ADR for Frontend

- **Summary** - In order to create this large web application that is scalable and runs fast, we are using React for front end development.
- **Problem** - The client did not have a specific framework in mind and we had a variety of options to choose from.
- **Constraints** - The client hopes to expand this platform with future developers so the stack must be adaptable, scalable, and popular. The frontend must be visually appealing and compatible with the backend framework of choice.
- **Options** - Here are the following options and the pros and cons of each:
    - Pure HTML/CSS and Javascript
        - Pros- Well known, easily implemented and debugged
        - Cons- does not scale well as the application continues to grow
    - React
        - Pros- very popular framework, scalable, great for large applications, reactive rerendering aligns with Firebase realtime database model
        - Cons- Solid learning curve, SEO problems, crappy documentation, large library size=higher hosting costs
    - Vue.js
        - Pros- very tiny library size, virtual dom performance, two-way data binding, easy to learn
        - Cons- Less popular in US/English, reactivity complexity (re-render misses at times), lack of scaling
    - Angular 2
        - Pros- two-way data binding, dependency injection, directives, massive community
        - Cons- performance, very steep learning curve
- **Rationale** - We choose React for the front end because the pros outweigh all other cons. Our team is very familiar with the framework and overall it will work best with the Firebase managed backend service.

ADR for Backend

- **Summary -** We are using Firestore Cloud Functions to implement the backend logic for a serverless architecture and easy compatibility with the backend.
- **Problem -** We need to create backend services that are fast and stable to secure critical business logic.
- **Constraints -** N/A
- **Options -** Here are the following options and the pros and cons of each:
    - Firebase Cloud Functions
        - Pros- Serverless framework that will be automatically managed, scaled, and secured by the Firebase team. Great compatibility out of the box with the Cloud Firestore database.
        - Cons- Cold starts can hurt user experience, no scheduler cron support, Javascript only
    - Java spring framework
        - Pros- Type safety and object oriented
        - Cons-poor memory management, need to manage servers/go to separate provider
- **Rationale -** We decided to use Firestore Cloud Functions since it's serverless design makes less operational and setup load on the team as well as it's great out-of-box compatibility with the Firebase suite.

ADR for Database

- **Summary -** In order to efficiently store user and sales data, we decided to use Firebase.
- **Problem -** We need to log each payment and keep track of the sales historically for data analytics purposes so that small businesses can make better decisions on their pricing and the number of staff on duty.
- **Constraints -** N/A
- **Options -** Here are the following options and the pros and cons of each:
    - MySQL
        - Pros- Lightweight
        - Cons- SQL!, Cumbersome schema that we need to abide by
    - Firestore Cloud Firestore
        - Pros- JSON structured, schema free, real time event notifications
        - Cons- Complex queries become difficult
- **Rationale -** We decided to use Firebase Cloud Firestore because the data we are storing are less structured (payments, users' features, etc.) so we need a database that is less strict on that.

ADR for Web Server

- **Summary -** We are going to host the web app statically on Firebase Hosting
- **Problem -** We need a way to host our web app that is scalable and low-cost.
- **Constraints -** N/A
- **Options -** Here are the following options and the pros and cons of each:
    - Firebase Hosting
        - Pros- Built-in CDN network on SSD edge devices, fast, secure, reliable, auto SSL certificate, 1-click deploys, great for SPAs
        - Cons- literally none
    - Tomcat
        - Pros- easy to configure, popular, good for dynamic content
        - Cons- not good for multiple concurrent requests)
    - Nginx
        - Pros- good for static content, good for multiple concurrent requests
        - Cons- less ideal for dynamic content, hard to configure)
- **Rationale -** We choose Firebase Hosting to statically host our React SPA since we can easily leverage the massive google cloud CDN and it requires 0 setup.