SOFTWARE REQUIREMENTS SPECIFICATIONS

For

MakanGoWhere

-ZAND^2

Table Of Content

1. Introduction

1.1 Purpose

This document aims to provide detailed documentation on the features, interfaces and requirements of MakanGoWhere, an android-based application that provides a one-stop solution for users to search and discover places to eat based on their various requirements, give ratings for food location and update the menu of these food locations. ZAND[2] aims to serve the following purposes with MakanGoWhere:

1. Allow users to effortlessly narrow down food choices based on their requirements and preferences. It also allows them to quickly browse through the menu of the place as well as other useful information such as dietary preferences and reviews.
2. Give shop owners a platform to post their shops and give their shop exposure.

1.2 Document Conventions

The format of the document is outlined as such:

- Section headings in Arial; font size 18
- Subsection headings in Arial; font size 14
- All headings to be bolded
- All screenshots of the web application are labelled in italics.

1.3 Data Dictionary

Terms	**Definitions**
User	The person currently using the app
MakanSpot	Places that sell food (Anything ranging from Restaurants, food stalls to Hawker Centre Stalls)
Home Page	Default page of the app which shows the list of food places
Current Location	The current location of the user
MakanSpots Nearby	The recommended food places near the user's current location on the homepage after scanning an area on the map interface.

Filtered MakanSpots	The list of recommended MakanSpot according to the price range.
Reviews	An online evaluation and comment of performance of the restaurant.
View Reviews	A feature that allows us to see the number of reviews and the average rating of a MakanSpot.
Profile Page	A page for entry of one's personal information.
Settings Page	Settings page which allows users to change privacy settings and contact customer support
MakanDistance	Approximate distance between user and restaurant
BringMeThere	The path from the user's Current Location to the restaurant.
EditMakanSpot	A feature that allows customer to edit a restaurant's information
AddMakanSpot	A feature that allows customer to add a restaurant
EditMakanSpotMenuItems	A feature that allows customer to edit a restaurant's menu items' information
AddMakanSpotMenuItems	A feature that allows customer to add a restaurant's menu items' information
View Menu	A feature that allow users to view the list of food/dishes of a MakanSpot
MakanPoints	A reward system that encourages users to give accurate inputs and verify recommendations that are given.
Redeem Rewards	A feature that allow the user to redeem 3 types of benefits: 1.) Grab food voucher $10 2.) Grab food voucher $30 3.) ActiveSG 3 month membership based on the amount of MakanPoints the users have
Start Chat	A feature that allow you to communicate with other users using chat function

1.4 Product Scope

1.4.1 Project Mission Statement

Living in Singapore, we love our food and we are constantly spoilt for choices. Food is all around us and in a huge variety. More often than not, we are overloaded with information and suggestions of what to eat for a certain meal since there are just too many choices all around us. This results in difficulty in deciding what to eat. Moreover, people with dietary preferences face the headache of filtering what is available nearby that meets their requirements.

Our app, MakanGoWhere, aims to provide a one-stop solution to all our food needs. We believe that by first filtering based on price range, the huge number of choices can be lowered down and users can then decide on what to eat by looking at the other requirements such as the menu and the dietary preference of the store.

MakanGoWhere aims to go beyond just providing suggestions for food, but also give users a platform to meet other foodies and bond over food with the chat functionality, which allows users to chat with one another whom they have met on the app.

Users will be able to earn rewards for updating the platform with food options, menu information and reviews. This is to incentivise users to continuously expand the database.

All Singaporeans stand to benefit from MakanGoWhere. ZAND^2 aims to provide users a more comprehensive and complete discovery of Singapore food with MakanGoWhere.

1.4.2 Target Audience

Foodies looking to share obscure food locations and connect with other foodies in Singapore and store owners who want a platform to give their stores more exposure.

This document is intended for developers, project managers, testers and users of the application. We have included a table for specific users and respective sections that will be more useful below:

Developers	Uses Case, pages 38 to 96
Users	System Features, pages 26 to 34
Testers	Testing, pages 101 - 117

2. Overall Description

2.1 Product Perspective

The newly developed MakanGoWhere app makes use of google maps' Places API to source for places to eat for the user. The user can apply a price filter to narrow their search. The app also displays specific information on a MakanSpot to allow the user to decide easier.

MakanGoWhere will also allow users to review places and chat with one another, ultimately allowing the user to discover new places to eat.

2.2 Product Features

- **Create Account** : A new user will have to sign up for an account and indicate their dietary preferences/requirements.
- **Map** : The user can use the circle shown on the map to scan the map for MakanSpots. The MakanSpots that fall within the area of the circle will be scanned.
- **Filter** : The user can narrow down what is displayed on the map by inputting the price range.
- **MakanSpot Info** : The user can view specific information regarding a MakanSpot such as opening hours, dietary preferences and reviews.
- **Bring Me There** : The app can launch Google Maps which will show the user how directions to the makanSpot from the user's current location
- **Profile Page** : The user can view their profile details, view their points and change password here.
- **View MakanSpot menu** : Users can view the menu of the makanSpot here. Users can also add a menu item or edit a particular menu item's details here.
- **Reviews** : Users can view the reviews of the MakanSpot here.
- **Chat Function** : Users can start a chat with another user who have left a review.
- **AddMakanSpot** : The user can add new places to eat to the database.
- **EditMakanSpot** : Should a user see an inaccuracy in the MakanSpot information, or that there are changes to it, the user can then edit it.
- **Redeem Item** : The user can exchange points earned from editing, adding and reviewing MakanSpots and menu items for gift cards.

2.3 User Classes and Characteristics

For our application, we have outlined several types of users (as shown in the table below). For implementation, we only used one user class. This has been done as all functionalities and requirements for these two users are mostly similar. Splitting this class into various subclasses will only introduce unnecessary complexity.

Type Of User	Characteristics	Required Functionalities
Customer	- Predominantly looking to buy food	- Identifying MakanSpots - Chat function to communicate with other users - Reading reviews to find out what others think about the place - Price range function to see the approximate spending
Hawkers	- Predominantly looking to sell food	- Advertising their MakanSpots - Showcasing their menu - See what other stalls around them are doing
Food Reviewers (or similar)	- Predominantly looking for less popular food places (not listed on google maps) to review	- Identifying MakanSpots there are not listed on Google Maps

2.4 Dataset

MakanGoWhere makes use of Google's Map API and Places API for the location of the user and the location of the MakanSpot. We also make use of these data to create the Bring Me There function.

2.5 Operating Environment

MakanGoWhere is an android app with a minimum sdk of 16. Internet connection is required for users to log in and access the database of makanSpots, hence all functionality. Location services are needed for users using the map functionality in BringMeThere and scanning a location for makanSpots.

2.6 Design and Implementation Constraints

- Timing Requirements: 11 weeks
- Incorporation of publicly available government data from https://developers.google.com
- All the features in MakanWhere require member-only access. A self-imposed constraint we have is to have registered users with verified email addresses/mobile phone numbers only.
- The application requires Google Firebase.
- The app is written on Android Studio in Flutter (dart), and hence targets Android users.

2.7 Assumptions and Dependencies

The following assumptions were made in the development of MakanWhere:

- MakanWhere will be built for Android and thus must be compatible with the latest Android software.
- MakanGoWhere relies on the location information of the user to present the nearest MakanSpots on the map display and allow the Bring Me There function to work

MakanGoWhere has the following dependencies:

- Dart
- Java
- FireBase
- Places API

3. External Interface Requirements

3.1 User Interfaces

The user interface for the software will be compatible with any smart phone running on the android operating system.

MakanGoWhere adopts a simple and interactive layout that includes a navigation bar that is present on every page. On top of using the back key on android devices, users can jump straight to a page using the navigation bar and navigate the app with ease.

The interface has been designed such that it is possible for normal functionality without the use of the back key on their devices as there is a back button on pages that are not found on the navigation which users can use to navigate back.

When users enters a price range that no makanSpot fall under, or if they have entered a radius on the map where no MakanSpots is found, they will simply be greeted with a blank screen on the home page. This signifies that no makanSpot matches either (or both) of these criterias.

3.1.1 Bottom Navigation Bar

The bottom navigation bar (figure 1) will be present at the bottom of all pages of the app. It will provide convenient access to frequently used pages:

- Home Page: List of MakanSpots which is found within selected area and matches the price range selected
- Add Makan Spot: Add a new MakanSpot.
- Maps: shows makanSpots closest to the user
- Chat: Access to the chat page which
- Profile Page: Shows the details of the user and other settings

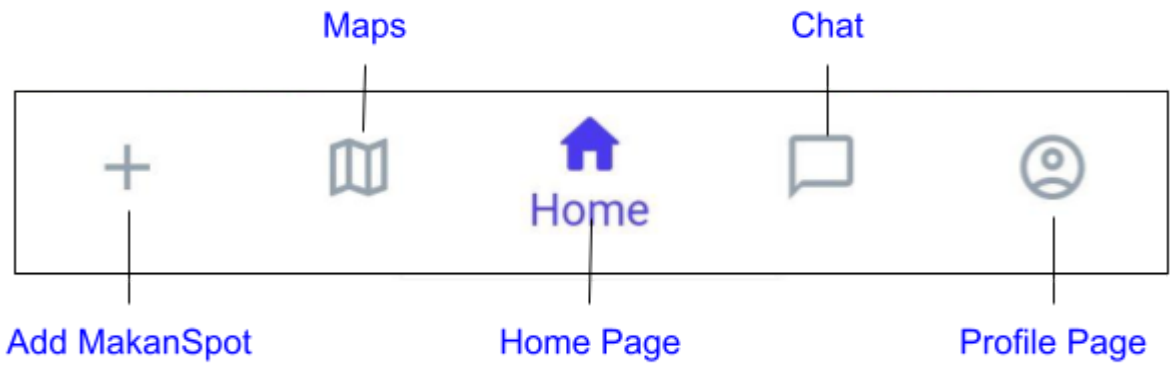


Figure 1: The bottom navigator bar

3.1.2 Home Page

At the top of the home page, there is a filter, which allows users to filter the MakanSpots based on price range. If no filters are selected, then, all MakanSpots which falls in the map area will be shown.

