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

GeographicalArea - CircularEconomyProviderReport Accosiation

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

CircularEconomyProviderReport - ThematicExpertise Accosiation

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

Services - CircularEconomyProviderReport Accosiation

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

CircularEconomyProviderReport - Expertise Accosiation

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

Actor - CircularEconomyProviderReport Accosiation

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

Actor - FileData Accosiation

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

Actor - Actor Accosiation

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

Actor - DigicircUser Accosiation

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

CircularEconomyReport - Product Accosiation

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

CircularEconomyReport - Product Accosiation

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

Actor - Address Accosiation

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

Actor - EntityType Accosiation

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

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

Match - Actor Accosiation

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

Match - Actor Accosiation

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

Material - DigicircUser Accosiation

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

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

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

ApplicationUser - ApplicationUserLogin Accosiation

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

ApplicationUserClaim - ApplicationUser Accosiation

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

Profile - ApplicationUser Accosiation

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

Address - Country Accosiation

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

Product - Address Accosiation

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

SearchQuery - SectorType Accosiation

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

Product - Material Accosiation

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

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

GraphQuery - Product Accosiation

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

GraphQuery - Product Accosiation

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

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"/>

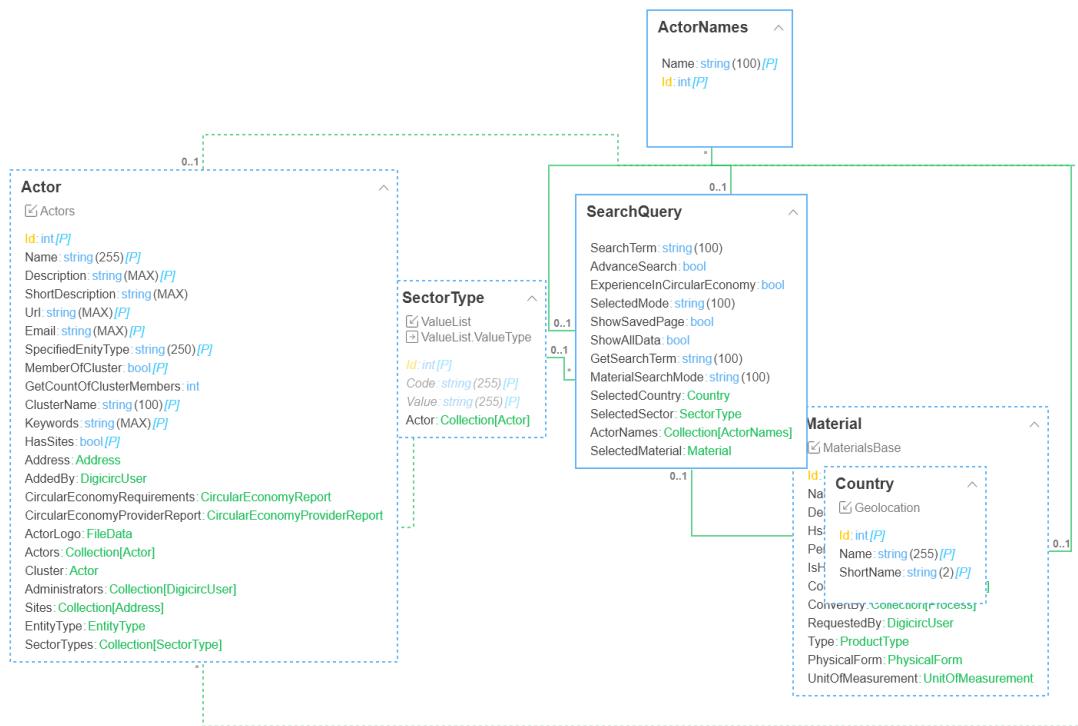
Activities Class

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

SearchQuery - SectorType Accosiation

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

SearchQuery - ActorNames Accosiation

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

SearchQuery - Material Accosiation

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

Address - Country Accosiation

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

Actor - Address Accosiation

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

Actor - DigicircUser Accosiation

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

CircularEconomyReport - Actor Accosiation

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

Actor - CircularEconomyProviderReport Accosiation

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

Actor - FileData Accosiation

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

Actor - Actor Accosiation

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

Actor - DigicircUser Accosiation

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

Actor - Address Accosiation

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

Actor - EntityType Accosiation

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

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

Match - Actor Accosiation

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

Match - Actor Accosiation

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

CircularEconomyReport - SectorType Accosiation

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

GraphQuery - ActorNames Accosiation

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

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

Material - ProductType Accosiation

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

Material - PhysicalForm Accosiation

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

Material - UnitOfMeasurement Accosiation

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

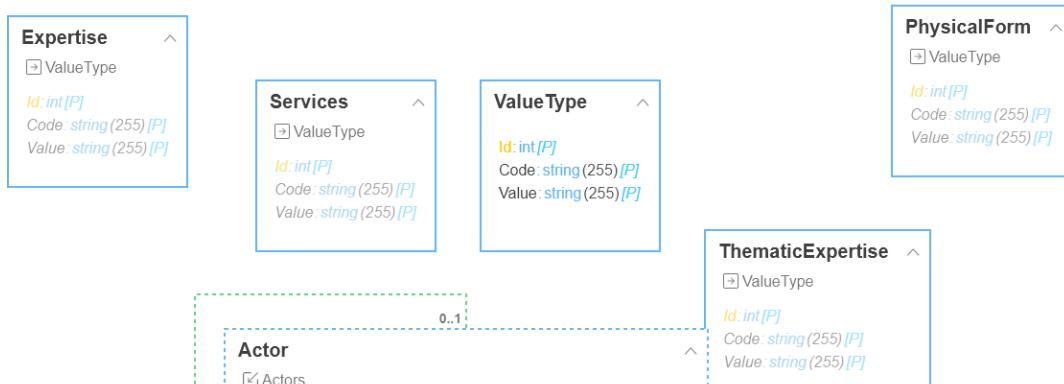
Product - Material Accosiation

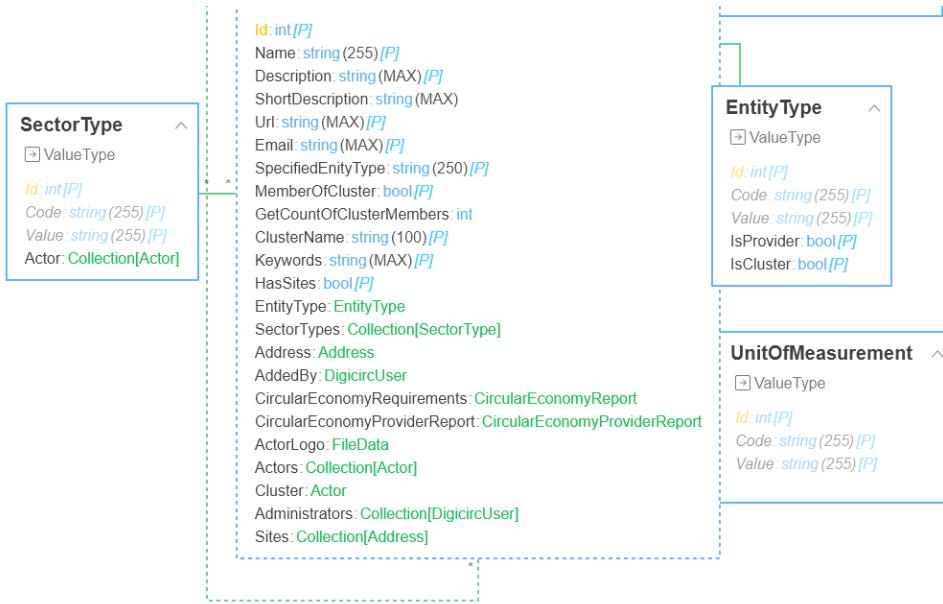
	Member	Navigable	Multiplicity
Product	<i>Product</i>	☒	*****
Material	<i>Resource</i>	<input checked="" type="checkbox"/>	0..1

Match - Material Accosiation

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

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

SectorType - Actor Accosiation

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

CircularEconomyReport - SectorType Accosiation

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

SearchQuery - SectorType Accosiation

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

Actor - Address Accosiation

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

Actor - DigicircUser Accosiation

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

CircularEconomyReport - Actor Accosiation

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

Actor - CircularEconomyProviderReport Accosiation

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

Actor - FileData Accosiation

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

Actor - Actor Accosiation

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

Actor - DigicircUser Accosiation

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

Actor - Address Accosiation

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

GraphQuery - Actor Accosiation

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

Match - Actor Accosiation

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

Match - Actor Accosiation

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

CircularEconomyReport - ThematicExpertise Accosiation

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

CircularEconomyProviderReport - ThematicExpertise Accosiation

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

Services - CircularEconomyProviderReport Accosiation

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

CircularEconomyProviderReport - Expertise Accosiation

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

Material - PhysicalForm Accosiation

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

Product - PhysicalForm Accosiation

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

Material - UnitOfMeasurement Accosiation

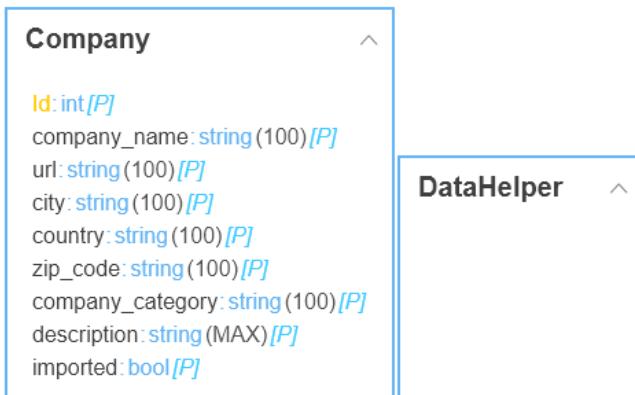
	Member	Navigable	Multiplicity

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

Product - UnitOfMeasurement Accosiation

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

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

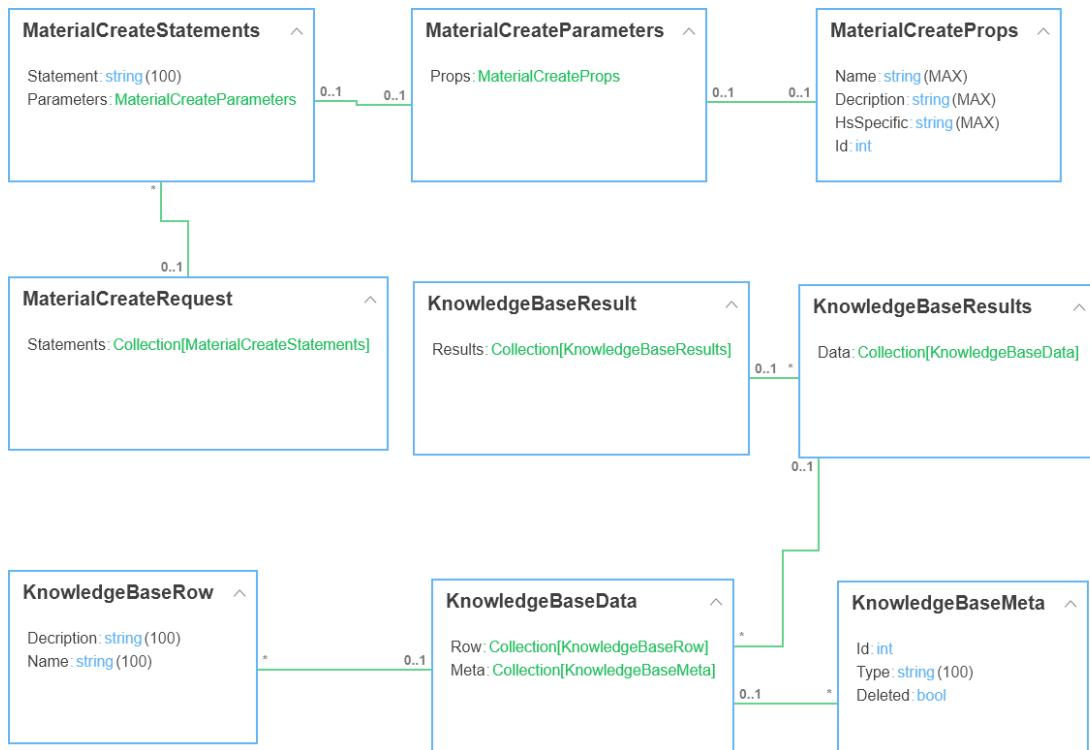
Name	Datatype	Persisted	Required	Encrypted

