

# Documentation & Project Diary

Innovation Lab 1-2  
Year 2021-2022

Project: **WebApp for social Impulse analysis**

Team: **Group 14**

# General Information

**Project name:** WebApp for social Impulse analysis

**Supervisor:** Florian Eckkrammer

Innovation Lab 2, summer term 2022

## Project team:

Brandenburg Jonas, if20b243@technikum-wien.at

Dohnalek Raphael, if20b075@technikum-wien.at

Duskanich Markus, if20b078@technikum-wien.at

Duskanich Michael, if20b077@technikum-wien.at, project manager

Meindl Tobias, if20b125@technikum-wien.at

## Management Summary of the Project

This project concerns itself with the topic of social impulse analysis. Social impulse analysis is all about recording, visualizing, and evaluating social connections within a group. A software developed in 2010 already provides a basis for this project:

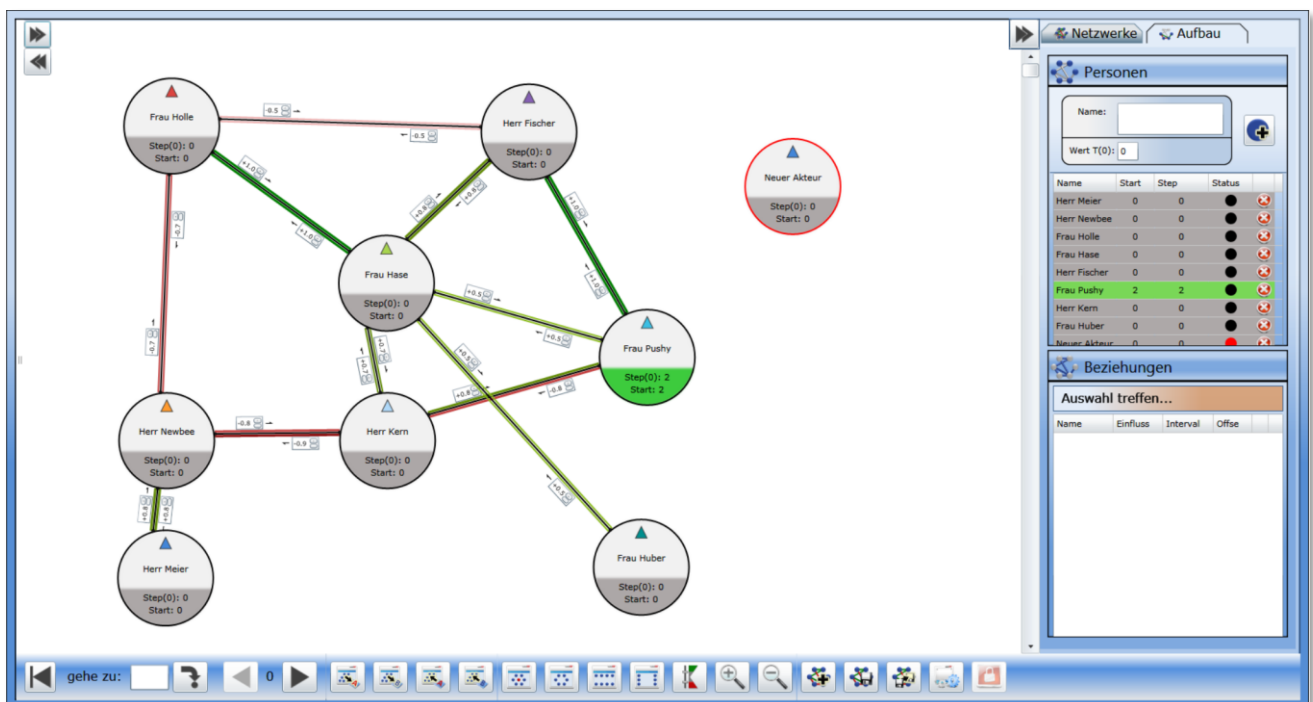


Figure 1 The 2010 developed A-SIA interface

As it turned out, the software, A-SIA, became outdated, leading to the main task: Redesigning the frontend with a modern framework, as well as outsourcing resource heavy computation to a server-side application.

## Framework Conditions and Project Environment

This project requires a full stack development. As the project might be expanded in the future, it must be well documented.

### Frontend:

- Angular
- Jasmin / Karma
- Compodoc
- SCSS

As there already exists prior knowledge with the MEAN stack, Angular was chosen as a frontend Framework.

Angular natively includes Jasmin and Karma, being the obvious choice for unit testing. For automatic code documentation, Compodoc was chosen – a popular open source, Angular-friendly tool.

### Backend:

- C#
- Neo4J

As there will be a lot of calculation and processing related to graphs, a graph database is the obvious choice. Neo4J provides a free and easy to use community edition.

## Semester-Roadmap

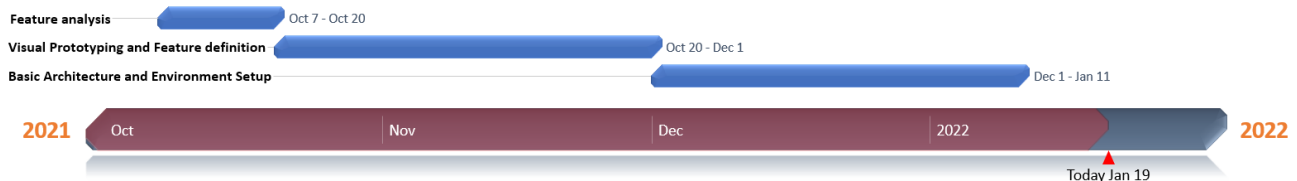


Figure 2 Semester Roadmap INNO1

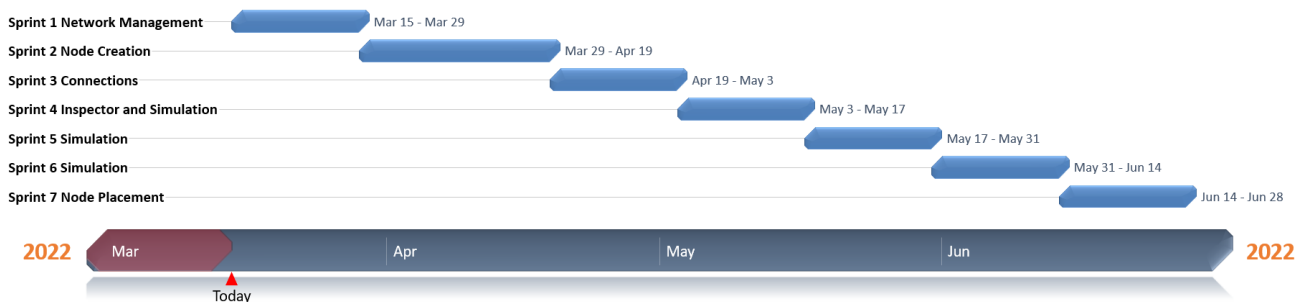


Figure 3 Semester Roadmap INNO2

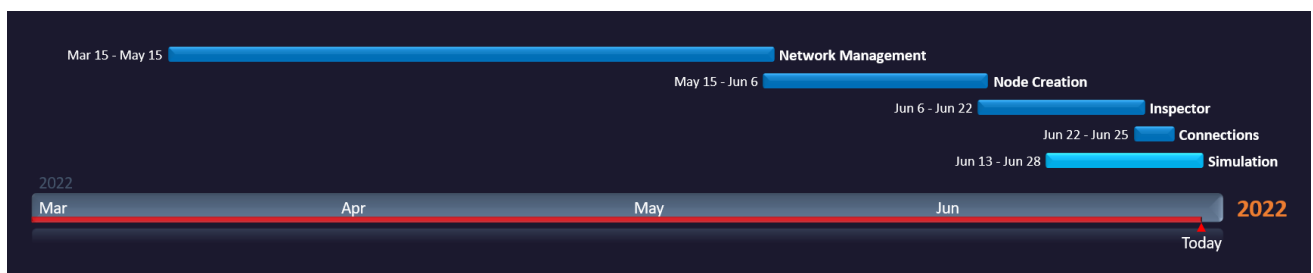


Figure 4 Real Road INNO2

## Collaboration & Tooling

- ALM - [Azure DevOps](#)
- Communication - [Zoom](#)
- VC - Azure DevOps (Snapshot available on [GitHub](#))
- Mockups - [Figma](#)

# Brief Description of the Project

- What is the theoretical background of this app?
  - A-SIA was a software for modeling teams and social networks. The tool supported the recording of the influencing structures, which could either be entered manually or be automatically generated from answered questionnaires. The relationship patterns collected in this way were graphically displayed and support the processing of social and group dynamic phenomena in teams. Furthermore, it simulated *What-If* scenarios, visualizing group dynamic effects.
- Why does it have to be reworked?
  - A-SIA was developed on *Microsoft Silverlight*, now unsuitable for modern app development. Moreover, a lot of computation is done client-side. A visual refactoring, as well as a redistribution of server-side calculation is required.
- Who is the target audience / customer?
  - A-SIA was used by a multitude of different companies. This project, however, does not necessarily need to be deployment ready with multiple pricing options. The focus lies on a well-documented and futureproof web application.
- What are the main features of the application?
  - *Visualizing a sociogram:*  
A sociogram consists of actors and relations. Those will have to be created manually (and possibly generated via an auto generated form) within a user interface, which is displayed as an undirected graph. Trust, mistrust, irritation, insistency and *echodampening* are metrics that can be helpful for a more descriptive display. Different arrangements and filtering options should be available.
  - *Simulation:*  
The prominent premise of this application is, that it can simulate social dynamics. For example, an actor can be selected to propose an idea. Then, the user of the software can step through the simulation.  
The algorithm for this simulation will be provided.
  - *Backend:*  
Data (sociograms) can be stored and loaded from a database. Furthermore, the computation will be done server-side, reducing the load on the client.
- What are non-targets of this project?
  - Improving the simulation algorithm.
  - A mobile first approach
  - Keeping the calculation client-side

# Specification of the Solution

## Prototyping and User Stories

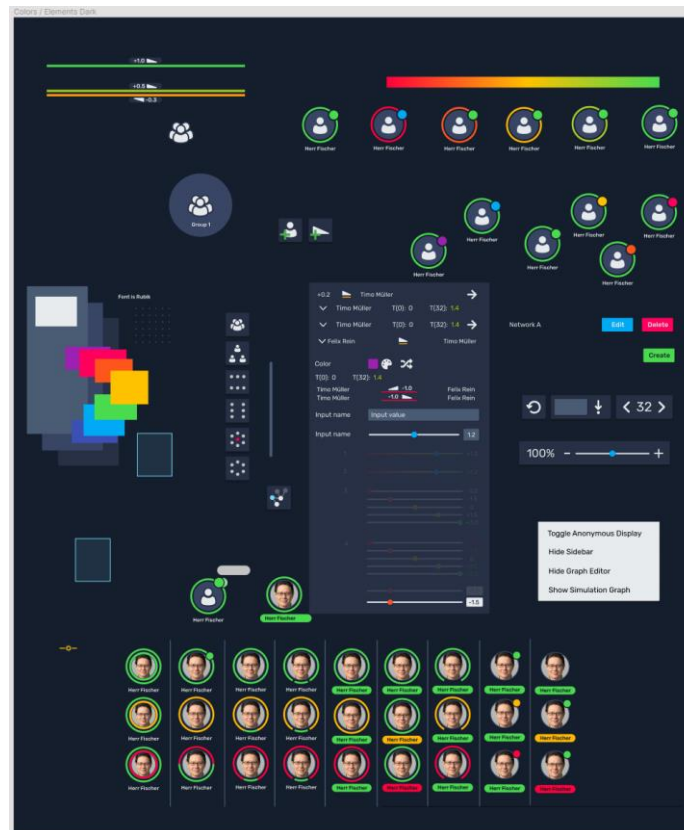


Figure 5 Collection of prototyped elements

## User wants to build their network from external data

ID	User Story	Description	Story Points
604	User wants to generate network from survey data	A user can generate a network from existing (self made) survey data. This story concerns itself with visual design, frontend logic code, backend code and database access.	-
605	User wants to generate network from external tool	A user can generate a network from external survey data (sources TBD). This story concerns itself with visual design, frontend logic code, backend code and database access.	-

