Technical Requirement Report for the “Expertise Finder” project

This report outlines the reasoning behind the selected technical requirements for the “*Expertise Finder*” project. The decisions are categorized into three main areas: hardware for hosting the website and database on, back-end software, and the graphical interface for the node graph.

Back-end

Self-hosted docker container:

Utilizing Docker containers offers a significant advantage to the project maintainability and scalability. Docker containers streamline the installation of Neo4j or any other service because conflicting dependencies are isolated and cannot cause problems. Additionally, container configuration can be easily distributed across different servers regardless of their operating system using Dockerfile or docker-compose.yml files. Docker Compose simplifies the process of hosting multiple communicating services, such as a Neo4j back-end container and a web server container. Neo4j provides extensive documentation for using its Docker containers.

Enterprise edition:

The Enterprise edition is preferred over the Community edition due to its enhanced authentication and authorization capabilities such as role-based access control. The Community edition only supports admin user accounts, which poses a security risk by using the admin account for data retrieval. Additionally, the Enterprise edition offers clustering and therefore high availability and scalability. Currently the Enterprise edition has a (currently) unknown license cost. There is a free license, but it is for “*educational*” purposes which is not applicable here. Contacting Neo4j for potential university-specific arrangements did not result in further information. However, the Community edition more than suffices to start the project

Newest version:

Adopting the latest version of Neo4j ensures the use of the most secure release. New Neo4j versions typically release monthly. Upgrading to a newer version involves stopping the running instance (also backup), removing the earlier version, and downloading the newest version using the package manager. With Docker, this process includes stopping and removing the container, followed by pulling the latest image, which is always tagged as "latest" to ensure the newest version is used.

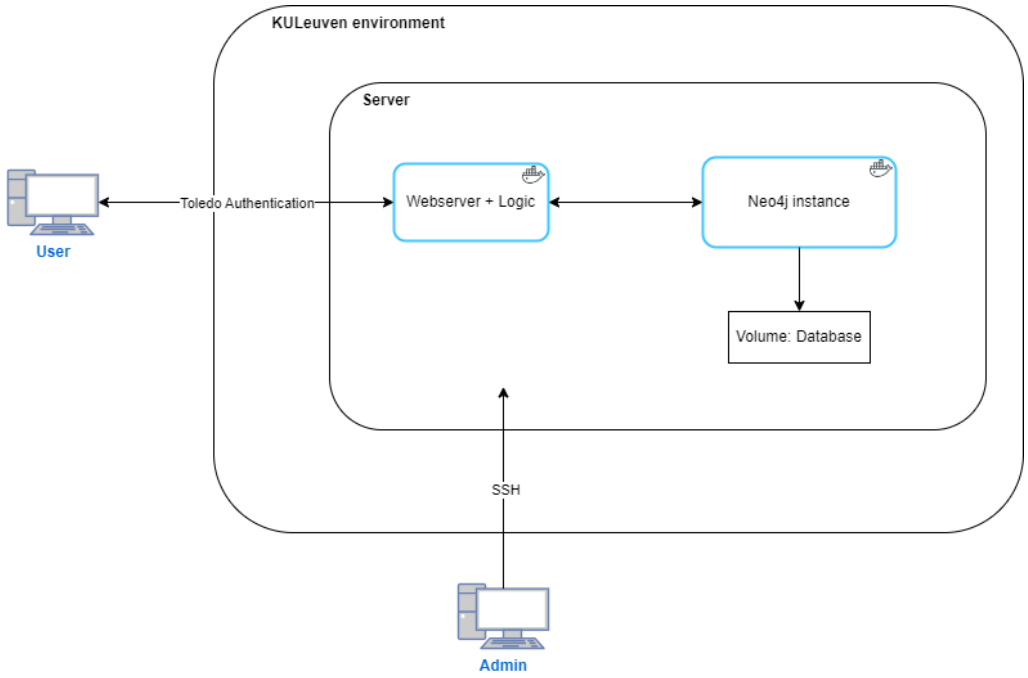
Web and Logic Backend Server:

To enhance security, data should not be pulled directly from the database to the front end. Instead, a web server should be used to access the database and forward the data to the front end. Currently, the prototype HTML pages expose the database's username and password by pulling data directly from it. For the webserver, Apache or Nginx suffices. The choice of web application framework heavily depends on the expertise of future developers working on the project.

Hosting Hardware

The KULeuven offers various hosting services. One of them is CaaS or Container as a Service. CaaS is the most cost-effective option compared to other KUL services or Neo4J's AuraDB. CaaS aligns perfectly with the project's intention of using Docker containers. The KUL services also provide ways to expand storage and processing power if needed. It costs €66.57/year and €45.08/100GB for storage extensions.

If CaaS is not feasible, Neo4j's AuraDB provides a free cloud solution for limited data. Other Cloud services are also available but require admins to also manage the server and be familiar with the provider's platform. Additionally, unexpected or excessive costs can arise when using cloud providers.



