Grubite

By

Osahon David Osemwegie

In

The Irving K. Barber School of Arts and Sciences

(Honours Computer Science Major Computer Science)

THE UNIVERSITY OF BRITISH COLUMBIA

(Okanagan)

April 2019

Table of Contents

Abstract

Grubite is an online platform that was built to make the process of choosing what to order at a restaurant simpler. The software is made up of 3 different components: the RESTful API, the web application and the mobile application. The node server API was developed using Node.js, the Web Application was built using React.JS and the mobile application with built with the react-native JavaScript framework. The Web and Mobile applications are the two components that the user will be able to access. The web application is what the restaurant owners will use to upload their menu items. The Mobile application is the main and largest component of this project. Through the mobile application, a user can view a restaurants full menu with photo and reviews and ratings from other users.

Acknowledgment

I would like to thank Dr. Ramon Lawrence for helping supervising and helping me through the development of this project.

1. Introduction

Restaurants are part of our everyday lives. People go to restaurants for several different reasons such as birthdays, business meetings, dates, anniversaries, etc. One thing that every person has to do when they go to a restaurant is decide what food on the menu they want to order. Most of the time, especially if it's a restaurant that you have not been to before, looking at the large menu with all the different options and trying to pick something to order can be an extremely overwhelming process for many reasons. Here is a short list of reasons why this can be an overwhelming process:

- The menu may not be in a language that you understand.
- The meals on the menu are quite expensive and people don't want to end up wasting their money on something that they may not enjoy.
- Most menus are just words and don't have an image of what the food item looks like, so the individual has no idea what the final product of their order will look like.
- There are too many options, leading to choice paralysis.

During the summer of 2018, while at a friend's birthday dinner party at the Old Spaghetti factory the idea came to me. I had been to that restaurant once before, but had only ordered a salad. This time would be different. Looking through the menu trying to decide what to order, the decision was extremely difficult due to the fact that the names of the food were all in Italian. I had absolutely no idea what I was looking at. I had no idea what the dish was, what it looked like, what it tasted like and definitely didn't know if it would be worth the money. In an attempt to see if there was better information on the menu, I went online to the company website. No luck. I tried to find an app to download that could possibly have a digital version of the menu on it, still no luck. I ended up getting the only thing that sounded familiar on the menu; chicken alfredo. Throughout the evening, it really bothered me that in the technology filled world that we live in today, there was no app that I could download on my phone where one could look at a digital version of a restaurant's menu.

1.1. Motivation

The motivation for the project was to build an online platform that would make the process of selecting what food to order at a restaurant less complicated.

1.2. Goals and Objectives

For this project, there are 3 different components that I will develop:

Web Application:
This is going to be a web portal that restaurant owners will use to digitize their menu. This means that they will be able to upload different food items to their menu that belong to one of the main 4 categories: Appetizers, Mains, Desserts and Drinks.

Mobile Application:
The mobile application is the main and largest component of this entire project. The Grubite app is what restaurant customers will use to view the menu information for their restaurant of choice. They will be able to see an image of the menu item, the price, a description and reviews. Customers will be able to leave their own review of each particular food item.

Server (RESTful API):
The API is the backend of the entire system. The API is the only component in the entire system that has a direct connection to the database. The API is a set of URLs that the web application and mobile application use to send and receive data to and from the database.

2. Background

2.1. Development Process

The development process was split up into 3 different phases. This helped to clearly plan out all of the tasks to be completed.

2.1.1. Planning Phase

This was the to-do list that was followed during the planning phase:

- Create the user interface mockups
- Design database schema
- Decide on and learn the different programming languages that were going to be used in the building process
- Design the overall software architecture and understand how all the different components of the platform would interact with each other
- Decide on exactly what features were going to be implemented on the web and mobile application
- Set up development environment

2.1.2. Development Phase

The development phase was the longest out of the 3 phases. During this phase these were the items completed:

- Created and setup the MySQL database (SQL DDL)
- Built web application using React.js
- Built node backend (API) with node.js
- Built mobile application with React-Native.js
- Directly translated the user interface mockups to code

2.1.3. Deployment Phase

The deployment phase of this project was to deploy the web application, the mobile application and the server on the internet.

2.2. User Interface Design

User interface (UI) design in a mobile application is an extremely important part of the whole process of developing software. The UI is the means that a user uses to interact with the program or software that has been developed. *User interface design* is the process of making interfaces in software or computerized devices with a focus on looks or style ("What is User Interface (UI) Design?"). The essence of UI design is to create designs that users will find easy and straightforward to use. For this thesis paper, there is development of a graphical user interface design for the web application and for the mobile application. When designing the UI for programs there are a lot of factors that come into play such as:

- button feedback
- font weight, style, and size
- color palettes

There are a lot more things that have to be factored in, but the few listed above are very common factors. Another very important factor is following modern industry standards. For example when a user is opening the login screen on a mobile application it is part of the industry standards to have the login button below the fields where the username/email and password is entered. It would throw the user off if the login button was positioned above the input fields. The best type of user interface is one where the learning curve is very small. This means that a new user should not have to think about the purpose of particular user interface features.

Adobe XD was used to create the initial design for the user interface. Adobe XD is a desktop application that is typically used to create and design user interfaces for websites and mobile apps. After the rough drafts of the user interfaces were designed, they were then employed as a starting point to develop the real UI for the different applications.

2.2.1. Colour

Colours plays a massive role in user interface design. It is a key component of the psychological impact of a design on users. ("The Role of Color in UX"). A properly chosen color palette can elevate a UI design from average to exceptional; but in the same way good can turn into great, a poorly thought out UI design colour palette can

totally ruin the entire user experience. The psychology of colour shows that certain colours, mainly the primary colors have meanings associated with each of them.

