BookingApp

Evolution Document

Advanced Software Engineering

Professor McKee

Notional Implementation

BookingApp was to be a web-based application that was composed of a frontend component and a backend component. The frontend component was to consist of two subcomponents that interact with the Firebase backend. The first, used by diners that wish to reserve a table at a desired restaurant and the second, used by restaurants to manage their reservations and analytics/statistics.

The application separates itself from other table reservation software already on the market by displaying a floor plan of the restaurant and allowing users to select a specific table that they wish to reserve by its location or style.

BookingApp should integrate with a few well-known systems to enhance usability and familiarity. Yelp will be incorporated into the restaurant selection process, allowing users to assess reviews before booking their reservation. For authentication services, Google, Twitter, and Facebook will be provided for users to sign in with their already established accounts.

For the selection of restaurants, we wanted to be able to pull from a list of all of the restaurants that matched the diners specifications such as cuisine type, wait time, and rating.

Issues Encountered/Discoveries Made

Our biggest issue right away was lack of knowledge of the technologies that we selected to use. We wanted to turn this project into a learning experience for our team, so we purposefully picked technologies and languages that we did not know. Some examples include Angular which is a Typescript-based front-end web application platform developed at Google, Firebase which is a back-end mobile and web application platform developed at Google, and Node.js which is a cross-platform JavaScript runtime environment for servers and applications. Npm, the package manager for Javascript, was the source if many headaches a few of our team members. They particularly struggled when pulling in new changes from the git repository where the project would compile but not serve the app correctly, with no traceable errors or bugs.

For table selection, we wanted to use Konva but due to inexperience using this javascript canvas library, and having to implement multiple callbacks for each table we were not able to complete this task.

Due to so much lost time learning all of these technologies we were never able to catch up in the design, development, and testing phases. We had to prioritize which of the functionality was going to be implemented and what we would leave for the future.

Current State

The frontend component was supposed to have two subcomponents, one for the diners and one for the restaurant management but, we were not able to achieve the subcomponent for the restaurant management. Since we were not able to implement the restaurant management subcomponent, we were not able to implement the statistics page because there was no parent component to allow access.

We also were not able to accomplish displaying the floor plan of each restaurant for the diners to be able to select their tables. Since we struggled with displaying the floor plan, we made this design simpler by just listing the available tables so that the user can still select their table, just not be able to view where it is located.

We were not able to integrate BookingApp with any other the systems listed above due to time restraints and prioritization of functionality. One system that we were able to use was Google for login authorization to the website since we used Firebase for our database, this functionality was built in and free to use.

We did not prioritize the restaurant selection feature since we were running out of time so we made a drop down that was pre-populated with a few restaurants that we selected. This was a quick fix that made the website testable, but not yet usable to the public. There was no way to filter the restaurants or to look at any information about the restaurant.

We are, not, however, using any canned data. The restaurants, tables, users, and reservations are all retrieved from/written to the actual database. We wanted to ensure that as we added new functionality, we hadn't deferred any key architectural implementation. Other than some convenient web forms to facilitate data entry, the architecture we developed should scale nicely with additional features.

Future Enhancements

What we would like to do in the future is implement all of the things that were discussed above to match our original requirements. In the first couple weeks of design and development we hit a bunch of the hurdles that held us back so we think that given a couple more weeks, these things would be possible.

Some other things that we would like to implement after that would be

- A tool that allows the proprietor of the restaurant to easily build and edit the floor plan by dragging and dropping styles of tables to define their custom layout to allow for changing restaurants.

- An online waitlist for restaurants that don’t allow reservations so that diners can join that waitlist online and wait on their own time before they get to the restaurant.
- Online payments so that diners can pay for their meal through the website.
- A favorites list for the diners most frequently reserved restaurants.
- Restaurant suggestions based on previously booked reservations.