Project Approach

Frontend

EJs

Pros:

- Simple syntax for embedding JavaScript into HTML
- Integrates easily with Node.js

Cons:

- Limited functionality compared to more modern frameworks

React

Pros:

- Component-based, can re-use components
- Virtual DOM rendering for responsiveness
- Good documentation

Cons:

- Steep learning curve
- Requires additional tools for server-side rendering

Flutter

Pros:

- Single codebase for all types of applications
- Fast and responsive
- Predefined widgets

Cons:

- Mainly used for mobile (although it is multi-platform)
- New and not much documentation (still developing)

Backend

Node.js

Pros:

- Huge amount of documentation
- Works well with other JavaScript frameworks/libs

Cons:

- Scalability issues
- Inconsistent standard library for naming conventions

PHP

Pros:

- Strongly typed therefore increases readability
- A lot of documentation

Cons:

- Older language therefore slower technology and outdated
- Takes a long time to set up an environment compared to other backend languages

Java

Pros:

- Strongly typed so it is very good for readability
- A lot of good frameworks like Spring
- Good scalability
- Uses JVM so very portable

Cons:

- Leads to boilerplate code
- Long to set up environment
- No Asynchronous programming

Database

MongoDB

Pros:

- NoSQL therefore flexible data modeling
- Scalable for high-performance apps
- JSON formatted documents

Cons:

- Data consistency can be challenging

Firebase

Pros:

- Easy to setup and integrate into a project
- Has authentication libraries for you

Cons:

- Limited query capabilities
- Have to pay for more access

MySQL

Pros:

- Readable and widely used relational database
- Mature tooling and a lot of documentation

Cons:

- Not flexible for modeling data
- Migrating databases can be complex

Detailed Explanation

We decided to choose React in the frontend, Node.js for backend, and MongoDB for our database. We chose Node.js since we are all familiar with JavaScript and there is a lot of documentation on the language and especially Node.js. Also, there is an abundance of frameworks for it, and we are going with Express.js to implement the MVC design pattern. Secondly, it is a lightweight, easy to setup backend language which is perfect for this project since we do not want to spend a lot of our time and resources setting up the backend. Secondly, we chose React since it integrates nicely with Node.js and it has a big community and documentation behind it in case of us getting stuck. We chose this over Flutter or EJs because of the amount of documentation it provides us. Lastly, we chose to go with a Non-Relational Database, MongoDB since we want flexible data modeling. If we were to choose a relational database such as MySQL, it would have been hard to change the data tables if we needed to