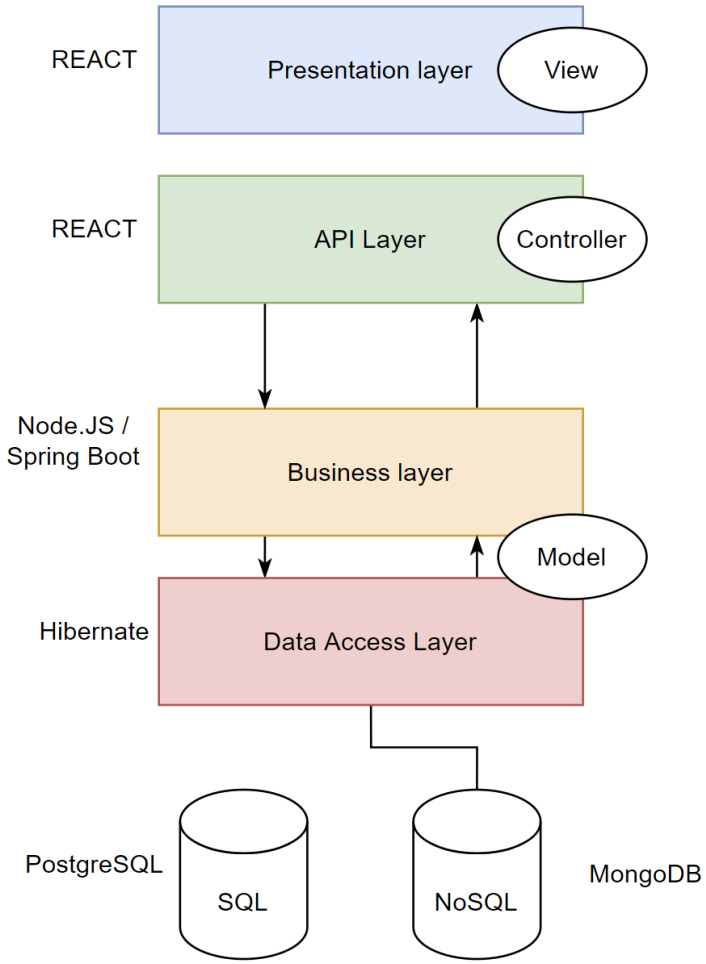
Project description

The application surrounds a simple chat application. It is designed to be a replica of the famous dating application Tinder. The user can create a profile and define values such as profile pictures and text. From there they receive other user profiles matched by their gender and their set gender preference. If both users like each other they can start a chat room and communicate with each other.

Architecture



Architecture is the same as defined in my first LB1 document. The business layer uses Spring Boot after finalization.

Used Frameworks and API

https://upload.wikimedia.org/wikipedia/commons/thumb/4/44/Spring_Framework_Logo_2018.svg/1200px-Spring_Framework_Logo_2018.svg.png

1. Spring Boot

The framework is used for all backend services and queries performed on the two databases using **ORM** and **OGM.**

https://stomp.github.io/images/project-logo.png

2. Spring Boot STOMP + SockJS WebSocket

Use in connection with React and Spring Boot to establish WebSockets.

https://logos-download.com/wp-content/uploads/2016/09/React_logo_wordmark.png

Used for all Client side logic and graphical components. Additionally, manages sending requests and filtering responses.

3. React

https://upload.wikimedia.org/wikipedia/commons/thumb/5/5b/Logo_de_Auth0.svg/2560px-Logo_de_Auth0.svg.png

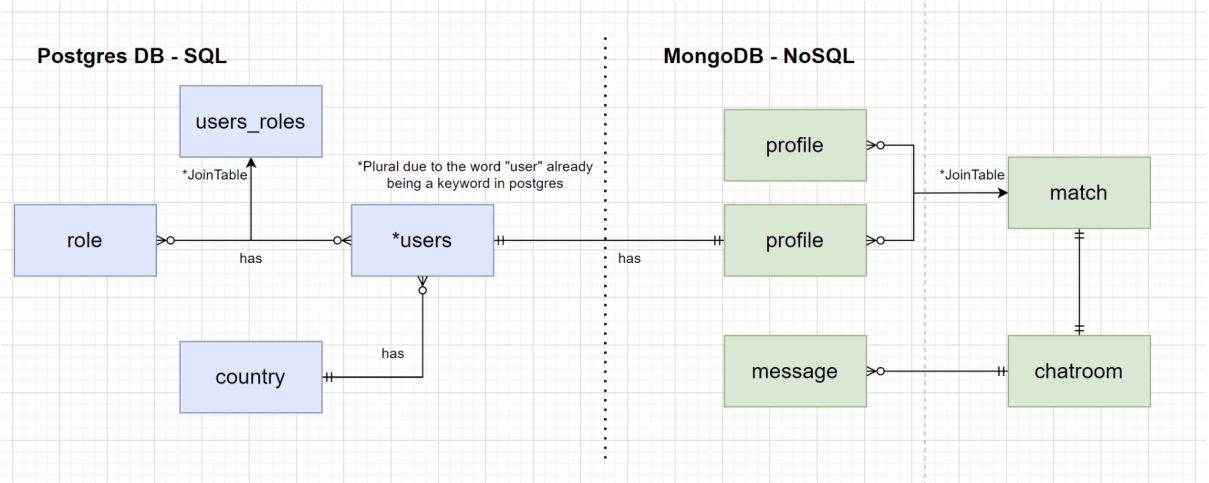
4. Auth0 Provider

Used for **role/permission distribution**, **Login** / **Sign Up** / **Logout** features, and **JWT token generation**.

ERM

The ERM contains two database connections. One is the primary database on the left. It is contained inside a Postgres DB which is hosted by a server. It deals with main functions such as **users**, **user roles**, and **login/logout** functions.

On the right is the second database structure. It as well is hosted by a server and is contacted via my repository objects. It is based on MongoDB and is a NoSQL server. Important by the NoSQL databases is the lack of real datatypes and foreign key mechanisms.



Source-Sascha Buthelezi

Reflection

Looking back at the whole module I have mixed feelings. From one side it was a blast creating another entire application(***As of now already imitated Netflix***) from the ground up. I also decided to add some new topics which I didn't have any prior experience with. That is why it was a great chance to get some necessary experience. The downside was the lack of experience became an ever-growing problem during the module time. At the end of the module, I couldn't be happier to finally have a break from working on the project. I spent too much time and especially free time working on the project.

As for my reflection on my work proficiency. I sadly wasn't able to achieve my dream goal of the application state. The reasoning is unsure of new technologies and especially wasting too much important time on a specific part of the application. I am however still proud of the result I definitely feel that if I saved my time earlier and tried something new quicker I could definitely have achieved my goal for this application. The actual code quality can definitely be improved and worked on. I did wish however we would have received a bit more guidance or had a clue about what the end product should have looked like. I also would have wished that some criteria such as GIT branching strategies and frameworks would have been considered a bit earlier instead of the last days.

Test concept

Test Type	Tool
End-To-End testing	Postman
Unit testing	JUnit