# Menubar

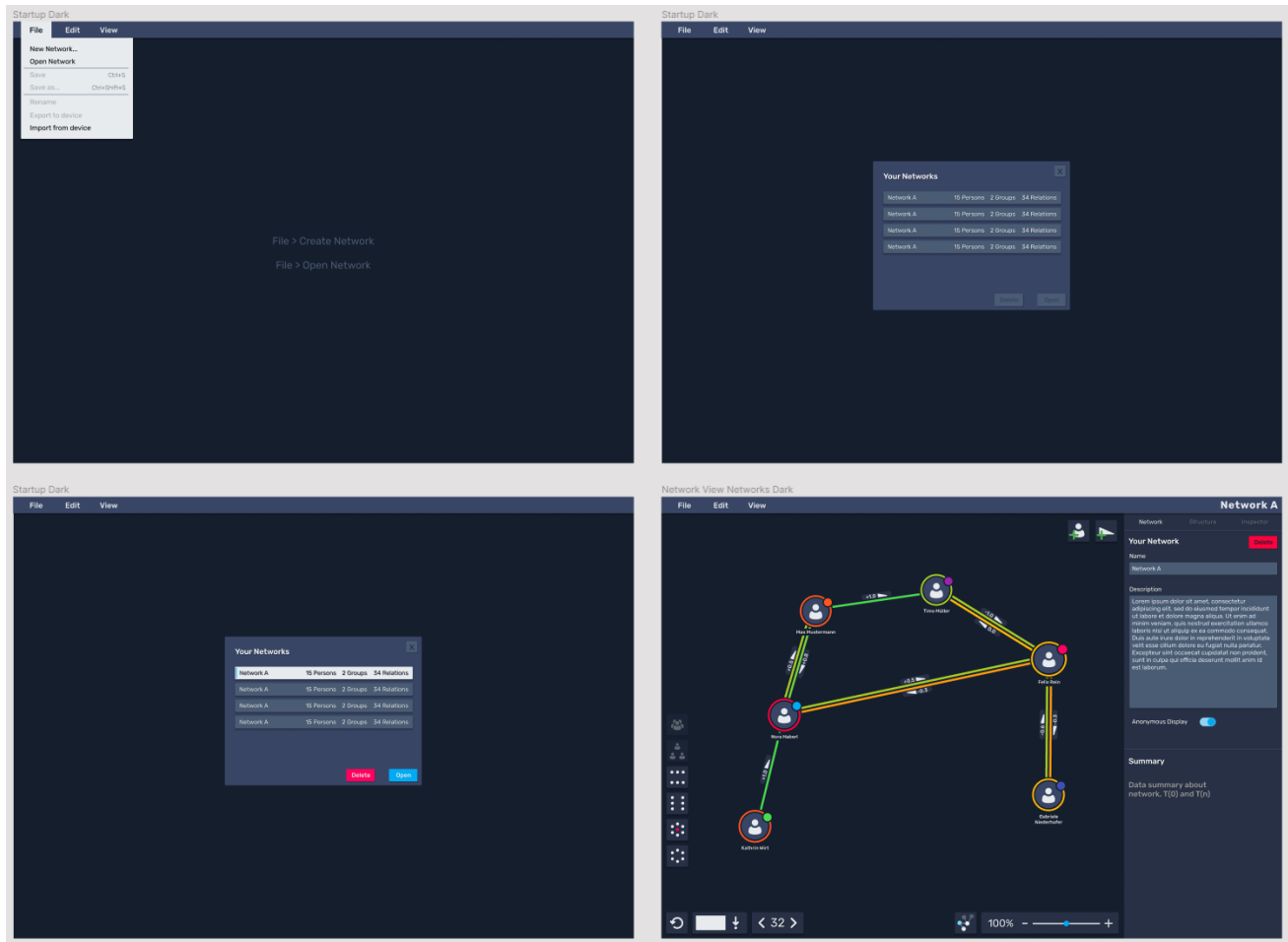


Figure 6 Creating a Network through the Menubar

ID	User Story	Description	Story Points
509	User can use File Menubar Menu	A user can open the File menu in the Menubar, seeing all its relevant entries. This story only encompasses the menu, and an access point through which the relevant function (e.g. New Network, Open Network, Save, Save as..., Rename, Export to device, Import from device,...) can be executed, but NOT their implementation. Menubar entries should be easy to extend.	2
510	User can use Edit Menubar Menu	A user can open the Edit menu in the Menubar, seeing all its relevant entries. This story only encompasses the menu, and an access point through which the relevant function (e.g. Undo, Redo, Preferences,...) can be executed, but NOT their implementation. Menubar entries should be easy to extend.	2
511	User can use View Menubar Menu	A user can open the View menu in the Menubar, seeing all its relevant entries. This story only encompasses the menu, and an access point through which the relevant function (e.g. Toggle Anonymous Display, Toggle Sidebar, Toggle Graph Editor, Toggle Simulation Diagram,...) can be executed, but NOT their implementation. Menubar entries should be easy to extend.	1

# Highlevel Network Management

ID	User Story	Description	Story Points
513	User wants to create Network	When a user navigates to File>Create Network in the menubar, they can create a network through the click of a menubar entry, followed by being automatically opening the new Network, with the sidebar being set to the Network tab. This story encompasses visual design, frontend logic code, backend code, and database connection.	2
514	User wants to open Network	When a user navigates to File>Open Network in the menubar, they can open a network through the click of a menubar entry, followed by a pop up containing a list of all their networks, Upon selection of a Network, a Open button will be clickable. This story encompasses visual design, frontend logic code, backend code, and database connection.	2
515	User wants to Export their network	When a user navigates to File>Export Network in the menubar, they can export their network through the click of a menubar entry. It will probably get exported as JSON, because the REST API also communicates through JSON. This story encompasses visual design and frontend logic code.	3
516	User wants to import a network	When a user navigates to File>Import Network in the menubar, they can import their network through the click of a menubar entry. It will probably get imported as JSON, because the REST API also communicates through JSON. This story encompasses visual design and frontend logic code.	3
622	Save Network As (NEW!)		

## Sidebar Network

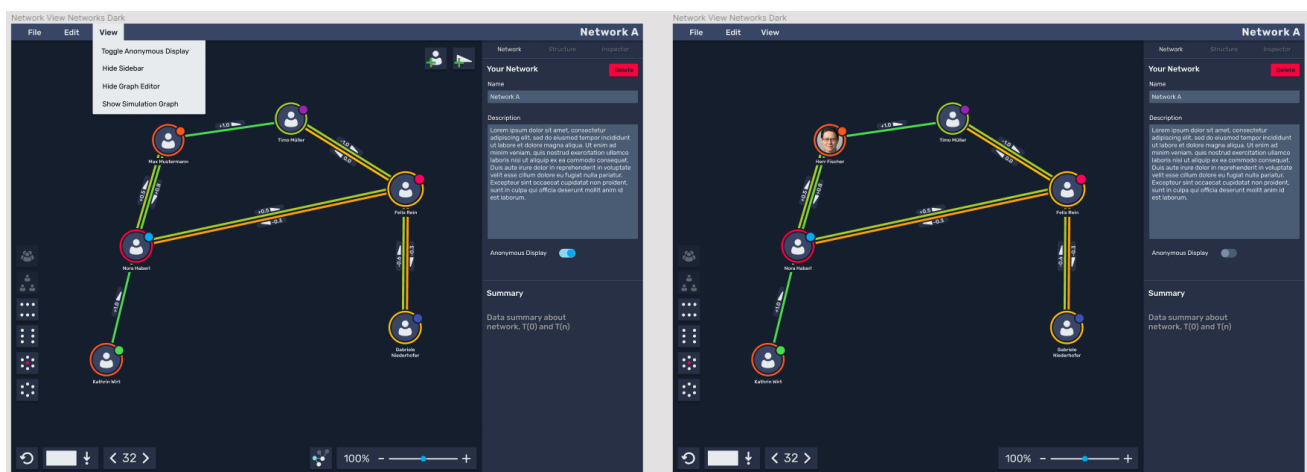


Figure 7 Toggling anonymity status

ID	User Story	Description	Story Points
520	User wants to change Network Name	The user can change the Network Name through the Network Tab in the Sidebar. This story encompasses visual design and frontend logic code, backend code and Database access.	5



521	User wants to change Network Description	The user can change the Network Description through the Network Tab in the Sidebar. This story encompasses visual design and frontend logic code, backend code and Database access.	5
522	User wants to Anonymize Network Display	The user can toggle the Anonymous Display through the Network Tab in the Sidebar. Upon clicking a toggle switch, all persons node's avatars and names should be anonymized. This story encompasses visual design and frontend logic code, backend code and Database access.	3
523	User wants to access reporting	The user access their reporting (overall summary and detailed customizable person level report) within the bottom part of the Network Sidebar (and Inspector for Person level reporting). This story encompasses visual design.	5
524	User wants to delete network	When a user navigates to Network in the sidebar, they can remove the network through the click of a button, followed by a confirmation pop up. This story encompasses visual design, frontend logic code, backend code, and database connection.	2

## Sidebar Inspector

ID	User Story	Description	Story Points
534	User wants to access person data through inspector	The user can access a person's data within the Inspector Sidebar. This story encompasses visual design.	1
535	User wants to access group data through inspector	The user can access a group's data within the Inspector Sidebar. This story encompasses visual design.	2
536	User wants to access connection data through inspector	The user can access a connection's data within the Inspector Sidebar. This story encompasses visual design.	1

## Sidebar Structure

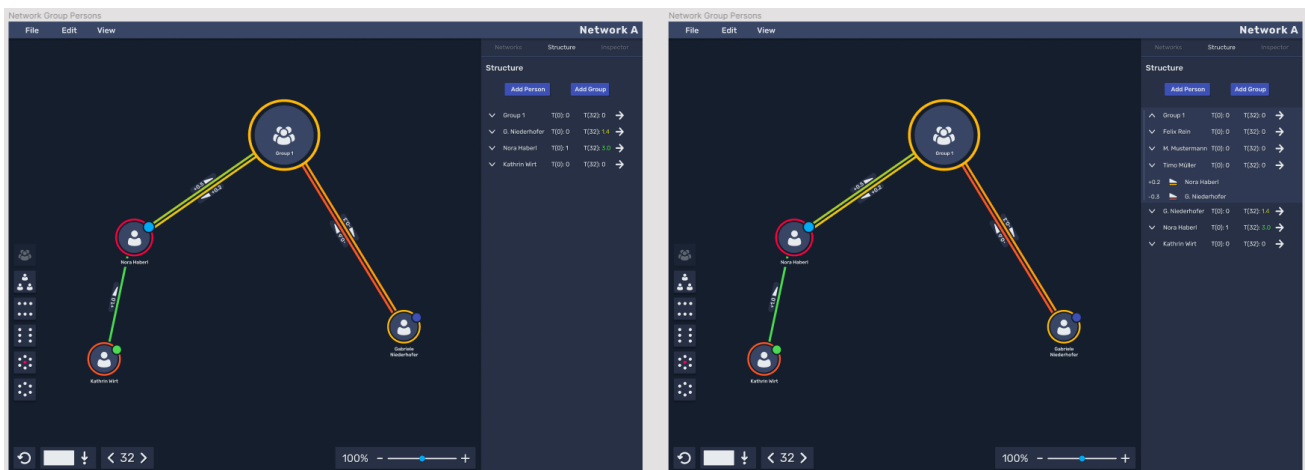


Figure 8 The Structure Tab (1)