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 ClassPersisted: 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 ClassPersisted: 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
MaterialCreateRequest	<i>MaterialCreateRequest</i>	<input checked="" type="checkbox"/>	0..1
MaterialCreateStatements	<i>Statements</i>	<input checked="" type="checkbox"/>	*****

MaterialCreateStatements - MaterialCreateParameters Accosiation

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

MaterialCreateParameters - MaterialCreateProps Accosiation

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

KnowledgeBaseResult - KnowledgeBaseResults Accosiation

	Member	Navigable	Multiplicity

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

KnowledgeBaseResults - KnowledgeBaseData Accosiation

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

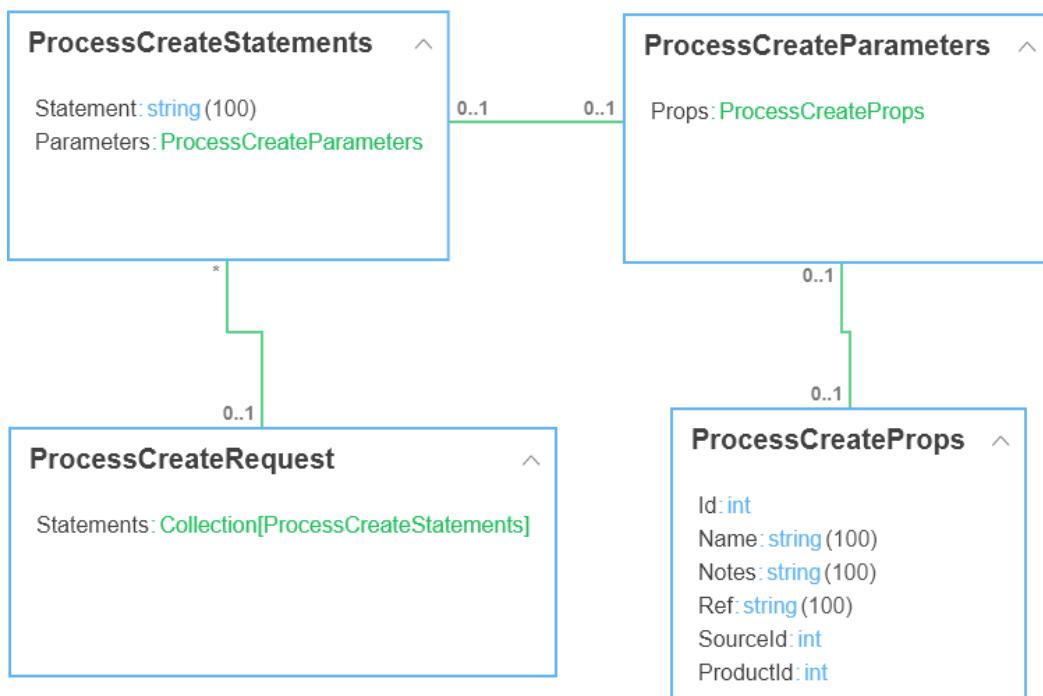
KnowledgeBaseData - KnowledgeBaseRow Accosiation

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

KnowledgeBaseData - KnowledgeBaseMeta Accosiation

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

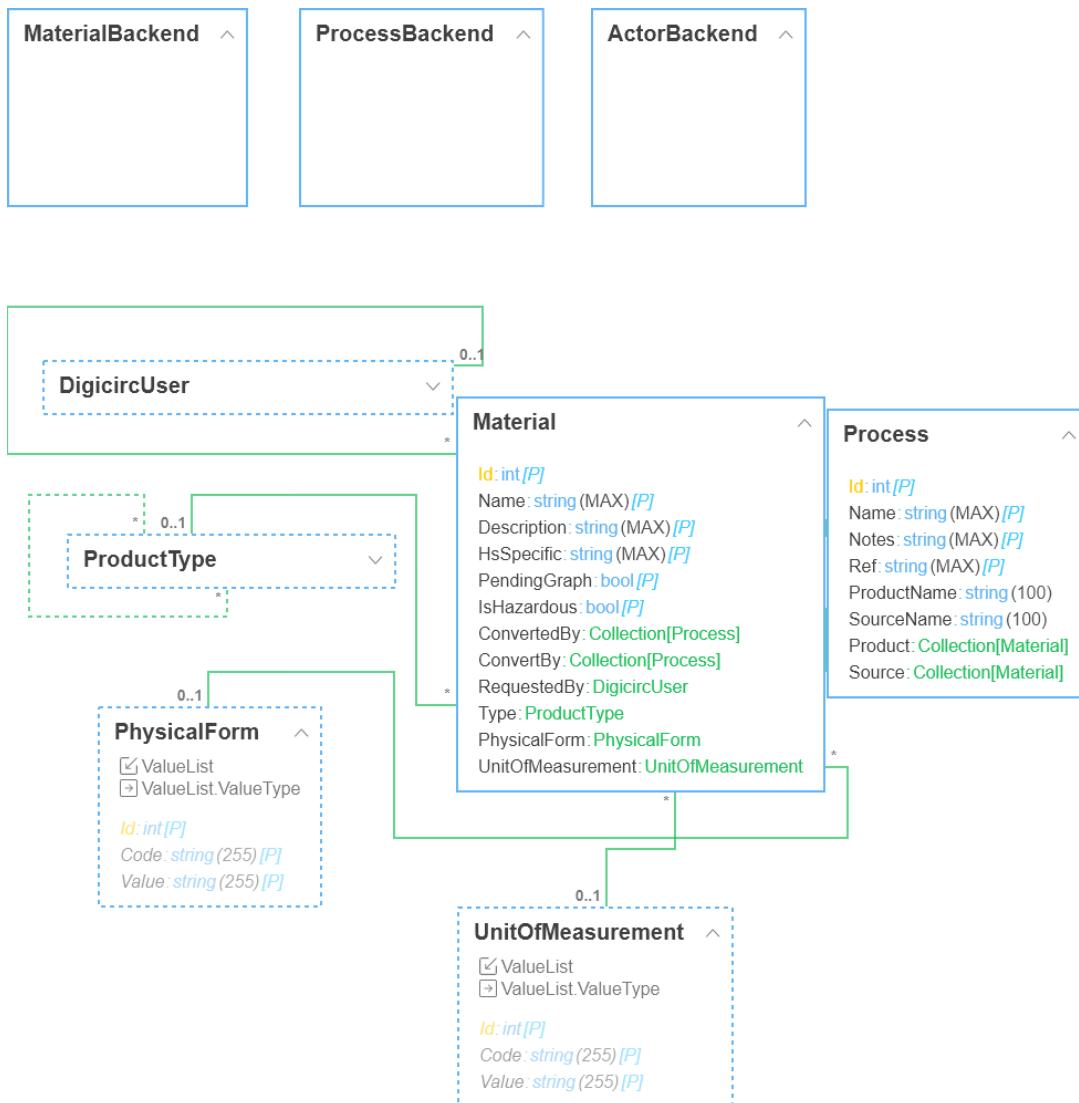
ProcessCreateStatements - ProcessCreateParameters Accosiation

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

ProcessCreateParameters - ProcessCreateProps Association

	Member	Navigable	Multiplicity
ProcessCreateParameters	<i>ProcessCreateParameters</i>	☒	0..1
ProcessCreateProps	<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"/>	0..1