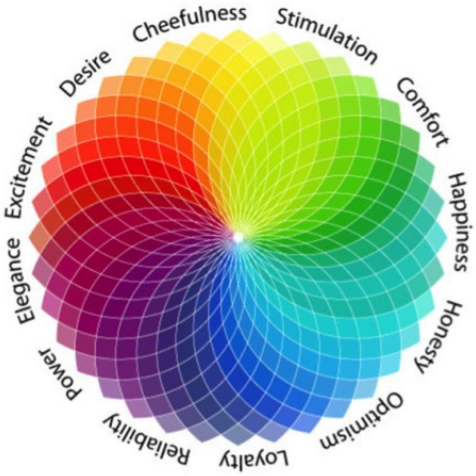

Figure 1: Psychology of Color Wheel ("The Psychology of Color in Web Design", 2016)

For the Grubite platform, it was developed using very simple and basic colours. The main color scheme is white, light grey and black.

2.2.2. Web User Interface

The user interface mockup for the web application was quite simple to design and convert to code. The image below is the initial UI mockup for the dashboard page that was designed during the planning phase of the project.

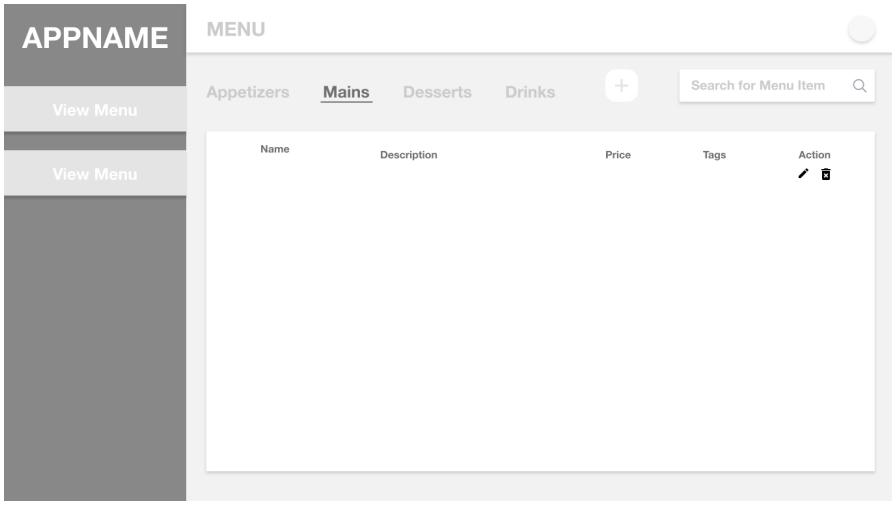

Figure 2: UI Design for Web Application

The next image is what the web application UI ended up looking like.

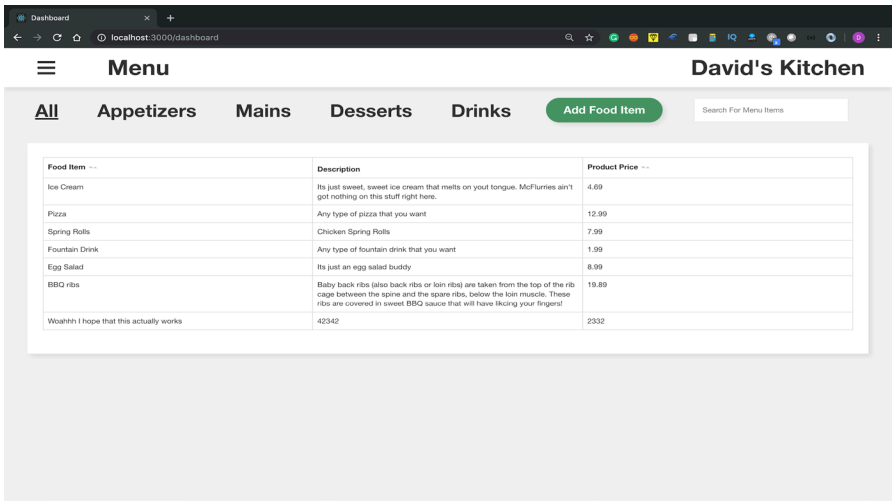

Figure 3: Final Dashboard UI

2.2.3. Mobile User Interface

Designing the user interface for the mobile application (app) was one of the more difficult challenges of this project for various reasons. The UI for a mobile application is so important that it can make or break your application. A user can download a mobile application from the app store and when the user opens the app for the first time, depending on the user interface, it can make the user love or hate using the app. For the design of the Grubite mobile application, all of the user interface elements were based off successful apps that millions of people use today such as Facebook Messenger, Yelp and OpenTable.

2.3. Technologies Used

Github

Github was used to store/backup all of the code for this project. It was also used for version control to view different versions of code files.

Google Cloud Platform

Google Cloud Platform is a cloud computing solution provided by Google to host web applications and store data. For this project the RESTful API was hosted on GCP, but later transferred to Netlify because GCP is very expensive to maintain.

MySQL

MySQL is the type of relational database that was used for storing the data for this platform.

Netlify

Netlify is a cloud computer software that hosts serverless websites and static web pages. Netflify has a free tier that you can use to host a limited number of projects. I ended up using Netlify to host the web application and the RESTful API.

Node.js

Node.js is a JavaScript framework that runs JavaScript code outside of a web browser. It's mainly used to code backend functionality of a website and to build a RESTful API. This project used Node.js to build the API.