There are 3 options. 1 $ means price less than $10, 2 $ means prices between $10 and $50, and 3 $ means price more than $50.

MakanSpots which fall into the chosen price range will be displayed. Within each individual MakanSpot cell, the MakanSpot's opening hours and price range will be displayed.

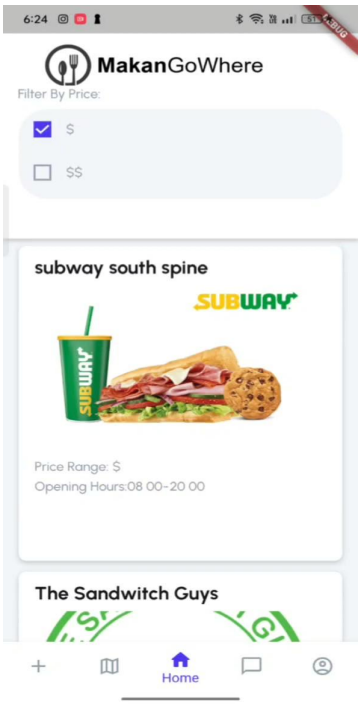


Figure 2: Feed where users can access newMakans and places to verify

3.1.3 Map

The map will display the user's current location. The scan function will display the radius of an area, allowing users to find food places within that proximity. It will then bring the users to the Home Page which will display a list of MakanSpots within that area after the user confirms using the blue button.

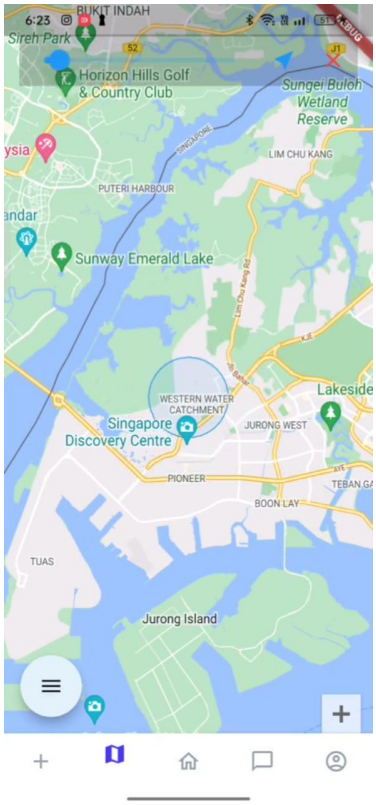


Figure 3: Map, the homepage of the app

3.1.4 MakanSpot Information

The makanSpot information page will display the detailed information of the makanSpot such as menu, opening hours, ratings, dietary preferences and reviews.

The reviews and menu items can be seen in individual cells of the page where users can swipe left to see more. Added reviews and items are automatically added to these cells.

Dietary preferences are shown as icons. Should an icon appear, it will represent that that particular MakanSpot supports that dietary preference.

Should a user see anything that is inaccurate, he can toggle the edit button which will lead to the makanSpot editing page. MakanSpot and allow users to favourite a particular MakanSpot.

The user is also able to toggle the “Bring me There” function which will take the user from the current location to this MakanSpot.

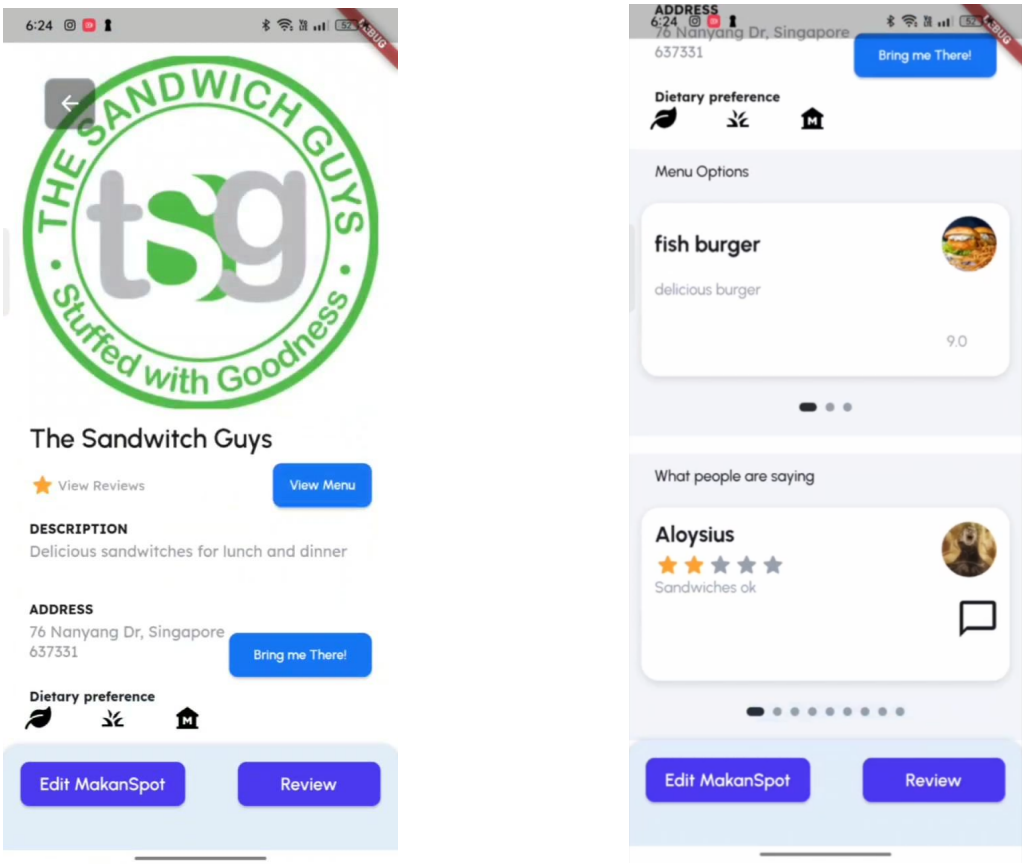


Figure 4 and 5: MakanSpot Details. The figure on the right shows how menu items and reviews on this MakanSpot is displayed. Swiping left will allow the user to view more.

3.1.5 Reviews Page

The reviews page will display a list of reviews of a particular MakanSpot. The number of ratings and average rating is also displayed on the reviews page to provide better insights for the users.

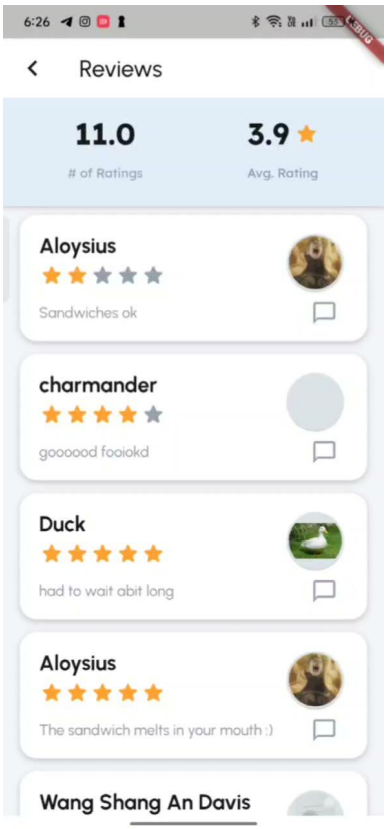


Figure 6: Reviews Page

3.1.6 Login/Sign Up Page

Users will be required to register an account before using the functions of the application. Users will need to upload their profile photo and key in their personal details like name,email as well their phone number. After logging in, the user will then be able to add MakanSpots to the database and earn points. Also, the user will be directed to the Home Page, which is the feed page.

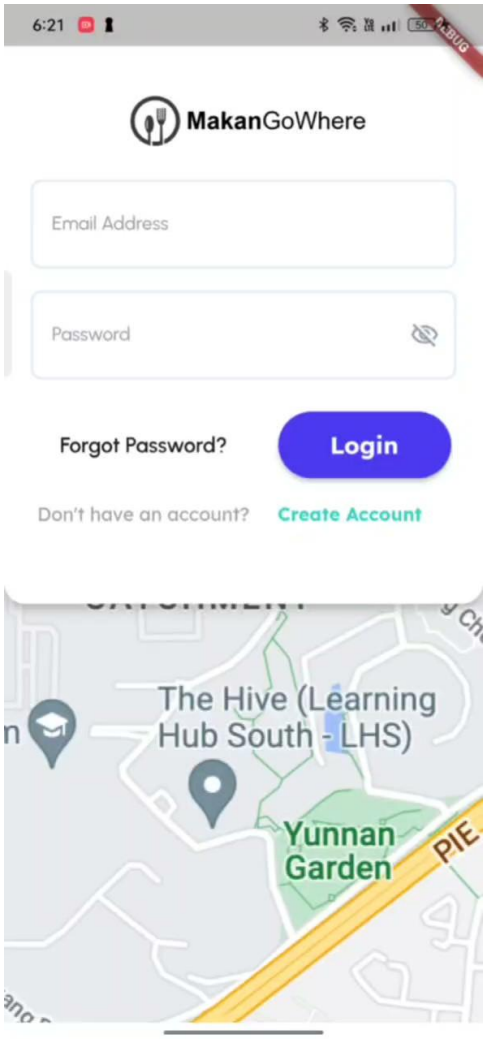


Figure 7: Login Page

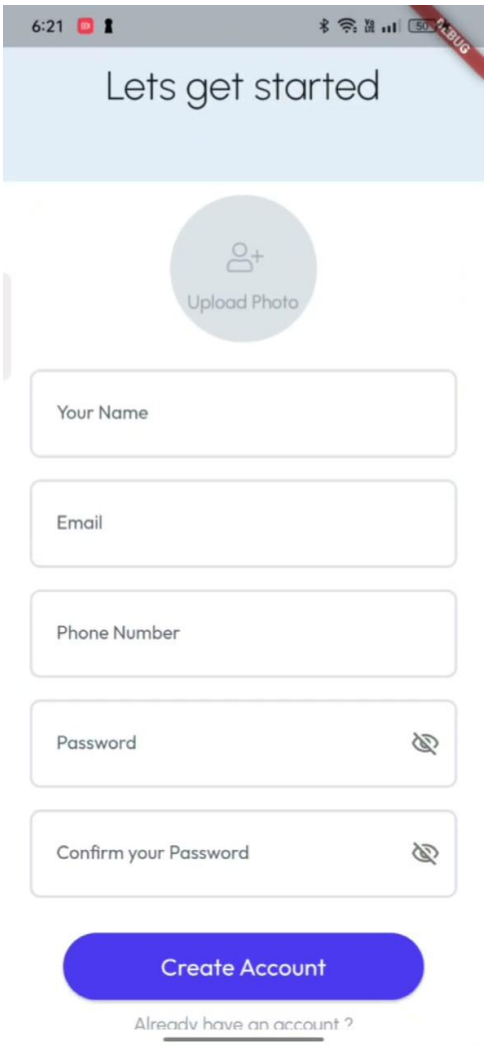


Figure 8: Sign Up Page

3.1.7 Add MakanSpot

The AddMakan Spot page will allow users to add a new MakanSpot that is not yet in the system. The main aim of this is to add food places that might not be already in the google maps database. Our team realised that this happens more than expected as food in hawker centres and coffee shops are not usually found on google and yet highly popular.