Material - ProductType Accosiation

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

Material - PhysicalForm Accosiation

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

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

Material - UnitOfMeasurement Accosiation

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

Product - Material Accosiation

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

Match - Material Accosiation

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

Actor - DigicircUser Accosiation

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

Actor - DigicircUser Accosiation

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

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

ApplicationUser - ApplicationUserLogin Accosiation

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

ApplicationUserClaim - ApplicationUser Accosiation

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

Profile - ApplicationUser Accosiation

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

ProductType - ProductType Accosiation

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

Product - ProductType Accosiation

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

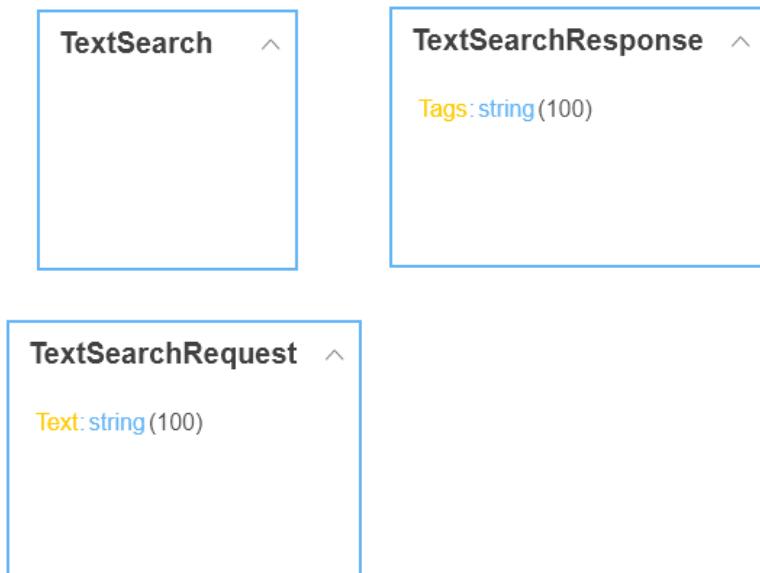
Product - PhysicalForm Accosiation

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

Product - UnitOfMeasurement Accosiation

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

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"/>

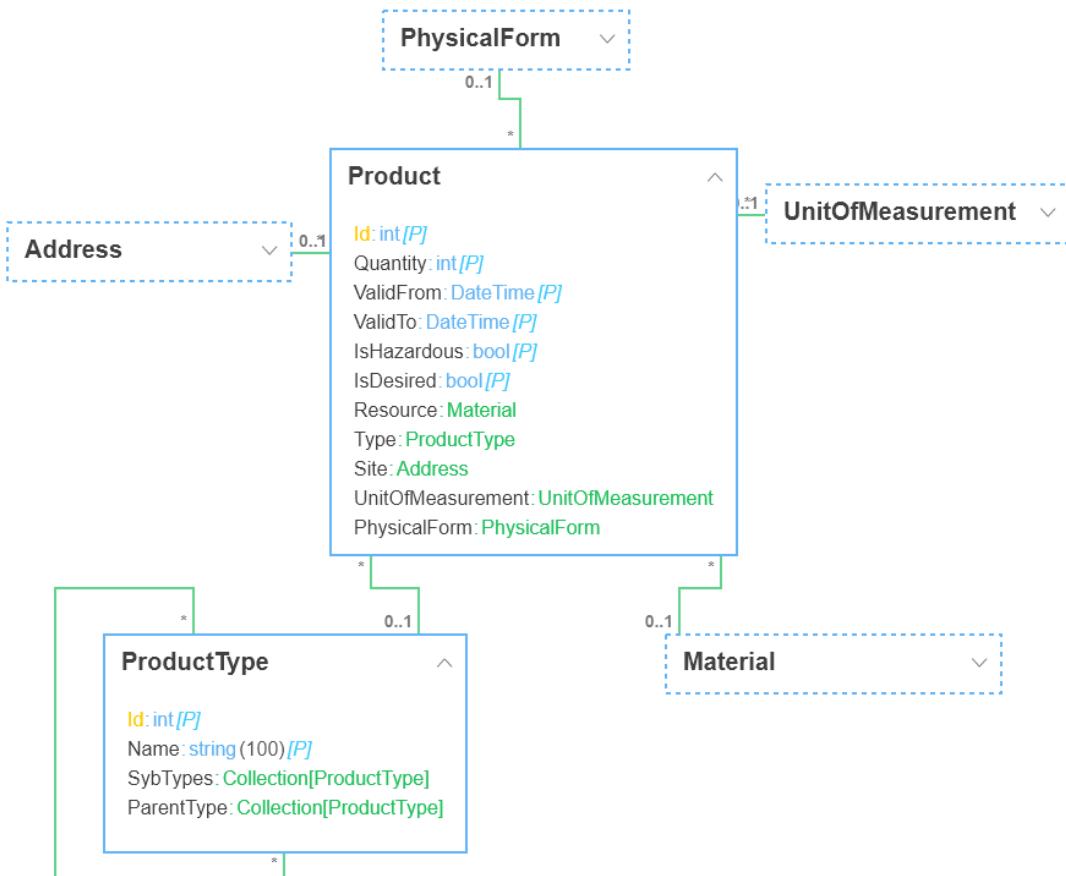
TextSearchResponse Class

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

ProductType - ProductType Accosiation

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

Product - ProductType Accosiation

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

Product - Address Accosiation

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

Product - UnitOfMeasurement Accosiation

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

Product - PhysicalForm Accosiation

	Member	Navigable	Multiplicity

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

CircularEconomyReport - Product Accosiation

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

CircularEconomyReport - Product Accosiation

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

GraphQuery - Product Accosiation

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

GraphQuery - Product Accosiation

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

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

Match - Material Accosiation

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

Actor - Address Accosiation

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

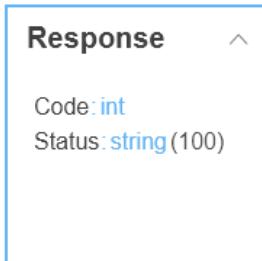
Actor - Address Accosiation

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

Address - Country Accosiation

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

ActorAPIHelper Class Diagram



ActorAPIHelper Classes

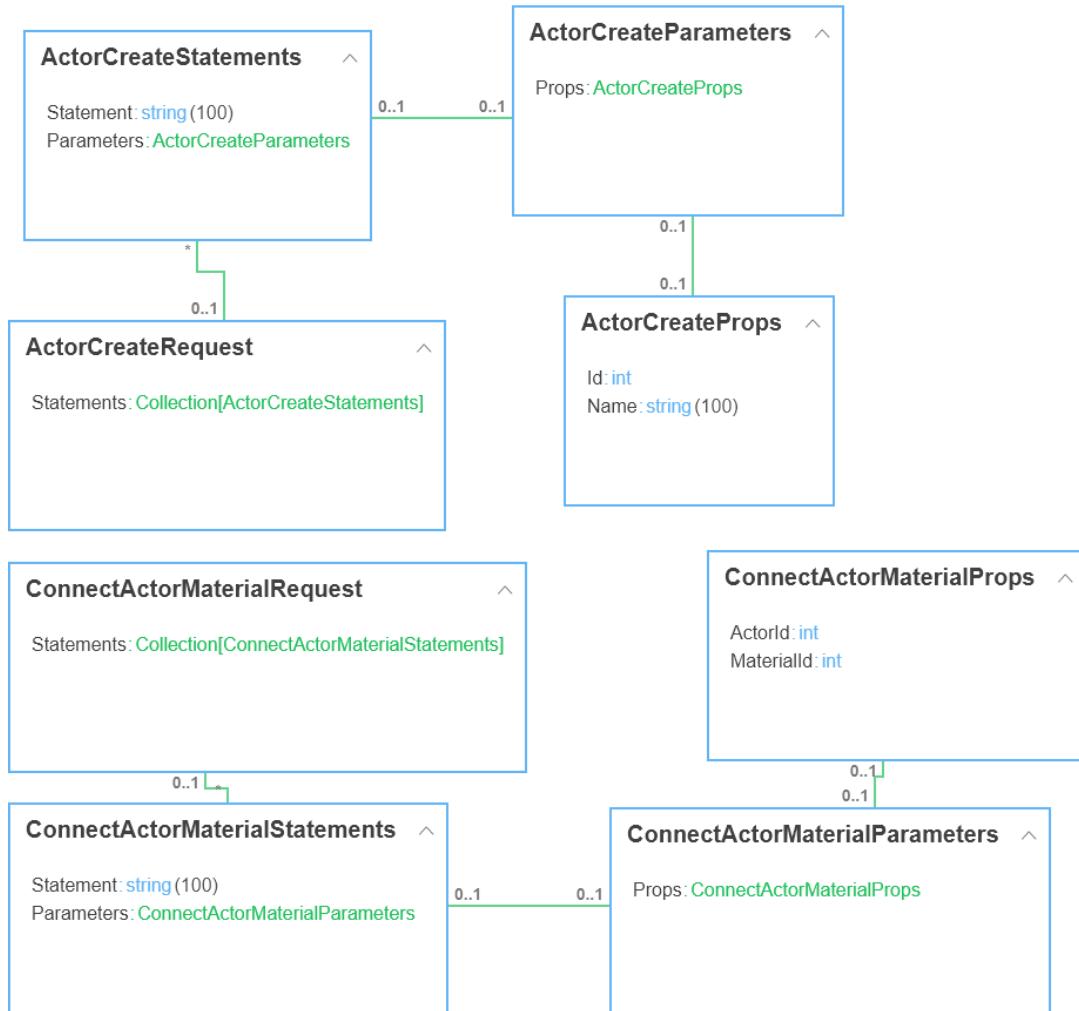
Response Class

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 ClassPersisted: Identity: **Attributes**

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

ConnectActorMaterialStatements ClassPersisted: Identity: **Attributes**

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

ConnectActorMaterialParameters ClassPersisted: Identity: **Attributes**

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

ConnectActorMaterialProps ClassPersisted: 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
ActorCreateRequest	<i>ActorCreateRequest</i>	<input checked="" type="checkbox"/>	0..1
ActorCreateStatements	<i>Statements</i>	<input checked="" type="checkbox"/>	*****

ActorCreateStatements - ActorCreateParameters Accosiation

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

ActorCreateParameters - ActorCreateProps Accosiation

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

ConnectActorMaterialRequest - ConnectActorMaterialStatements Accosiation

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

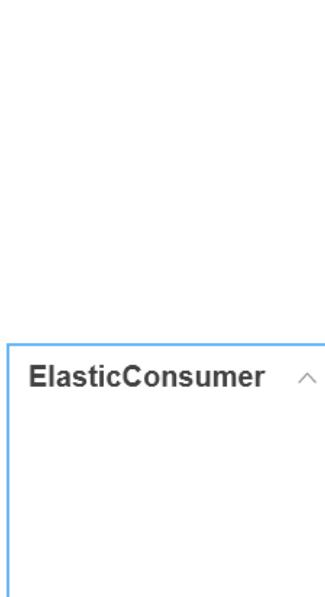
ConnectActorMaterialStatements - ConnectActorMaterialParameters Accosiation

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

ConnectActorMaterialParameters - ConnectActorMaterialProps Accosiation

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

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);
    }

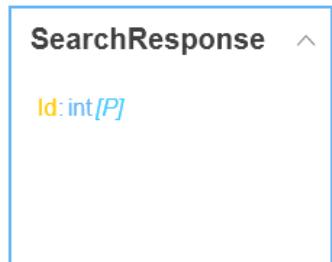
    if(filterResource && modelQuery.SelectedMaterial.Name != null)
    {
        Interfaces.ElasticSearch.Must mustSearchTerm;
        Interfaces.ElasticSearch.MultiMatch matchSearchTerm;
        matchSearchTerm.Query = modelQuery.SelectedMaterial.Name;
    }
}

```

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

    return mustCollection;
}
```

ElasticModel Class Diagram



ElasticModel Classes

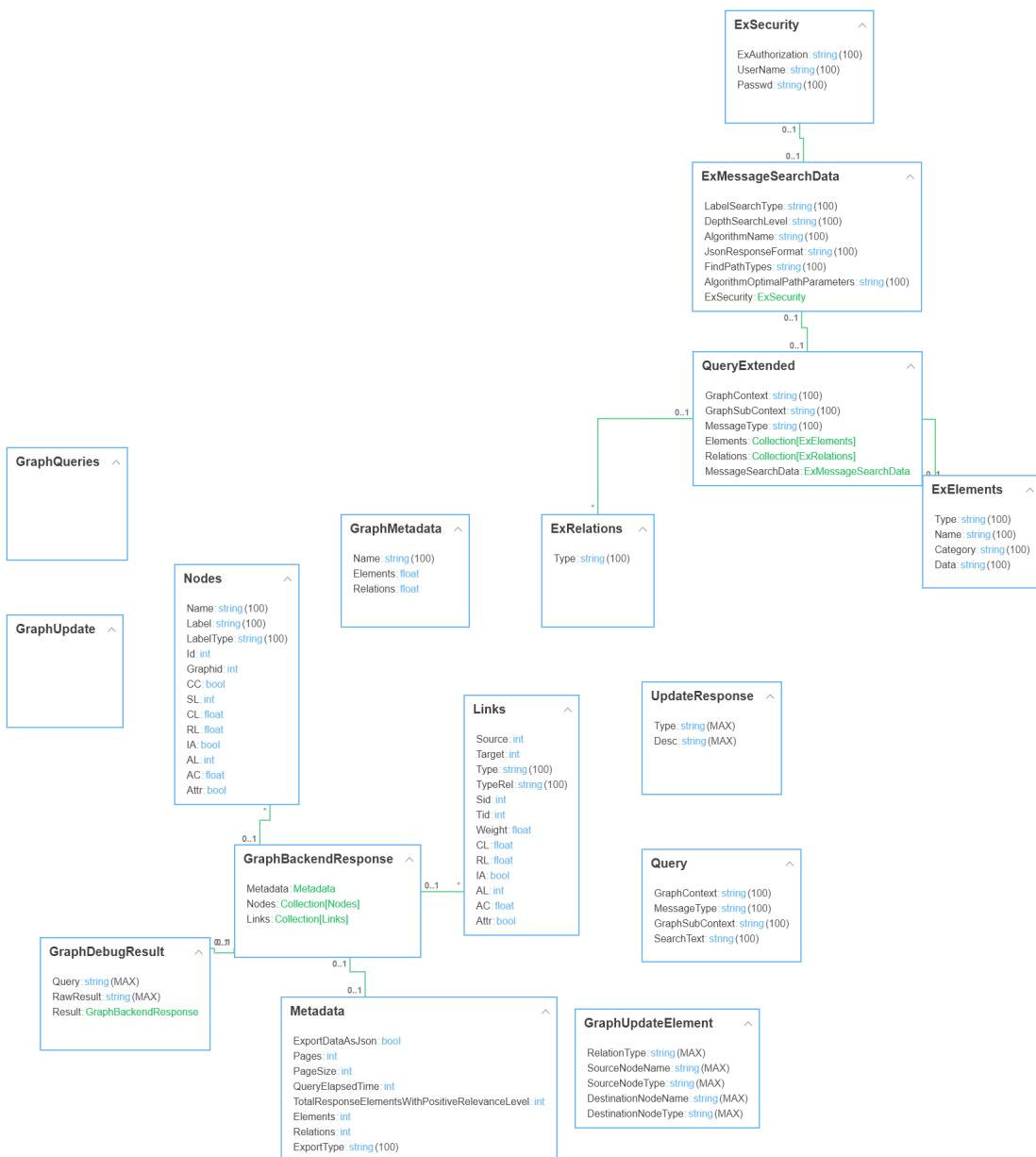
SearchResponse Class

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"/>
QueryElapsedTime	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
GraphBackendResponse	<i>GraphBackendResponce</i>	<input checked="" type="checkbox"/>	0..1
Metadata	<i>Metadata</i>	<input checked="" type="checkbox"/>	0..1