React.js

React.js is a JavaScript framework that makes it easy to build user interfaces for websites and web applications. React allows users to make reusable components that have props attached to it that can be used over and over on different pages. For example, using regular HTML and CSS to create a header on a web page, requires copying and pasting the exact same HTML code into every single page that use that specific header on and change the text every single time. Using React.js made it possible to simply create a file that contained the code HTML and CSS code for the header components and add a 'headerText' prop to it that would determine what the text in the header would say. For every page that used that same header, all that was required was to import the file to the page and set the 'headerText' props to what it needed to say. This process makes it easy to build web pages because all that is required is fitting the different components together like pieces of a puzzle. React.js was chosen over other frameworks like Angular.js and Vue.js because React is focused primarily on user interface and is one of the easiest JavaScript web development frameworks to learn and implement.

React Native

React Native is extremely similar to React.js except that it is solely designed for the development of iOS and Android mobile applications.

RESTful API

REST stands for Representational State Transfer ("What is RESTful API? - Definition from WhatIs.com"). API stands for Application Program Interface which allows two computer programs to communicate with each other. A RESTful API is an application program interface (API) that uses HTTP requests to GET, PUT, POST and DELETE data. With an API, different end points (URLs) are created to perform different functions. For example the '/login' endpoint is used to log a user in and the '/users/3' endpoint could be used to retrieve the information for user id #3.

Trello

Trello is a project management software that allows the user to create and make different lists and boards to organize various tasks that have to be completed. For this project Trello was used to organize the different tasks that needed to be completed for each of the different components of this project. The image below is a screenshot of what the Trello board looked like.

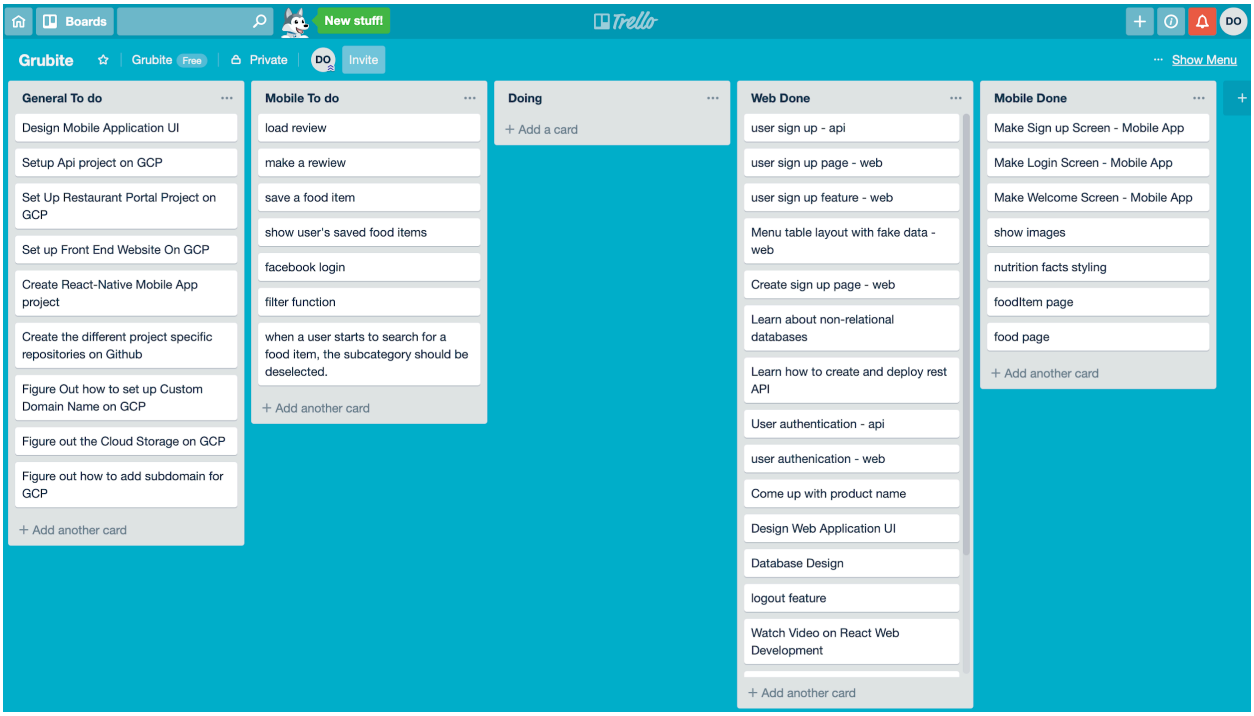

Figure 4: Trello Board

3. Software Architecture

The software architecture of the whole system is quite simple. It is made up of 4 main components:

- Database
- Server (API)
- Mobile Application
- Web Application

Figure 5 shows how the different software components interact with each other.

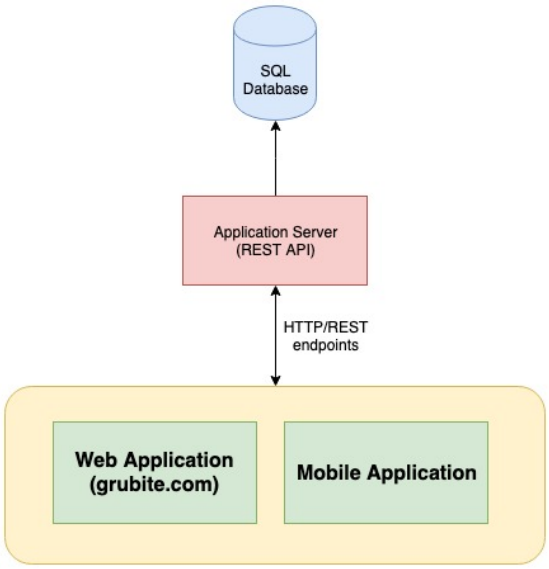

Figure 5: Overall Software Architecture

The web and mobile applications both connect to the application server via different RESTful API endpoints to access data from the database. The functions of the different endpoints will be expanded upon in a latter section of this paper.