Graphical Interface

Admin management

Neo4j browser:

Neo4j browser is an ideal tool for admins to manage the database and users, excluding server-side processes such as container management. It supports data modification, user management and regular cyphers. It is bundled with every version of Neo4j.

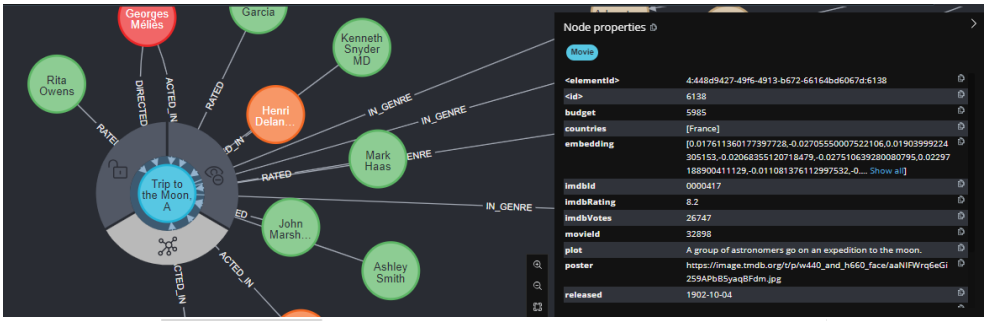
Graph library

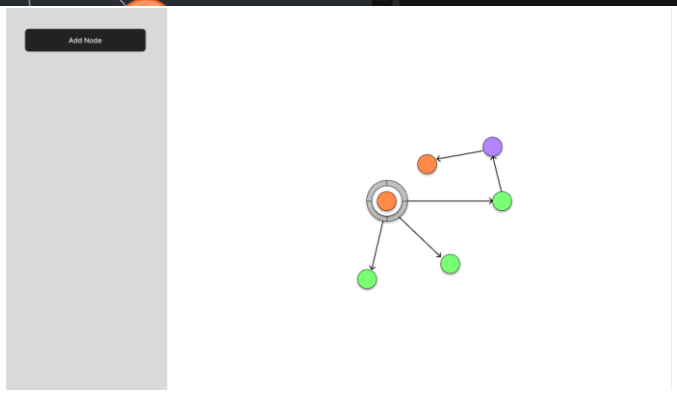
Neo4jD3:

The Neo4jD3 library, built upon D3.js, has been selected due to its robust foundation. Although D3.js is complex to learn, it is highly flexible and allows for dynamic graphical manipulations. Despite Neo4jD3 not being fully up to date with the latest data structures of Neo4j, it provides a solid base that can be modified to suit the project's needs, combining the ease of use of Neo4jD3 with the flexibility of D3.

User Interaction

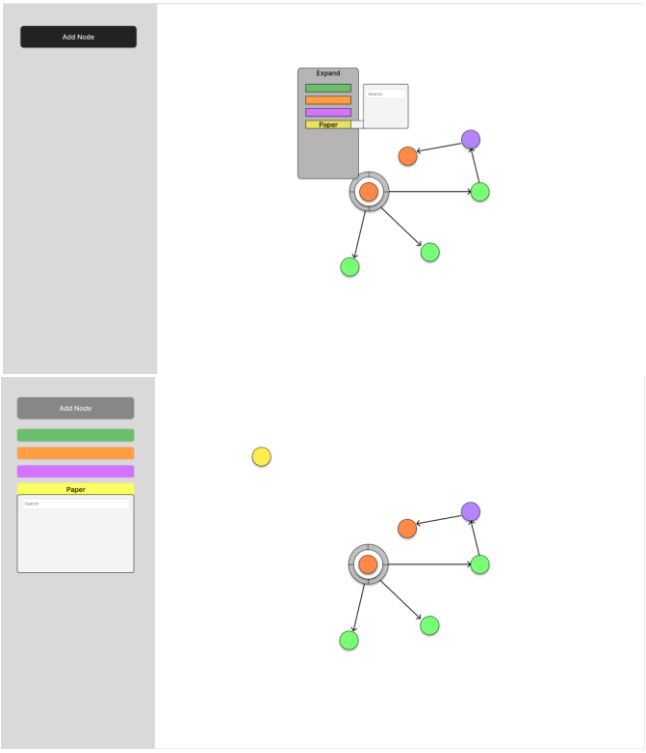
The goal is to replicate the core functionalities of Neo4j Browser and Bloom. Both Neo4j Browser and Bloom are known for their intuitive yet comprehensive design. When a node is selected, a panel appears displaying various options and node values.





No Cypher:

To minimize the use of the Cypher Query Language, nodes should be extended with restrictive parameters, such as limiting the number of expanded nodes. Users should be able to choose the type of extended nodes and optionally include specific keywords. Additionally, users should have the option to add specific nodes individually.



Keeping graphs composed:

The simplest way to keep graphs composed and organized is to open a new graph or tab for the expanded node. This should occur at the user's request or when a certain node limit is reached.