GraphBackendResponse - Nodes Accosiation

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

GraphBackendResponse - Links Accosiation

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

GraphDebugResult - GraphBackendResponse Accosiation

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

QueryExtended - ExElements Accosiation

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

QueryExtended - ExRelations Accosiation

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

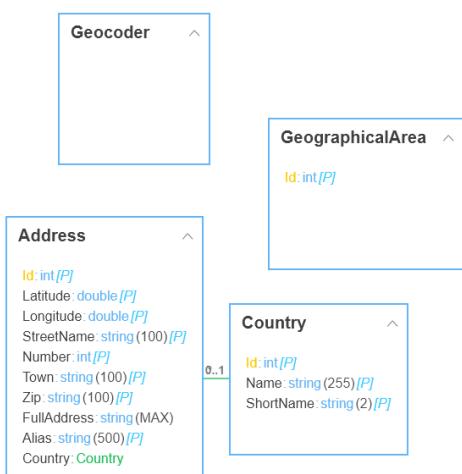
ExMessageSearchData - ExSecurity Accosiation

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

QueryExtended - ExMessageSearchData Accosiation

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

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

Actor - Address Accosiation

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

Actor - Address Accosiation

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

Product - Address Accosiation

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

SearchQuery - Country Accosiation

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

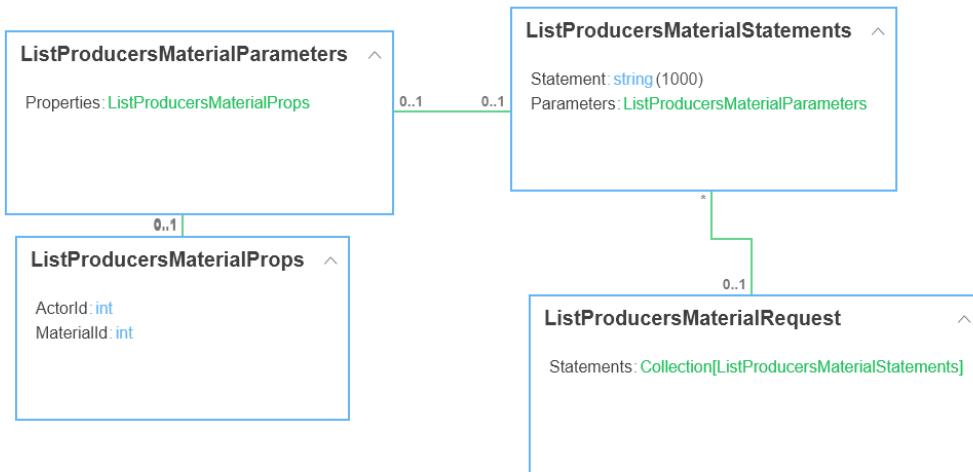
CircularEconomyReport - GeographicalArea Accosiation

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

GeographicalArea - CircularEconomyProviderReport Accosiation

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

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"/>	0..1
ListProducersMaterialParameters	ListProducersMaterialParameters	<input checked="" type="checkbox"/>	0..1

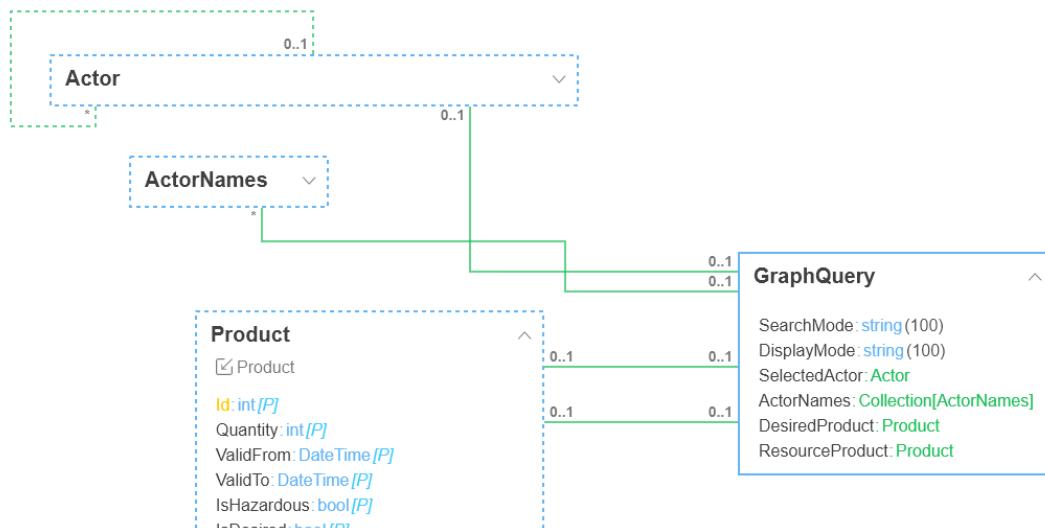
ListProducersMaterialStatements - ListProducersMaterialParameters Accosiation

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

ListProducersMaterialRequest - ListProducersMaterialStatements Accosiation

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

GraphQuery - ActorNames Accosiation

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

Actor - Address Accosiation

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

Actor - DigicircUser Accosiation

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

CircularEconomyReport - Actor Accosiation

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

Actor - CircularEconomyProviderReport Accosiation

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

Actor - FileData Accosiation

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

Actor - Actor Accosiation

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

Actor - DigicircUser Accosiation

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

Actor - Address Accosiation

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

Actor - EntityType Accosiation

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

SectorType - Actor Accosiation

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

SearchQuery - ActorNames Accosiation

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

CircularEconomyReport - Product Accosiation

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

CircularEconomyReport - Product Accosiation

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

Product - Material Accosiation

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

Product - ProductType Accosiation

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

Product - Address Accosiation

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

Product - UnitOfMeasurement Accosiation

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

Product - PhysicalForm Accosiation

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

GraphQuery - Product Accosiation

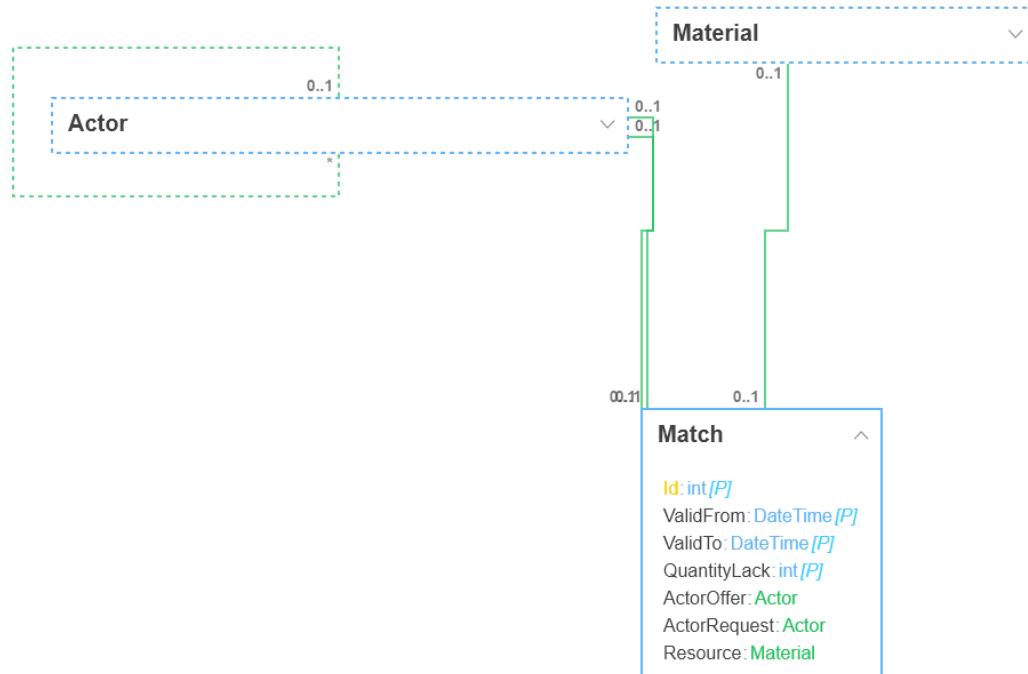
	Member	Navigable	Multiplicity
GraphQuery	<i>GraphQuery</i>	<input checked="" type="checkbox"/>	0..1
Product	<i>DesiredProduct</i>	<input checked="" type="checkbox"/>	0..1