4. Database

For the development of this project it is was quite difficult to determine what sort of database would function best for this particular type of project. During the planning phase of this project, while researching online I discovered that non-relational databases were the most common type of database that were employed in the development of mobile applications. Due to projects that I have worked on in the past, I only have experience with the MySQL relational database. The type of the data that would be stored on the database is mainly relational data, so using a MySQL relational database would be my best option.

The next issue that I encountered was that with most mobile applications, the process of connecting your mobile app directly to the MySQL database was extremely complicated and tedious. It was a challenge to work to resolve this difficulty, but I was able to work through it.

5. RESTful API

5.1. Development Process

Developing the API for the Grubite platform was a very complicated process due to the fact that I had never created something like this before so I had to spend a lot of time learning and understanding exactly what an API does.
There are various functions that an API can perform; everything from user authentication and data security to uploading files and modifying data on the database. The development process had to be done simultaneously with the development of the web and mobile application. As different functions for the web and mobile application were developed, I would have to create the necessary backend function/endpoint on the API that would allow me to send and/or retrieve the information as needed. For example, during the development of the mobile application I created the login screen on the mobile app and then created the login and authentication function on the backend that would allow the user to authenticate the login credentials that they entered into the mobile app.

5.2. Current bugs / Issues

There are currently no known bugs in the API.

5.3. Future Improvements

A major future improvement to the API would be to have proper user validation using web tokens. This would prevent unauthorized users from accessing data from the API.

6. Web Application: grubite.com

6.1. Development Process

The development process for the web application was relatively complicated because although I have built numerous websites in the past, they had all been built using basic HTML, CSS and JavaScript. I had never built a web application using a web development framework. The first task I tackled was figuring out the design what I wanted the user interface to look like. The image below is a screenshot of the initial UI mockup design.

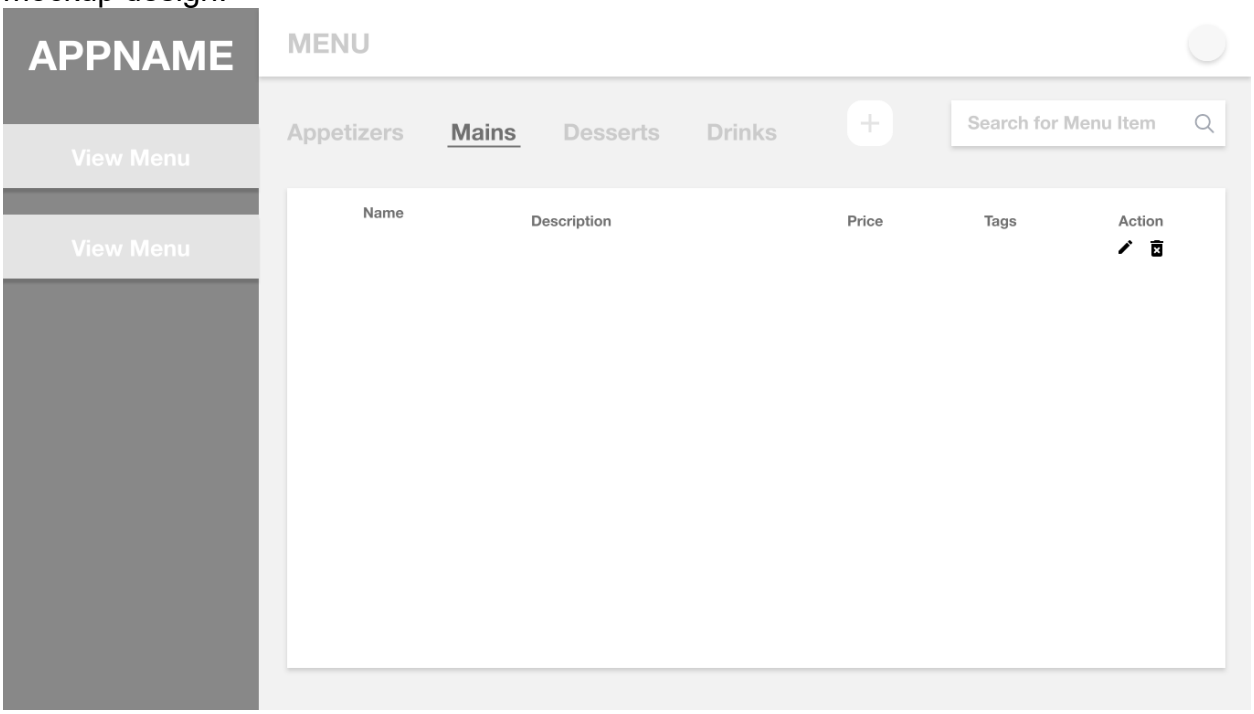

Figure 6: Screenshot of Web App UI Design

The next step after designing the interface of the web application was figure out what programming language was going to be used to build it. This required learning React.js and everything about React from scratch. This required many tutorial videos on YouTube. During the learning process while following the tutorials, I built other smaller React web applications, with each project teaching me a how to accomplish different types of functions in React. It took about 2 weeks to get a good understanding of how to use the React.js framework.

After learning React, I set up my development environment for the Grubite web application and began the coding phase. Here is the list and description of the process I went through while coding:

- **Creating a Login Page:** This is the page that a restaurant owner will use to log in to the web application. Creating this page took a lot longer than expected because I initially wanted to code everything from scratch my by myself but I figured it would be more effective to create the different components of the page using a CSS library such as Bootstrap. I did my research on the internet to find how to implement Bootstrap CSS into a React web application. It was a bit of a

learning curve because it was very different from working with basic HTML, CSS and JavaScript.