Figure 9 The Structure Tab (2)

ID	User Story	Description	Story Points
525	User wants to have a hierarchical list of their network structure	The user can access a hierarchical list of the networks structure within the Structure Sidebar (only a bland list of data). This story encompasses frontend logic code, backend code, and database access.	5
526	User wants to expand / collapse network structure	The user can expand / collapse the network structure within the Structure Sidebar. This story encompasses frontend logic code and visual design.	5
527	User wants to navigate to node via a structure entry	The user can navigate to a node within the Structure Sidebar (move / zoom graph editor, select node, open inspector) . This story encompasses frontend logic code and visual design.	5
528	User wants to add a person through the structure tab	The user can add a person (node) through a button within the Structure Sidebar (auto selection, but no switch to inspector). This story encompasses frontend logic code, visual design, backend code and database access.	3
529	User wants to add a group through the structure tab	The user can add a Group (node) through a button within the Structure Sidebar (auto selection, but no switch to inspector) . This story encompasses frontend logic code, visual design, backend code and database access.	3

## Sidebar

ID	User Story	Description	Story Points
531	User wants to Navigate between Network / Structure / Inspector	Within the sidebar, the user can switch between Network / Structure / Inspector. This story encompasses visual design and frontend logic code.	2
532	User wants to toggle Sidebar	The user can hide / show the Sidebar through View>Show/Hide Sidebar . The Graph Editor / Diagram should automatically fill out the empty space. This story encompasses visual design and frontend logic code.	1
533	User wants to resize Sidebar	The user can resize the Sidebar through a handle along the element. The Graph Editor / Diagram should automatically resize accordingly. This also means, that all content has to be designed responsively. This story encompasses visual design and frontend logic code.	2

## Inspector Person

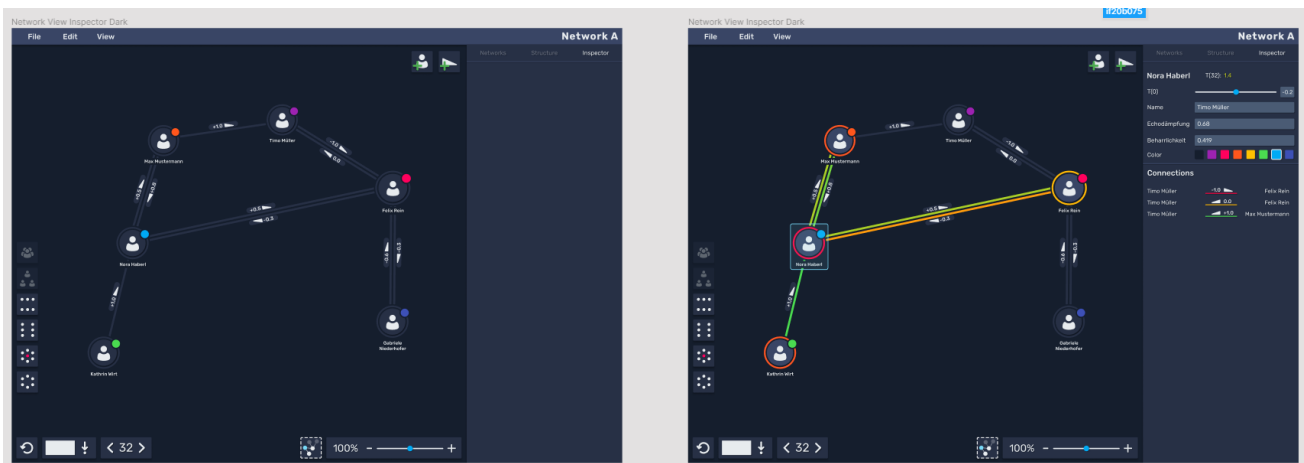


Figure 10 Selecting a node in filter mode

ID	User Story	Description	Story Points
540	User wants to change T(0) through slider in inspector	The user can change T(0) through a limited slider or manual input (min and max values) within the Inspector Sidebar. This story encompasses frontend logic code, visual design (if not already done), backend code and database access.	2
541	User wants to change person name through in inspector	The user can change the person name through a text field within the Inspector Sidebar. This story encompasses frontend logic code, visual design (if not already done), backend code and database access.	1
542	User wants to change person echodämpfung / beharrlichkeit in inspector	The user can change echodämpfung / beharrlichkeit through input fields within the Inspector Sidebar. This story encompasses frontend logic code, visual design (if not already done), backend code and database access.	1

<b>543</b>	User wants to change person color randomly in inspector	The user can change a node's color randomly with the click of a button within the Inspector Sidebar. This story encompasses frontend logic code, visual design (if not already done), backend code and database access.	2
<b>544</b>	User wants to see all adjoining connections to the selected Person in inspector	The user can see all connections going in / out of a person's node. This story encompasses frontend logic code, visual design, backend code and (maybe) database access.	2
<b>545</b>	User wants to see all names and connecting connections between multiple persons in inspector	The user can see all connections between multiple selected person nodes within the inspector. This story encompasses frontend logic code, visual design, backend code and (maybe) database access.	2
<b>546</b>	User wants to see persons special role(s) in inspector	User can see persons special role(s) in inspector. This only concerns itself with the visual display (see #607 #609 ). This story encompasses visual design and frontend logic code.	5
<b>547</b>	User wants to change person color through a color selector in inspector	User can change the color through a color selector (pop up) in the inspector. This story encompasses visual design, frontend logic code, backend code and database access.	1
<b>589</b>	User wants to change Avatar in inspector	User can change their avatar in the inspector. They can also choose a general icon. This story encompasses visual design, frontend logic code, backend code and database access.	5
<b>607</b>	Developer wants to have an expandable calculation process of KPIs	A Developer can add calculations for special roles and metrics concerning persons and the network. The effort of this story will get evaluated / split up as soon as the team has informed themselves about special roles and their calculations. This story encompasses backend code and database access (and the whole calculation of special roles as a whole).	-

# Inspector Group



Figure 11 Selecting a Group

ID	User Story	Description	Story Points
551	User wants to see averaged $T(0)$ , $T(t)$ , Echodämpfung, Beharrlichkeit of the group in the inspector	As written in title. This story encompasses visual design, frontend logic code, backend code and database access.	3
552	User wants to change color of the group in the inspector	As written in title. Same options as a Person node (random, selector). This story encompasses visual design (should mostly be done), frontend logic code, backend code and database access.	1
553	User wants to see averaged connections of the group in the inspector	The user can see the averaged outgoing / ingoing nodes of a group in the inspector. This story encompasses visual design (should mostly be done), frontend logic code, backend code and database access.	3
554	User wants to see containing nodes of the group in the inspector	The user can see the containing nodes of a group in the inspector. This story encompasses visual design (should mostly be done), frontend logic code, backend code and database access.	1

555	User wants to see containing relations of the group in the inspector	The user can see the containing relations of a group in the inspector. This story encompasses visual design, frontend logic code, backend code and database access.	1
556	User wants to see expand / collapse group through a button in the inspector	The user can expand / collapse a group through the click of a button in the inspector (only linking button with functionality; for the actual expanding / collapsing see #581 ). This story encompasses visual design, and frontend logic code.	1

## Inspector Connection

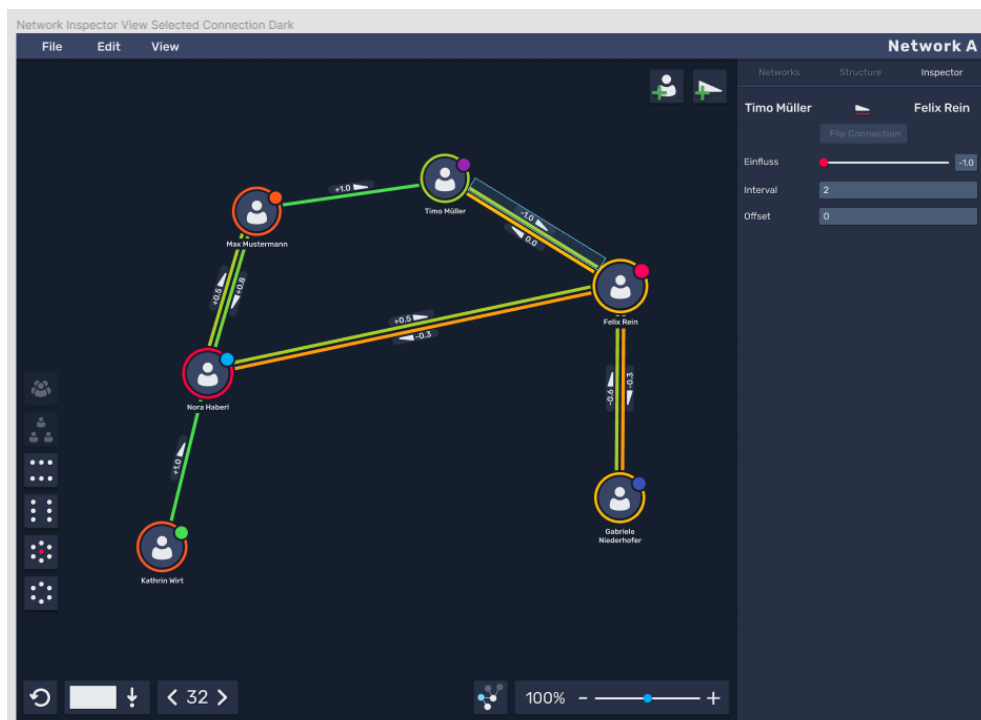


Figure 12 Selecting a connection

ID	User Story	Description	Story Points
548	User wants to change Connection direction through the click of a button	A User can click on a button to flip a selected collection (or swap if there is one in the opposite connection). This should be noticeable so the user does not miss it (animation, info pop up). This story encompasses visual design, and frontend logic code, backend code and database access.	2
549	User wants to change connection interval and offset via inspector	A User can change interval and offset (input field) . This story encompasses visual design, and frontend logic code, backend code and database access.	1

<b>550</b>	User wants to change connection Einfluss in inspector	A User can change influence through a slider (as number) or dropdown (as text). This story encompasses visual design, and frontend logic code, backend code and database access.	1
------------	---	--	---

## Database

ID	User Story	Description	Story Points
<b>595</b>	Developer wants to have a basic graph database they can access	The developer can access a relatively finished graph database (neo4j) (aka database / schema design). This story encompasses database access and backend code.	5
<b>606</b>	Developer wants to have a basic relational database they can access	The developer can access a relatively finished relational database (aka database / schema design). This story encompasses database access and backend code.	3
<b>611</b>	User wants to login (new Discussion)		2

