# IM3080 Design and Innovation Project

# (AY 22/23 Semester 1)

# Project Report

# 

# Title: Kung Food Hippo

# GitHub: <https://github.com/SpiderPigzz/Design-and-Innovation-Project>

# Submitted by: Group 2

# Supervisor: Prof. Teh Kah Chan

## Content Page

[**IM3080 Design and Innovation Project**](#_heading=h.gmzsfp8qz2l) **0**

[Content Page](#_heading=h.rzfl77y0y2ut) 1

[1. Background & Motivation](#_heading=h.3a33k9nteg1v) 2

[2. Objective](#_heading=h.fli4uo1oudt2) 3

[3. Review of Literature/Technology](#_heading=h.4y6q8gknkkqj) 5

[3.1. React Native](#_heading=h.jpkc9p72m8pu) 5

[3.2. GitHub](#_heading=h.ll9cjwy17kf2) 6

[3.3. SQLite](#_heading=h.aap8jnce5bvp) 6

[3.4. Flask/Gunicorn](#_heading=h.kqo0m0r0ooj2) 7

[3.5. Google API](#_heading=h.h6cvk713q2fh) 7

[3.6. Photoshop and After Effects](#_heading=h.dey6wg1bm3om) 8

[3.7. Canva & Figma](#_heading=h.t27ktpkxinw1) 8

[4. Design and Implementation](#_heading=h.xxu5inkucn6b) 9

[4.1. UI Design Consideration](#_heading=h.7m6eoe24g7ll) 9

[4.2. Map](#_heading=h.5stobz219ojk) 11

[4.3. Rating System](#_heading=h.nmstl3h6cv9a) 12

[4.4. Final Design (with block diagrams) (For software, use the proper Software Engineering Diagram, such as Use case diagram, Sequence diagram, etc.)](#_heading=h.38g7lx6ij6xg) 14

[4.5. Implementations](#_heading=h.jk42d4v07scx) 20

[5. Conclusion and Recommendation](#_heading=h.a6ok2uj97we8) 24

[5.1. Conclusion](#_heading=h.bpt7eqm8arie) 24

[5.2. Recommendation for Future Works](#_heading=h.poef5kcutbrz) 25

[6. Appendices](#_heading=h.h9j92y6x3w02) 26

[6.1. Design Diagrams](#_heading=h.sz0f1q9c66w) 26

[6.2. Guides](#_heading=h.lonxxpeh28zd) 26

[6.3. Source Code](#_heading=h.e5dhoxjp8gli) 26

[6.4. Project Management Schedule and Task Allocation/Breakdown](#_heading=h.axmyhd81inhm) 27

[6.5. Others](#_heading=h.acsz88eb1qu2) 28

[7. Bibliography](#_heading=h.mq8xt48sz7d) 28

## 

## Background & Motivation

This project starts with a question in mind: what troubles are we facing when using software technology? We brought this question to our lives, and we spotted a common frustration: food delivery apps. While there are quite a few delivery apps in Singapore, they failed to do their job well in satisfying user demands and providing a decent user experience.

After analyzing the popular delivery apps out there in the market, we identified several pain points which potentially diminish the delivery experience.

1. Branding does not clearly reflect company culture
2. User interface makes it difficult to access all in-app features
3. Poor review system with insufficient content and media confuses users
4. Users are unable to order food from multiple stores in one delivery

Therefore, these provided us the means to test out our capabilities as well as learn new things which will value-add to our future projects. Inspired by the FoodPanda delivery app, our main motivation for this project is to create a simple and user-friendly online food ordering app that focuses on user experience and ease-of-access. Circumventing the distracting and often confusing layout and features of the typical food delivery apps and integrating innovative new features that significantly improves the quality of life of the users. Features that provide accurate and intuitive user reviews, as well as creating a more optimized way of group ordering.

## Objective

There are generally 3 stages in this project and each stage carries its own objective.

The first stage included the research and development on the core features required for a food delivery app: what is the dominant user flow of the app. We identified that to make the app usable, we need to work on both the front end (UI design and page development) and backend (database setup and API support).

The second stage involved fine-tuning and integrating the app pages. On stage one we were only concerned with developing individual pages and utilized a drawer to contain them for navigation. So, integration was the key. We also added some tasks to further improve the app itself from weekly meetings where we went through the app together.

The third stage had the main objective of innovating the app. We collected ideas both from the research during previous stages and those from the biweekly presentation and came out with many forms of innovation. We added animation of the Hippo (running, cooking, delivering) to make the app branding look appealing to the user, leaving a positive impression. We also revamped the rating system by changing rating from 5 to 10 to give a higher accuracy and flexibility. Not only that, but we also added food reviews with images and texts to provide richer content to users. Moreover, we included the “recent reviews” for each page to reflect the current user satisfaction on the restaurant. Bundle delivery is another innovation we implemented. The app allows users to choose food from multiple restaurants and the user only needs to pay for one delivery fee. This gives our customers more choices in one delivery and motivates them to make the order from our app.

For detailed task breakdown and project management schedule, please refer to the [project management google sheet](https://docs.google.com/spreadsheets/d/1MQUd-J3Qi_bcFBElXk0JxB_pIy6PrecgUGDyyFp20-Q/edit#gid=0).

## Review of Literature/Technology

### React Native

Our DIP requires collaboration among all the team members, therefore we needed a framework that is not only robust, but allows easy merge deconfliction when it comes to GitHub collaboration and version control. With React Native, the development can be compartmentalized in such a way that merge conflicts are few and far in between, speeding up the process of development due to the lack of code dependencies.

React Native is a JavaScript framework for creating genuine, natively rendered mobile applications for iOS and Android. Although it is built on React, Facebook's JavaScript toolkit for creating user interfaces, it targets mobile platforms as opposed to browsers. In other words, using the familiar and beloved JavaScript library, web developers can now create mobile applications that actually look and feel "native." Additionally, React Native makes it simple to simultaneously build for both Android and iOS because the majority of the code you create can be shared across platforms.

React Native applications are created using JSX, a combination of JavaScript and XML-like markup, much like React for the Web. The native rendering APIs in Objective-C (for iOS) or Java are then invoked by the React Native "bridge" in the background (for Android). As a result, your app will display utilizing actual mobile UI components rather than webviews and will have the same appearance and feel as other mobile apps. Your React Native apps may access platform capabilities like the phone camera or the user's location thanks to React Native's exposure of JavaScript interfaces for platform APIs.

### GitHub

For the version control and collaboration platform, we naturally used

GitHub as it is the most widely used platform for Git-based version control and software development. It offers each project access control, bug tracking, software feature requests, task management, continuous integration, and wikis in addition to Git's distributed version control.

### SQLite

SQLite is a lightweight relational database management engine, RDBMS,. SQLite is suitable for our project as it can operate in both an embedded or server based. SQLite is widely used for a large number of mobile apps (Delgado, 2019). Its lightweight, portable and cross platform nature makes SQLite ideal for mobile applications. The relational database model makes data read and write extremely efficient. SQLite is a fully featured SQL database, but is much easier to set up and integrate than other RDBMS. The typing structure for SQLite is also weaker than most SQL based databases, while this can make it more flexible and easier to work with, this did lead to several type and value exceptions while programming.

Initially we planned to have the database file as a local asset on the users device. However the multi-user nature and high-read traffic of our application lead to several problems. Having both the model and the view on the same app cause several race conditions database operation are inherently sequential whereas the graphical user interface is asynchronous by nature. Another issue is that as the database scales in size transferring the file becomes fairly impractical. Due to the above issues we decided to implement a model view controller architecture, where the mobile app interacts with the backend via representational state transfer, REST. This implementation allows a database call to be approximately atomic.

### Flask/Gunicorn

Due to the above mentioned design consideration, we have a REST like architecture for our model. Therefore we decided on the following technologies to implement a centralised web server for the mobile apps to interact with via HTTP.

Flask is a micro framework for python web applications. We chose Flask as it is lightweight and does not require any additional library or setup. This allowed for rapid deployment of the web application. Other benefits of using Flask include innate support for a RESTful architecture style and native unicode support.

Gunicorn is a Web Server Gateway Interface Hypertext Transfer Protocol Server. It is a python port of Unicorn for Ruby. It is not good practice to deploy on a development server as it can lead to stability, performance and security issues. Therefore we decided our web application should run on a Gunicorn server.

### Google API

Our DIP is a multi user application that also requires geospatial data and visualisation. Both these features have alot of design and technical considerations. To support multiple users, there needs to be reliable authentication services with security features like password hashing and reset. Furthermore, to support the delivery routing and mapping, we need to be able to convert text based addresses into geographical coordinates. After the addresses are converted into coordinates, a route between these points needs to be calculated, then displayed to the user.

Both these tasks would be time consuming to implement to an acceptable standard. However, Google has already created a robust and feature rich application programming interface, API. So there is no need for our group to reinvent the wheel, we can use Google’s authentication, geocoding, and routing APIs.

### Photoshop and After Effects

As we wanted to include unique graphic components that are not taken from

other online sources and it is not achievable through react native frameworks alone, we made use of Adobe Photoshop and After Effect for these graphic design tasks.

Adobe Photoshop is a raster graphics editing software which can be used for graphic design and illustrations. We decided to use it to create our 2D logo designs and objects to be used for in-app animations as it has multiple fx effects and painting tools which would produce the most desired result.

After Effect is a digital visual effects, motion graphics, and compositing application which is capable of editing frame by frame animations. It is also compatible with Photoshop where psd files can be imported into After Effect and animated each of the different components created .Therefore we used After Effect to create the 2d animations we included for visual components.

### Canva & Figma

Figma is a free browser-based web application for interface design where we can work together collaboratively in real time and is easy to learn. We decided to use Figma for the initial prototyping of wireframes for the application pages and also the final poster.

Canva is a graphic design platform that is used to create presentations, videos, posters and other visual contents. With canva pro, it is easy to make high quality videos using their extensive library of ready-made templates without being a professional video editor. We decided to use Canva for our promotional app video.