- **Restaurant Owner Registration Page:** Creating this page was quite straight forward after coding the login page because they had the exact same form components. The registration page just had more form inputs.

- **Figure out how to submit form data in React.js:** After the login and registration page was created, the next task was to learn how to actually send the form data to be processed. Before that I needed the login and register user end points to be functional on the API.

- **Created the dashboard page:** This was the most time consuming task for the development of the web application because as you will see in the walkthrough section of the web application, the page has many different components.

- **Created the form to add new menu items:** This was the last thing coded during the development of the entire project because I was not 100% sure what the exact data was that each food item had to have associated with it. So after I was done building the mobile application I realized that there were definitely more pieces of the data

6.2. Walkthrough

Step 1

When a user opens the web application, they have the option to login with their login credentials or they can register with the necessary information.

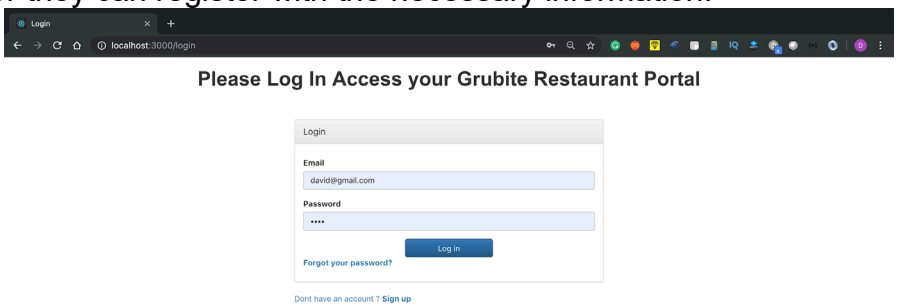

Figure 7: Web Application Login Screen

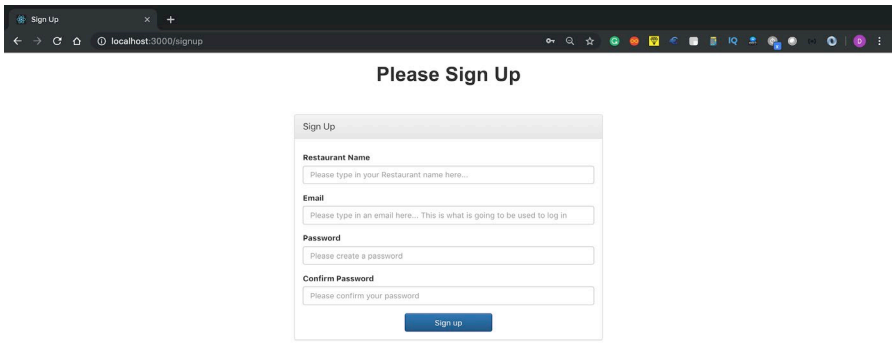

Figure 8: Web Application Registration Page

Step 2

After the user logs in, they are led to the dashboard. This is the main component of the web application. This is where the user can view all of the items on the menu; they can view the different menu categories and add different sub-categories as needed.

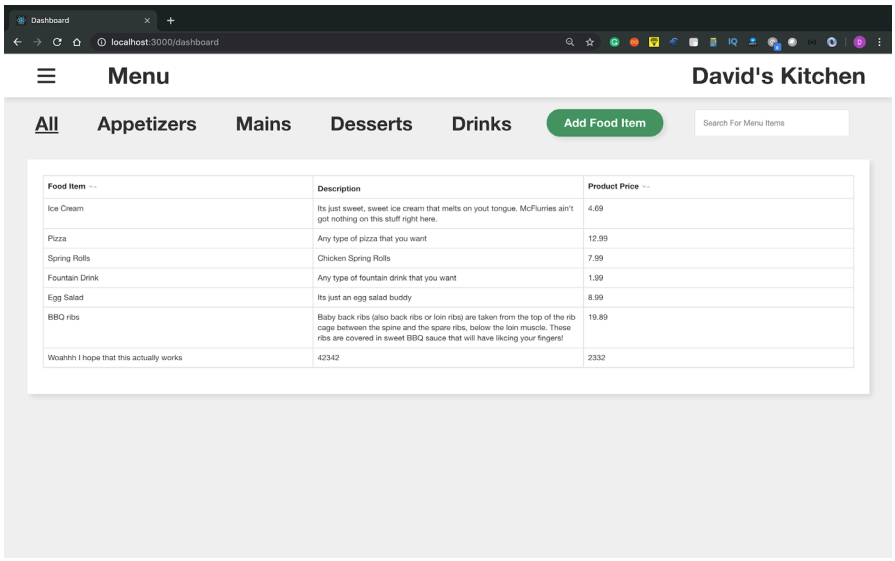

Figure 9: Web application dashboard

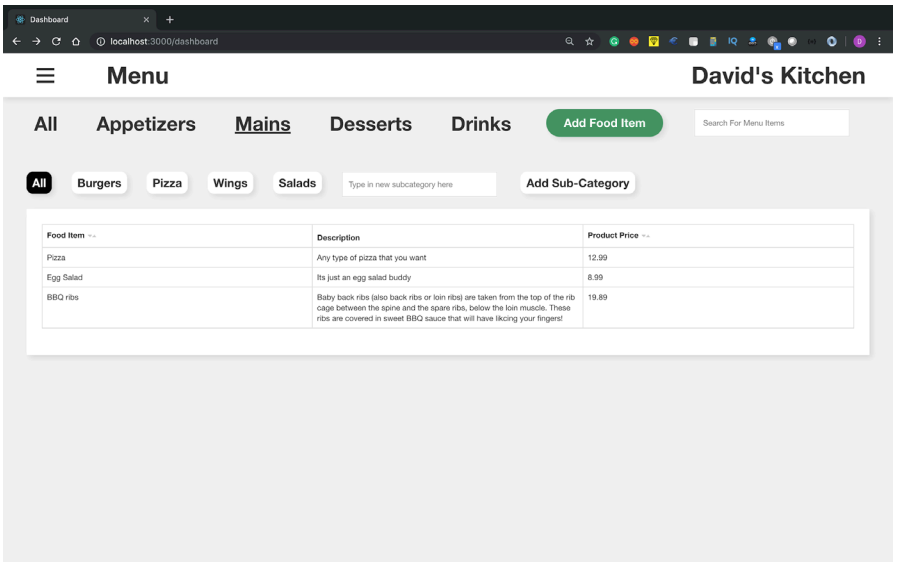

Figure 10: Viewing Menu Category and Sub-Categories

Step 3

When the user clicks the ‘Add Food Item’ button, this is the form that pops up on the users screen.