# Graph Editor Navigation + Selection

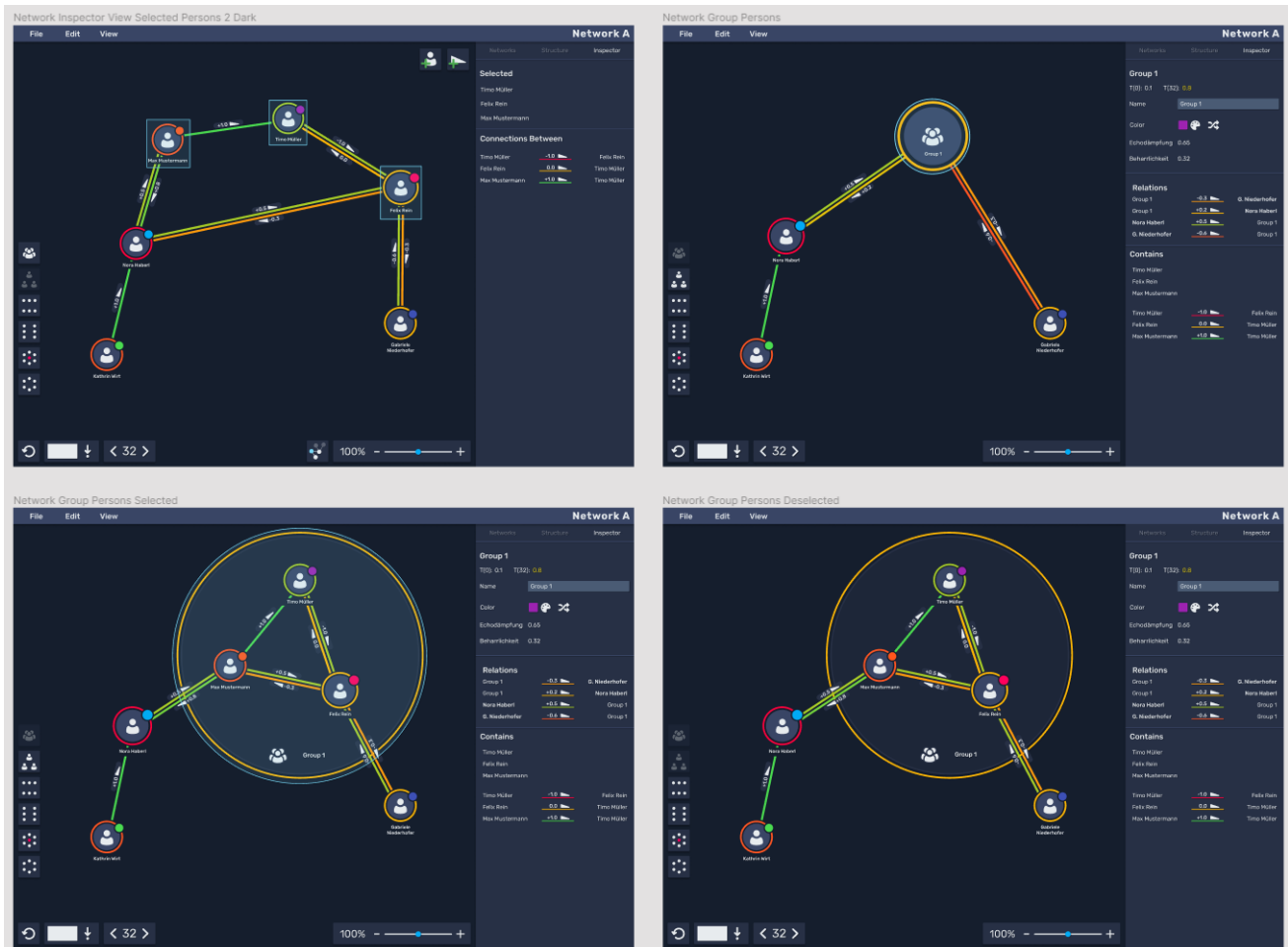


Figure 13 Grouping and ungrouping nodes

ID	User Story	Description	Story Points
574	User wants to zoom in graph editor	A User zoom by using their scroll wheel. There should be a function within the frontend that is usable for zooming to a fixed value / position. This story encompasses visual design, and frontend logic code.	5
575	User wants to pan in graph editor	A User pans by left click + dragging. There should be a function within the frontend that is usable panning to a fixed value / position. This story encompasses visual design, and frontend logic code.	5
576	User wants to select Nodes / Connections in graph editor	A User selects a or multiple nodes by (shift) + left clicking. Keep tablet controls in mind. There should be a function within the frontend that takes a node and selects it. This story encompasses visual design, and frontend logic code.	3
577	User wants to move Nodes in graph editor	A User moves a node by left click + dragging. There should be a function within the frontend that moves a node to a specific position. This story encompasses visual design, frontend logic code, backend code and database access .	5



578	User wants to move Nodes into groups in graph editor	A User moves a node into an existing group, which automatically adds it (and pushes it away from the border). This story encompasses visual design, frontend logic code, backend code and database access .	8
579	User wants to move Nodes out of groups in graph editor	A User moves a node out of an existing group, which automatically removes it (and pushes it away from the border). This story encompasses visual design, frontend logic code, backend code and database access .	8
582	User wants to toggle graph editor	The user can hide / show the graph editor through View>Show/Hide graph editor . The Sidebar / Diagram should automatically fill out the empty space. This story encompasses visual design and frontend logic code.	2
583	User wants to resize graph editor	The user can resize the Graph editor through a handle along the element. The Sidebar / Diagram should automatically resize accordingly. This also means, that all content has to be designed responsively. This story encompasses visual design and frontend logic code.	2

## Graph Editor Tools

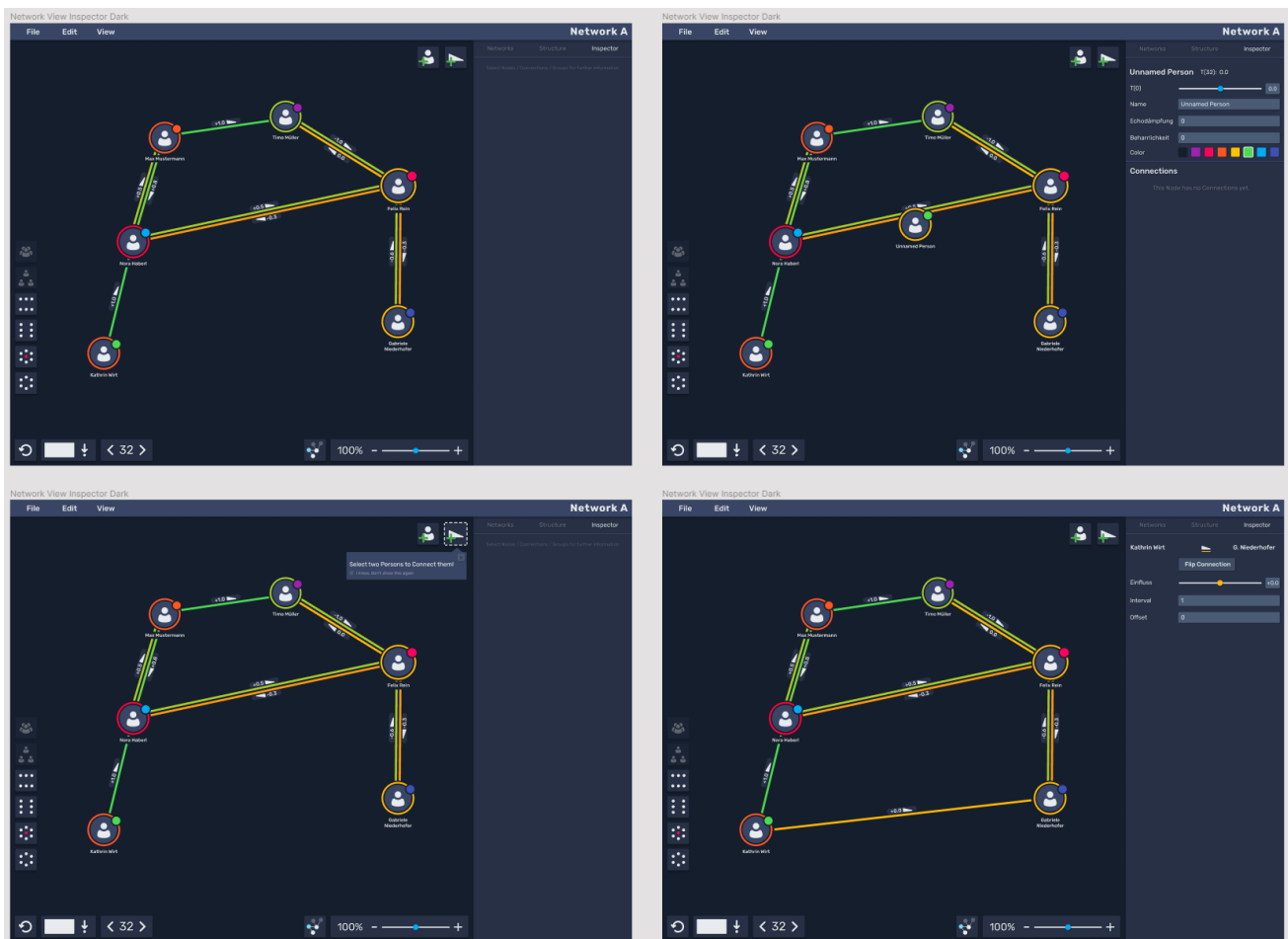


Figure 14 Graph Editor Tools

ID	User Story	Description	Story Points
564	User wants to reset simulation in graph editor	A User can reset the simulation through a click of a button in the graph editor. They should get prompted if a value changes, which would require them to rerun the simulation. This story encompasses visual design, frontend logic code, backend code and database access.	2
565	User wants to skip to simulation frame in graph editor	A User can skip to a specific frame in the simulation through a click of a button, and an input field in the graph editor. This story encompasses visual design, frontend logic code, backend code and database access.	8
567	User wants to navigate plus / minus one simulation frame in graph editor	A User can advance / go back one frame in the simulation through the click of two buttons in the graph editor. This story encompasses visual design, frontend logic code, backend code and database access.	2
569	User wants to create person in graph editor	A User can create a person through the click of two buttons in the graph editor. This story encompasses visual design, frontend logic code, backend code and database access.	3
571	User wants to create connection in graph editor	A User can create a connection, by activating the connect tool through the click of a button in the graph editor. The connection is created when they select two nodes (or hold and drag to another). This story encompasses visual design, frontend logic code, backend code and database access.	5
572	User wants to automatically Arrange nodes in graph editor (TODO!)	A User can automatically arrange their network (Formations: circle, circle around selected, opposite horizontal, opposite vertical). There should also be frontend functions for procedural positioning (e.g. when creating nodes, groups finding a free space). This story encompasses visual design, frontend logic code, backend code and database access.	13
573	User wants to enable "connected selection filtering" in graph editor	User wants to enable "connected selection filtering" through a button in the graph editor. Nodes / connections not directly connected with the selected node will be greyed out. This story encompasses visual design, frontend logic code, backend code and database access.	2
580	User wants to group / ungroup nodes in graph editor	User can to group / ungroup nodes in graph editor. If they select a node or multiple nodes, the group button will be enabled. Pressing automatically groups. A selected group can get ungrouped through an enabled ungroup button. This story encompasses visual design, frontend logic code, backend code and database access.	8
581	User wants to collapse / expand group nodes in graph editor	User can to collapse / expand group nodes. This brings multiple challenges with it, as it requires some procedural placement of surrounding, and maybe containing nodes. It also depends on the knowledge which connections are fully contained within the group, and which are outgoing. This story encompasses visual design and frontend logic code.	20

621	User can define Start Value of Nodes before/during simulation (NEW!)		-
641	User wants to simulate the network	The simulation gets calculated in the backend, and returns a step, or the whole simulation of $T(x)$ . The $T(x)$ value is also displayed through the node color.	-