## Design and Implementation

### UI Design Consideration

1. Colour Theme

Our team chose to employ various shades of red as the main colour of our application since warm hues like red are the best choices for food branding. Given that red-colored foods are so prevalent in nature, people are more likely to feel hungry when they see the color red, which increases consumer appetite and boosts restaurant sales. Additionally, warm colours also complement one another well, giving users a pleasant aesthetic appeal. (Ellis, M. 2019)

1. Logo

We decided to make our logo out of a hippopotamus since rounder shapes are perceived as a kinder and more friendly one than square or rectangular. (Ellis, M. 2019) Due to its vast size, the hippopotamus could express sensations of fullness and happiness from the food it consumed, persuading people to look forward to the gratifying feeling they will experience if they choose to use our app to order food.

1. Visuals

Our team decided to incorporate many visual images and animations to make the user experience more self-explanatory and user-friendly as human brains can process visuals 60,000 times faster than text. The brain processes 90% of its information visually and also creators who offer visual material get a 12 times greater increase in traffic than those who do not. (29 Incredible Stats That Prove the Power of Visual Marketing) This also keeps our screen from being overly wordy, which could confuse users and scare them away from using our app.

1. Home

The main function of the home page is to be user-friendly, making it easier for them to browse our app. We implemented many features with graphic elements and color solutions to allow people to search in an intuitive and visual way. Users can find nearby restaurants with offers, hot deals and good reviews/ratings. We use high quality photos in our UI design to make app dishes more attractive.

### Map

Maps are one of the key elements in food delivery. Our team realized several features by utilizing the react native map components.

1. Delivery and Pick Up

Considering the Kung Food Hippo app offers the option for users to choose between delivery and self-pickup, different routings on the map should be provided. By switching the mode under MapViewDirections from ‘driving’ to ‘walking,’ google map API will recalculate the optimal route for users.

1. Bundle Delivery

This is one of the innovations in the Kung Food Hippo app. Before, if a user wants to order from multiple shops, he or she must make separate orders. However, this leads to paying multiple delivery fees. Understanding the user’s pain point, our team decided to come out with a proper solution: bundle delivery. We implemented this feature by generating a route across all restaurants which users place the order from, and then directing the route to the delivery destination. Our team also considered the possibility of this feature being abused. Therefore, we added constraints to the bundle delivery. The first restriction is that users can only order from 3 shops at a maximum. The other one is more complicated where we let the app decide whether this order is possible based on restaurant locations. For example, if the restaurants are too far away from each other, this order will not be successfully placed and users will be asked to amend the order.

### Rating System

* 1. One to Ten Scale

For reviews and ratings, most applications employ a one to five scale. In our application, we employed a one-to-ten scale instead because of its greater granularity. A one-to-five scale does not allow for much differentiation between a good shop from an excellent one. As a result, several shops have a similar score.

* 1. Tree Rating Categories

Rating a store solely on one numerical score is too reductive. As one store may excel in one area and fail in others. We implemented a system where shops are rated on three separate criteria. food, value, and packaging.

* 1. Removing Outliers

There are two main statistical methods to remove outliers - Z-scores and percentile based ranges. (Bonthu, H. 2022) Z-scores cannot be used for our application because it requires the data to be normally distributed. We cannot assume that the rating data is normally distributed nor can we approximate it with a normal distribution. Therefore we will use percentile based to detect outliers.

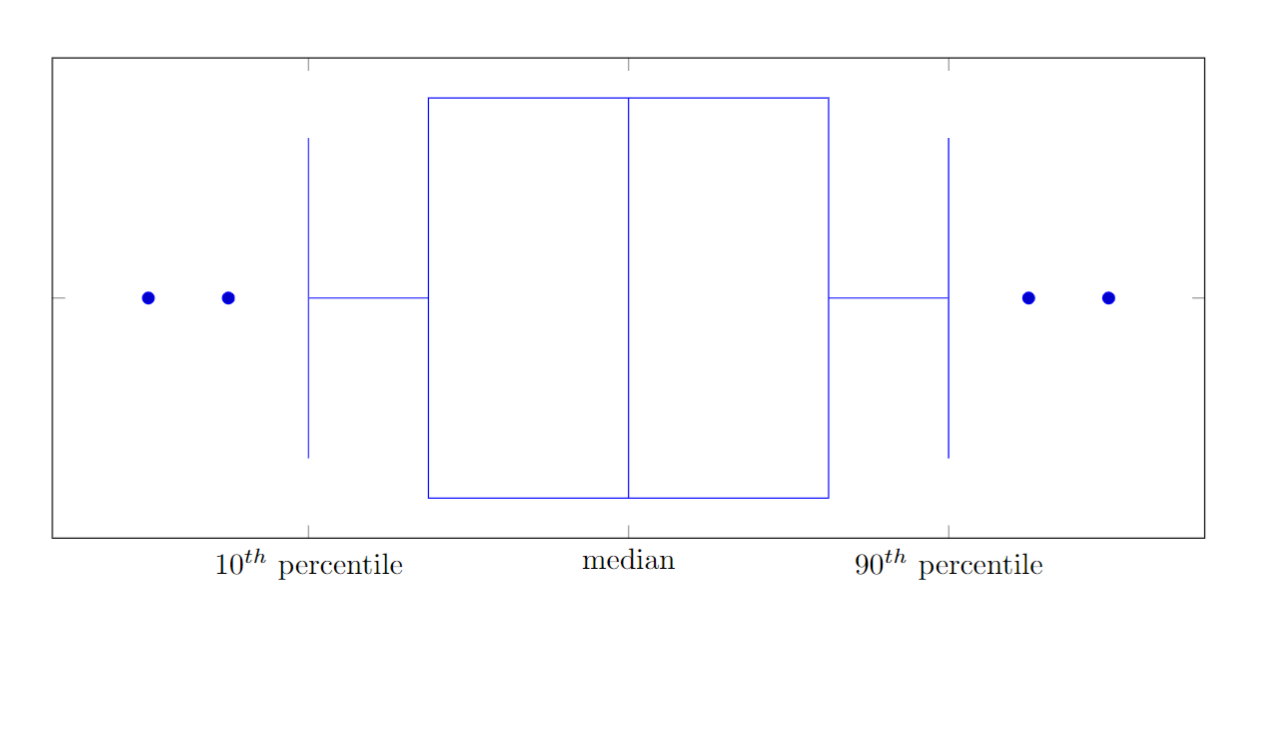


Figure 1 - Boxplot

As pictured above, all data points lower than the 10th percentile and higher than the 90th percentile are considered outliers. Since ratings are split into 3 categories, we will floor and ceiling the outliers instead of simply removing them. This ensures all reviews have the same number of categories. The function used is as follows:

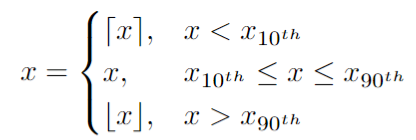


Figure 2 - Review Outlier Function

* 1. Comparing Recent Ratings

A shop's rating is fluid and changes over time as the shop itself changes. For our system to reflect such changes, such as a sudden drop in food standard, we subtract the overall rating over all time from the overall rating over a recent window of time, such as the past month. If the result is a positive number, it means that the shop has improved recently likewise for a negative number. the larger the magnitude the more drastic the increase or decrease.

### Final Design (with block diagrams) (For software, use the proper Software Engineering Diagram, such as Use case diagram, Sequence diagram, etc.)