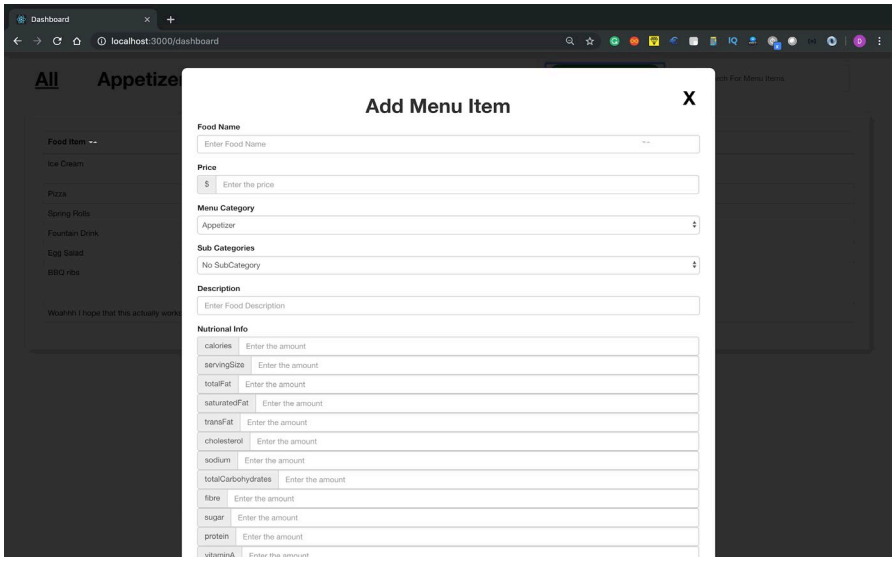

Figure 11: Add Food Item Form

6.3. Current bugs / Issues

The only current bug with the web application is that initially when the browser is loaded, after the user logs in, their menu does not load until the page is reloaded.

6.4. Future Improvements

Two things that will make the user experience better is improving the overall user interface of the web application and adding activity indicators to give the user feedback on what is going on behind the scenes.

7. Mobile Application

7.1. Development Process

Developing the mobile application was by far the most time consuming portion of this project. I had no previous app development skills that could have been used during the development of the mobile application because all the skills and knowledge that were required to build this app was new territory. The Grubite app was built using the React-Native JavaScript framework. To be able to use this programming language, I bought an online React-Native development course and spent about three weeks going through the course content.

The thing that was done to start the process of building the mobile app was to design that user interface of the mobile application. This step was really important because as discussed earlier, the user interface is one of the most important elements when building a mobile applications because it can make or break your application. For the user interface, the goal to make it simple, easy to use and understand. The image below is a screenshot of the user interfaces that was designed at the start of this project.

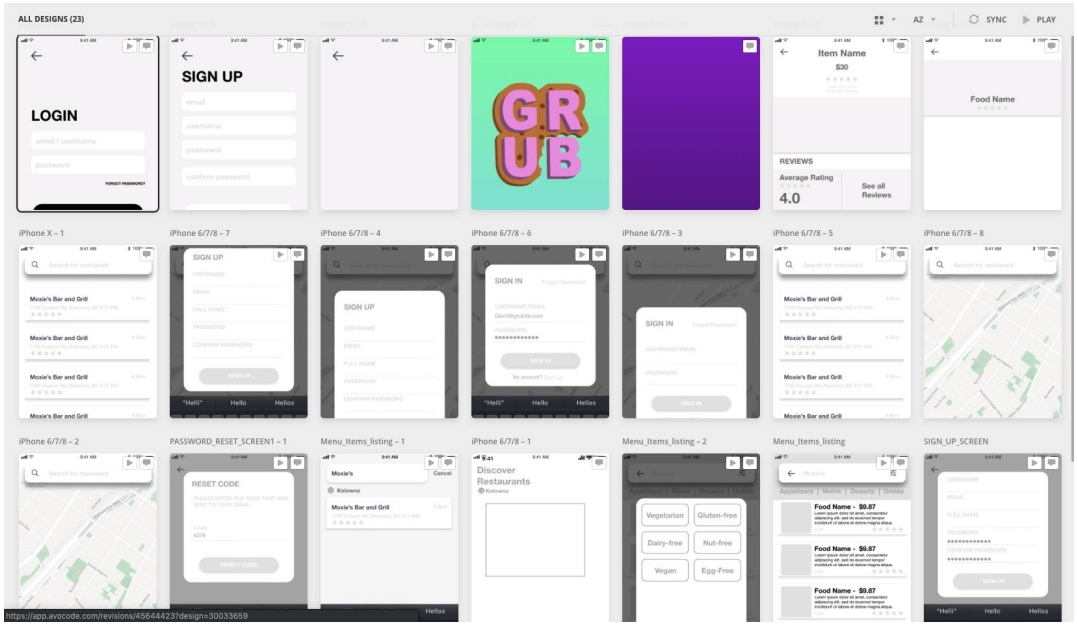

Figure 12: Screenshot of UI mockups for Mobile Application

After the user interface, the development environment was setup and actual development began. These are the steps below; listed in chronological order, describing the coding process:

- **Created reusable components**: React-native is based on React.js which is built around the concept of reusable components. The first thing that was done was to design and code a small library of reusable components like buttons, form inputs, cards, page headers etc. As every new page was created, these ‘common’ components were imported, used and modified as necessary.