## Simulation Diagram



Figure 15 Simulation diagram

ID	User Story	Description	Story Points
585	User wants to toggle Simulation Diagram	The user can hide / show the Simulation Diagram through View>Show/Hide Simulation Diagram . The Graph Editor / Sidebar should automatically fill out the empty space. This story encompasses visual design and frontend logic code.	2
586	User wants to resize Simulation Diagram	The user can resize the Simulation Diagram through a handle along the element. The Sidebar / Graph Editor should automatically resize accordingly. This also means, that all content has to be designed responsively. This story encompasses visual design and frontend logic code.	2
587	User wants to see a legend for the Simulation Diagram	The user can see a legend for their diagram, similar to the structure tab. This story encompasses visual design, and frontend logic code.	2

<b>588</b>	User wants to expand / collapse groups through the legend within the Simulation Diagram	The user can collapse / expand groups in the legend, the diagram will update accordingly (similar to the structure tab). This story encompasses visual design, and frontend logic code.	1
<b>591</b>	User wants to see the simulation diagram	As in title. This story encompasses visual design, and frontend logic code.	5

## Graph Editor Display

ID	User Story	Description	Story Points
<b>592</b>	User wants to see their person in the Graph editor	This story only encompasses the visual design of the person node.	3
<b>609</b>	User wants to see persons special role(s) in graph editor	A User can see special roles and metrics concerning persons and the network. If enabled, they can notice special icons overlaid above the nodes. The effort of this story will get evaluated / split up as soon as the team has informed themselves about special roles and their calculations. This story encompasses visual design, frontend logic code, backend code and database access (and the whole calculation of special roles as a whole).	-
<b>593</b>	User wants to see their connection in the Graph editor	This story only encompasses the visual design of the connection. Be aware of automatic placement of labels.	5
<b>594</b>	User wants to see their group in the Graph editor	This story only encompasses the visual design of the group node. (expanded + collapsed view, without positioning)	5
<b>623</b>	User wants to see network specific metrics (NEW)		-

## Menubar Tools

ID	User Story	Description	Story Points
597	User wants to access / change settings	The user can access and change settings through edit > settings . This opens a popup, which lists relevant settings (anonymize, word / number labels for influence, previously config options ). This story encompasses visual design, frontend logic code, database access and backend code.	5
598	User wants to undo / redo	The user can undo / redo operations through edit>undo and edit>redo. They should get warned if they do an irreversible action. This story concerns itself with visual design and frontend logic code.	5
610	User can use common hotkeys	Common hotkeys like ctrl+z, ctrl+y, ENTF, etc., should be usable. They will be listed at edit>keymap. This story concerns itself with visual design, frontend logic code, (and maybe backend code / database access if they are changable)	3

## User wants to generate a report

ID	User Story	Description	Story Points
600	User wants to have a network level report	A user can generate a network level report, through a existing reporting framework. They can choose which metrics to include ( #603 #523 #597 ). This story concerns itself with visual design, frontend logic code, (and maybe backend / database access if there are some calculations that are not already stored / done)	-
601	User wants to have a person level report	A user can generate a person level report, through a existing reporting framework. They can choose which metrics to include ( #523 #597 ). This story concerns itself with visual design, frontend logic code, (and maybe backend / database access if there are some calculations that are not already stored / done)	-
602	User wants to define template	A user can choose a template to be used for their reporting. This story concerns itself with visual design, frontend logic code, (and maybe backend / database access if they should be able to store their template)	-
603	User wants to define indicators (FRAGE)	A user can choose which existing metrics to include ( #597 ) in the reporting. This story concerns itself with visual design, frontend logic code, backend code and database access.	-

# Delivery

**Documentation & Repository:** [https://github.com/MichaelDusk2361/INNO2\\_A-SIA](https://github.com/MichaelDusk2361/INNO2_A-SIA)

The Angular documentation was automatically generated using **compodoc**. It can be found in the repository at **src/A-SIA2FrontendAngular/documentation**. There you can find lists of the implemented modules, components, and services as well as diagrams and the source code.

A detailed documentation of our API endpoints can be generated by executing the backend solution. Here are some screenshots of it. For more details and Information on the schemas run the A-SIA2WebAPI.BL.API.exe.

Group	
POST	/Group/{networkId}
DELETE	/Group/{groupId}
PUT	/Group/{groupId}
Network	
POST	/Network
PUT	/Network/{networkId}
DELETE	/Network/{networkId}
NetworkStructure	
GET	/NetworkStructure/{networkId}
PUT	/NetworkStructure/{networkId}
Person	
POST	/Person/{networkId}
PUT	/Person/{personId}
DELETE	/Person/{personId}
POST	/Person/{personId}/Group/{groupId}
DELETE	/Person/{personId}/Group/{groupId}
PATCH	/Person/{personId}/Group/{groupId}
Relation	
POST	/Relation/Person/{fromPersonId}/Influences/{toPersonId}
DELETE	/Relation/Person/{fromPersonId}/Influences/{toPersonId}
PUT	/Relation/Person/{fromPersonId}/Influences/{toPersonId}
User	
GET	/User/{userId}
PUT	/User/{userId}
DELETE	/User/{userId}
POST	/User/Login
POST	/User/Logout
POST	/User
GET	/User/{userId}/Networks

Figure 16 Rest-API

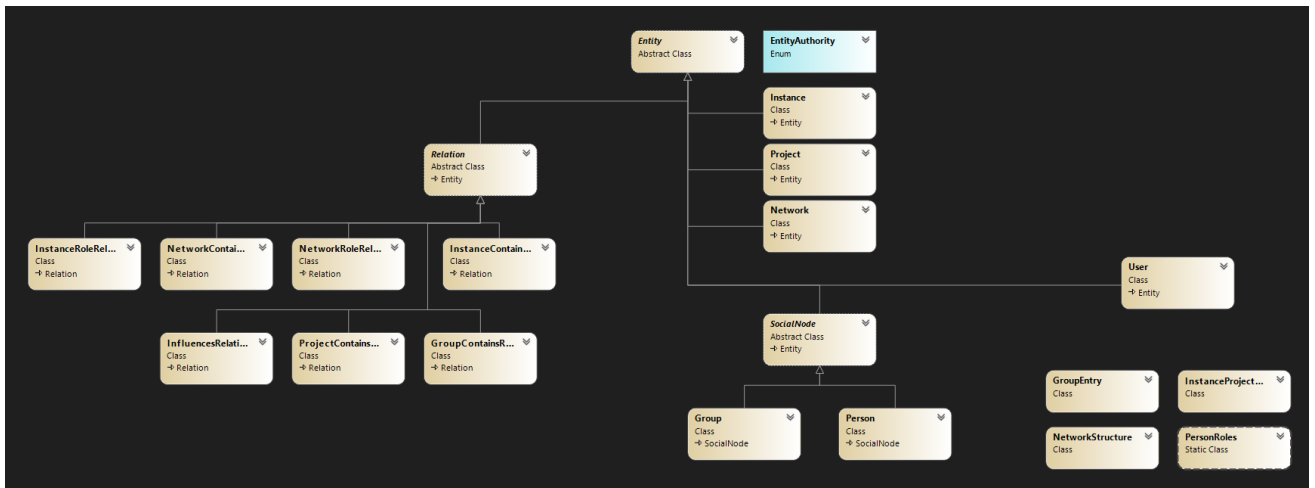


Figure 17 Class diagram with an overview of all the models used

Detailed view can be found at **A-SIA 2.0\src\A-SIA2WebAPI\A-IA2WebAPI.Models\ClassDiagram.cd**

You can also find the currently developing technical documentation at **A-SIA 2.0\docs\Documentation\_A-SIA2.0.docx**

# Our Project Diary

Initial meeting 07.10.2021

Summary of features and discussion of functionality.

Meeting 2: 11.10.2021

Discussion of first user stories and important project metrics like irritation, persistence, and echo dampening.

We started to research and make decisions for the tools and frameworks we wanted to use. We began to flesh out the first user stories and started designing prototypes.

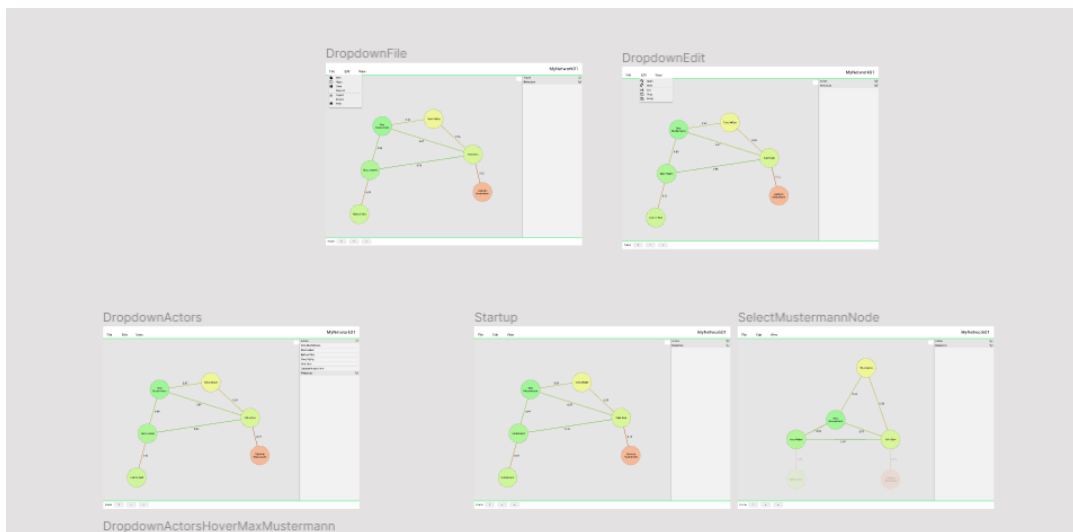


Figure 18 Early prototype

Meeting 3: 30.10.2021

Discussion about configuration values and if the simulation should be precomputed. We concluded that they should be precomputed.

Meeting 4: 03.11.2021

Demonstration of first mockup prototypes. UML diagrams for the software architecture of the project were created.

Meeting 5: 11.11.2021

Demonstration of mockup and discussion about visual design. After we came to a common denominator how certain details should be designed, we kept working on the prototype.

Meeting 6: 25.11.2021

Demonstration of mockup progress. Work on the backend server started with DAL and definition of the API.



Group		
POST	/Group/{networkId}	
DELETE	/Group/{groupId}	
PUT	/Group/{groupId}	
Network		
POST	/Network	
PUT	/Network/{networkId}	
DELETE	/Network/{networkId}	

Figure 19 Part of the Rest-API

```

C:\GitHub\A-SIA 2.0\src\A-SIA2WebAPI\A-SIA2WebAPI.BL.API\bin\Debug\net5.0\A-SIA2WebAPI.BL.API.exe
info: A_SIA2WebAPI.BL.PluginSystem.PluginLoader[0]
      No Plugin dlls were provided
info: Microsoft.Hosting.Lifetime[0]
      Now listening on: http://localhost:5000
info: Microsoft.Hosting.Lifetime[0]
      Application started. Press Ctrl+C to shut down.
info: Microsoft.Hosting.Lifetime[0]
      Hosting environment: Development
info: Microsoft.Hosting.Lifetime[0]
      Content root path: C:\GitHub\A-SIA 2.0\src\A-SIA2WebAPI\A-SIA2WebAPI.BL.API
  
```

Figure 20 running server

## Meeting 7: 01.12.2021

Presentation of finalized prototype. User stories for the entire project were created and rated. We also set up a pipeline to automatize testing for both frontend and backend. We started to make progress on implementing the frontend side of the project.