This page will request information about the MakanSpot from the users:

- Name of MakanSpot
- Photo of MakanSpot/Menu
- Address
- Postal Code
- Opening Hours
- Dietary Preferences

The postal code has to be input separately so that the system can store it in a separate database of Postal Codes. This allows the system to be able to automatically input this location as the destination when the user clicks Bring Me There.

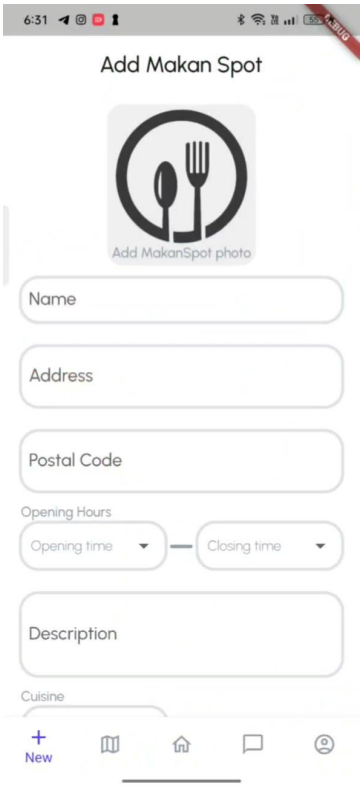


Figure 9: Add makanSpot page where users can add new MakanSpots

3.1.8 Menu

Description: The menu displays a list of menu items with its description and price of each item for that MakanSpot. Clicking into a particular item will allow users to edit that menu item.

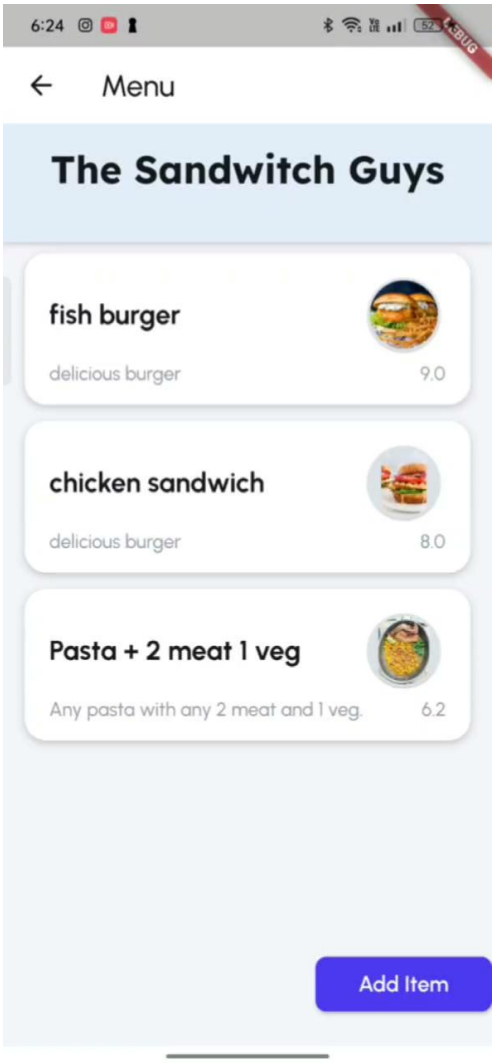


Figure 10: Menu page of the MakanSpot information

3.1.9 Add Menu Item

Users can add menu items to the MakanSpots if the Menu has not been updated. Users would need to upload an image of that Item with the name, description, as well as the price of the menu item. They will be awarded points for this action.

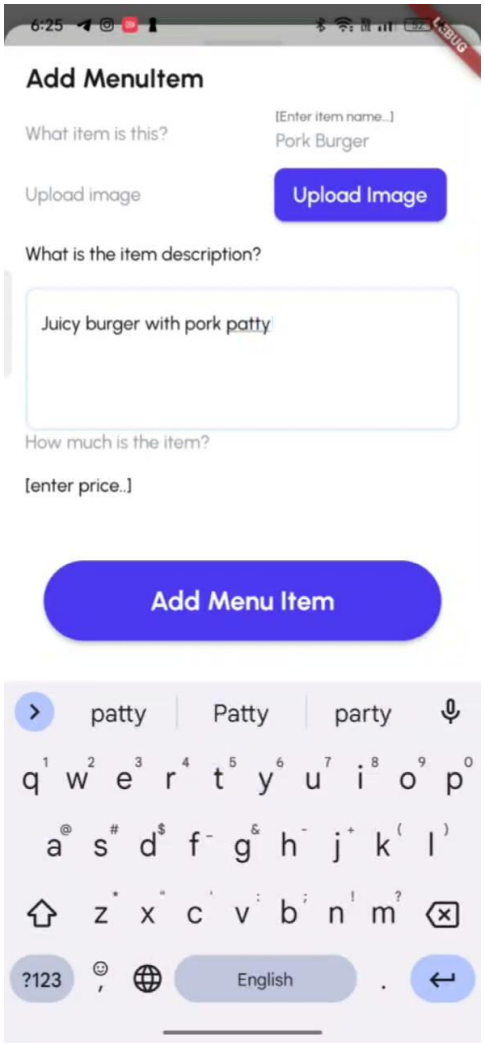


Figure 11: Add menu item

3.1.10 Edit Menu Item

Should a user realise that the details of a particular menu item is wrong, he can edit it by clicking into the particular menu item. He can then choose to edit whatever fields that are incorrect. He will earn points for this action.

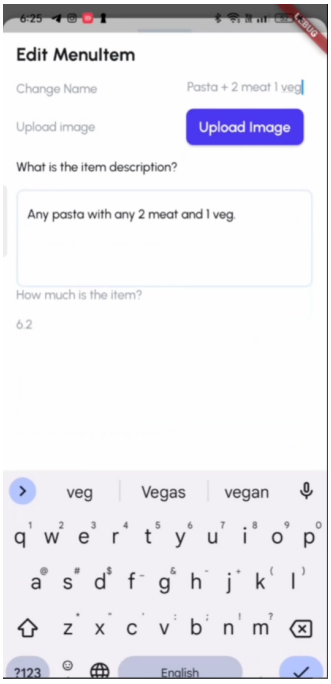


Figure 12: Edit Menu Item

3.1.11 Start Chat

From the reviews page(Figure 5), users can connect with other like minded foodies by starting a chat with them. This would allow users to find out more about that particular makanspot by talking to other users who have tried and reviewed that MakanSpot. The chat is real time and completely in-app.

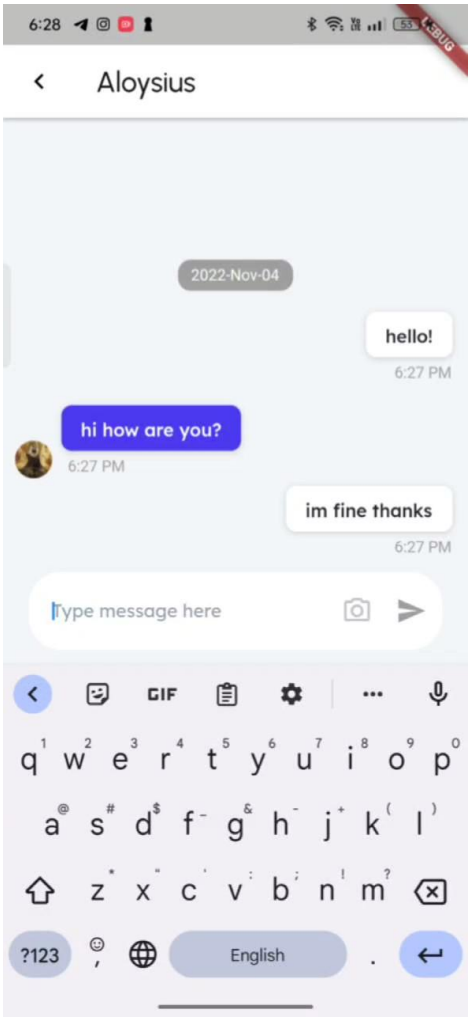
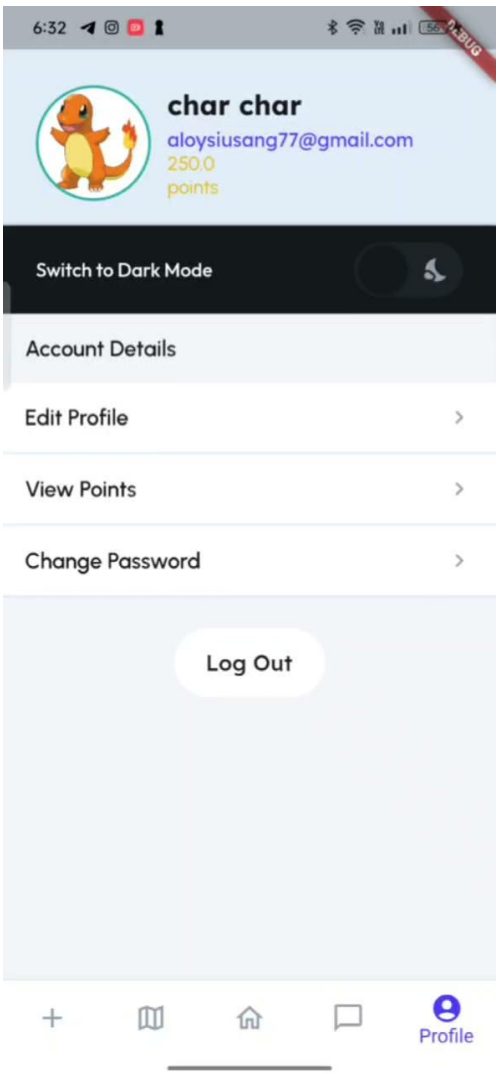


Figure 13: Chat Function

3.1.12 Profile Page (Switching Between Dark/light mode)

The profile page displays the users profile details (Eg. profile picture,name,email, number of points) as well as other functionalities. Users are also able to toggle between light and dark mode to suit their preferences. Here users can log out from their account.