Figure 3 - Use Case Diagram

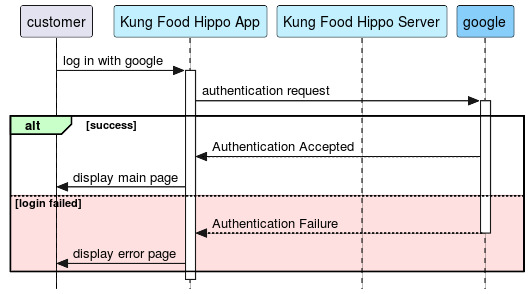


Figure 4 - Login Page Sequence Diagram

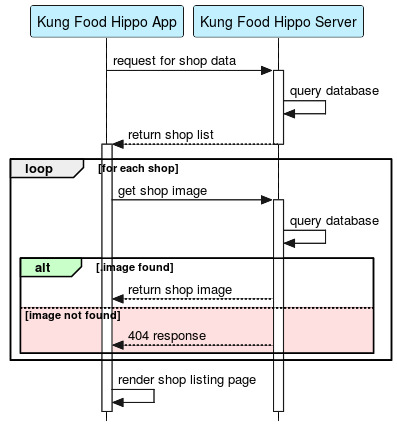


Figure 5 - List Shop Sequence Diagram

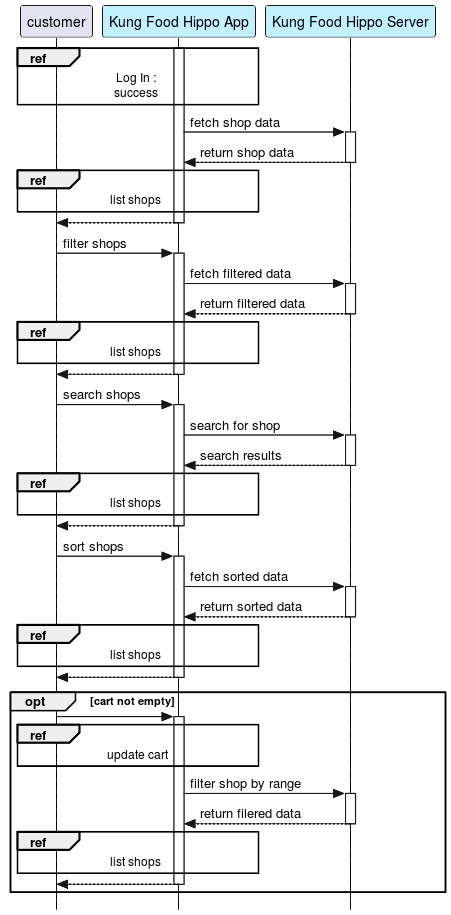


Figure 6 - View Shop Sequence Diagram

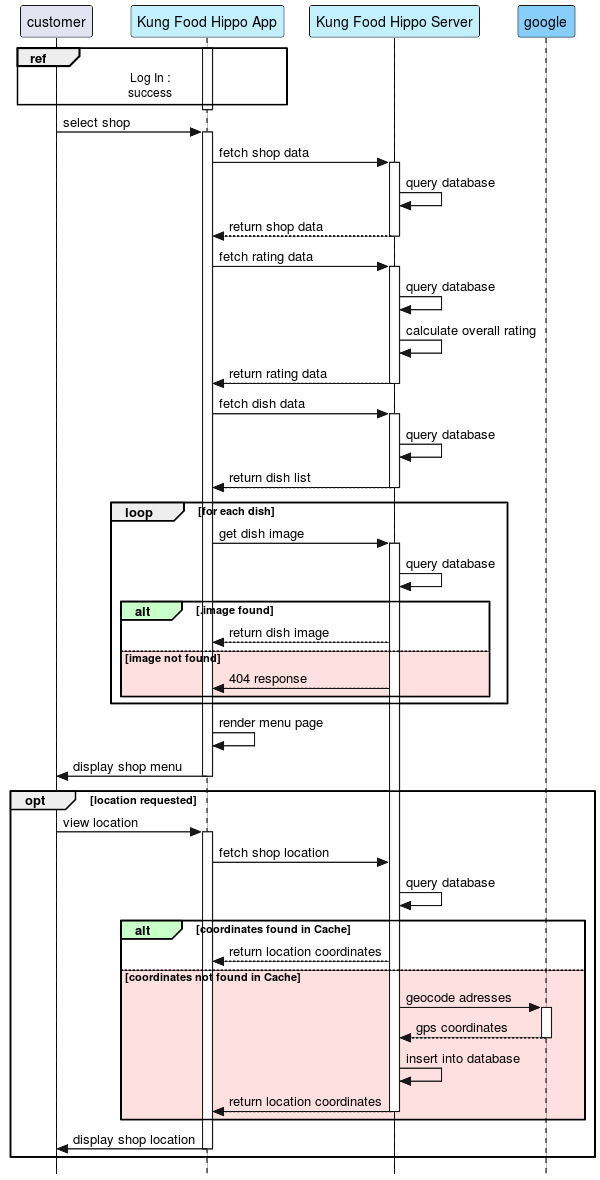


Figure 7 - Get Shop Sequence Diagram

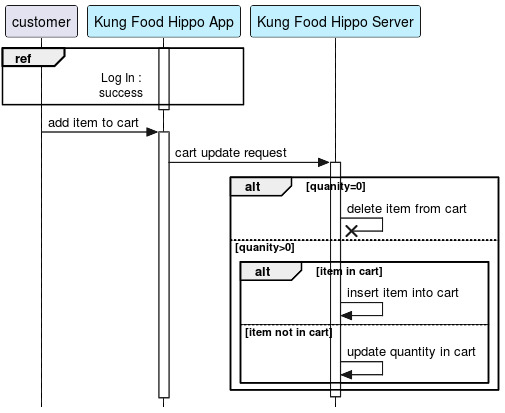


Figure 8 - Update Cart Sequence Diagram

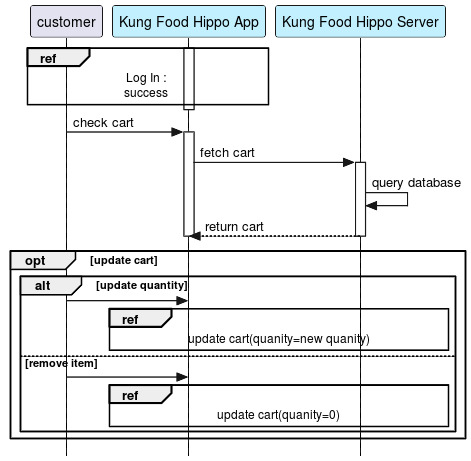


Figure 9 - View Cart Sequence Diagram

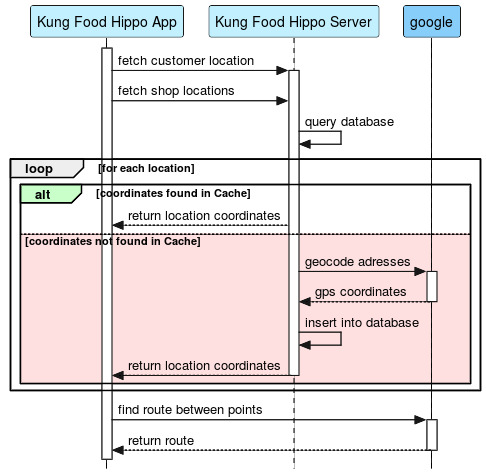


Figure 10 - Delivery Route Sequence Diagram

### Implementations

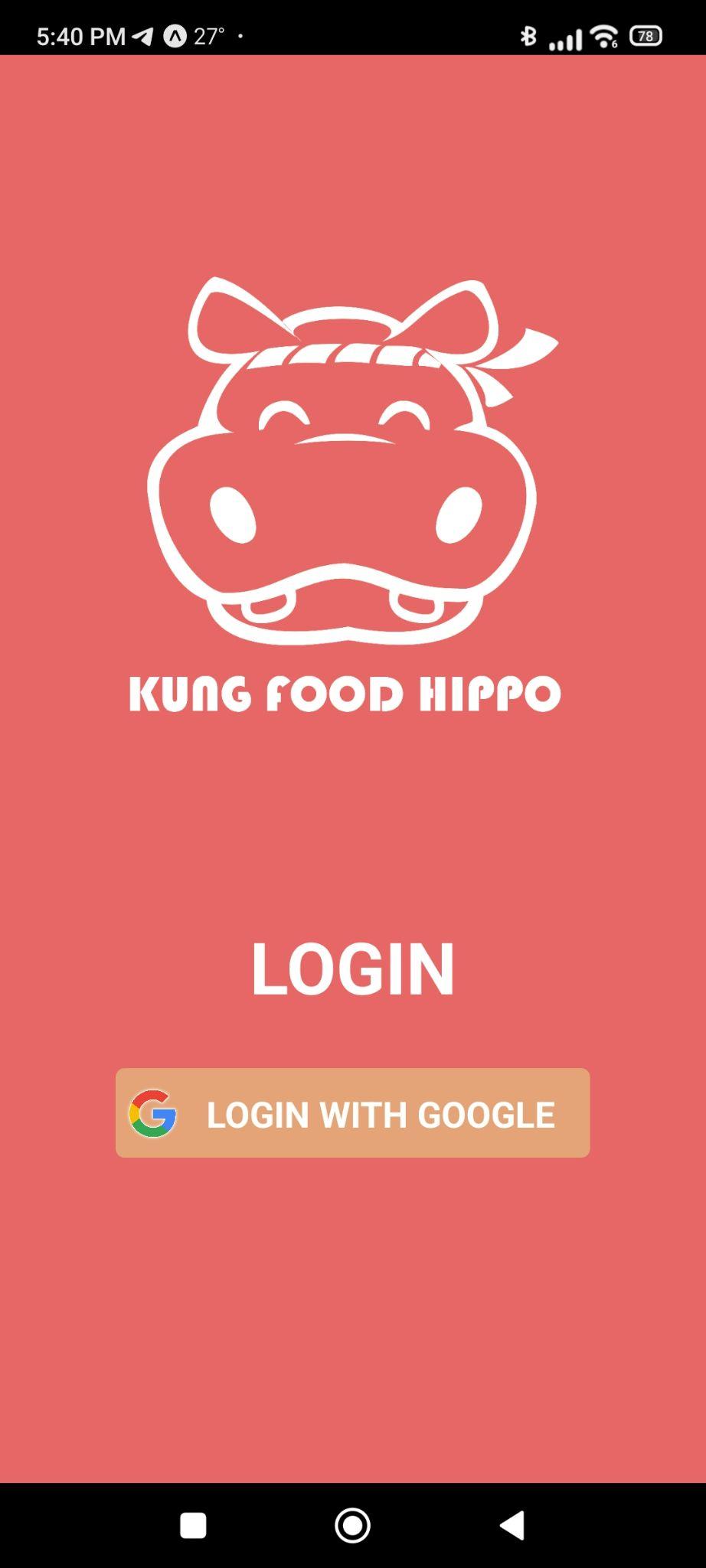
 

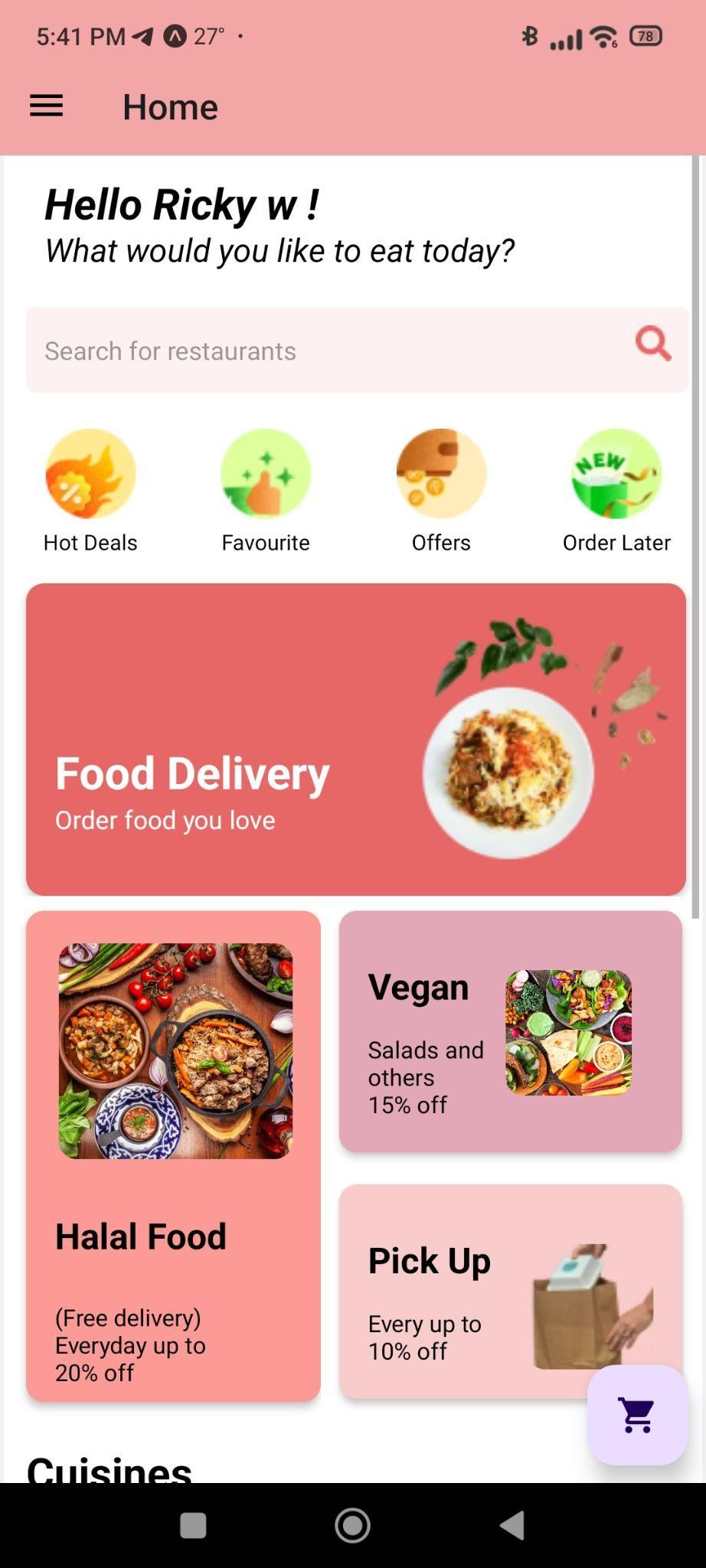
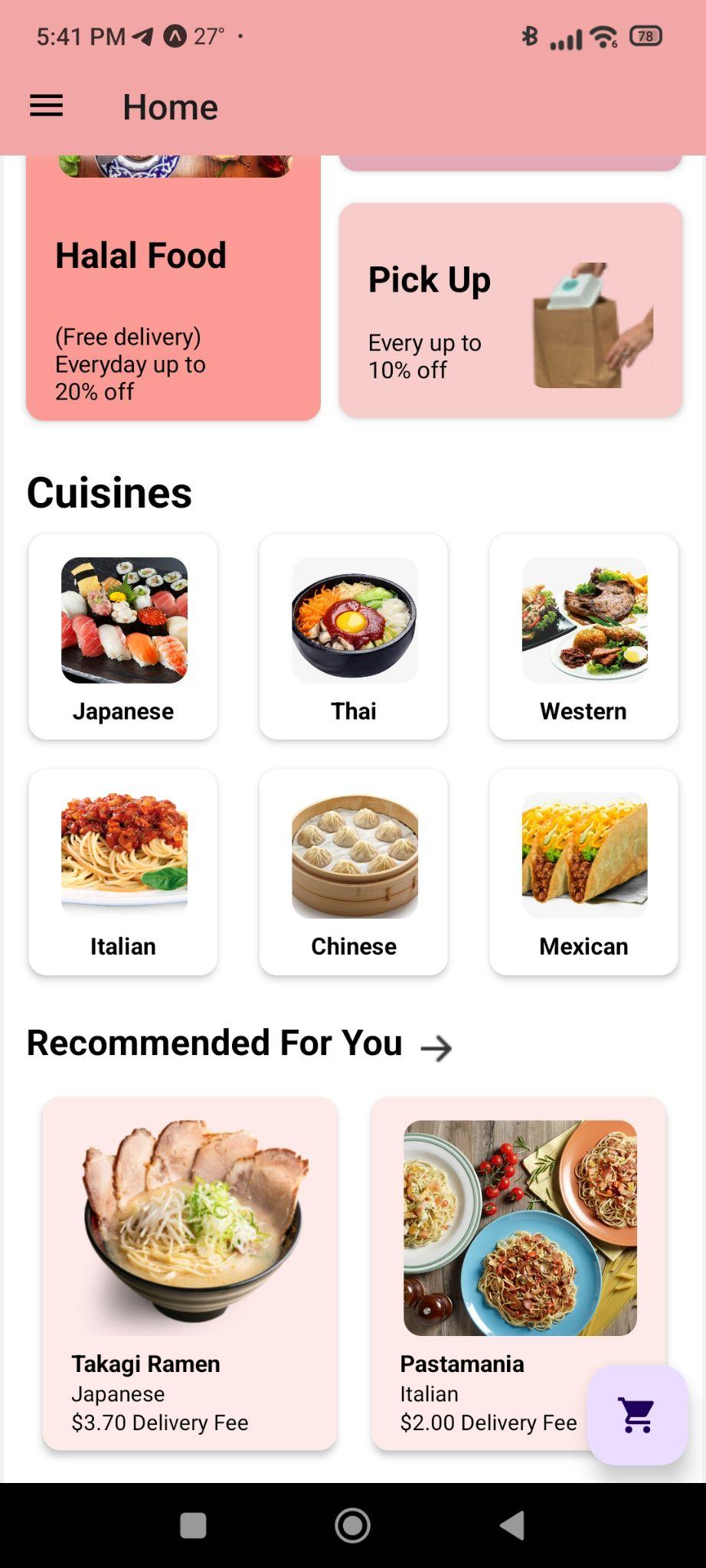
Figure 11 - Login Page Figure 12 - Animation for Login Page 

Figure 13 - Home Page Figure 14 - Home Page 2

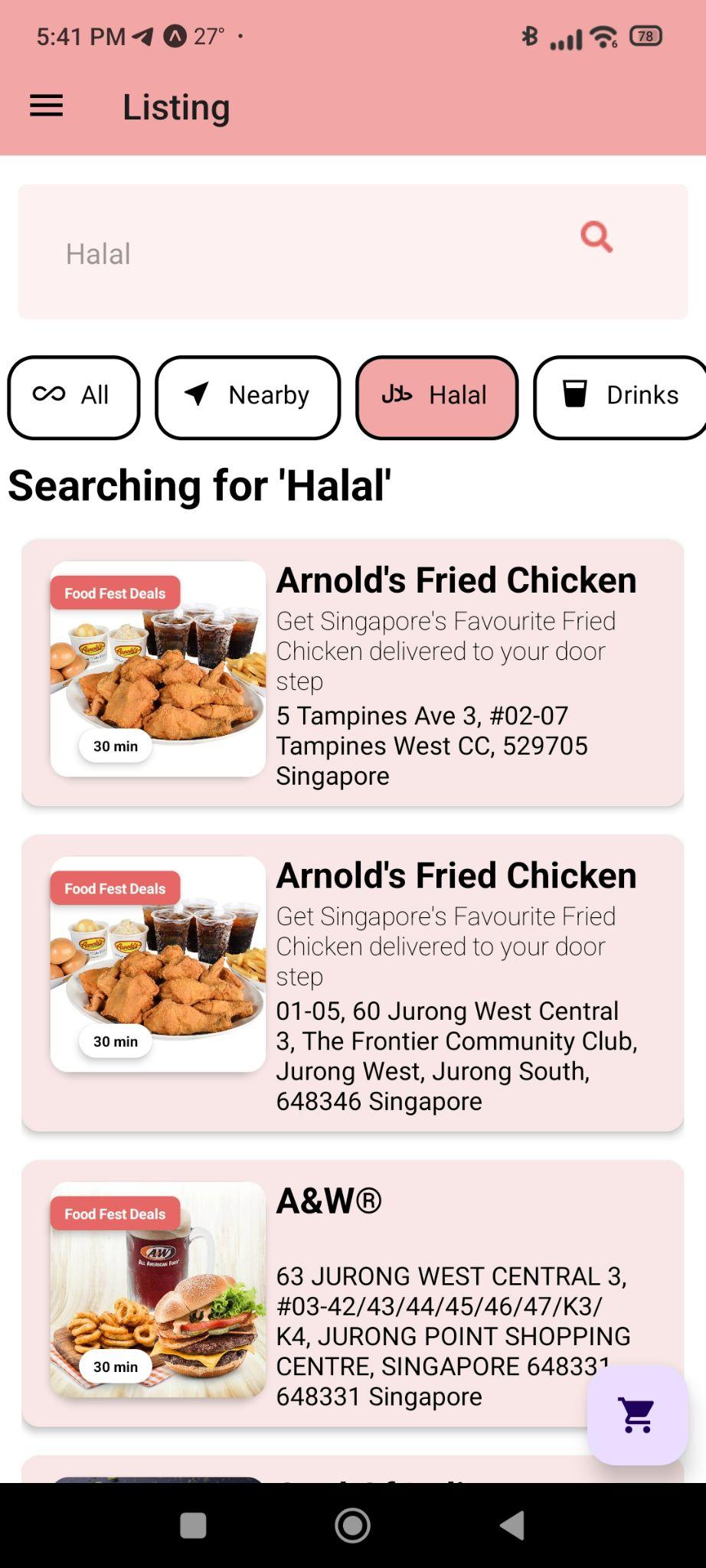
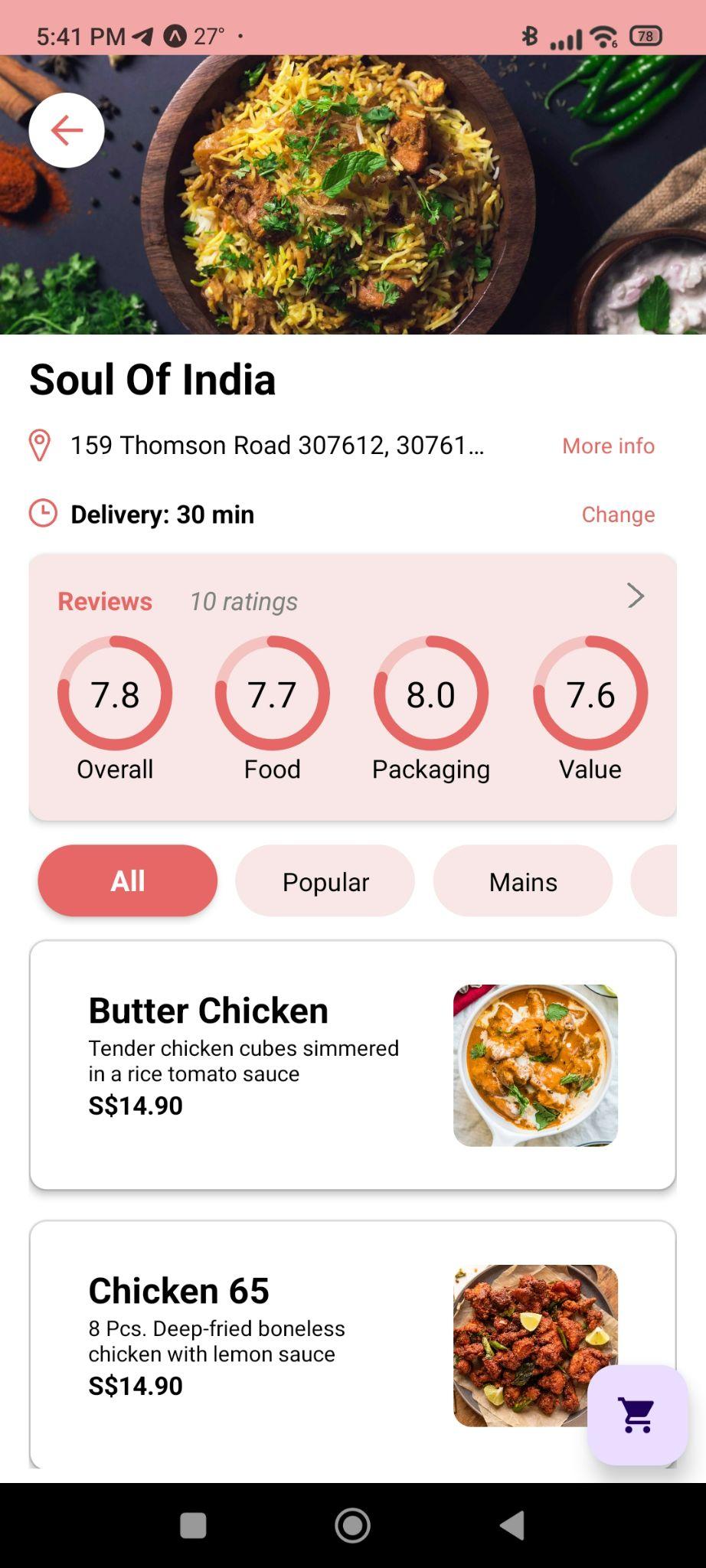
 

Figure 15 - Listing Page Figure 16 - Restaurant Page

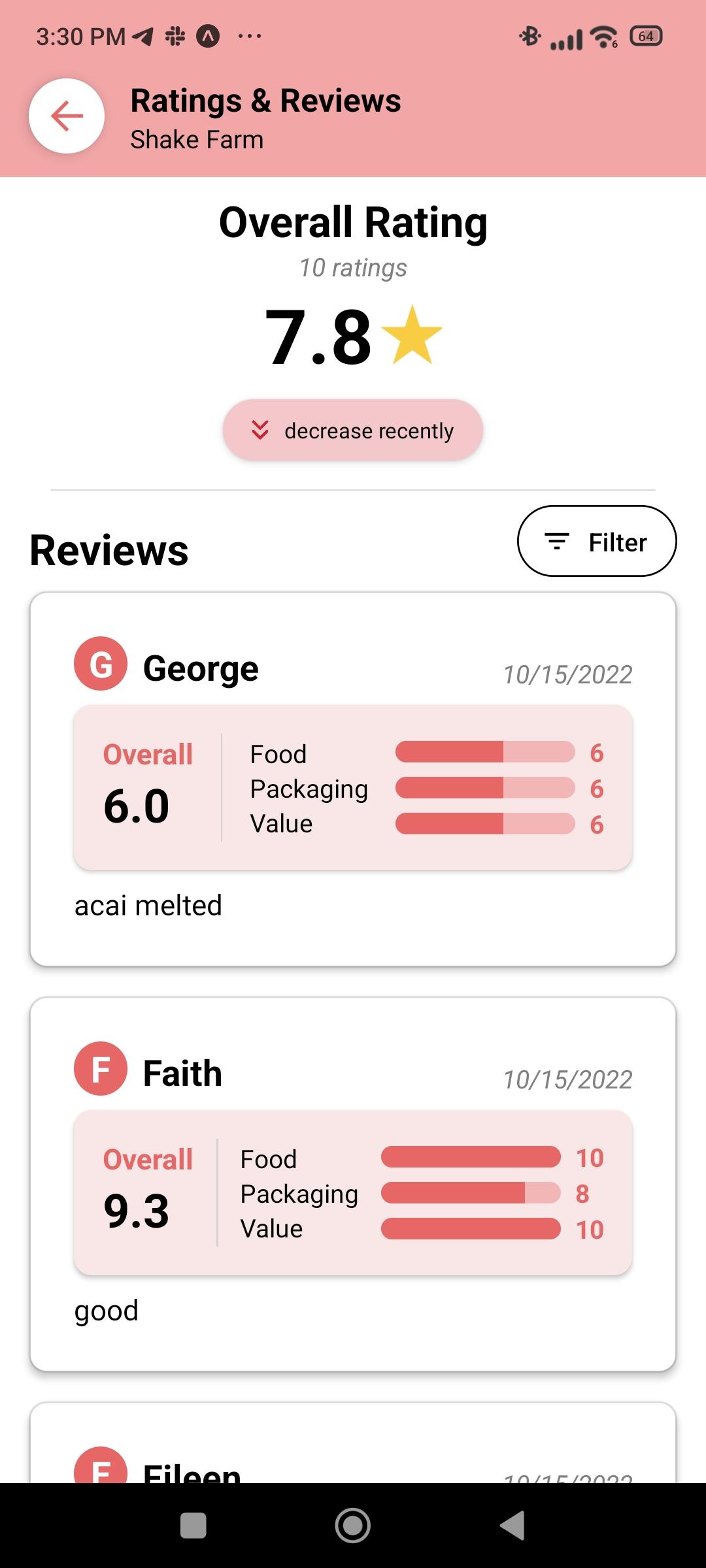
 

Figure 17 - Review Page Figure 18 - Map 1

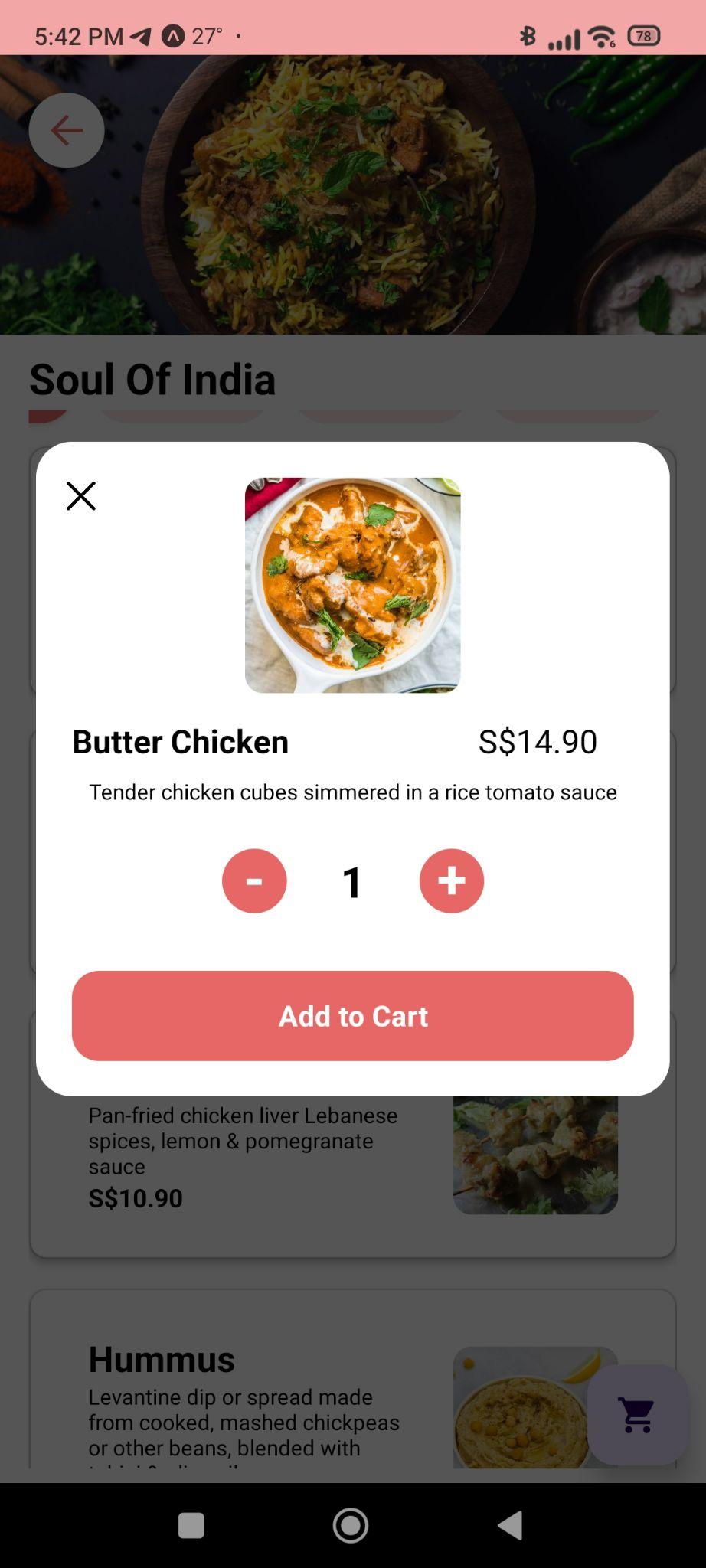
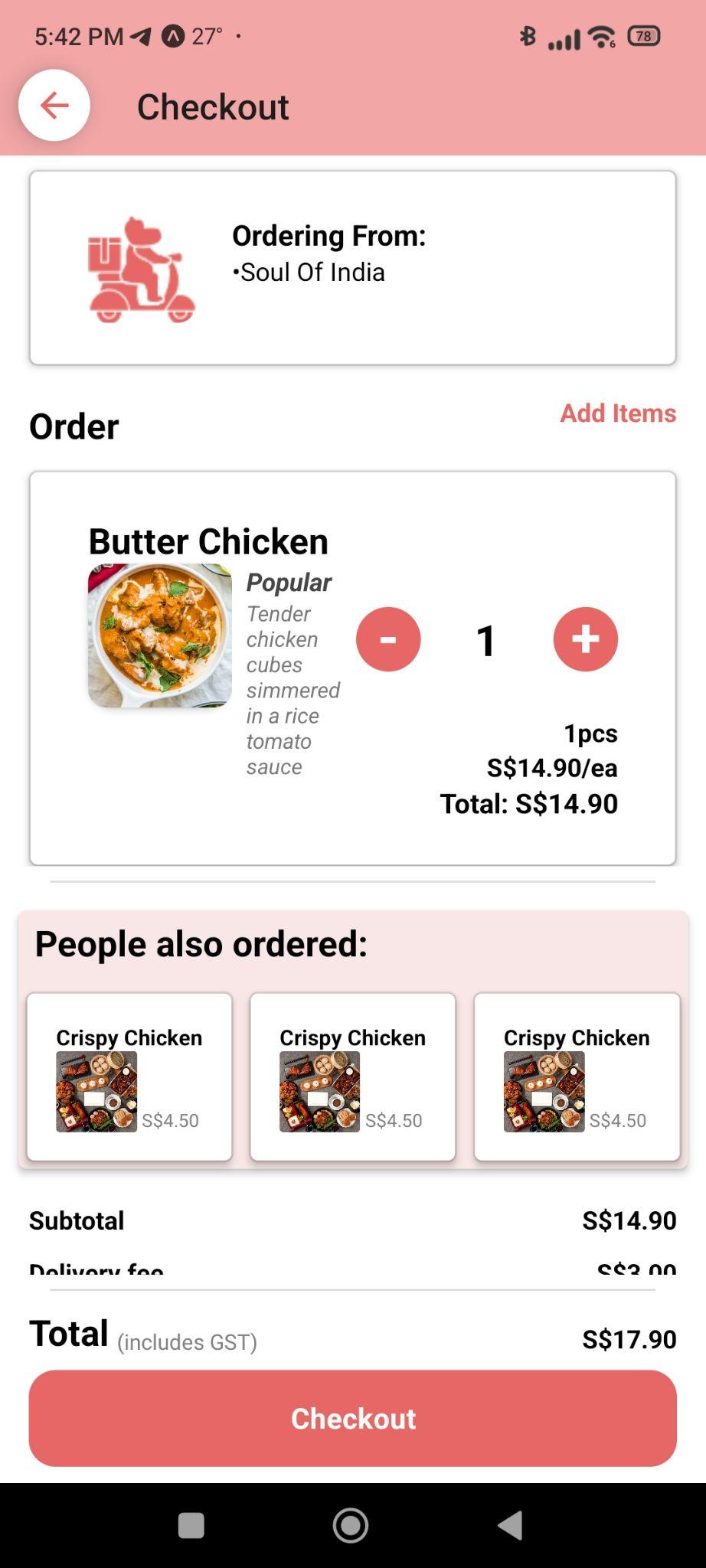
 

Figure 19 - Adding to Cart Figure 20 - Checkout Page

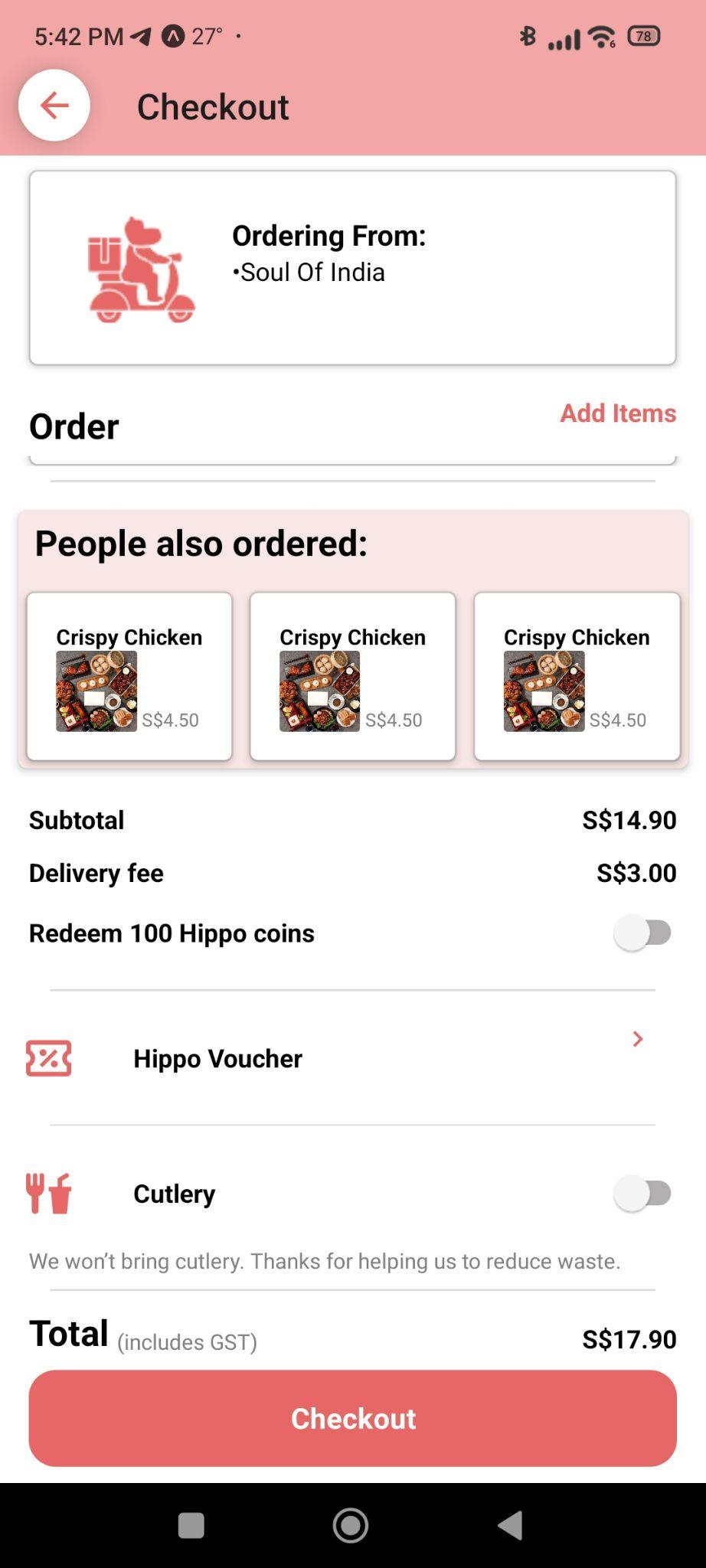
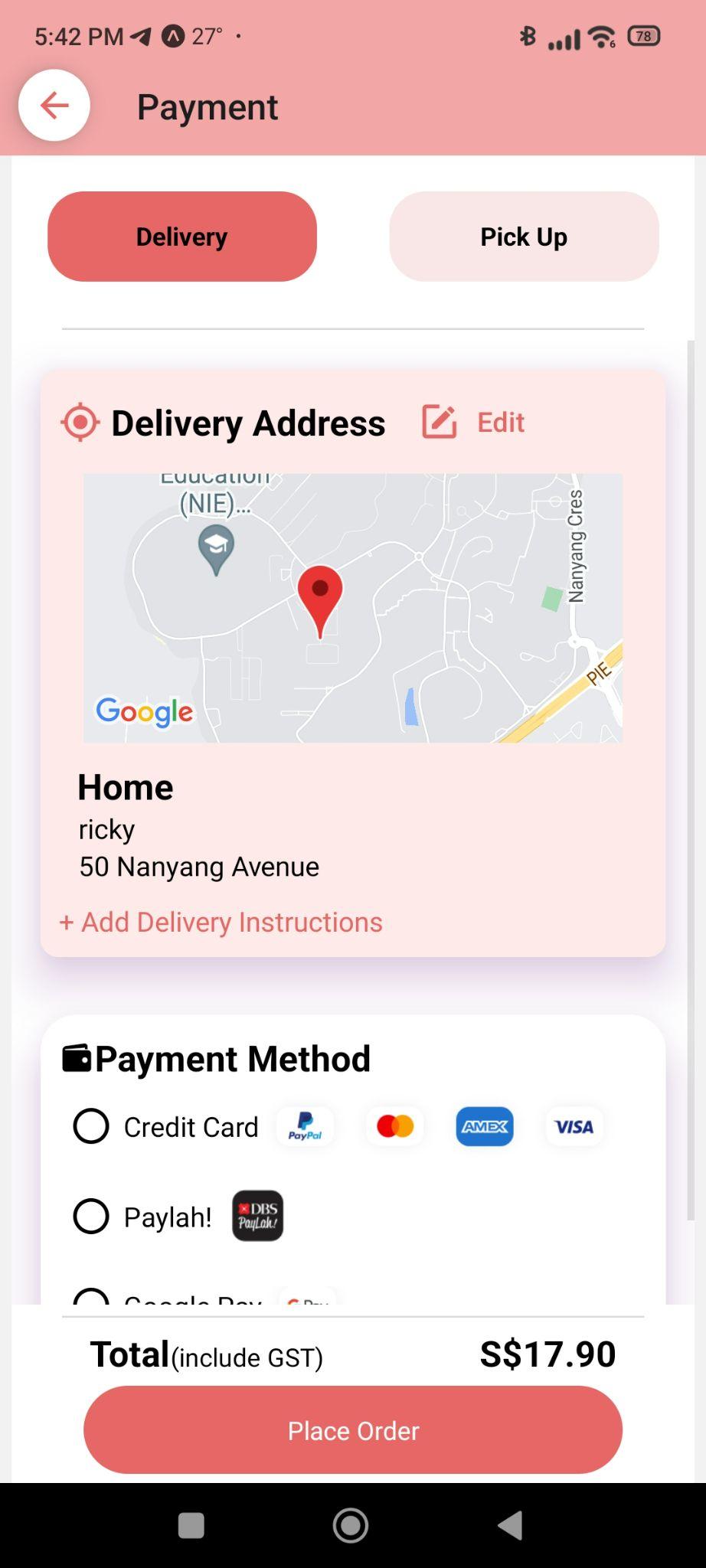
 

Figure 21 - Checkout Page 2 Figure 22 - Payment Page

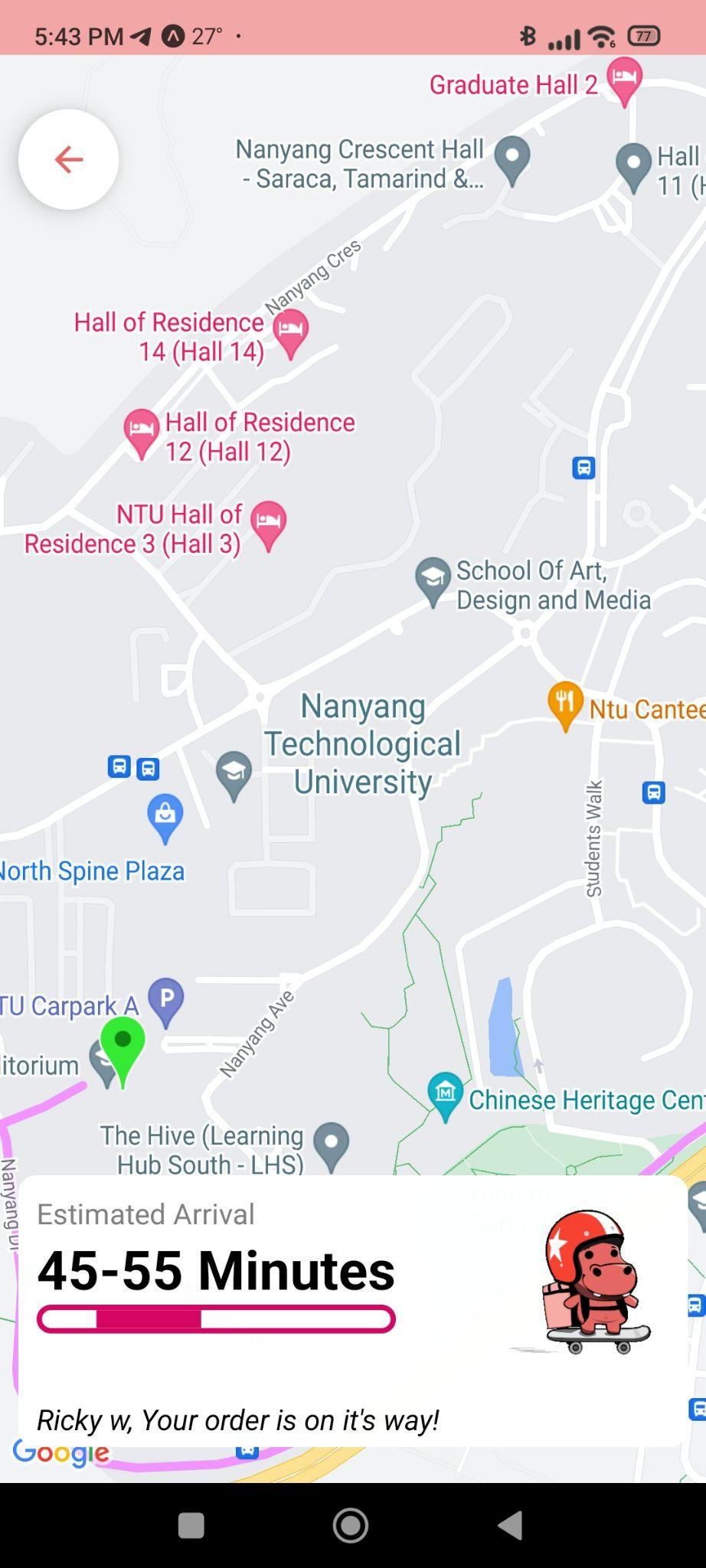
 

Figure 23 - Animation after Payment Figure 24 - Delivery Page

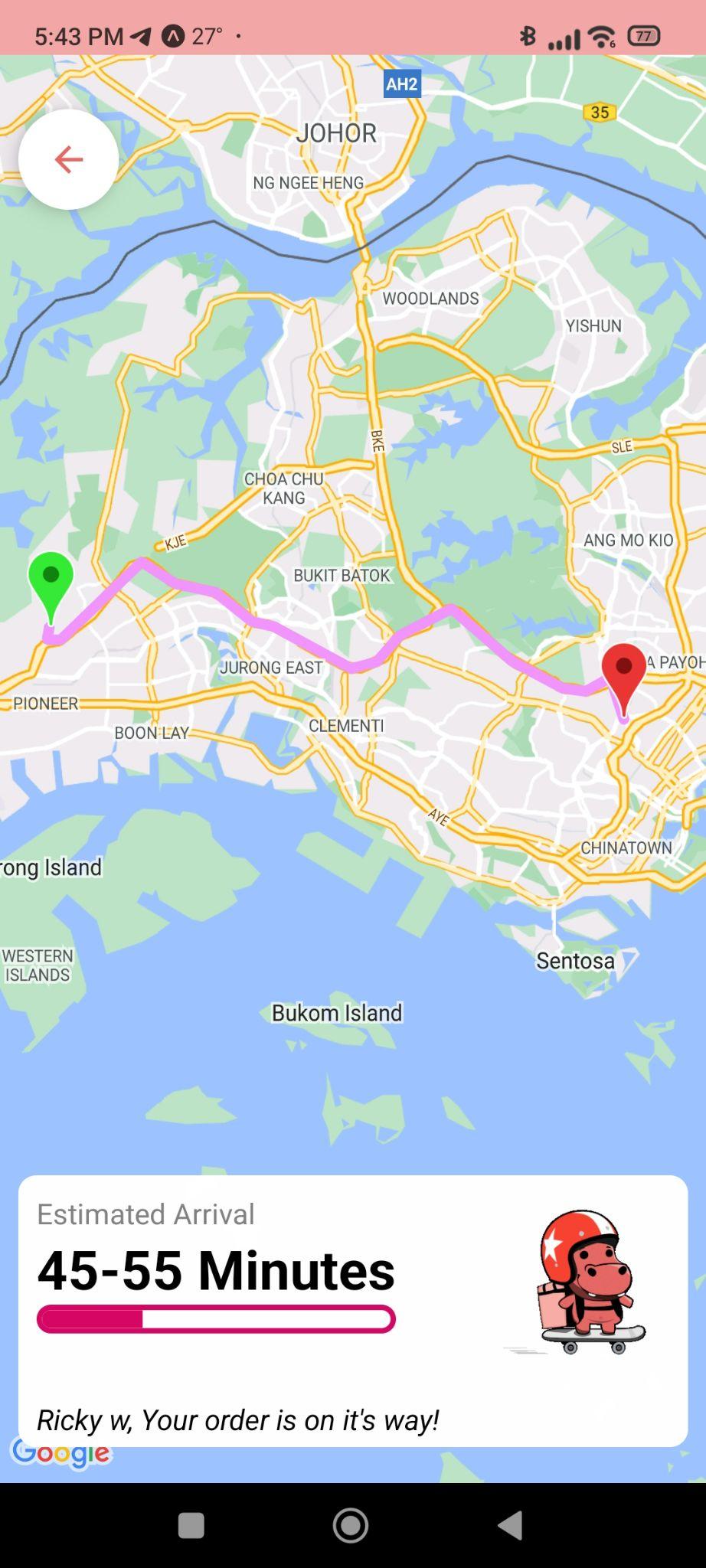
 

Figure 25 - Estimated Arrival Page 2 Figure 26 - Estimated Arrival Page 3

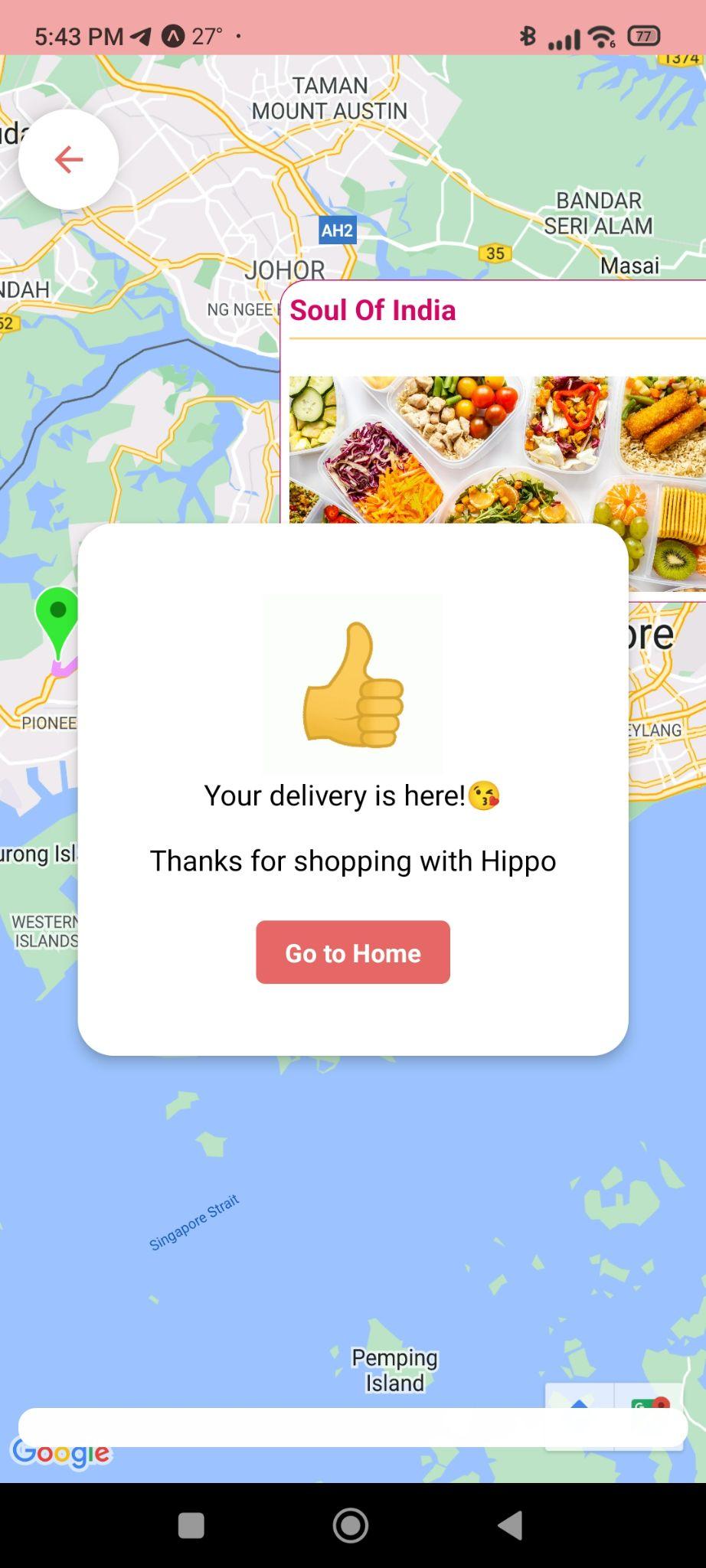
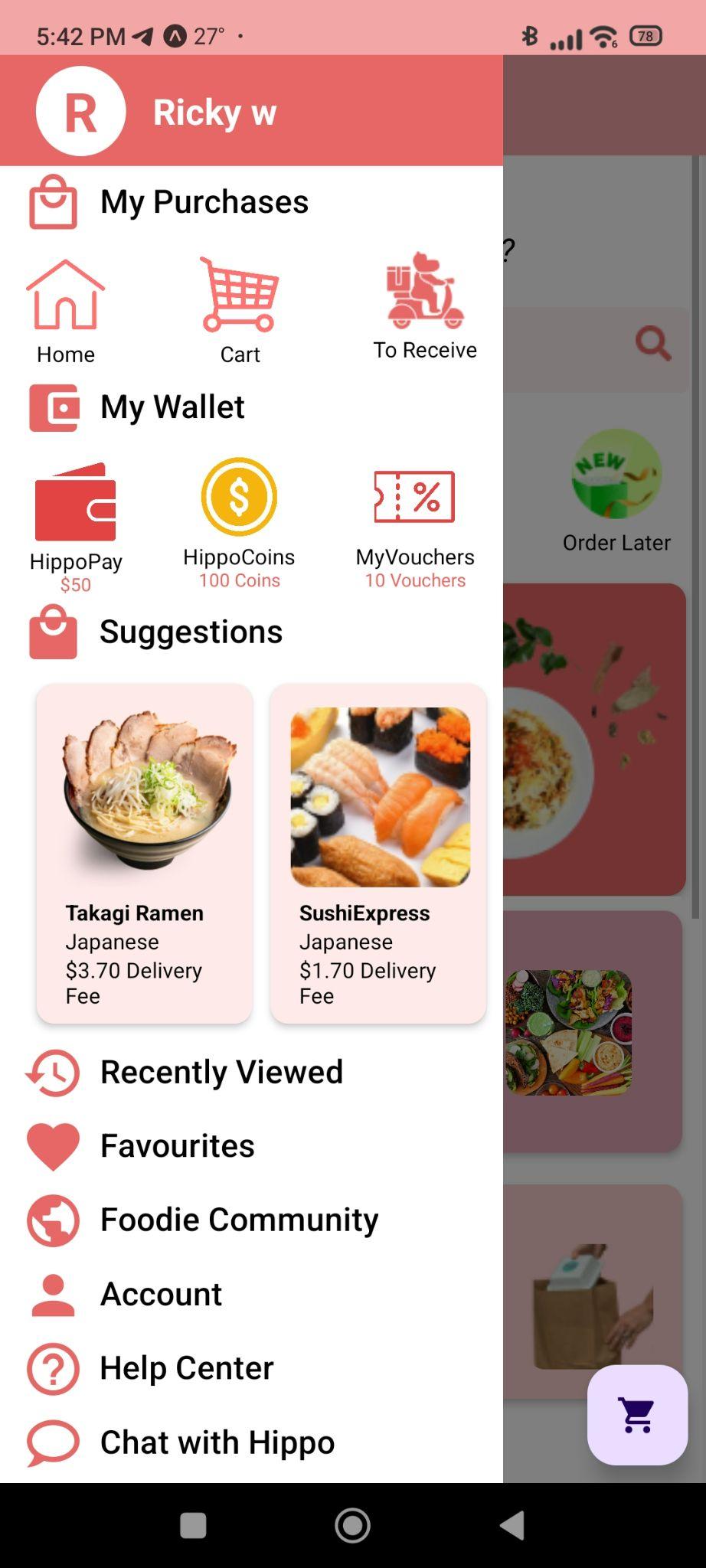
 

Figure 27 - Order Completed Page Figure 28 - Home Page 3

## Conclusion and Recommendation

### Conclusion

Our project has successfully satisfied its motivations and objectives. We have learned a new framework for mobile app development as well as gained insightful knowledge in the intricacies of software project development. But more so than that, we have learned how to realize and analyze pain points that customers of an application might face which not only includes the features of the app itself, but the user interface and the user experience. However, all this would not be achievable without building a robust back-end framework wherein we explored different ways to create an API and to host a database framework on the the web.

### Recommendation for Future Works

In the course of 13 weeks, we have achieved a lot in terms of features and innovations. However, there were many areas of improvements as well as features that we had hoped to implement if time would have sufficed. A feature we were unable to implement in time was the user recommendation engine.

For the scope of our project, it was not feasible in the limited timespan as for us to train the model for the user recommendation, we have to gather a large number of data points that is specific to our application by utilising user activity. Unfortunately, it is not realistic for us to get hundreds of users on board to our application. This perhaps can be an avenue of research for future works of this project.

Other features that would be great to implement is an in-app currency to keep track of rewards as well as to purchase vouchers. The reward system would offer vouchers to users based on user interaction/purchase and incentivises the user to continue using the app. Or a Foodie Community for food sharing and seller promotion that better engages users, and provides reliable reviews and recommendations by the people.

### 

### 

## Appendices

### Design Diagrams

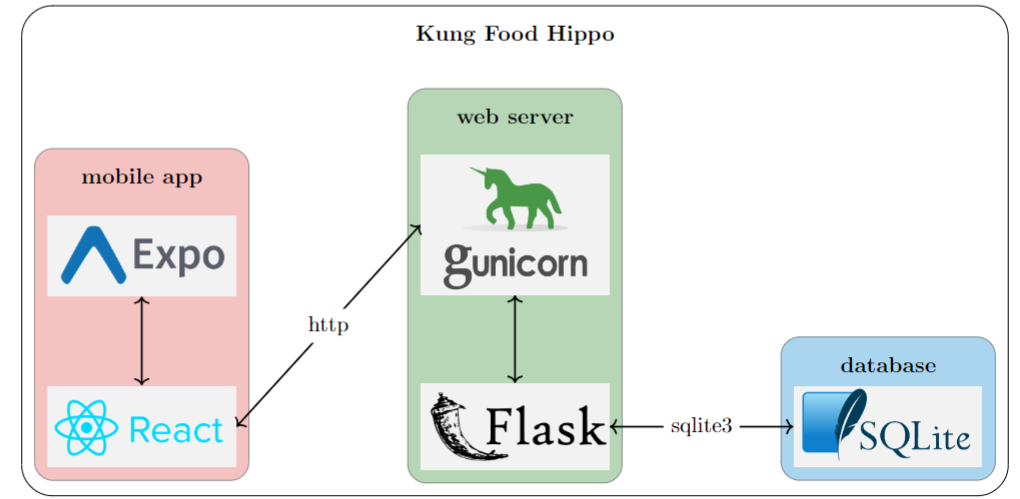


Figure 29 - Architecture diagram of Kung Food Hippo App

Software Engineering Diagrams can be viewed in this drive:

[Software Engineering Diagrams](https://drive.google.com/drive/folders/19fz853c69SqWFC-1r3ogDtxCjOkdigmo?usp=share_link)

### Guides

Web Server guide can be viewed here:

[Web server Guide](https://drive.google.com/file/d/13uKoRlm6GrEdI9sS21Ay-5H2MBwqFyUZ/view?usp=share_link)

Web api documentation can be found here:

[Documentation](https://drive.google.com/file/d/1kui0_NXyAbGykJCo2RadC8wR_SzJtf6f/view?usp=share_link)

### Source Code

All resources can be viewed on the repository: <https://github.com/SpiderPigzz/Design-and-Innovation-Project>

To request access, please email the project leader Ricky Winarko at [RICK0016@e.ntu.edu.sg](mailto:RICK0016@e.ntu.edu.sg) with your GitHub account name.

### Project Management Schedule and Task Allocation/Breakdown

Figure 30 - Project Gantt Chart

For more detailed information regarding the schedule, you can view it through [this](https://docs.google.com/spreadsheets/d/1MQUd-J3Qi_bcFBElXk0JxB_pIy6PrecgUGDyyFp20-Q/edit?usp=sharing) [link](https://docs.google.com/spreadsheets/d/1MQUd-J3Qi_bcFBElXk0JxB_pIy6PrecgUGDyyFp20-Q/edit?usp=sharing).

### Others

We are grateful for one another and for having such a wonderful team!!! :D

Figure 31 - Kung Food Hippo Family

## Bibliography

Bonthu, H. (2022) *Detecting and treating outliers: How to handle outliers*, *Analytics Vidhya*. Available at: https://www.analyticsvidhya.com/blog/2021/05/detecting-and-treating-outliers-treating-the-odd-one-out/ (Accessed: November 1, 2022).

Ellis, M. (2019) *Logo & branding - bring your brand to life | 99designs*, *99designs*. Vista. Available at: https://99designs.com.sg/blog/logo-branding/food-branding/ (Accessed: November 1, 2022).

Glick, K. (2019) *I/O 2019: New features to help you develop, release, and grow your business on Google Play*, *Android Developers Blog*. Available at: https://android-developers.googleblog.com/2019/05/whats-new-in-play.html (Accessed: November 1, 2022).

Grinsted, T., Lin, S. and Tat , Y.K. (2021) *Making ratings and reviews better for users and developers*, *Android Developers Blog*. Available at: https://android-developers.googleblog.com/2021/08/making-ratings-and-reviews-better-for.html (Accessed: November 1, 2022).

Valve (2016) *News - customer review system updated to show recent reviews and summary*, *Customer Review System Updated To Show Recent Reviews And Summary*. Valve. Available at: https://store.steampowered.com/oldnews/21695 (Accessed: November 1, 2022).

Wadbude, Rahul & Gupta, Vivek & Mekala, Dheeraj & Karnick, Harish. (2018). User bias removal in review score prediction. 175-179. 10.1145/3152494.3152520.

Hady Wirawan Lauw, Ee-Peng Lim, Ke Wang. Summarizing Review Scores of Unequal Reviewers. In Proceedings of the Seventh SIAM International Conference on Data Mining, April 26-28, 2007, Minneapolis, Minnesota, USA. pages 539-544, SIAM, 2007.

29 Incredible Stats That Prove the Power of Visual Marketing. https://movableink.com/blog/29-incredible-stats-that-prove-the-power-of-visual-marketing. Accessed 7 Nov. 2022.

Delgado, C. (2019) Everything you need to know about sqlite mobile database, Our Code World. Our Code World. Available at: https://ourcodeworld.com/articles/read/737/everything-you-need-to-know-about-sqlite-mobile-database (Accessed: November 7, 2022).