Order	Work Item...	Title	State	Story Points	Area Path	Iteration Path	Team Project
	Feature	▼ User want to generate a report	● New		A-SIA 2.0	A-SIA 2.0	A-SIA 2.0
	User Story	■ User wants to have a network level report	● New		A-SIA 2.0\Mid	A-SIA 2.0	A-SIA 2.0
	User Story	■ User wants to have a person level report	● New		A-SIA 2.0\Mid	A-SIA 2.0	A-SIA 2.0
	User Story	■ User wants to define template	● New		A-SIA 2.0\Large	A-SIA 2.0	A-SIA 2.0
	User Story	■ User wants to define indicators (FRAGE)	● New		A-SIA 2.0\Large	A-SIA 2.0	A-SIA 2.0
	Feature	▼ User wants to build their network from external data	● New		A-SIA 2.0	A-SIA 2.0	A-SIA 2.0
	User Story	■ User wants to generate network from survey data	● New		A-SIA 2.0\basic	A-SIA 2.0	A-SIA 2.0
	User Story	■ User wants to generate network from external tool	● New		A-SIA 2.0\basic	A-SIA 2.0	A-SIA 2.0
+	Feature	▼ Menubar	...	● New	A-SIA 2.0\basic	A-SIA 2.0\Iterati...	A-SIA 2.0
	User Story	> ■ User can use File Menubar Menu	● Closed	2	A-SIA 2.0	A-SIA 2.0\Iterati...	A-SIA 2.0
	User Story	> ■ User can use Edit Menubar Menu	● Closed	2	A-SIA 2.0	A-SIA 2.0\Iterati...	A-SIA 2.0
	User Story	> ■ User can use View Menubar Menu	● Closed	1	A-SIA 2.0	A-SIA 2.0\Iterati...	A-SIA 2.0
	Feature	> ▼ Highlevel Network Management	● New		A-SIA 2.0	A-SIA 2.0	A-SIA 2.0
	Feature	> ▼ Sidebar	● New		A-SIA 2.0\basic	A-SIA 2.0\Iterati...	A-SIA 2.0
	Feature	> ▼ Sidebar Network	● New		A-SIA 2.0	A-SIA 2.0	A-SIA 2.0
	Feature	> ▼ Sidebar Inspector	● New		A-SIA 2.0	A-SIA 2.0	A-SIA 2.0
	Feature	> ▼ Sidebar Structure	● New		A-SIA 2.0	A-SIA 2.0	A-SIA 2.0
	Feature	> ▼ Inspector Person	● New		A-SIA 2.0\basic	A-SIA 2.0	A-SIA 2.0
	Feature	> ▼ Inspector Group	● New		A-SIA 2.0	A-SIA 2.0	A-SIA 2.0
	Feature	> ▼ Inspector Connection	● New		A-SIA 2.0	A-SIA 2.0	A-SIA 2.0
	Feature	> ▼ Graph Editor Navigation + Selection	● New		A-SIA 2.0	A-SIA 2.0	A-SIA 2.0
	Feature	> ▼ Graph Editor Tools	● New		A-SIA 2.0	A-SIA 2.0	A-SIA 2.0
	Feature	> ▼ Graph Editor Display	● New		A-SIA 2.0	A-SIA 2.0	A-SIA 2.0
	Feature	> ▼ Simulation Diagram (Frage)	● New		A-SIA 2.0	A-SIA 2.0	A-SIA 2.0
	Feature	> ▼ Database	● New		A-SIA 2.0\basic	A-SIA 2.0	A-SIA 2.0
	Feature	> ▼ Menubar Tools	● New		A-SIA 2.0	A-SIA 2.0	A-SIA 2.0

Figure 21 Some user stories in Azure

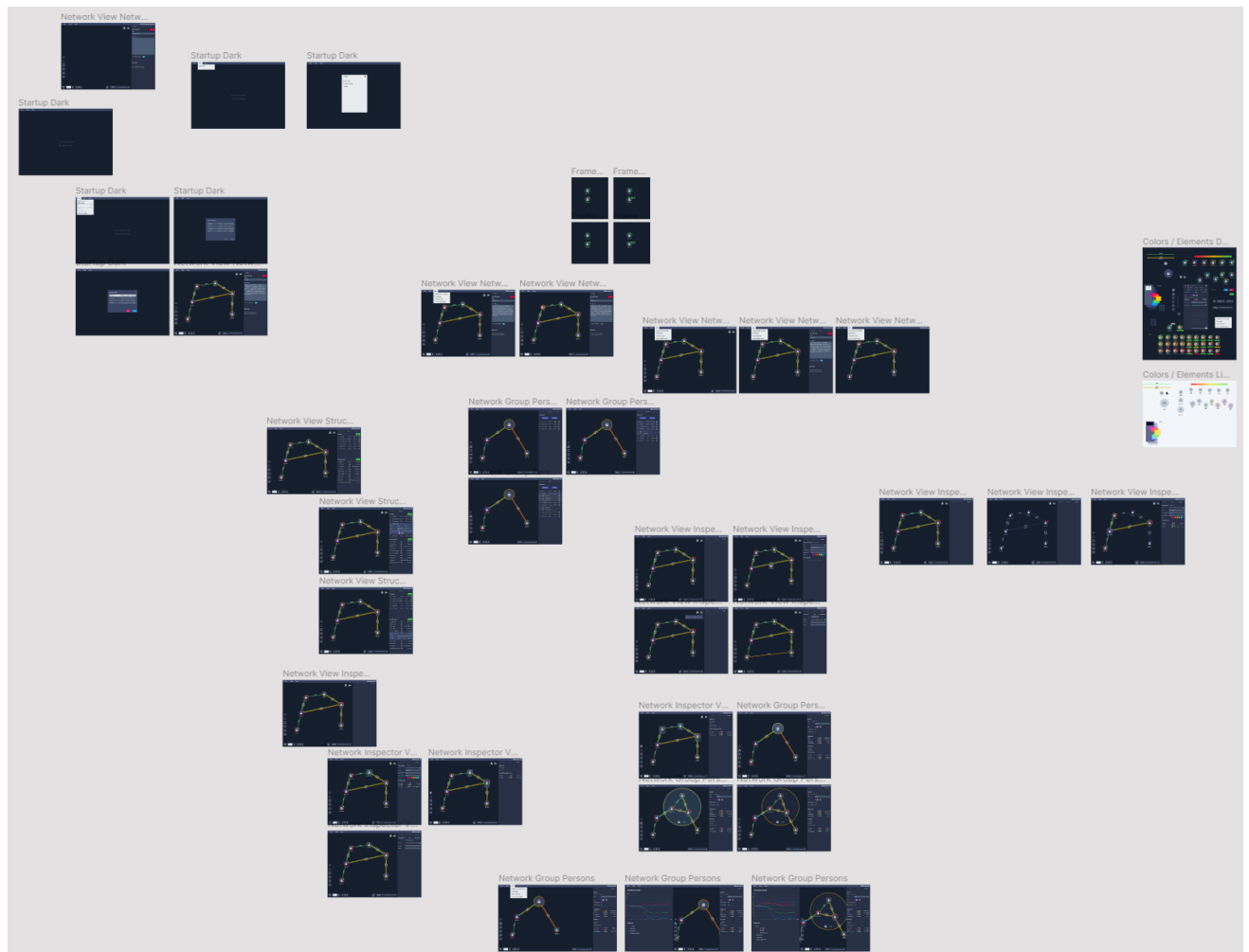


Figure 22 Finalized prototype

Meeting 8: 15.12.2021

Coding continued and the menu bar was implemented. The backend database was setup. Also, the first http round trip for the login was completed.

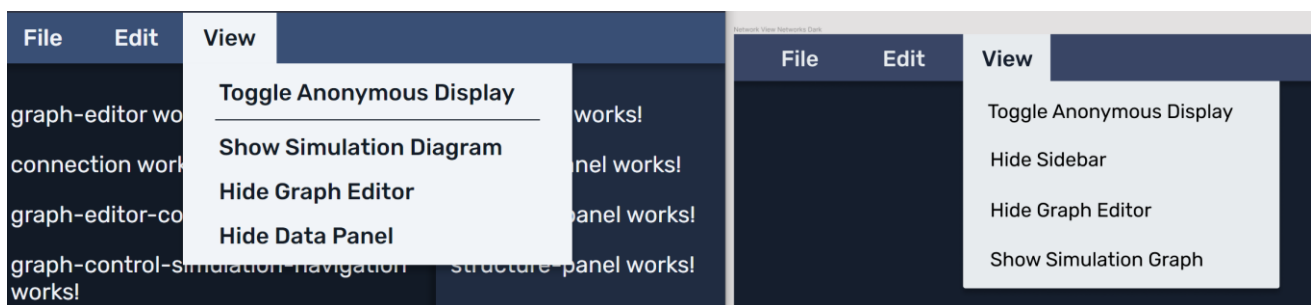


Figure 23 Implemented menu bar on the left compared to prototype on the right

Meeting 9: 11.01.2022

Final meeting before project presentation. We talked about the status of the project and which items will be required for the final hand in. Finally, we got access to the book in which the algorithms for computing the simulation are described.

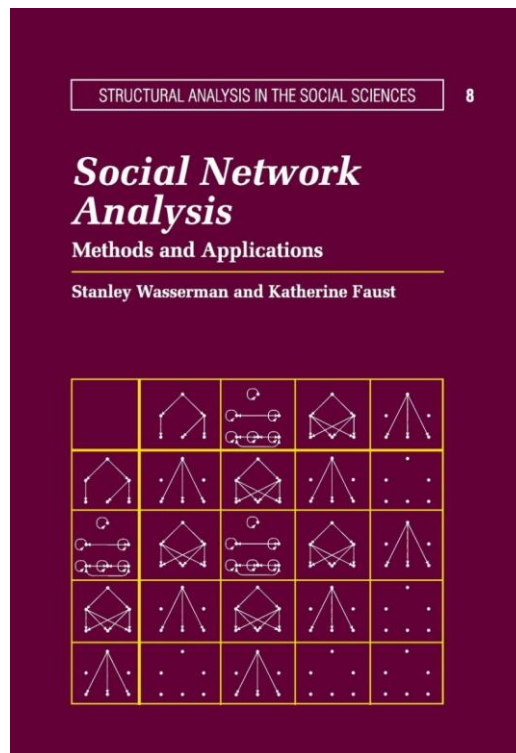


Figure 24 This book will be our companion in the next semester

28.2.2022

A component for resizable panels was implemented. This allows resizing the data- / graph editor- / inspector panel. Also works on mobile. Furthermore, the PowerPoint and video was created. Automatic deployment pipelines were set up, and Jwt authentication was implemented in frontend and backend. Navigation in the data panel is possible, and the server provides a nice framework to implement the upcoming API requests.

9.3.2022

Deployment now is fully automated and works with HTTPS (both C# server and Angular frontend). The Graph editor is also zoomable (except mobile two finger pinching) and pan-able. It was also decided to expand the requirements, as a proper way for handling the Instance>Project>Network hierarchy with shared permissions was necessary.

29.3.2022

Work on this sprint already started before schedule. Fully frontend features, like an undo / redo history, a hotkey system, and an endpoint for settings were completed. Besides establishing to be used endpoints, like opening networks, the first end-to-end calls were being made after implementing network management (creating deleting modifying instances, projects and networks). The following image was prototyped and implemented afterwards (frontend and backend).