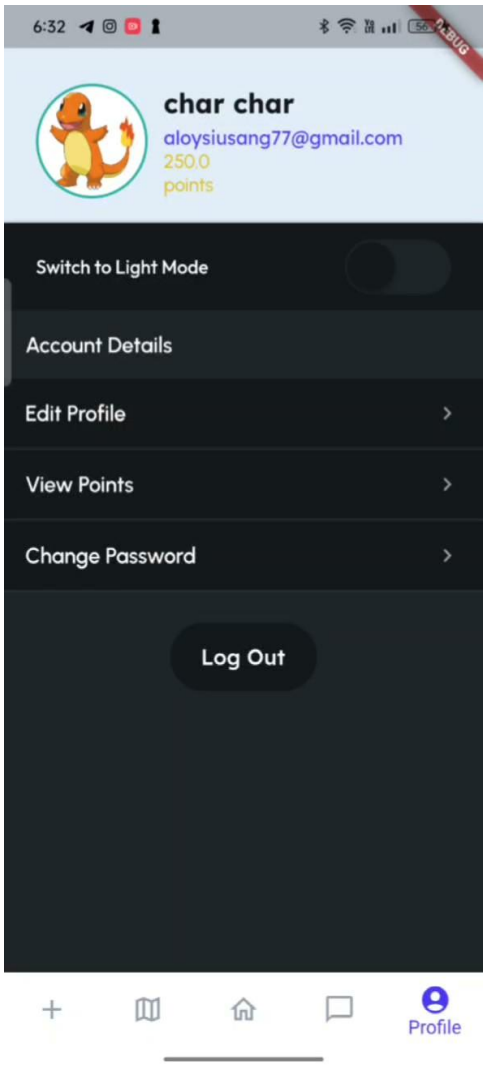
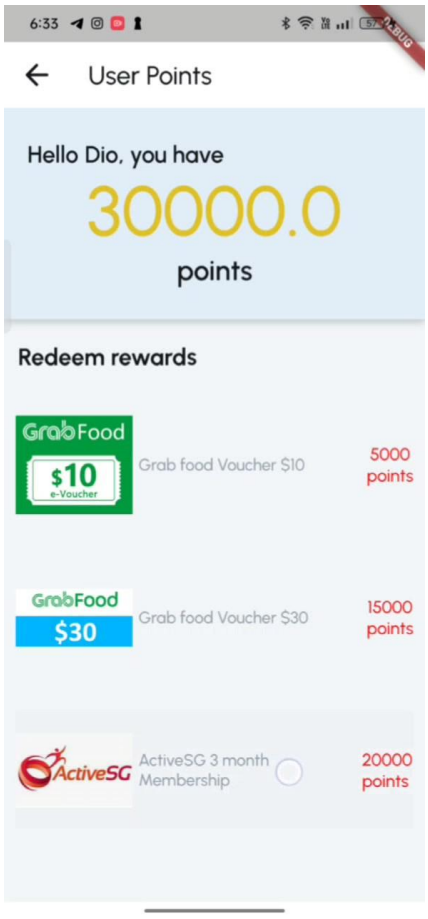
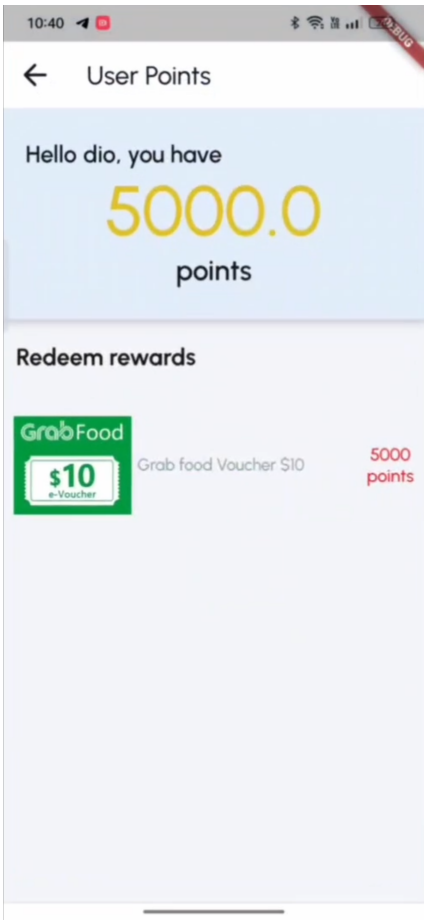


Figure 14 and 15: Profile Page after light and dark themes have been toggled

3.1.13 Points Redemption

At the user points page, the user is able to see the total points that he has. The rewards he is eligible for are displayed. If he has insufficient points, the rewards will not be displayed.





Figures 16 and 17: User Points page. Figure on the right shows what happens when there is insufficient points for a reward.

3.1.14 Bring Me There

From the MakanSpot Details Page(Figure 4), users can tap the “Bring Me There” Button which will automatically direct users to google maps with the provided directions from the users current location.

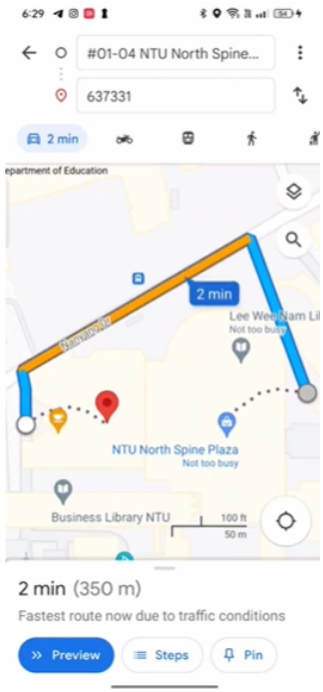


Figure 18: Bring me there, after Google Maps has been launched.

3.2 Hardware Interfaces

The application accesses the database through the internet. Hence, the hardware interfaces for the system will be all the hardware required to connect to the internet. For example: Modem, Router, WAN-LAN, Ethernet Cross-Cable.

3.3 Software Interfaces

This is a mobile application and will interface with android-based software components

Frontend Components

- All frameworks and libraries will fundamentally interact with Hyper Text Markup Language, Cascading Style Sheets,
  Javascript and Dart in Flutter.
- The official tool used to code the app providing all necessary tools and resources is Android Studio and the languages used to create content was Java
- The app uses Google maps API and Places API to enable the “Bring me there” feature which integrates Google Maps and the App. It provides users the incentive to be guided to the Makanspot with a single click of a button

Backend Components

- App is developed in Java and its APIs are designed to be called from Java
- Firebase is used for the persistent storage of user information and data.

4. System Features and Non-Functional Requirements

4.1 System Features

4.1.1 System Feature 1: Login

4.1.1.1 Description

Upon first entering the app, the user will be prompted to log in using their Email and password.

4.1.1.2 Stimulus/Response Sequences

User action: The user inputs their Email and password

System Response: The system checks the Email and password with the user database. If the Email and password matches, the user will be successfully logged in. Else, the system will display an error message and prompt the user to attempt login again, repeating step 2.

User action: The user inputs the wrong Email and password again

System Response: The system lets the user know that log in has failed.

4.1.1.3 Functional Requirements:

1. The system must store login information into a database application.
2. The system must be able to retrieve the information stored from the database.
3. The system must be able to match the information that the user has input to that of the database and conclude whether the information that the user has input is valid.
4. The system must be able to do the above efficiently, such that the user can know whether the login is successful or not in five seconds.
5. The system must be able to prompt the user to retry login if the previously entered login info is not found in the database.

4.1.2 System Feature 2: Sign up

4.1.2.1 Description

The system will require a new user to sign up for a new account. The following details will be required:

- Email
- Password

- Phone Number
- Profile photo
- Dietary Preferences

4.1.2.2 Stimulus/Response Sequences

User Action: New users who do not have an account, click on create account.

System Response: The system brings the user to create a new account page.

User Action: The user enters the required information into the various text fields to sign up an account

System Response: The system updates the database of existing users by adding in the newly registered account

4.1.2.3 Functional Requirements:

1. System must be able to verify that the email is in the correct form, eg.xxx@gmail.com
2. The system must verify that the input email address is not already present in the system.
3. The system must verify that the password and confirm password text fields are the same.
4. The system must verify that the password fields have a minimum of 6 characters.
5. Upon successful sign up, the system must enter the new details into the database.

4.1.3 System Feature 3: AddMakanSpot

4.1.3.1 Description

The system allows users to add new MakanSpots that are not already inside the database. The user will enter details of the MakanSpot earn points in return.

- Photo of the MakanSpot
- Name of the MakanSpot
- Location
- PostalCode
- Operating Hours
- Cuisine
- Dietary Preferences
- Price Range

4.1.3.2 Stimulus/Response Sequences

User Action: The user inputs a new MakanSpot

System Response: The system prompts the user to input details of the new MakanSpot and provide a review.