GraphQuery - Product Accosiation

	Member	Navigable	Multiplicity

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

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

Match - Actor Accosiation

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

Match - Material Accosiation

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

Actor - Address Accosiation

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

Actor - DigiCircUser Accosiation

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

CircularEconomyReport - Actor Accosiation

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

Actor - CircularEconomyProviderReport Accosiation

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

Actor - FileData Accosiation

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

Actor - Actor Accosiation

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

Actor - DigiCircUser Accosiation

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

Actor - Address Accosiation

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

Actor - EntityType Accosiation

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

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

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"/>	0..1

Material - ProductType Accosiation

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

Material - PhysicalForm Accosiation

	Member	Navigable	Multiplicity

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

Material - UnitOfMeasurement Accosiation

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

Product - Material Accosiation

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

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

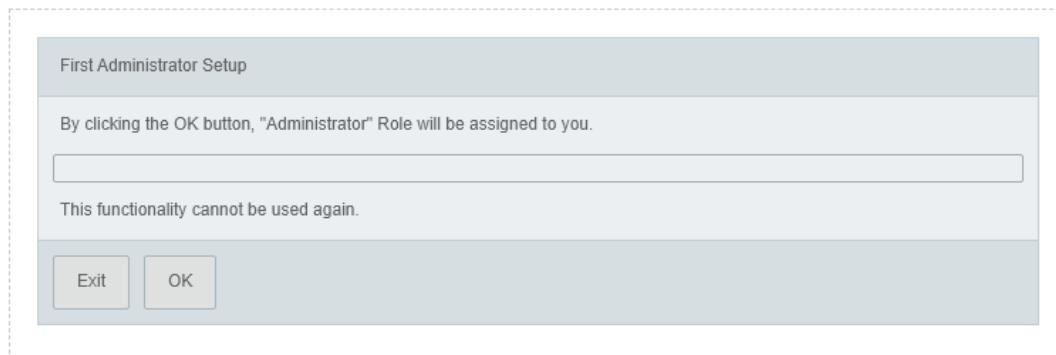
```

|-- 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

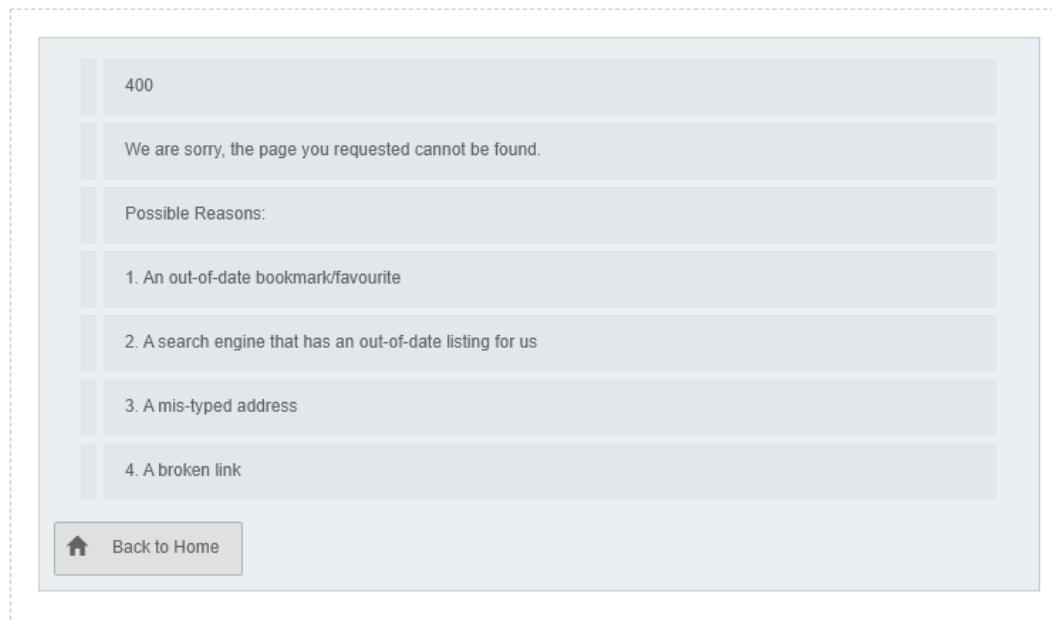
Is Entrypoint: Enabled Access Log: Causes Validation:

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****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

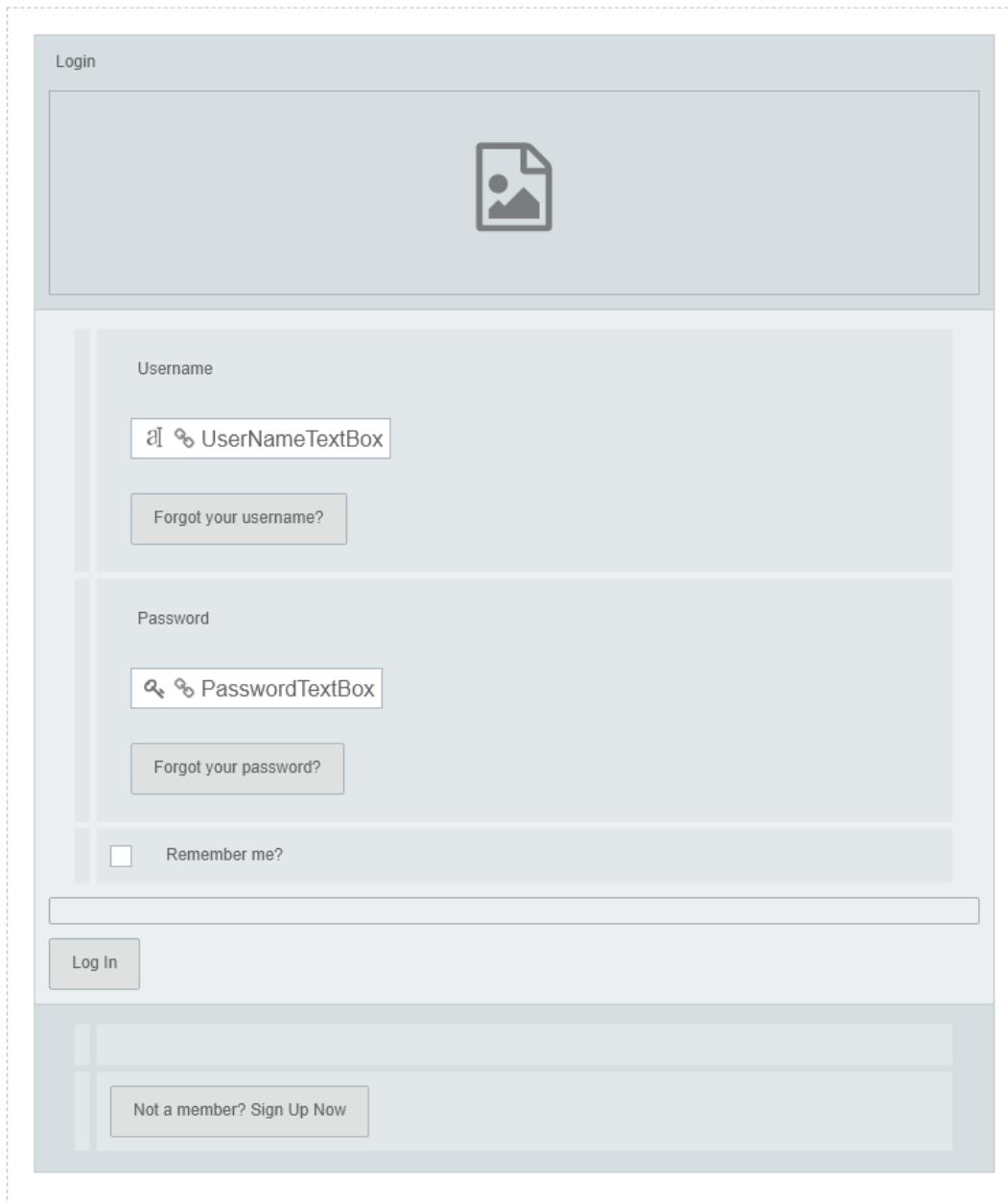
```
function void Execute() {  
}  
}
```

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

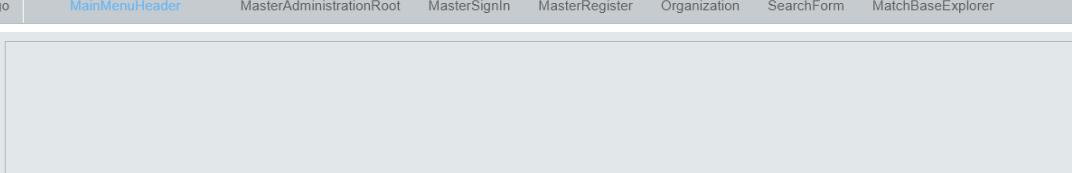
Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

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">

</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></div> <div></div> <div class="active"></div> <div class="active"></div> <div class="active"></div> <div class="active"></div> <div href="https://fasttrack.vc/"></div> <div href="https://draxis.gr/"></div> <div href="https://www.clmsuk.com/"></div> <div href="https://www.arthurslegal.com/"></div> <div href="https://www.inspiringculture.org/"></div> <div href="https://www.europeflag300x300.png" alt="image"></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"> Crafted by <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="M1000.71,178.43c0-20.47-3.32-52.39-29.11-76.92C949.9,8.106,922.48,76.6,901.2,76.6c-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,-0.22,9.2-4.88-27-9.7-7.7-11.47-20.85-11.49-30.66l136.27,-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> Privacy Policy</p></div> </div> <div><p class="copyright">Copyright © 2020 CLMS. All Rights reserved. Privacy Policy</p></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 ActionIs 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 ActionIs 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 ActionIs 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 says 'Change Password'. The form has three text input fields: 'Current password', 'New password', and 'Confirm password', each with validation messages below it. At the bottom are 'Change' and 'Cancel' buttons.

Field	Type	Validation Message
Current password	Text	Validations.CurrentPasswordEmpty.Message
New password	Text	Validations.NewPasswordEmpty.Message
Confirm password	Text	Validations.RepeatPasswordEmpty.Message

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"/> *
Validations.Required_Name.Message	
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 ActionIs 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 ActionIs 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 ActionIs 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. 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 icons for file operations like upload, search, and refresh. The main area displays a grid of data. The grid has a header row with columns: "Edit", "Parent Controller Name", "Name", "Is Available To All Authorized Users", and "Is Available To Anonymous". There are three data rows below the header:

Edit	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 buttons for "Page 1 of 10 pages" and "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

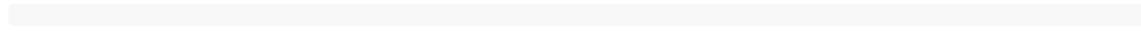
Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ManageUsers

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 Terms
--------------------------	----------------------------------

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 processes "Table_CurrentItem" from "ActorDataSet1", displaying icons for "ActorDataSet1.Item.Name" (document), "ActorDataSet1.Item.ShortDescription" (image), "ActorDataSet1.Item.EntityType.Value" (person), and "ActorDataSet1.Item.Address.Country.Name" (location pin). 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 ActionIs 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 ActionIs 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 ActionIs 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 ActionIs 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 and a main content area. The main content area contains a table-like structure with three columns: 'Name', 'Email', and 'Country'. Each row represents a record from the CompanyDataSet. An 'Edit' button is located in the top right corner of the table area. A footer bar at the bottom displays the text 'Records: % CompanyDataSet.TotalItems'.

Name:	Email:	Country:
% CompanyDataSet.Item.Name	% CompanyDataSet.Item.Email	% CompanyDataSet.Item.Address.Country.Name

Records: % CompanyDataSet.TotalItems

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

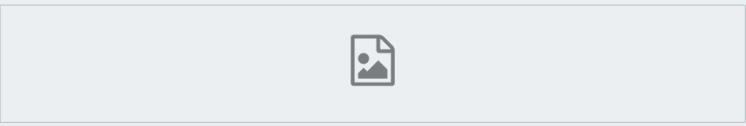
```

View

% Title Unsaved Changes!