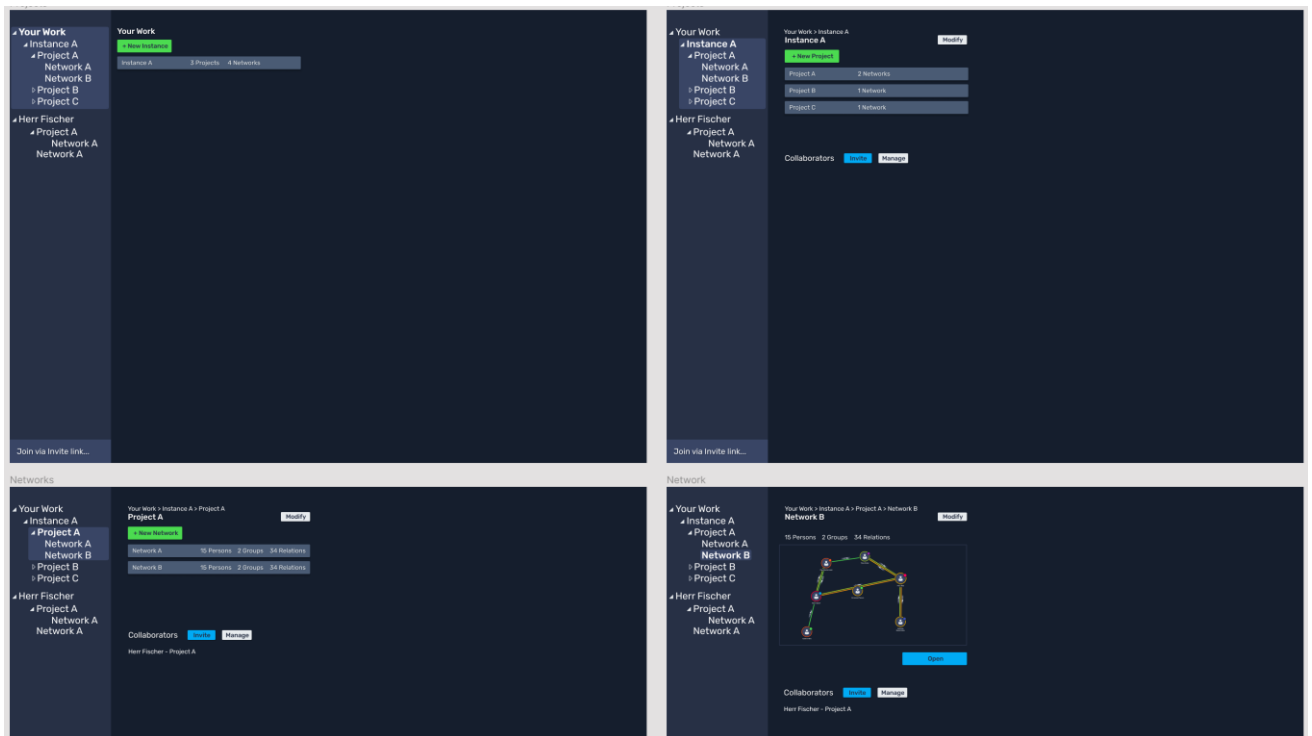


Figure 25 Instance>Project>Network management prototype

Through making our first large scale requests, we discovered that there was terribly high latency. A fetch of 77 instances resulted in a 6.57s request.

<input type="checkbox"/> instances	200	xhr	zone.js:2863	4.3 ...	6.57 s	
<input type="checkbox"/> projects	204	preflight	Preflight	0 B	1 ms	
<input type="checkbox"/> networks	204	preflight	Preflight	0 B	1 ms	
<input type="checkbox"/> projects	200	xhr	zone.js:2863	21...	9.36 s	
<input type="checkbox"/> networks	200	xhr	zone.js:2863	21...	9.40 s	
<input type="checkbox"/> favicon.ico	304	vsync	image.js:1	23...	2 ms	

Figure 26 A fetch of 77 instances

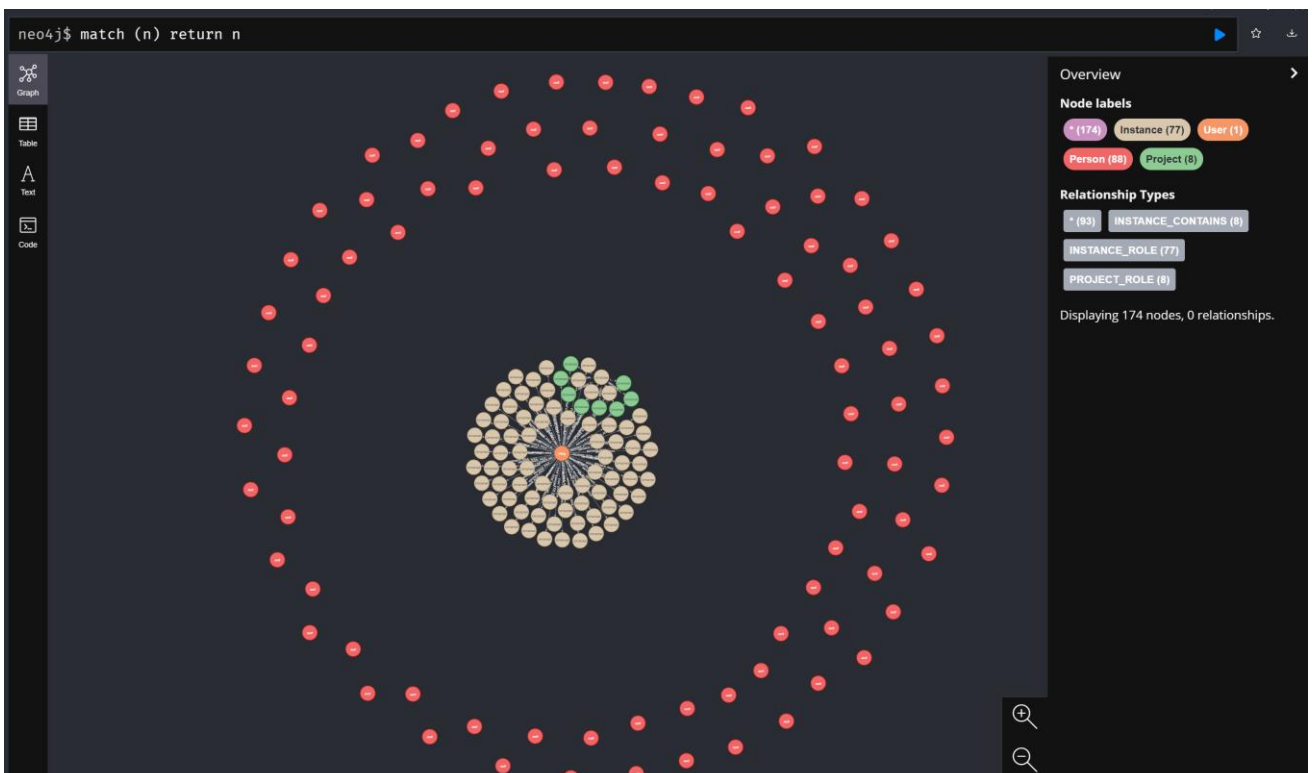
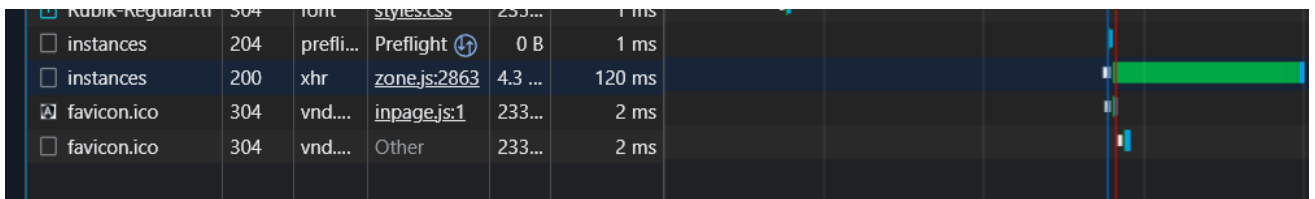


Figure 27 Neo4J containing a User (orange), Instances (beige), Projects (green), and Persons (red)

After some testing, it was discovered that our current approach of fetching data was too granular. In our case, when fetching a set of nodes, we fetched each node individually. This may have given us the highest amount of architectural flexibility, without having to worry about any new queries, but of course puts a high load on the database. After changing our queries, and caching some responses, like the logged in user, we eventually got down to ~120ms response time.



File Name	Size	Type	URL	Size	Time
instances	204	prefli...	Preflight	0 B	1 ms
instances	200	xhr	zone.js:2863	4.3 ...	120 ms
favicon.ico	304	vnd....	inpage.js:1	233...	2 ms
favicon.ico	304	vnd....	Other	233...	2 ms

Figure 28 Fetch of 77 Instances after rewriting more general queries

24.4.2022

Over the last month, we have concerned ourselves with state management. It is necessary to have a layer between the requests that are sent to the backend, and the data that is displayed in the components. Therefore, it is recommended to cache data, to minimize unnecessary server requests. We investigated the main framework fit for purpose, called ngRx. It was build specifically for angular, inspired by the redux pattern.

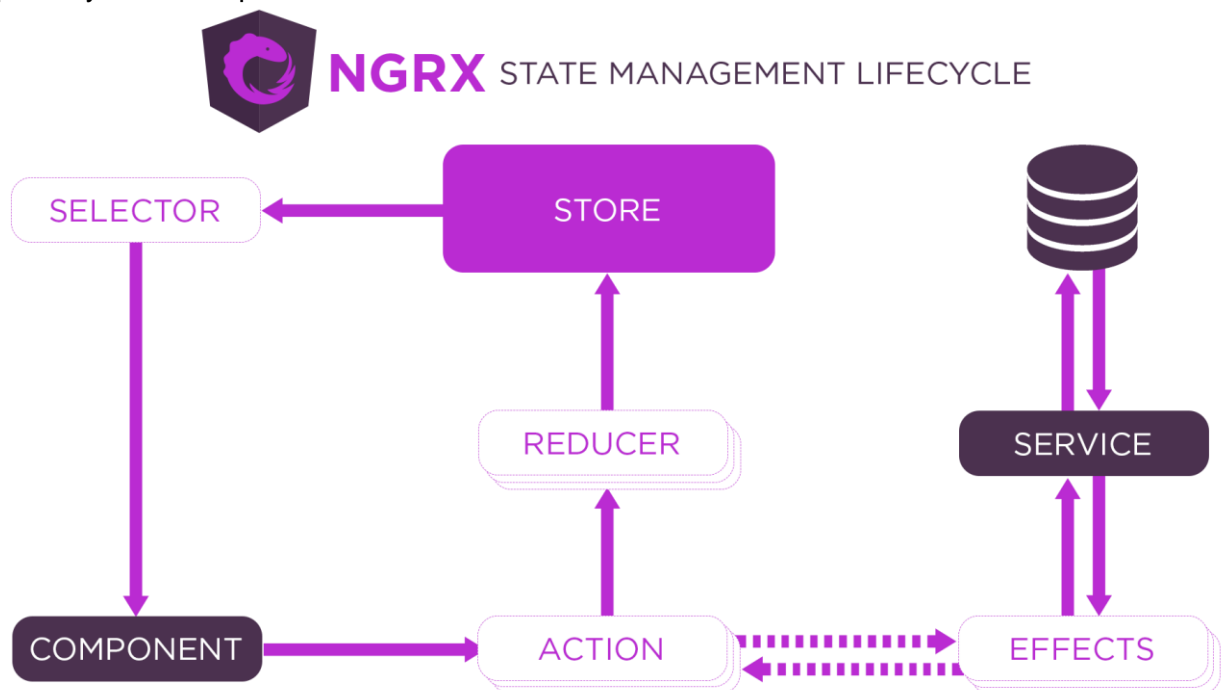


Figure 29 <https://ngrx.io/guide/store>

However, we decided to not use ngRx, as it uses many features that we didn't initially see the purpose of. In order to gain the most of this project, we built a simplified version of ngRx ourselves, which allows for a bit more flexibility. In our case, we have a store for each type of data, (e.g., a store for networks, a store for persons, etc.). Every store loads their initial data (e.g. all current networks of the user) when created. We separate the Actions into their own class, where we update, create, or delete the relevant data. The actions expose the new update through either BehaviourSubjects or ReplaySubjects which are located in the store. We also have a service which contains all current requests and actions that are running, in case a component wants to inquire the loading status.

