Wentworth Institute of Technology
Senior Project

TruckIt Design Document

Prepared By
Group 7
Thomas Michael michaelt@wit.edu
Nick Faro faron@wit.edu
Instructor: Salem Othman

1. Introduction

1.1. Project Overview

TruckIt aims to revolutionize the food truck industry by implementing a mobile application to make food and ice cream trucks more accessible. By using real-time location tracking, menu viewing capabilities, and a user-friendly interface, TruckIt will provide a seamless experience for users to discover and interact with street food vendors.

TruckIt serves as a food and ice cream truck locator, with profile and menu information corresponding to the truck. It also gives trucks ability to customize profile and menu, and track location.

Users will be able to view the location of all active trucks in their area as well as their menu. Real-time location tracking ensures that users can always find their favorite trucks while menu viewing capabilities allow for easy browsing. On a truck-to-truck basis, doorstep delivery and events can be scheduled upon user request, which allows a flexible way of finding the truck or requesting for the truck to come to you. This aims to provide customers with a way of knowing where to get street food since, at times, it can be difficult to come across one as they commonly drive and park in various locations.

The application will be built using Flutter which allows a single codebase to be used for development in both iOS and Android, and Firebase will be used for backend services. The real-time location updates and communication will be handled by Google Maps and WebSocket, which shares data with Firebase.

TruckIt targets food trucks who may not be getting consistent customers throughout the day. Many trucks find themselves spending most of the day waiting for or searching for customers while they could be driving to hot spots or delivering directly to their customers for more consistent sales. Ultimately, TruckIt serves as a dual-purpose app: for customers who cannot find food trucks and food trucks who cannot find customers.

1.2. Components

1.2.1. Flutter - UI development kit
1.2.2. GitHub - Version control
1.2.3. Google Maps API - Location tracking/display
1.2.4.iOS and Android Simulators for testing
1.2.5. Flutter framework (framework provided by Google that allows the same code to be ran on any mobile device or computer)
    1.2.5.1. Dart Language
1.2.6. Google Maps API
1.2.7. Firebase (Google's backend cloud service for application development)
1.2.8. User Auth Firebase's "Auth" automatically takes input as sign in credentials and verifies upon previously created accounts. It stores this information here, and also allows you to configure the sign-in options. Anonymous was used for customers as they do not have to make an account, which enabled some reading and no writing permissions.
1.2.9. Firestore Database The main database, containing data in a hierarchical structure. It is broken up into collections, which contain other collections, etc. Some collections contain documents, in which their fields hold the data. The two main collections are users and companies, where user contains company id to correlate to corresponding company.

Within the company collection, it contains a truck collection, profile and menu collection, and menu attributes such as section, item, and item information.

1.2.10. Realtime Database
1.2.11. Storage Holds images which correspond to URL in Firestore, allowing the application to store, retrieve, and display the images

Scope

1.1. Milestone 1

Static UI implementation for the style of the UI. This includes account information, settings, filters, menu display, all on the customer app. The same will be implemented for the driver app, with the addition of a truck profile creation and menu creation page. (Estimated completion date: 6/17)

1.2. Milestone 2

Integrate the Google Maps API and get the GPS to display on the app. (Estimated completion date: 6/24)

1.3. Milestone 3

Implement database interactions send and retrieve data. This includes interactions between drivers and customers and any other user input displayed on the UI. (Estimated completion date: 7/08)

1.4. Milestone 4

Implement the tracking of a truck, uploading images. User’s data needs to correspond with account information to view only their data, and ensure they are only ones with access to write. (Estimated completion date: 7/25)

2. System Requirements

2.1. Functional Requirements

2.1.1. Account Management

2.1.1.1. Upon system startup, a user shall have the option to create an account.

2.1.1.2. Upon system startup, a user shall have the option to login to an existing account.

2.1.1.3. User shall be able to logout.

2.1.2. Customer App

2.1.2.1. Users shall have a map view displaying the real-time location of active trucks in a user specified radius.

2.1.2.2. Users shall have a list view of all trucks and their menus. Menu should be available by clicking on corresponding truck on the map.

2.1.2.3. Users shall be able to search for food and apply filters (active or not, distance, ratings).

2.1.2.4. Users should be able to click on trucks expanding their data.

2.1.2.5. Users shall be able to navigate to a truck's location.

2.1.3. Truck App

2.1.3.1. Users shall be able to verify that they are a real food truck.

2.1.3.2. Users shall be able to create and edit a public truck profile and menu.

2.1.3.3. Users shall be able to toggle their active status. When toggled on they will be visible to customers.

2.1.3.4. Users shall have the option to allow/disallow delivery and accept/deny the order.

2.1.3.5. Users shall be able to view hotspots on the map.

2.1.3.6. Users shall receive a notification when a customer requests an order and can accept/deny it.

2.1.3.7. Users shall be able to view customer ratings of their truck, as well as respond.

2.2. Non-Functional Requirements

2.2.1. The app shall load in under 2 seconds under normal conditions
2.2.2. Real-time updates and location tracking shall be consistent and accurate.
2.2.3. App shall be able to handle a growing number of users, including trucks who are being tracked.
2.2.4. The app shall have an interface that is user-friendly and does not require a tutorial to navigate through.
2.2.5. The app shall be able to run on both iOS and Android devices.
2.2.6. The app shall support different screen sizes and resolutions.
2.2.7. User login data shall be encrypted.
2.2.8. System should allow email and password login through truck app.
2.2.9. System should automatically authenticate user upon customer app startup

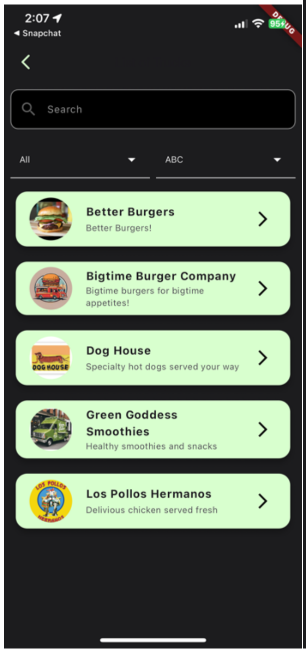
3. System Design

3.1. Design Decisions

The project will use a client server architecture. The client side will be the customer and truck applications. The backend services, such as the Google API and Firebase, correspond to the server side. The client applications will send requests to the backend such as real-time location updates and account editing/viewing, image upload, truck viewing, and map viewing. The server side will process the requests made by the client, perform the logic, update/retrieve data from the database, and send it back to the UI to be viewed by the user. Firebase handles this in the cloud, so the application has virtually unlimited space to store things barring any costs.

3.1.1. App Layout

TruckIt will consist of 2 separate apps. One that is solely for customers, and one that is solely for the truck drivers. We have isolated the 2 because they will be completely different experiences. For the customer app, they will be brought to a map screen without necessity to sign in. Live trucks will be viewable here. An icon with will redirect the user to a separate list page, where they can see the activity of nearby trucks including their menus and filter options. Each truck is clickable bringing you to a menu screen. For the truck app, they will have to sign in to an account, as user data is being stored. When initially logged in, users will be presented with the list of trucks they have. If they have not created a truck, they can hit create truck at the bottom of the screen. Once created, each truck is its clickable widget, allow editing and going live.



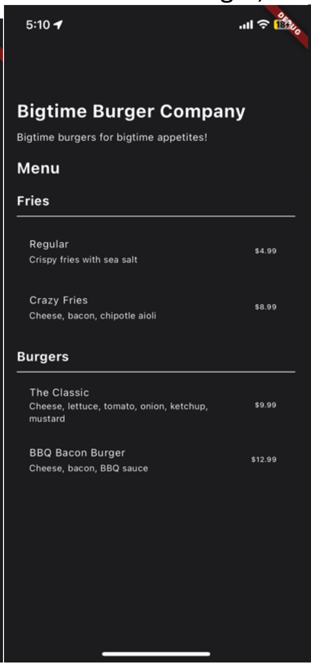
Better Burgers
Better Burgers!

Bigtime Burger Company
Bigtime burgers for bigtime appetites!

Dog House
Specialty hot dogs served your way

Green Goddess Smoothies
Healthy smoothies and snacks

Los Pollos Hermanos
Delivious chicken served fresh



Bigtime Burger Company
Bigtime burgers for bigtime appetites!

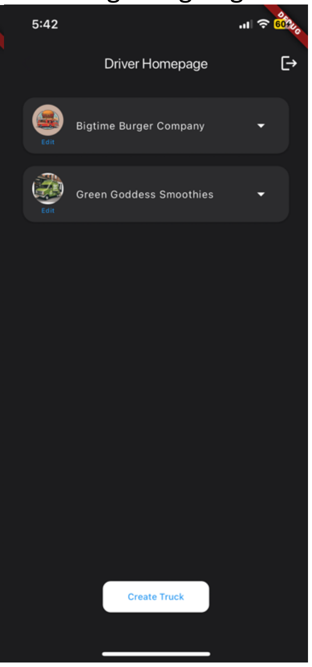
Menu

Fries

Regular Crispy fries with sea salt	$4.99
Crazy Fries Cheese, bacon, chipotle aioli	$8.99

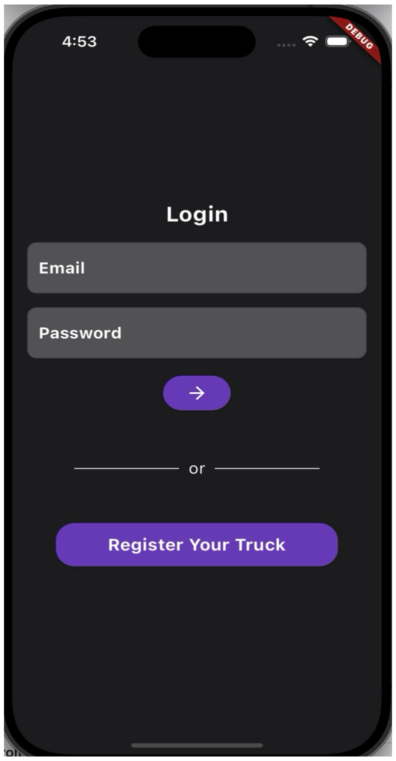
Burgers

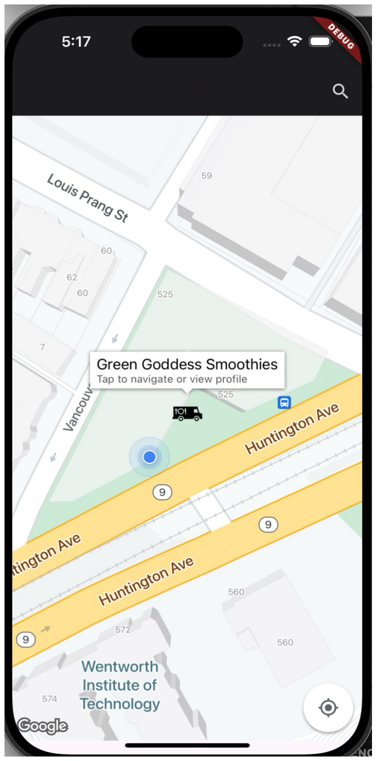
The Classic Cheese, lettuce, tomato, onion, ketchup, mustard	$9.99
BBQ Bacon Burger Cheese, bacon, BBQ sauce	$12.99



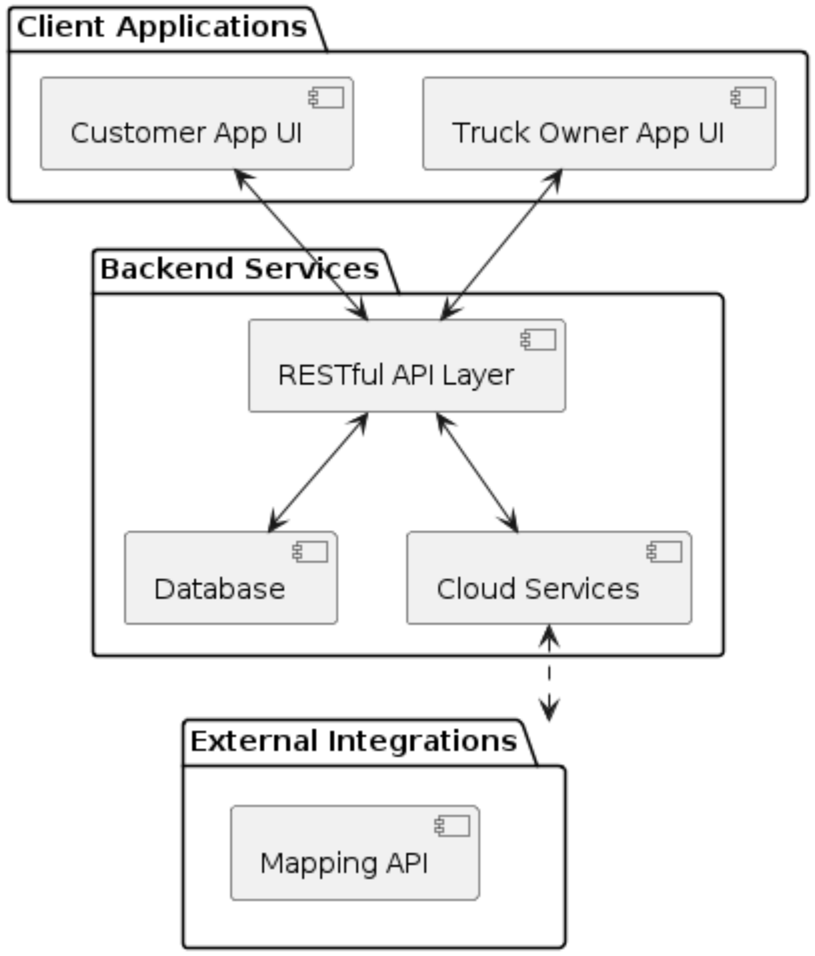
Bigtime Burger Company

Green Goddess Smoothies


4:53
DEBUG
Login
Email
Password
or
Register Your Truck


5:17
DEBUG
Louis Prang St
Green Goddess Smoothies
Tap to navigate or view profile
Huntington Ave
Huntington Ave
Huntington Ave
Wentworth Institute of Technology
Google

3.1.2. System Architecture Diagram



4. Project Goals

4.1. During Semester

To have a successful project, we aim to implement all functions of the app, but it can't be confirmed that we will have actual food trucks using the app upon initial release. The following all contribute to the goal of having a working app:

4.1.1. Successfully implement all functional requirements.
4.1.2. Test and ensure non-functional requirements are met.
4.1.3. Implement use of a cloud service to host and scale the project.

4.2. Additional Goals

4.2.1. Releasing TruckIt on App Store and Google Play Store.
4.2.2. The main additional goal of the project is to get trucks on board with our app so that it becomes populated with trucks, menus, etc. Without any active trucks, there is no app, so this will be the first step in officially releasing our app on the App Store and Google Play store.

4.2.3. An automated authentication process for new trucks to sign up for the driver app. This is crucial as we cannot allow anybody with a phone to become a “driver.” Until this is complete, we will be manually verifying the authenticity of trucks when they sign up by looking at things like business licenses. The automated process will consist of receiving submission of a permit from Local Board of Health (LBOH) and matching it with the user’s credentials.

5. Project Outcomes

5.1. The project goals during semester were all completed and exceeded.

5.2. Firebase served as our cloud service, providing numerous functions such as data retrieval, data input, image storage, real-time location updates via coordinates, and user authentication/credential storage.

5.3. All functional requirements were met, except those that fit under our additional goals which were not meant for the scope of this semester.

5.4. Testing and completion of back-end implementation went smoothly after everything was configured. Google’s console gave us a UI which shows all the data being stored and integrated into our application. This made it very easy to see how data was stored, what type of data we can store, and how to correlate data between services. This was especially useful in image URL storing in Firestore, corresponding to an image in Storage. Firestore acted as the bridge between getting and displaying the images and the storage of these images.