[Back](#) [Delete](#) [Edit](#) [Save](#)

Logo



Please choose images with size 256x96

Description

Actor.Description

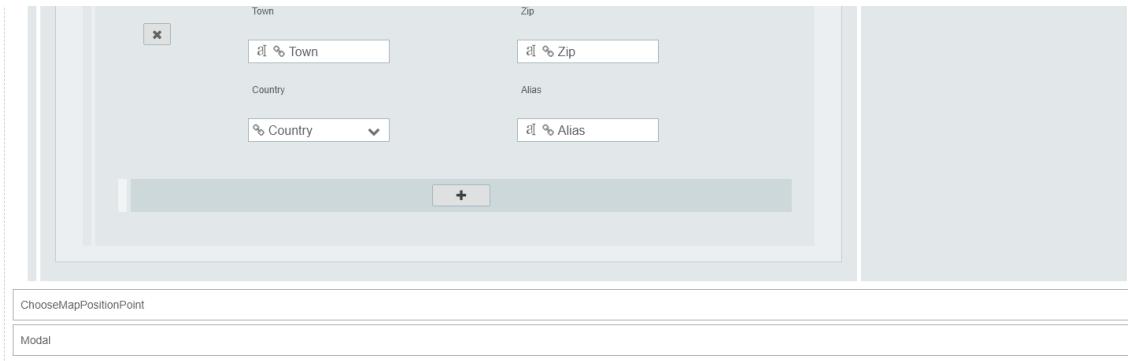
Description

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"/> Add New
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(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;
    }

    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

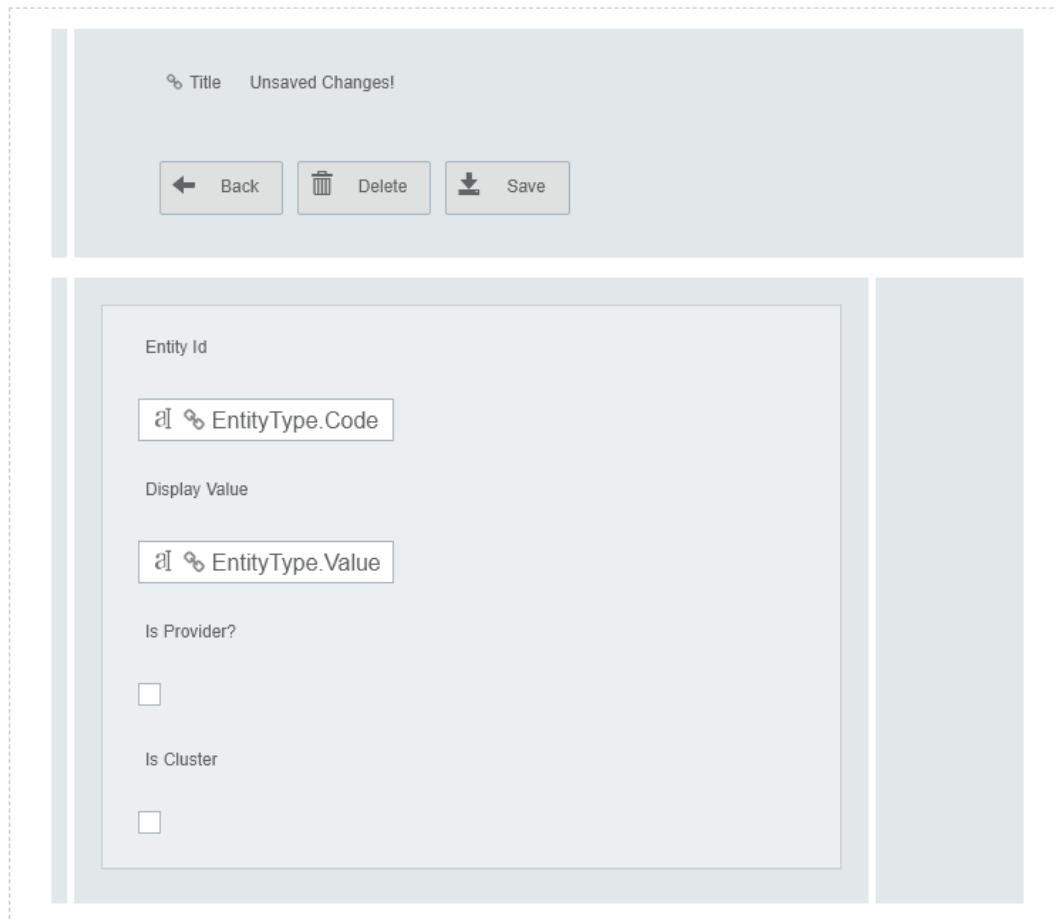
```
function void AddNewCluster()
{
    FormControls.Modal.Show();
}
```

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

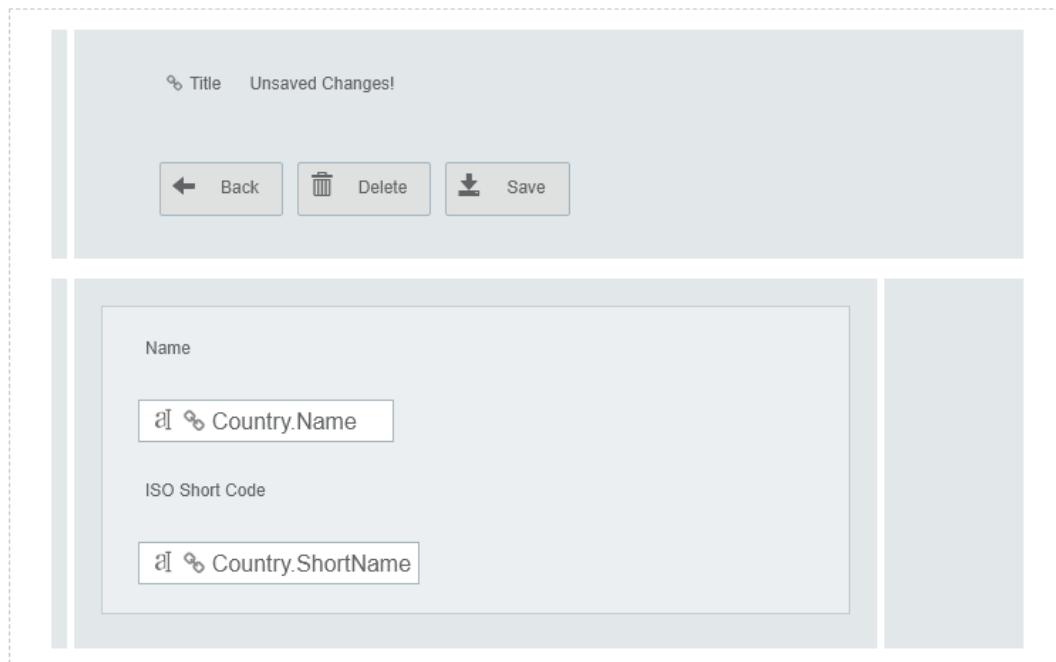
Model

```

|-- 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

View screenshot showing a list of entities. The interface includes a title bar, a toolbar with various icons, and a grid-based data table.

Toolbar: Add, Edit, Download, Search, Filter, Refresh, *.

Data Grid:

	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

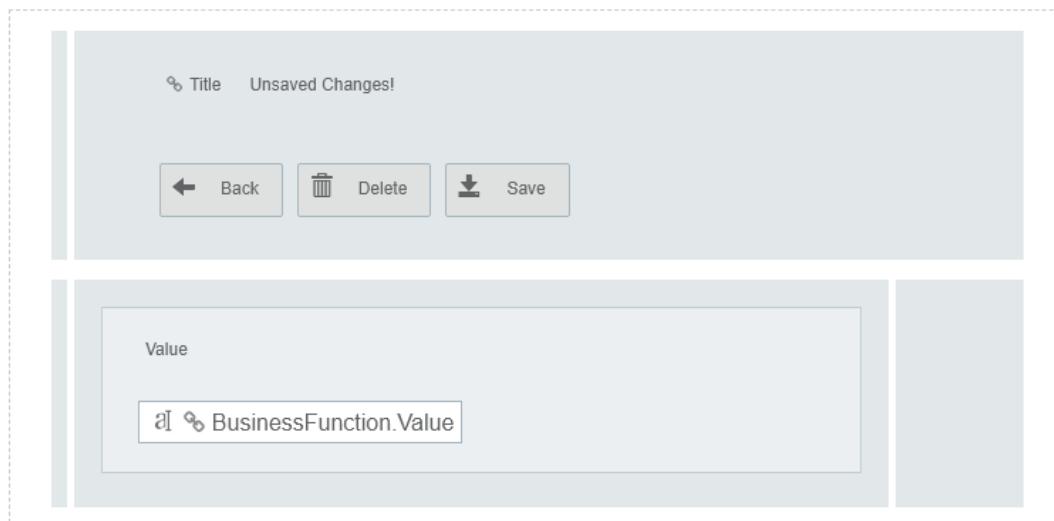
```
function void Index() { Model.Title = LocalResources.RES_PAGETITLE_Index; }
```

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**

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

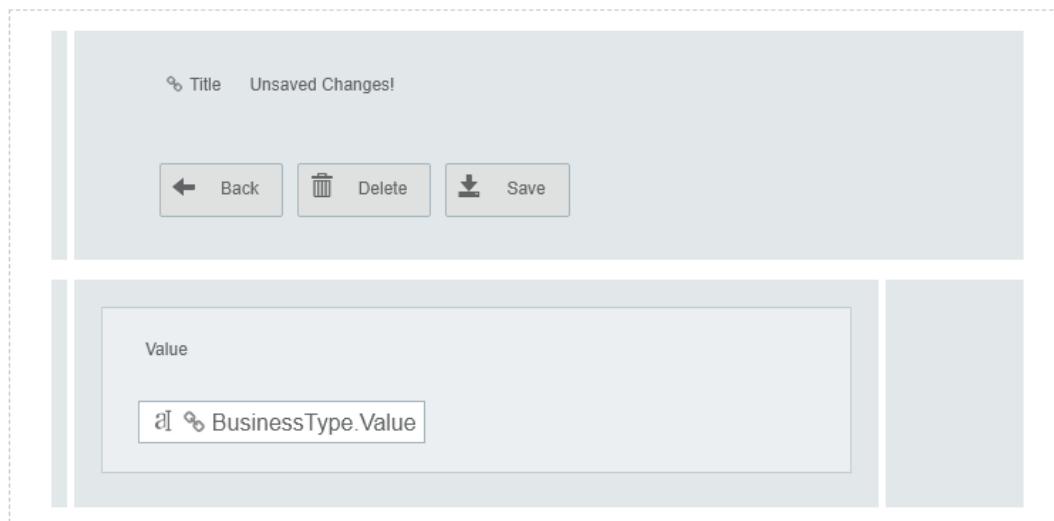
```
function void Index() { Model.Title = LocalResources.RES_PAGETITLE_Index; }
```

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

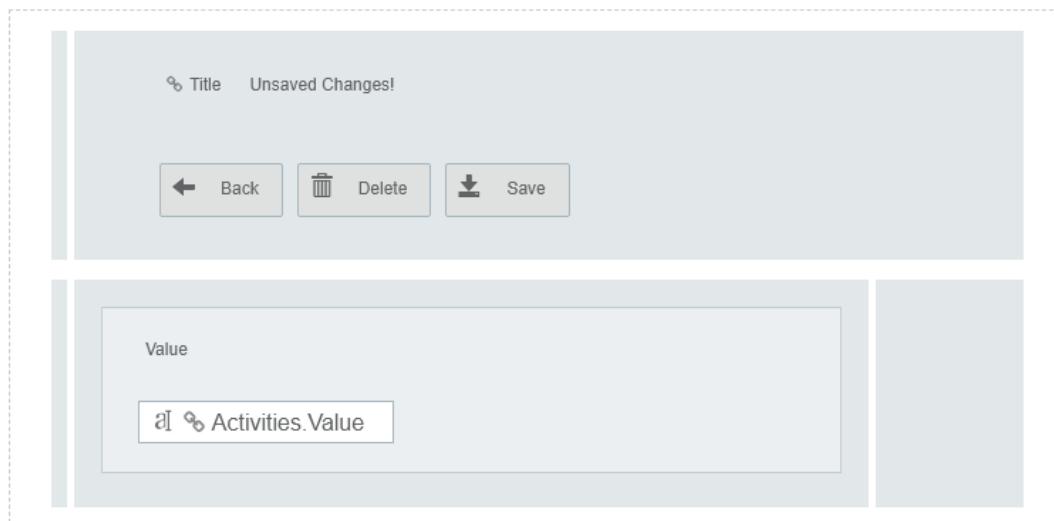
```
function void Index() { Model.Title = LocalResources.RES_PAGETITLE_Index; }
```

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**

	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 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

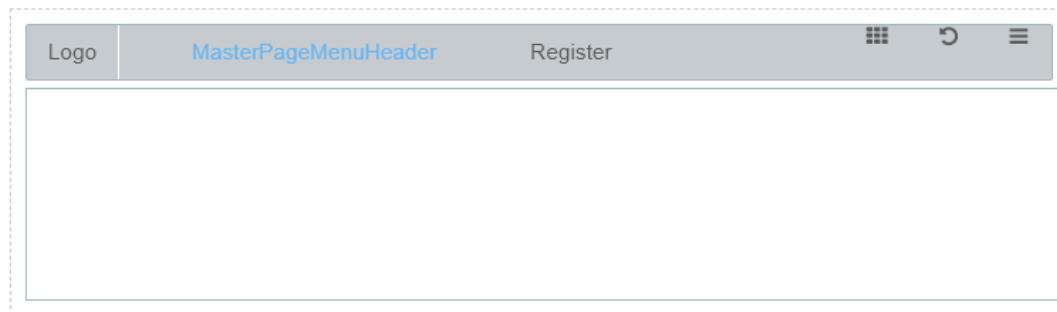
```
function void Index() {
    Model.Title = LocalResources.RES_PAGETITLE_Index;

}
```

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														
Nodes Links Results +														
Page 1 of 10 pages < > <> >>														
25 per page														
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

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

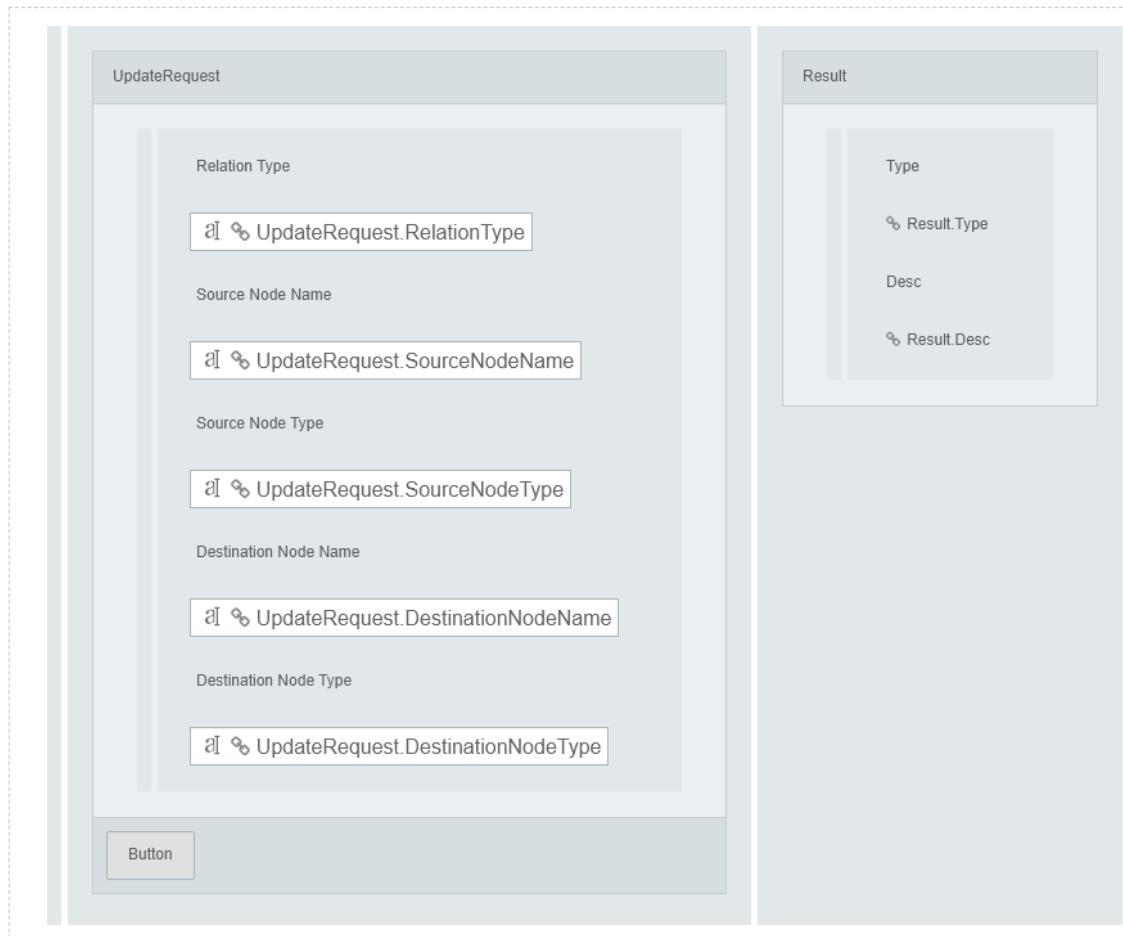
Model

```

|-- 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

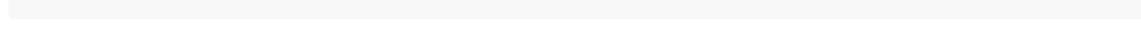
Allow Anonymous	Allow All Authenticated	All Groups	Selected Groups	Permissions
<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		Developer

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	
Expertise.Code	
Display Value	
Expertise.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.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

% 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

The screenshot shows a user interface for editing a sector type. At the top, there is a header bar with the title "Title" and a message "Unsaved Changes!". Below the header are three buttons: "Back" (with a left arrow icon), "Delete" (with a trash can icon), and "Save" (with a downward arrow icon). The main content area contains two sections: "Sector Id" and "Display Value". The "Sector Id" section contains a field labeled "SectorType.Code" with a value "aI % SectorType.Code". The "Display Value" section contains a field labeled "SectorType.Value" with a value "aI % 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

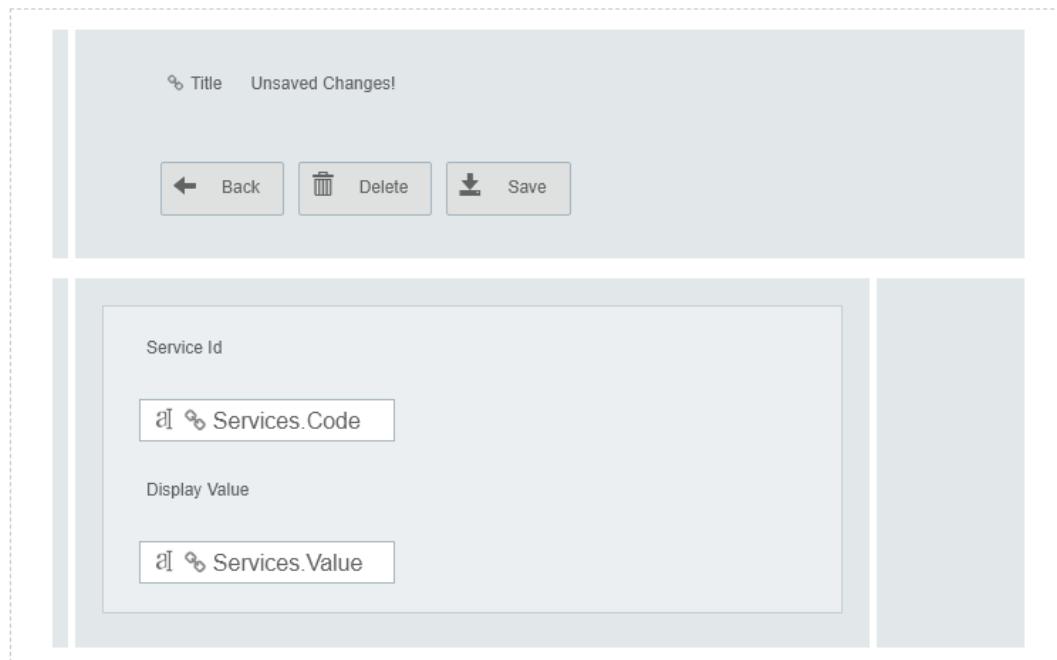
```
function void CleanDuplicates(Domain.SectorType sector)
{
    Domain.DataHelper.CleanDuplicateSectors(sector);
    FormControls.SectorTypeList.Refresh();
    ShowMessage("Done");
}
```

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 Thematic Expertise. 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 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 ActionIs 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 ActionIs 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', both with dropdown menus. 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

 % Title % Actor.EntityType.Value Visit Website	<div style="margin-bottom: 10px;"> Please Specify Type % Actor.SpecifiedEntityType </div> <div style="display: flex; justify-content: space-between;"> <div style="width: 45%;"> Member Of Cluster </div> <div style="width: 45%;"> % Actor.Cluster.Name % Actor.ClusterName </div> </div> <div style="display: flex; justify-content: space-between;"> <div style="width: 45%;"> Sector </div> <div style="width: 45%;"> % SelectedSector.Value </div> </div> <div style="display: flex; justify-content: space-between;"> <div style="width: 45%;"> Digital Expertise </div> <div style="width: 45%;"> % Actor.CircularEconomyRequirements.DigitalExpertise </div> </div> <div style="display: flex; justify-content: space-between;"> <div style="width: 45%;"> Describe Experience </div> <div style="width: 45%;"> % Actor.CircularEconomyRequirements.SpecifyExperienceInCircularEconomy </div> </div> <div style="display: flex; justify-content: space-between;"> <div style="width: 45%;"> Experience in Circular Economy? </div> <div style="width: 45%;"> % Actor.CircularEconomyRequirements.GetExperienceInCircularEconomy </div> </div> <div style="display: flex; justify-content: space-between;"> <div style="width: 45%;"> Available Testing Facilities </div> <div style="width: 45%;"> % Actor.CircularEconomyProviderReport.AvailableTestingFacilities </div> </div> <div style="text-align: center; margin-top: 20px;"> % Value </div> <div style="text-align: center; margin-top: 10px;"> Expertise </div>
---	---

Thematic Areas

% Value

Services

% Value

What are you looking for?

Request Digital provider?
Request Thematic Expertise?

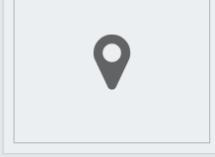
Desired Sector

% Actor.CircularEconomyRequirements.DesiredSMESector ▾

About us

% Actor.Description

Contact us



% Actor.Address.FullAddress

% Actor.Email

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; }
```

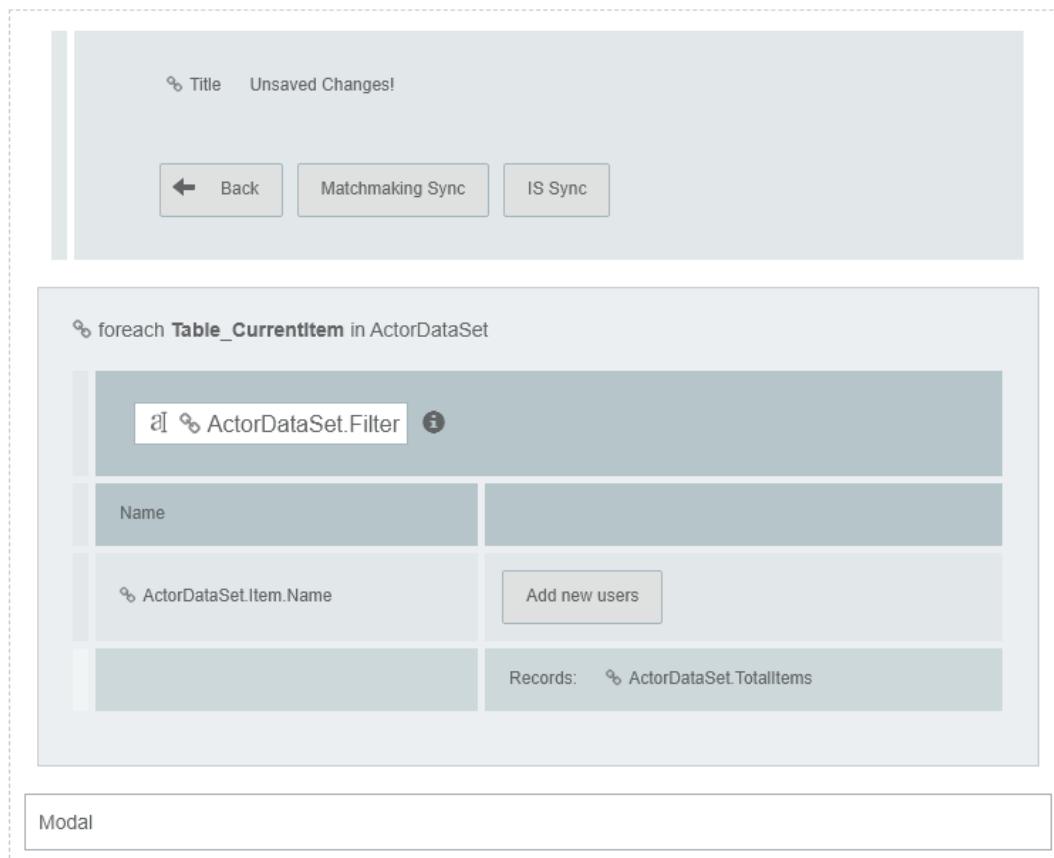
ManageActors Form

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 ActionIs 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 +		
@ % Material.Name	Name	Product	
Type	% Name	% ProductName	Delete Edit
Physical Form			
% Material.PhysicalForm			
Unit Of Measurement	Converted By +		
% Material.UnitOfMeasurement	Name	Source	
Is Hazardous	% Name	% SourceName	Delete Edit
Description			
% % Material.Description			

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																																																																			
Model																																																																			
View																																																																			
<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;"> Add Edit Sync Reset all </div> <div style="display: flex; justify-content: space-between;"> Page 1 of 10 pages </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;"> Add Edit Sync Reset all </div> <div style="display: flex; justify-content: space-between;"> Page 1 of 10 pages </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;"> Add Edit Sync Reset all </div> <div style="display: flex; justify-content: space-between;"> Page 1 of 10 pages </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

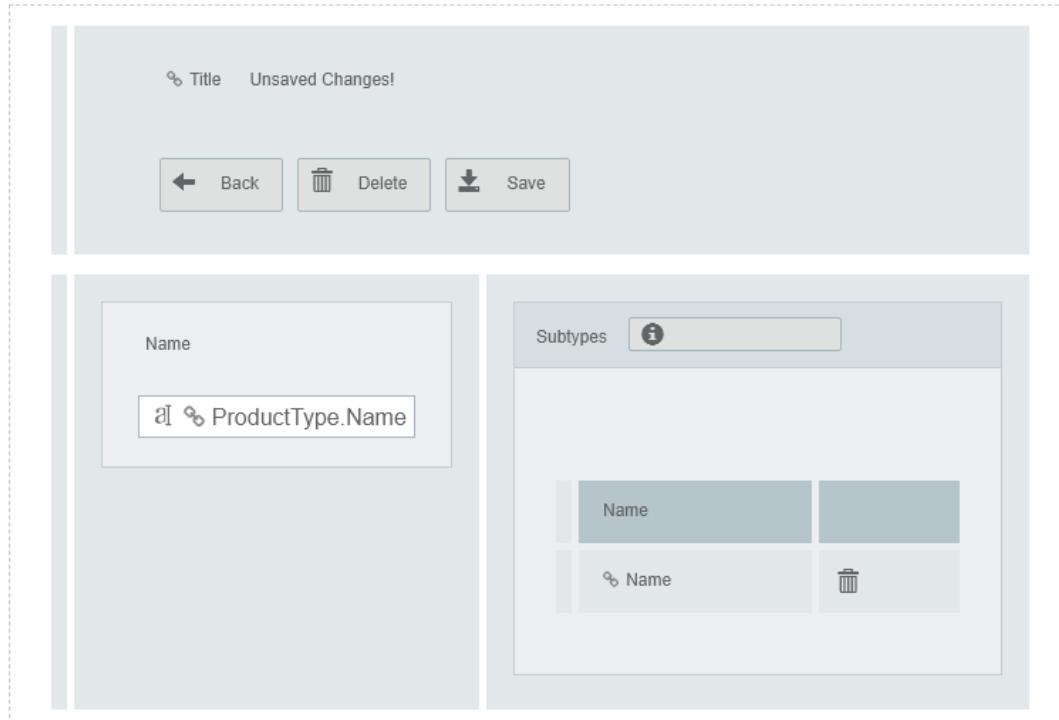
Model

```

|-- 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 page navigation section showing 'Page 1 of 10 pages' and '25 per page'. A table follows, with the first column labeled 'Name'. The table contains three rows of data:

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>Site</p> <div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 10px;"> % % Product.Site ▼ </div> <p>Valid From</p> <div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 10px;"> 📅 % Product.ValidFrom </div> <p>Quantity</p> <div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 10px;"> �� % Product.Quantity </div>	<p>Material*</p> <div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 10px;"> % % Product.Resource ▼ </div> <p>Valid To</p> <div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 10px;"> 📅 % 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

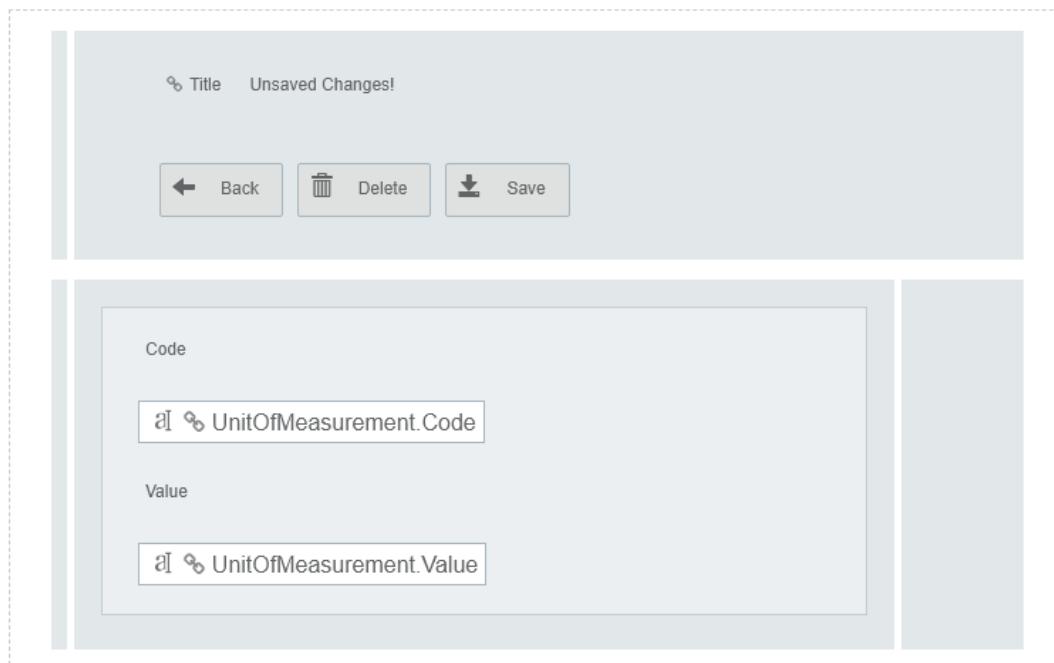
Model

```

|-- 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 next to the input field. Below 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"/>		

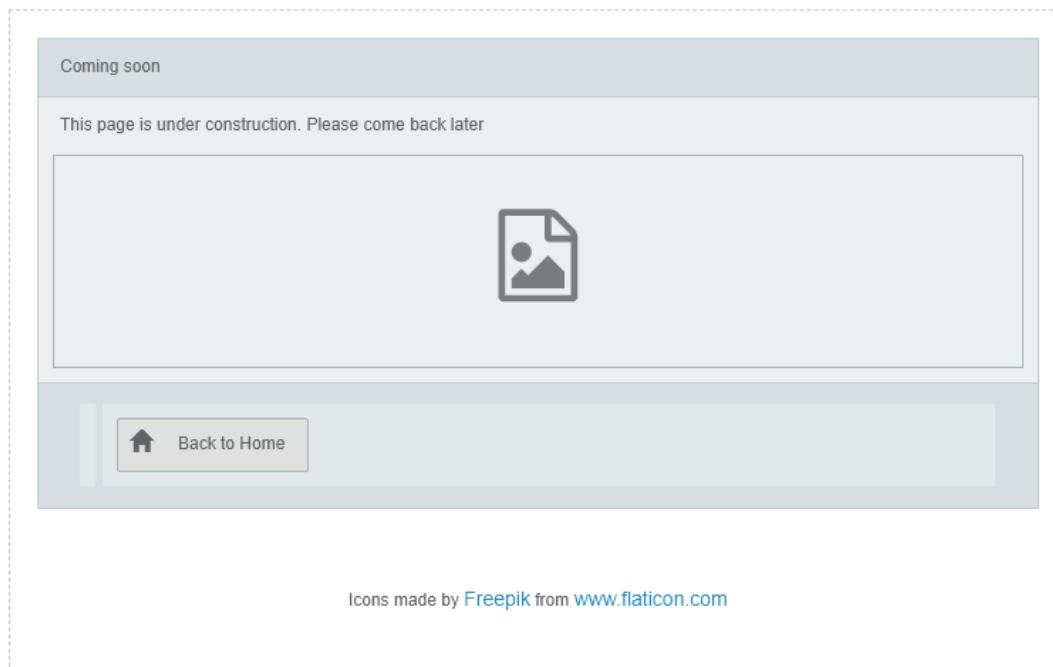
CODE

```
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 ActionIs 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);
    }

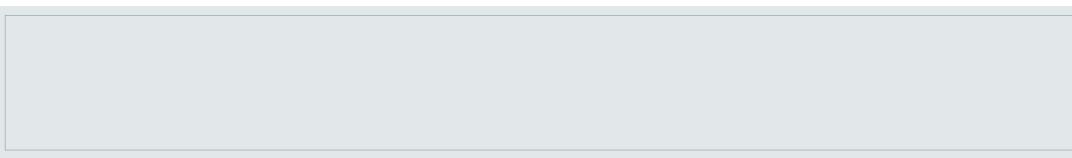
    ShowMessage("Finished");
}
```

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-6.17,5.6-16.17,6.19-21.46,6.19H414l0-48.78,17.35,0c6.45,0,15.86,88,21.76,7.5,29,5.28,5.87,12.34,5.87,17.64,0,4.7-28,12.62-6.43,17.92m92.69,100.29c-6.17,5.6-16.17,6.2-21.47,6.22h-17l0-48.83H524c6.45,0,15.87,88,21.75,7.5,29,5.28,5.89,12.35,5.89,17.64,0,4.7-28,12.63-6.45,17.92"/> <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-10.3,0-08-109.65,12.35,0c12.64,0,28.51,1.15,40.58,12.3,9.72,9.12,15.31,24.71,15.31,42.93,0,21.74-8.48,34.09-15.40,28"/> <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.