4.5.2022

We switched the whole frontend and backend to GUIDs, so we can create IDs in the frontend, and don't need to wait for the backend / database to respond with the id. We also added detailed logging to make debugging easier.

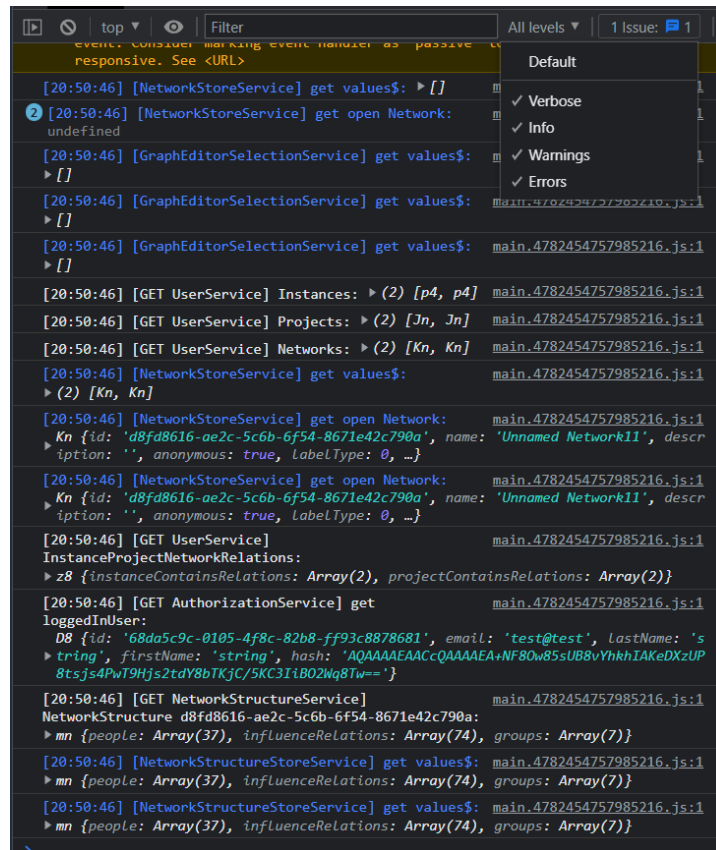


Figure 30 Logs created by the frontend

5.5.2022

Thanks to switching to GUIDs, we could implement optimistic UI updates. Optimistic UI updates can be used, when you know how the data will change, and the server request will probably not fail. With optimistic UI updates, you update the UI before the server got the change. In case the server does fail a request, we must roll back the UI change. In our case we are achieving that, by having a revert function in every action, that rolls back the data, and is called when the request sends back an error.

10.5.2022

We updated the project hierarchy to work with optimistic UI updates. You can now create, modify and delete instances, projects and networks with seemingly no latency.

27.5.2022

The Network structure was implemented in the backend, and can also be seen in the neo4j database.

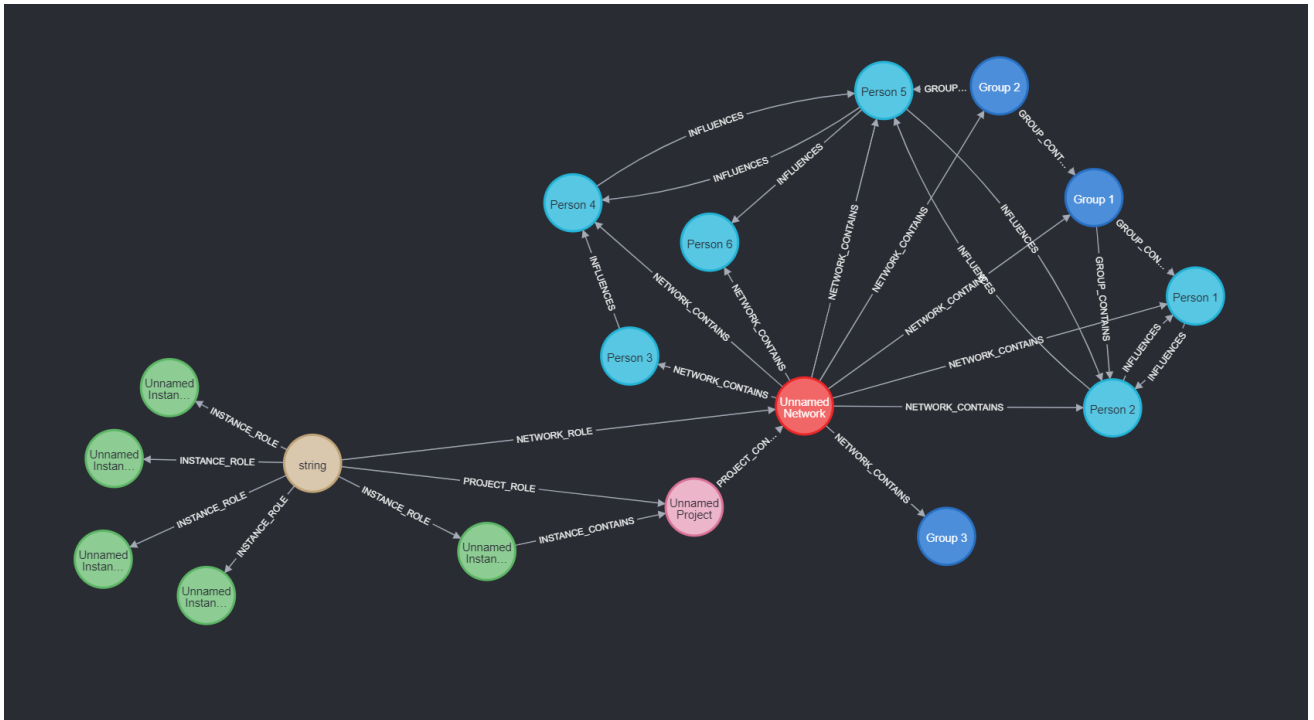


Figure 31 Example Network Entry

29.5.2022

Similar to the project hierarchy, the network structure was also implemented in the backend, and successfully fetched into a store in the frontend. Currently we send a list of all groups, persons, and connections to the frontend, which then uses this data to build the graph.

3.6.2022

Persons and groups can be added through the structure tab. If a group in the structure tab is selected, and a new person / group is created, it will automatically add it to the selected group.

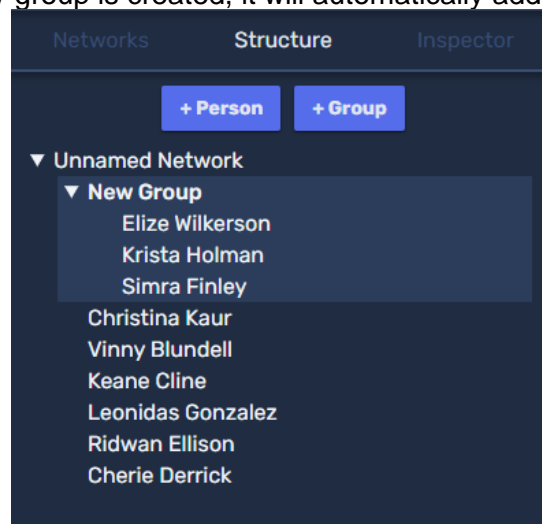


Figure 32 The structure tab with exemplary data

7.6.2022

Persons and connections are now displayed in the graph.



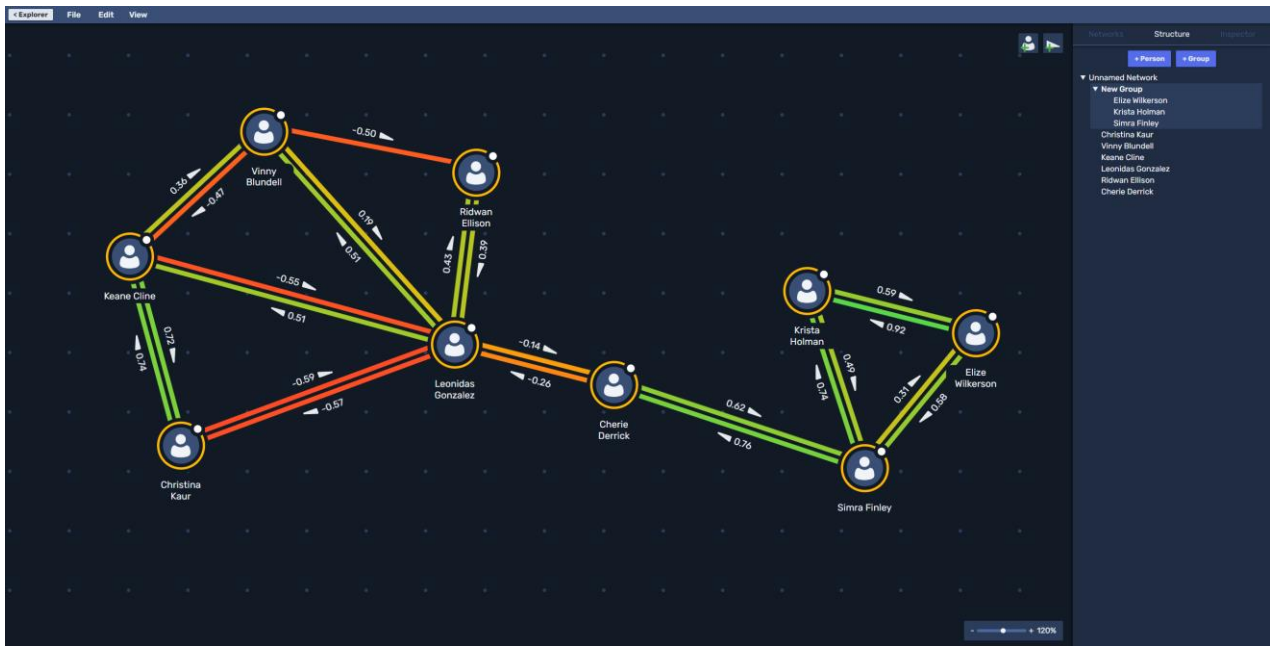


Figure 33 A graph being displayed in the editor

This currently works purely through HTML, CSS and variable binding. We did not use any existing frameworks, because we eventually want to do more complex visual work, like grouping and automatic placement. We felt it would be easier to create our own graph visualization system, instead of extending an existing one.

19.6.2022

Inspector for persons and connections were created. Now, you can select a node or connection by clicking on it, and the inspector tab will display some interesting data. It already updates in the backend whenever nothing the data changes.



Figure 34 A selected person's data in the inspector



Figure 35 A selected connection's data in the inspector



22.6.2022

Two tools have been added, one for creating persons, and one for creating connections. When hitting the connection creation button in the top right, it will draw connections between selected nodes.



Figure 36 A connection being created

23.6.2022

Nodes can now be moved. Furthermore, some special roles can already be calculated in the backend. For example, we have the Alpha and Omega roles, whereas the Alpha is a person who practices a lot of influence, and an Omega a person which receives a lot of influence. There are also the Carrier, Cut-Point, and Isolator roles. The Carrier has many ingoing and outgoing relations, the Cut-Points are person nodes that, if removed, split the graph into two separate graphs, and the Isolator has only ingoing and no outgoing relations.

Over the course of the Semester our models evolved into following structure:

