ADR – Decision of Backend

**(significantly changed from original by Kim Nyaguse)**

**Title:**

Decision of Backend

**Context + Problem Statement:**

The application needs a backend that can handle authentication, data storage and integration within its features. It should support scalability, good integration with front-end technologies and meet the project requirements.

**Decision Driver:**

* The backend needs to allow for scalability in the future due to a potential increase in users.
* The backend needs to be easy enough to integrate with different front-end technologies.

**Option:**

* Firebase
* SQLite
* MongoDB
* Oracle
* MySQL
* Node.js

**Considered Options:**

* Firebase
* SQLite
* MySQL
* Node.js
* Express

**Decision Outcomes:**

Chosen Option: “Firebase with Node.js” because firebase has many built in features such as authentication, real-time database capabilities and a general ease of use. “Node.js” allows for event driven architecture which supports our team goals of scalability and real time user interaction. They are both the most compatible with our personal project requirements. It also offers a good free version that is easy to learn.

**Consequences:**

* Good because Firebase simplifies the development process which reduces the amount of time it takes to develop an application.
* Good because Node.js provides a strong backend framework by being able to handle real-time data and supporting scalability.
* Bad because changing to a different database with increased scalability would be very difficult.

**Confirmation:**

Monitoring tools will be able to validate real-time performance. Code reviews and integration testing will ensure that the communication between Firebase, Node.js and the frontend (JavaScript).

**Pros and Cons:**

SQLite

* Good because it is serverless, allowing for more flexibility.
* Good because it is relatively easy to set up for smaller projects.
* Bad because it isn't ideal for scalability or consistent usage.
* Bad because it doesn't have a lot of support regarding advanced queries.

Oracle

* Good because of its high-level performance and good security.
* Good because it contains extensive features that are very accessible and guides that can be found online.
* Neutral because it can be over the top for small/medium projects.
* Bad because the licensing fees are extremely high, especially for people just starting out.

Template used:

<https://adr.github.io/madr/decisions/adr-template.html>