- **Created user authentication process**: The user authentication process includes 3 screens. The welcome screen, the login screen and the sign up screen. All 3 pages use the exact same authentication button, just a different colour.

- **Created the bottom tab navigation**: A bottom tab navigation is the main navigation component that splits up the 3 different components. The discover section, the saved section and the user profile section.

- **Created the restaurant menu screens**: The pages that were included in this step were the search screen, the menu screen and the food item screen.

- **Created the food Items actions button:** The food item screen contains 3 buttons that lead to 3 different screens: the review screen, the view nutritional info screen and the leave review screen.

As the mobile application was being built, the necessary endpoints to retrieve and send different pieces of data were also created simultaneously in the API.

7.2. Walkthrough

Step 1

When a user opens the Grubite app on their phone, if they are not logged in previously, they are greeted with the welcome screen, in which they can choose to either login with their login credentials or they can choose to register for an account.

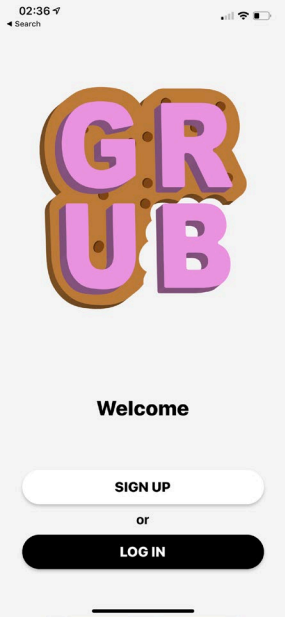

Figure 13: Welcome Screen

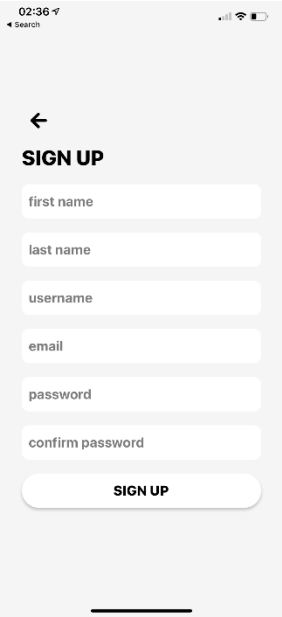

Figure 14: Sign Up Screen

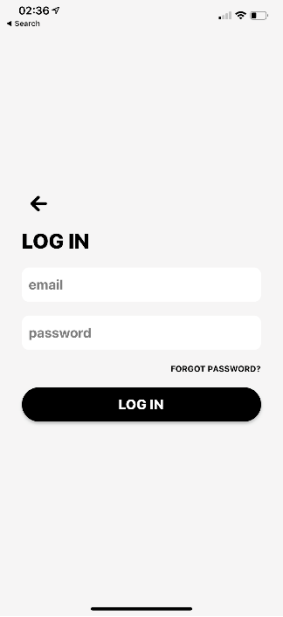

Figure 15: Login Screen

Step 2

After the user logs into the app, the first page that they see is the discover page. From here the user can search up all of the different restaurants that have been listed on the Grubite app.

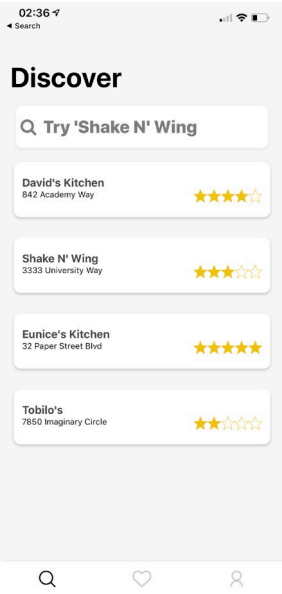

Figure 16: Discover Screen

Step 3

Once the user selects the restaurant menu that want to view, they are then taken to the 'view menu page' where they can see all of the menu items that this particular restaurant has on its menu. They can select that particular menu category that they want to view; either Appetizers, Mains, Desserts or Drinks. They can also search for

menu items using the search bar on the top of the menu page. Once they click on the menu item that they are interested in, they are taken to the food item screen.

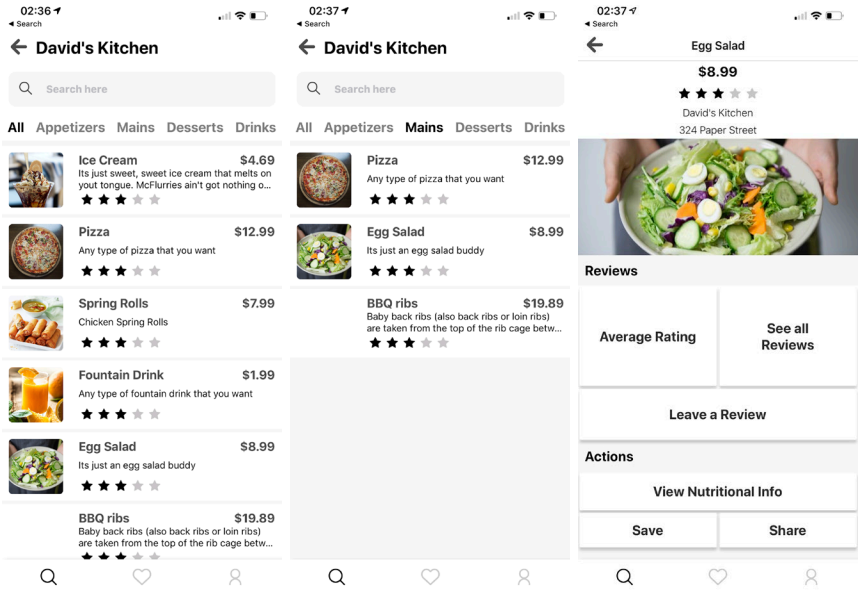

Figure 17: Menu Screen Figure 18: Selecting Category Figure 19: Food Item Screen

Step 4

Once the user is on the food item page, there are 3 buttons that they can click:

- The ‘see all reviews’ button will take them to the reviews screen where they can view all of the reviews for that particular food item.

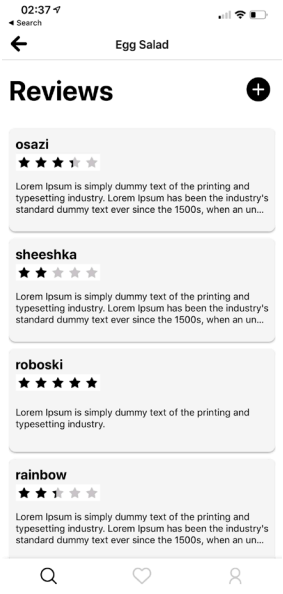

Figure 20: Reviews Screen

- The ‘leave a review’ button, will take them to screen where they can make and post their own review.

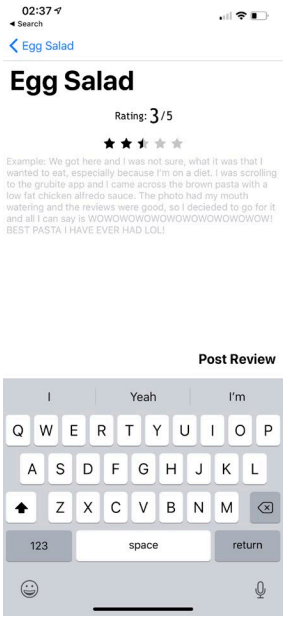

Figure 21: Leave Review Screen

- The ‘view nutritional info’ button will take the user to a screen where they can view a nutrition label (if available) for the selected menu item.

Figure 22: Nutritional Information Screen

Step 5

On the food item page, the user can also click the save button, which will add the food item to the user’s saved list, which they can access quickly in the future.

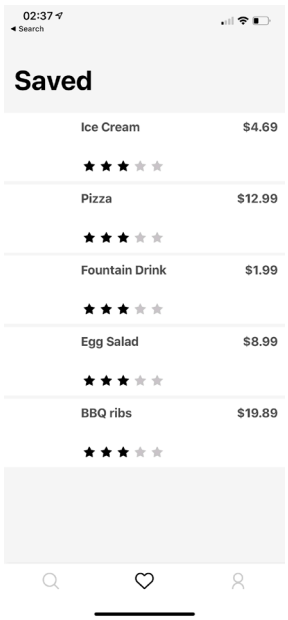

Figure 23: Saved Food Items Screen

7.3. Current bugs / Issues

One issue with the mobile application is that there are currently no activity indicators throughout the mobile application. So there is no user feedback for when an activity is going in the background.

7.4. Future Improvements

- Add activity indicators throughout the entire application.
- Add a pulldown to refresh for all of the different types of lists through the application.

8. Overall Future Improvements and Conclusion

8.1. Overall Future Improvements

There are a few more features that may or may not be implemented into the platform in the future.

Allow user to pay for meals through the mobile application

In the future, a user should be able pay for their meal at a restaurant within the Grubite mobile application itself. This is useful for a couple reasons:

- It would take the user experience to the next level because as a user if I am able to view the menu of a restaurant, naturally I would want to be able to pay for my meal through the app.
- It will allow the implementation of a rewards program in the future.

Filtering capabilities for the mobile application

In the future a user should be able to filter through the different food items to help narrow down their search to find the perfect item for them. There would be a filter button next to the search bar on the menu page. For example if a user is vegan, they should

be able to go on the mobile application and view only the vegan friendly options on the menu.

Data analysis tools on the web application for restaurant owner

In the future because users of the mobile application will be able pay for their meal through the mobile app, there will be a lot of data collected on the meals that are ordered. A restaurant owner will be able to see what meals are ordered the most and what meals users enjoy the most. They will be able to view ratings and reviews for each menu item, and they will also be able to see recommendations that users may leave for a particular menu item. For example, if there is a dish that a restaurant makes that is too salty, users on the Grubite app will say that the food is too salty; then the restaurant owner will be able to see this and make the necessary improvements to make the dish taste better or remove it from the menu entirely.

There are still a lot of micro-improvements that can be made but the 3 listed above are the main features that are going to be implemented and improved upon in the near future.

8.2. Conclusion

In conclusion, building this online platform has been a roller coaster of emotions. It was fun yet difficult. The skills that were acquired are skills that will be used in my professional career; now as well as in the future. There is still a lot of room for improvement with the software, and I hope to continue working on this even after my graduation.

References

The Psychology of Color in Web Design. (2016, April 26). Retrieved from https://www.vandelaydesign.com/the-psychology-of-color-in-web-design/

The Role of Color in UX. (n.d.). Retrieved from https://www.toptal.com/designers/ux/color-in-ux

What is RESTful API? - Definition from WhatIs.com. (n.d.). Retrieved from https://searchmicroservices.techtarget.com/definition/RESTful-API

What is User Interface (UI) Design? (n.d.). Retrieved from https://www.interaction-design.org/literature/topics/ui-design