User Action: The user inputs details and review

System Response: The system adds the details and the new MakanSpot to the database.

4.1.3.3 Functional Requirements

1. The system must add the new MakanSpot to the database
2. Users must be able to retrieve the new MakanSpot within 30s
3. The new MakanSpot must be presented in the home page within 30s

4.1.4 System Feature 4: Map

4.1.4.1 Description

The system will display the user's current location on a dynamic map. Users are able to use the blue circle to demarcate an area they would like to scan MakanSpots for.

4.1.4.2 Stimulus/Response Sequences

User Action: The user taps on the map

System Response: The system prompts the user with a blue circle covering a certain proximity of an area.

User Action: The user can drag the bar on the top to adjust the size of the circle

System Response: The system adjusts the size of the circle according to the users adjustments

User Action: The user adjusts the area which the circle scans.

System Response: The system adjusts the map accordingly.

User Action: The user can then tap the arrow key on the top right of the screen

System Response: The system then directs the users to the homepage with the list of makanspots within that area the user has selected

4.1.4.3 Functional Requirements

1. Google Maps and Places API are required for the maps to function

2. After the user adjusts the size of the circle using the slider, the system must reflect the change in size within one second.
3. After the user adjusts the area where he would like to scan, the system must reflect the change in area within one second.
4. The system must be able to match the area that the user has selected to the database and search for MakanSpots within that area within one second.

4.1.5 System Feature 5: Filter MakanSpot

4.1.5.1 Description

Users are able to filter the MakanSpots available according to price by selecting the number of '$' signs in the filter. There are 3 options. 1 $ means price less than $10, 2 $ means prices between $10 and $50, and 3 $ means price more than $50. After the user selects the number of '$' signs, the app will only show MakanSpots matching the number of '$' signs.

4.1.5.2 Stimulus/Response Sequences

User Action: The user selects his option to choose his desired price range

System Response: The app will only show MakanSpots within that price range

4.1.5.3 Functional Requirements

1. All MakanSpots are required to have 1, 2, or 3 '$' signs in their description when users add MakanSpot.
2. Once the user's selection has been made, the app should filter and display the desired MakanSpots within 2 seconds

4.1.6 System Feature 6: MakanSpot Info

4.1.6.1 Description

A more detailed description of the MakanSpot will be displayed. Information like the average rating, dietary preferences, scrollable reviews and menu, and other functionalities(Eg. View menu, Bring me there, leave a review Etc) are available on this page.

4.1.6.2 Stimulus/Response Sequences

User Action: From the HomePage, the user can tap on any MakanSpot they are interested in.

System Response: The system would then display the MakanSpot Information.

4.1.6.3 Functional Requirements

1. The MakanSpot, its menu and reviews related to it should already exist in our DataBase.
2. The system must bring the user to the MakanSpot Info within 2 seconds of clicking on the MakanSpot on the HomePage.

4.1.7 System Feature 7: Chat Functionality

4.1.7.1 Description

Users will be able to start a chat with other users from the reviews page. The chats page will show the ongoing conversations that the user has.

4.1.7.2 Stimulus/Response sequences

User action: The user clicks on the chat icon on the review that the other user has left.

System response: The system brings the user to a chat where a chat has been automatically started with the other user.

User action: The user sends a message.

System response: The system sends a message to the other user, and presents the message as a new conversation that can be found in the chat page of the other user.

User action: The other user clicks on the new conversation and starts sending messages to the first user if he wants.

4.1.7.3 Functional Requirements

1. The system must bring the user to a new conversation with the other user after he clicks on the chat icon within 2 seconds.
2. When a message is sent, if the receiver has no ongoing conversations with the sender, the message must still appear in the chat page of the receiver.
3. Messages must be sent to the receiving user within 2 seconds, and must appear on the receiver's chat page within 10 seconds.
4. All conversations must be stored within the database.

4.1.8 System Feature 8: Edit MakanSpot Info

4.1.8.1 Description

Users are able to edit the information of a particular MakanSpot if it is inaccurate. The information are all the parameters attributing to a MakanSpot (Eg. Operating hours, Dietary preferences, Operating Hours, Etc).

4.1.8.2 Stimulus/Response sequences

User Action: On the MakanSpot Info page, users can tap on the “Edit MakanSpot” button to edit the information.

System Response: The system will bring users to the list of text fields which they can edit.

User Action: After editing the text fields deemed required, user can tap on the “update button”

System Response: The Database of that MakanSpot will be updated and it will be reflected on the MakanSpot Info with the new information.

4.1.8.3 Functional Requirements

1. The system must be able to check if any inputs are missing, or if any inputs are invalid, such as Postal Code have to be an integer, and the image uploaded must be an appropriate file type
2. The system must have access to the information about the MakanSpot on the database and be able to edit it
3. When the user confirms the inputs, the system should be able to edit the information within 2 seconds
4. The system should be able to display the updated information when the user views the MakanSpot information again

4.1.9 System Feature 9: Bring Me There

4.1.9.1 Description

Directions to the MakanSpot from the user’s current location will be shown after the user toggles Bring Me There. The system will automatically open Google Maps and show the directions.

4.1.9.2 Stimulus/Response sequences

User Action: User selects ‘Bring Me There’ button from the MakanSpot Info page.

System response: Opens Google Maps with current location set as start point, MakanSpot set as destination automatically.

User Action: User toggles mode of transport (Walk, Drive, Public Transport, etc)

System response: The system changes the mode of transport, calculating and displaying the estimated time of arrival.

4.1.9.3 Functional Requirements

1. All MakanSpots should have a postal code (this is required when the users adds a MakanSpot)
2. When users click on the “Bring Me There” button, the app should be able to open Google Maps within 2 seconds. The duration of Google Maps loading will depend on the user’s device
3. The time taken by Google Maps to carry out the user’s actions in Google Maps will depend on Google Maps and the user’s device

4.1.10 System Feature 10: Redeem Rewards

4.1.10.1 Description

Users will earn points when they add or edit MakanSpots, menu items or add reviews. Adding MakanSpots or menu items rewards 100 points, editing MakanSpots or menu items or adding reviews rewards 50 points. Users are allowed to exchange the points they earned for discounts, like Grab vouchers. Users have to open the ‘View Points’ page from the profile page. There, all redeemable rewards will be displayed. Rewards that are unredeemable due to the lack of points will be hidden

4.1.10.2 Stimulus/Response sequences

User Action: Users earn points by editing or adding MakanSpots, menu items or adding reviews.

System Response: System awards users based on their actions. Add MakanSpot or add menu items rewards 100 points. Editing MakanSpot or menu items or adding reviews rewards 50 points.

User Action: User click on “View Points” button on the profile page

System Response: System shows the total number of points users have and displays the rewards that are redeemable by the user

User Action: User clicks on one of the rewards

System Response: System automatically deducts the user’s points and gives the user his reward

4.1.10.3 Functional Requirements

1. The app should collaborate with other companies like Grab to provide the rewards

2. The system should have access to the user's points, and have the ability to add, deduct and view the user's points
3. When the user presses the "View Points" button, the system should display the number of points within 2 seconds
4. When the user redeems a reward, the system should be able to deduct the corresponding amount within 2 seconds

4.2 Non-functional Requirements

4.2.1 Performance Requirements

The app should have a response time of ideally less than a second, and up to 2 seconds. The time taken for a user to receive a message in the chat function is allowed to take up to 10 seconds.

The app should be able to load within 5 seconds when the number of users using the app is more than 10,000.

The app should be able to display search results within 5 seconds after the user selects an area on the map to scan for MakanSpots.

The app should be able to display filtered MakanSpots within 2 seconds after the user uses the price filter.

The "Bring Me There" function should be able to open Google Maps within 2 seconds to guide the user on where to go.

The app should be able to update the database and display the updated information within 2 seconds after the user makes edit to MakanSpots, menu items and their profile.

The app should be able to add and display the user's reviews within 2 seconds after the user makes his review.

For the chat function, users should be able to receive text messages within 10 seconds after the other user sends a message.

The app should be able to switch between light or dark mode within 2 seconds.

The app should be able to send a reset password email within 1 minute after the user selects the "change password" button.

The app needs to be compatible in different platforms like IOS and Android.

User interfaces need to be clearly organised so that users are able to find their desired functions quickly and without hassle. The interface must be user-friendly and easy to use.

Text fields requiring input from the user should have example or hidden text as much as possible, to guide the users on what to input in the corresponding text field.

4.2.2 Safety Requirements

After a system reboot, the full system functionality must be restored within 5 minutes

The login page and create account page should only be shown at the start of the app

The app should be made up of modern themes to make it look more unique, new and modern and aesthetically pleasing.

4.2.3 Security Requirements

The app should not grant access to the user if the user does not create a password with more than 6 characters.

User data must be kept from the public. Only the users themselves are able to see

4.2.4 Software Quality Attributes

Theoretically, any faults reported should be fixed within the shortest time possible. App being down should not exceed more than 24 hours

4.2.5 Business Rules

User information should be kept confidential at all times, and be shown to only authorised personnel. User information is collected for verification purposes only, and should not be given to our app partners like Grab (Grab is one of the partners supplying rewards)

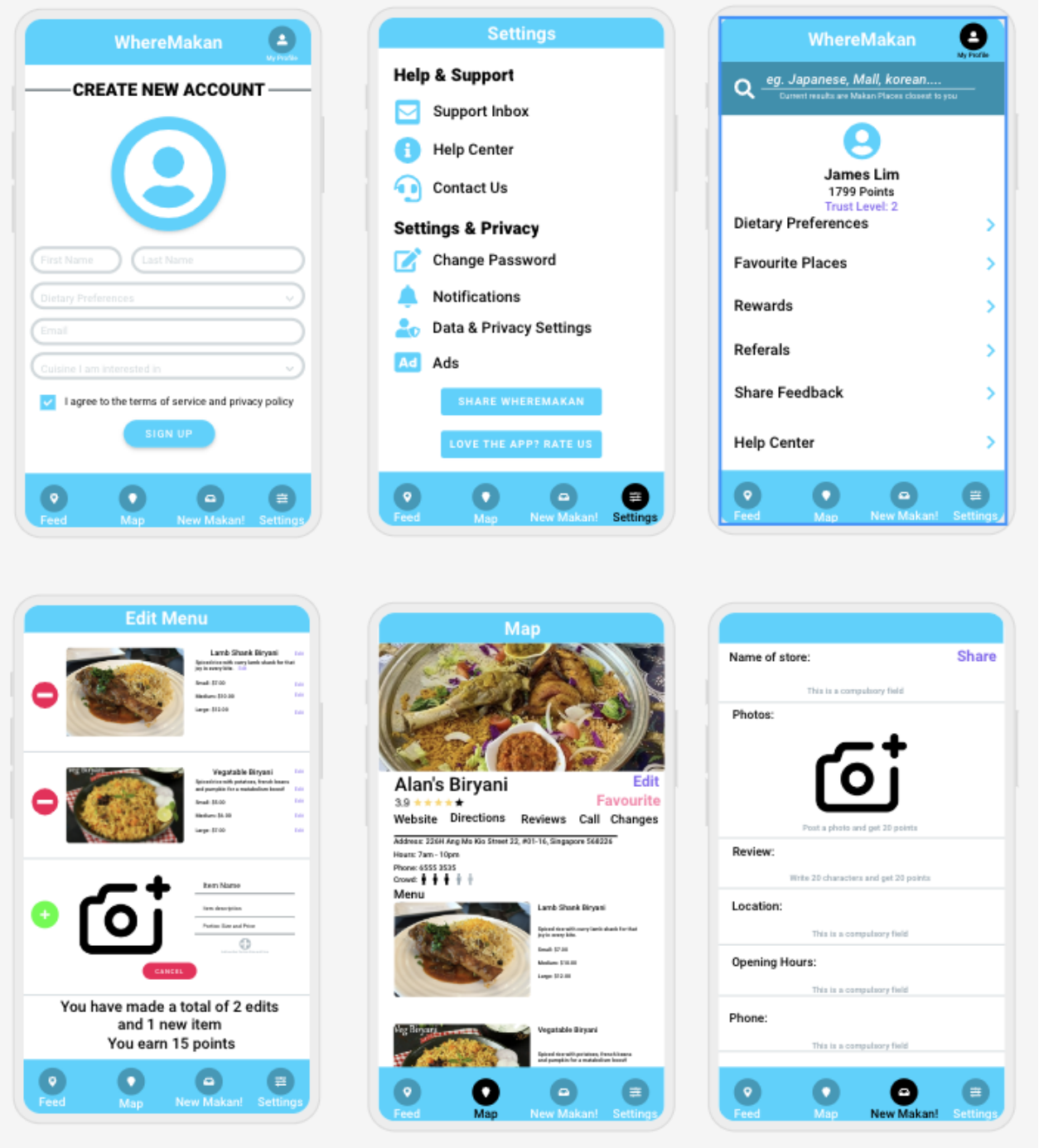
4.2.6 Other Requirements

The app’s back-end database uses Firebase and can only be accessed by authorised personnel

The database must be replaceable with any commercial product supporting standard SQL queries

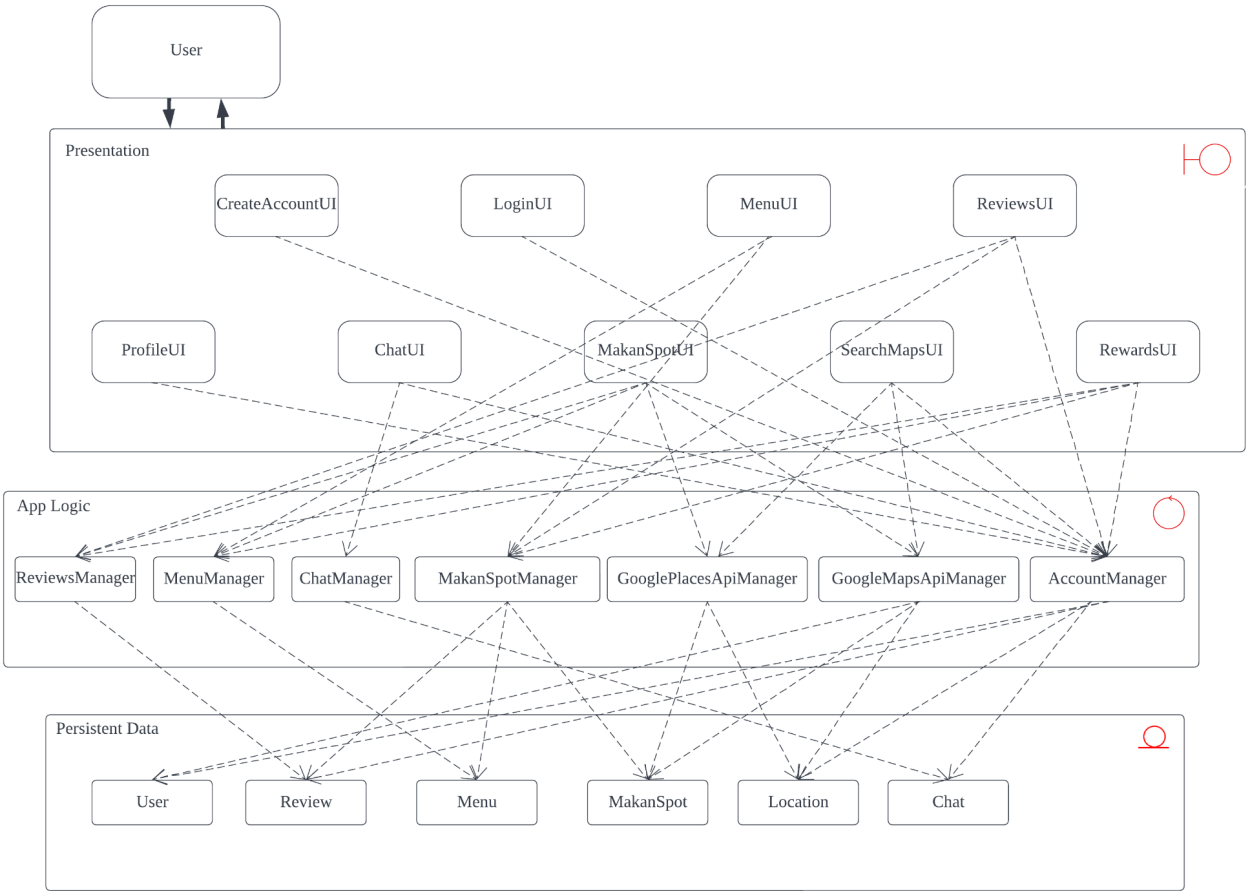
5. UI Mock-Up

The following diagram shows our initial UI during our planning phase. Much of it has been revised in the actual app.



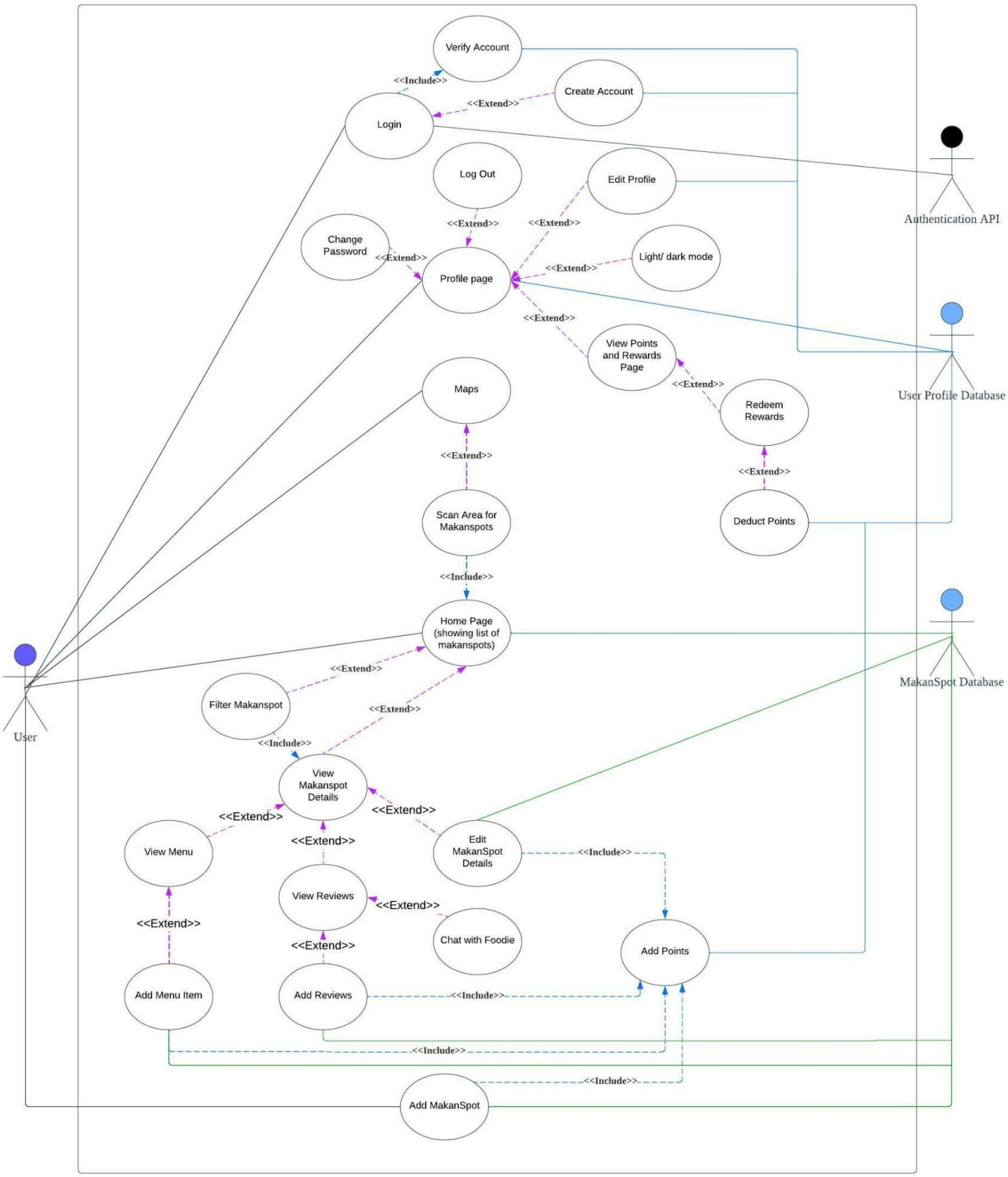

WhereMakan
eg. Japanese, Mall, korean....
1.28KM
KFC @ Thomson Plaza
Old Chang Kee @ Thomson Plaza
Feed
Map
New Makan!
Settings
Map
Foodelicia
A Hot H
Golden Spoon Eating House and Catering
Pen & Inc, NTU
The Sandwich Guys - NTU
Chinese Delights
A & I Hainanese Boneless Chicken...
Add filers
Map
Feed
Map
New Makan!
Settings
Map
Foodelicia
A Hot H
Golden Spoon Eating House and Catering
Pen & Inc, NTU
The Sandwich Guys - NTU
Chinese Delights
A & I Hainanese Boneless Chicken...
Feed
Map
New Makan!
Settings
ADD MAKANSPOT
Special Diet:
I hereby declare that the information provided is true and correct.
This submission grants you 350 points
Points will only be added after verification.
CANCEL
SUBMIT
Feed
Map
New Makan!
Settings
Map
Foodelicia
A Hot H
Golden Spoon Eating House and Catering
Pen & Inc, NTU
The Sandwich Guys - NTU
Price Range
Enter your price range eg: 0 - 4.5
Distance
Enter the maximum distance
Feed
Map
New Makan!
Settings
MY FEED
What's Popular
Alan's Biryani
284 Kway Chap
Verify for Points!
Cava Singapore
Edit
Verify
Feed
Map
New Makan!
Settings

6. System Architecture



7. Use Cases

7.1 Use Case Diagram



7.2 Specific Use Cases

7.2.1 Use Case 1: Login/ Logout

7.2.1.1 Use Case Description:

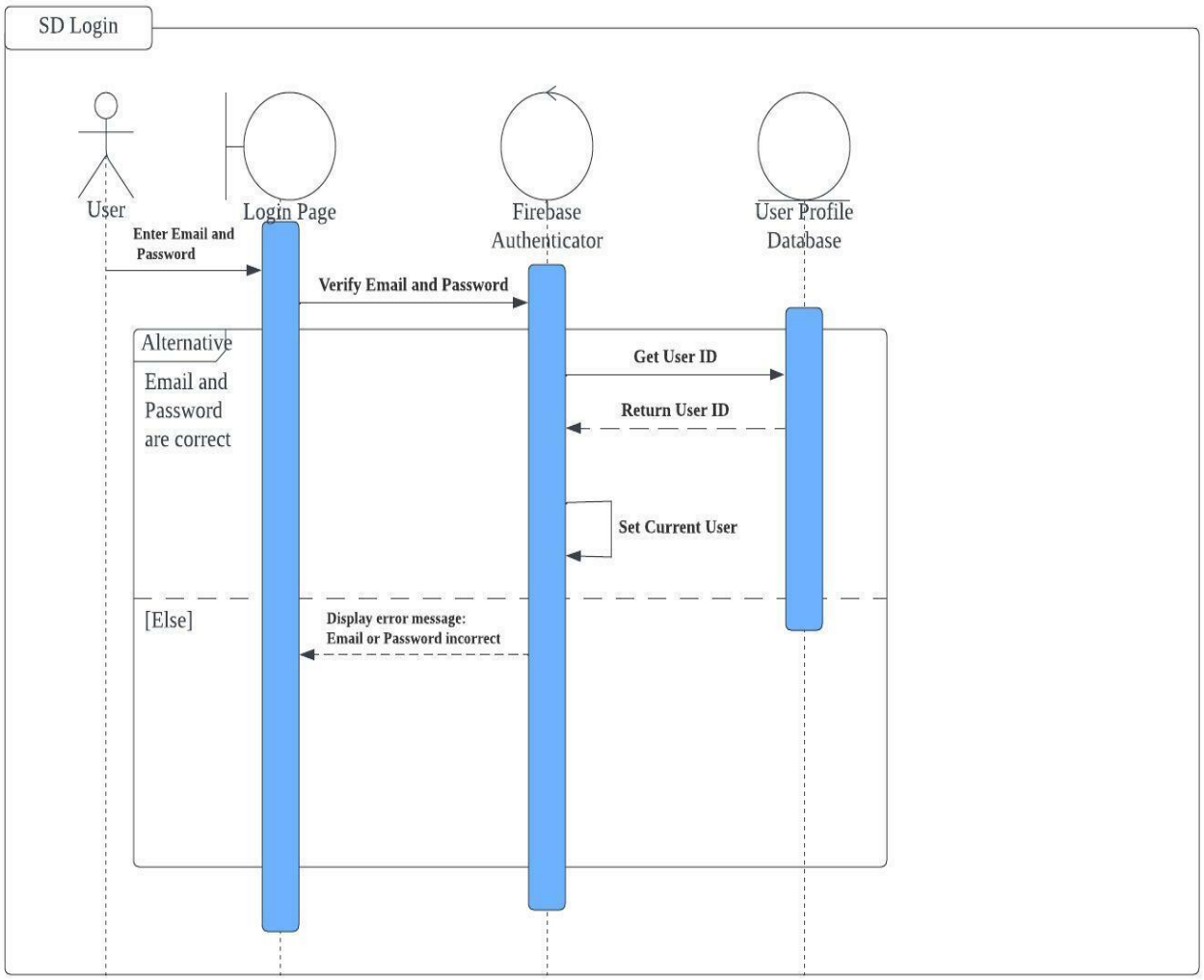
Use Case ID:	*1*		
Use Case Name:	Login/Logout		
Created By:	Aloysius	*Last Updated By:*	*Davis*
Date Created:	*9/1/2022*	*Date Last Updated:*	*28/10/2022*

Actor:	App users
Description:	Users can login to use the MakanGoWhere App and can log out after using the app.
Preconditions:	Users must have an account in order to login to the MakanGoWhere App.
Postconditions:	Users can now use the functionalities of the MakanGoWhere App.

Priority:	High
Frequency of Use:	Low
Flow of Events:	For users with accounts: 1. At the login page, users have to key in their email addresses and passwords 2. If both email addresses and passwords are correct, they will login and be directed to the home page. 3. After using the app, users can go to the profile page and logout.
Alternative Flows:	NIL
Exceptions:	1. If the email address keyed in is not formatted properly, an error message “Error: The email address is badly formatted” will be displayed. 2. If the password is keyed in incorrectly or the email address does not exist, an error message “ The password is invalid or the user does not have a password” will be displayed.
Includes:	NIL
Special Requirements:	NIL
Assumptions:	Users have already created their account on MakanGoWhereApp.

Notes and Issues:	0

7.2.1.2 Sequence Diagram:



7.2.2 Use Case 2: Create Account

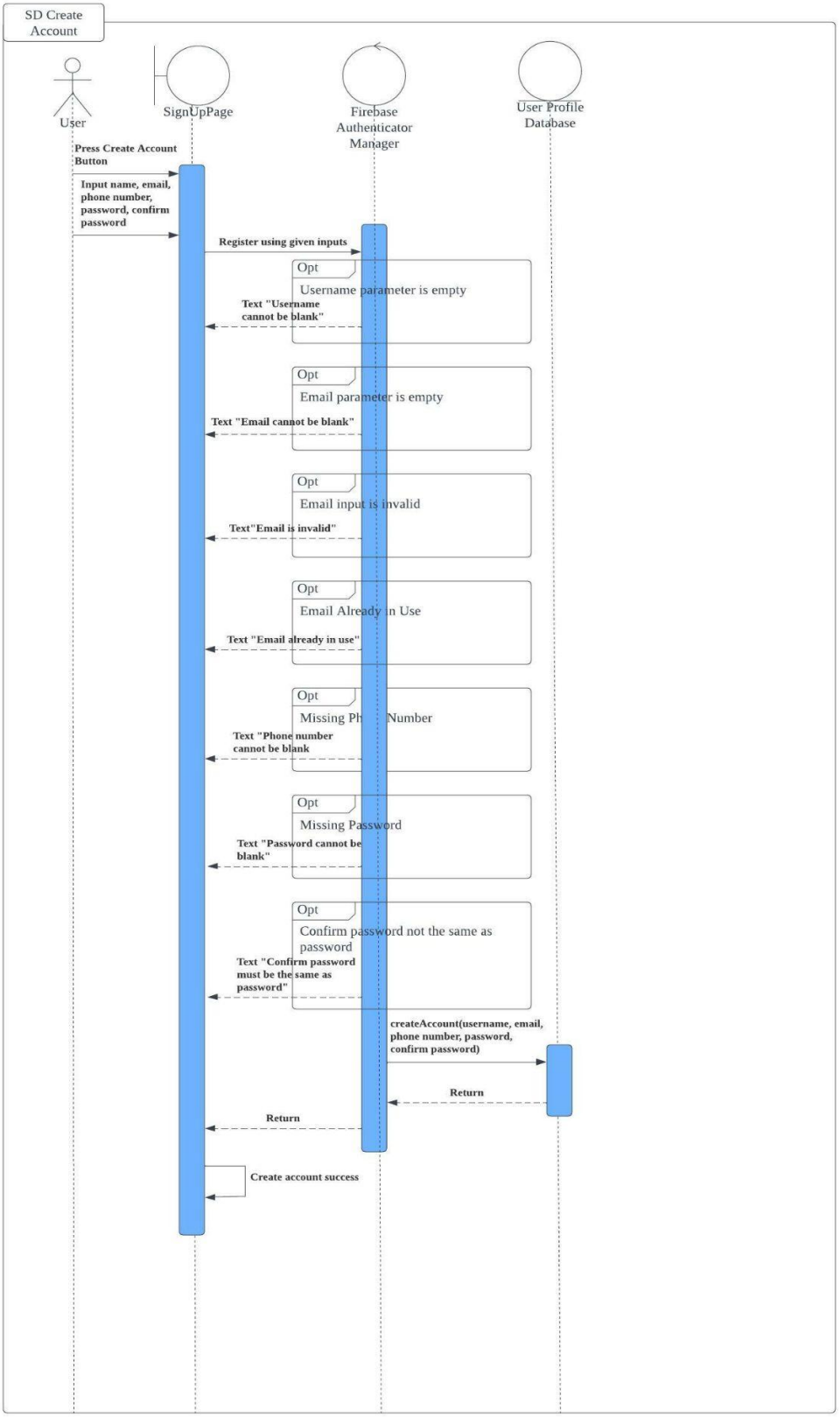
7.2.2.1 Use Case Description:

Use Case ID:	*2*		
Use Case Name:	Create Account		
Created By:	*Aloysius*	*Last Updated By:*	*Davis*
Date Created:	*8/18/2022*	*Date Last Updated:*	*28/10/2022*

Actor:	App users
Description:	Users set up their account using their email address and phone number.
Preconditions:	Users must have an existing email address and phone number.
Postconditions:	Users are now able to login to MakanGoWhere using their email address and password
Priority:	High
Frequency of Use:	Low

Flow of Events:	1. Users click on the button “Create Account”. 2. Users will now key in their desired account name, email address, phone number, password, and confirm password. 3. Users click on the “Create Account” button and the account is now created.
Alternative Flows:	NIL
Exceptions:	1. If the email address keyed in is already used in one of the existing accounts, an error message “The email address is already in use by another account.
Includes:	NIL
Special Requirements:	NIL
Assumptions:	Users have yet to create an account
Notes and Issues:	0

7.2.2.2 Sequence Diagram:



7.2.3 Use Case 3: Display Map

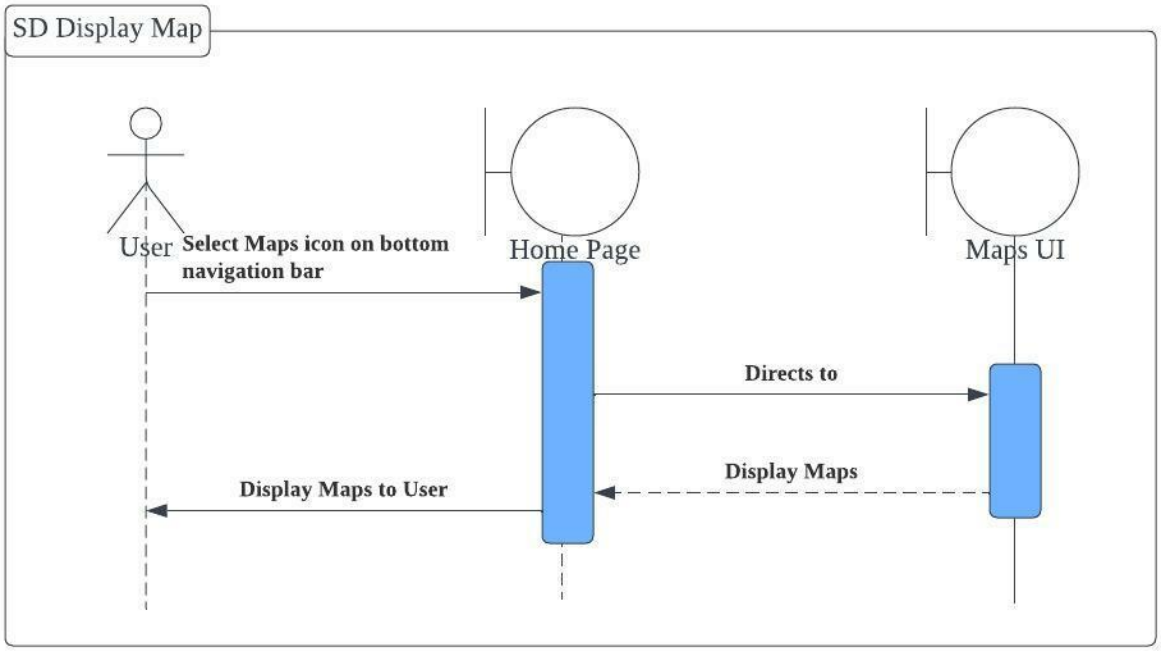
7.2.3.1 Use Case Description:

Use Case ID:	*3*		
Use Case Name:	*Display Map*		
Created By:	*Aloysius*	*Last Updated By:*	*Davis*
Date Created:	*8/18/2022*	*Date Last Updated:*	*28/10/2022*

Actor:	App users, database
Description:	Users will have access to map
Preconditions:	Users must have a MakanGoWhere account.
Postconditions:	The map will be displayed
Priority:	High

Frequency of Use:	When users want to scan an area on the map for makanspots.
Flow of Events:	1. Users click on the map icon on the navigation bar below.
Alternative Flows:	NIL
Exceptions:	NIL
Includes:	NIL
Special Requirements:	NIL
Assumptions:	NIL
Notes and Issues:	0

7.2.3.2 Sequence Diagram:



7.2.4 Use Case 4: Display list of MakanSpots

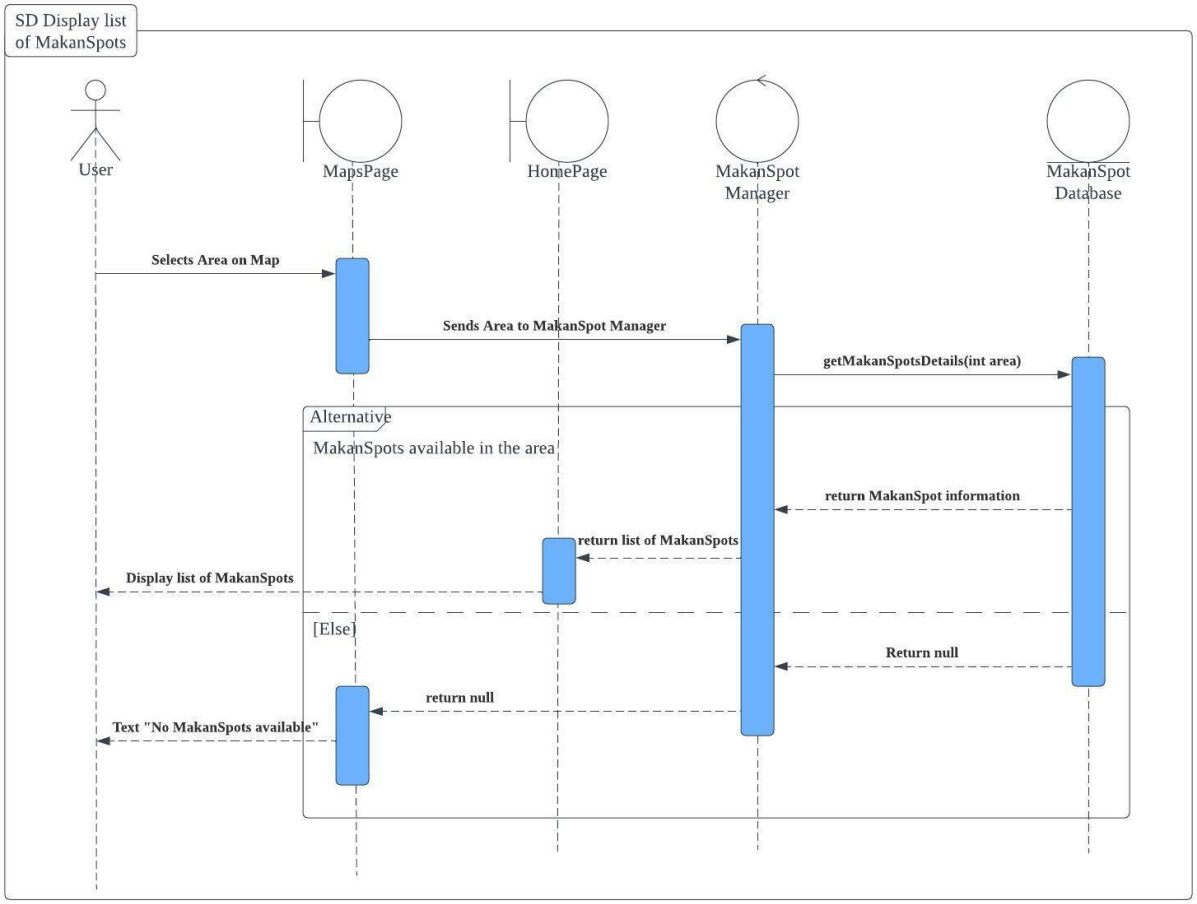
7.2.4.1 Use Case Description:

Use Case ID:	*4*		
Use Case Name:	*Display list of MakanSpots*		
Created By:	*Aloysius*	*Last Updated By:*	*Davis*
Date Created:	*8/18/2022*	*Date Last Updated:*	*28/10/2022*

Actor:	App users, database
Description:	*When the user wants to display a list of makan spots at an area, he will go to the map, tap at that area on the map, and scan the area for makanspots.*
Preconditions:	Users must scan at an area on the map
Postconditions:	Lists all available food choices based on the area on the map that has been scanned.
Priority:	High
Frequency of Use:	High
Flow of Events:	1. Users go to the map interface. 2. Users tap at an area on the map. 3. Users scan that area. 4. After scanning, users will be directed to the makanspots page where the list of makan spots in the area scanned will be displayed.
Alternative Flows:	NIL
Exceptions:	If there are no existing food places in the area scanned in the database, the makanspots page will be empty.
Includes:	NIL

Special Requirements:	NIL
Assumptions:	The area scanned will have existing makanspots in the database
Notes and Issues:	0

7.2.4.2 Sequence Diagram:



7.2.5 Use Case 5: Filter MakanSpot based on price

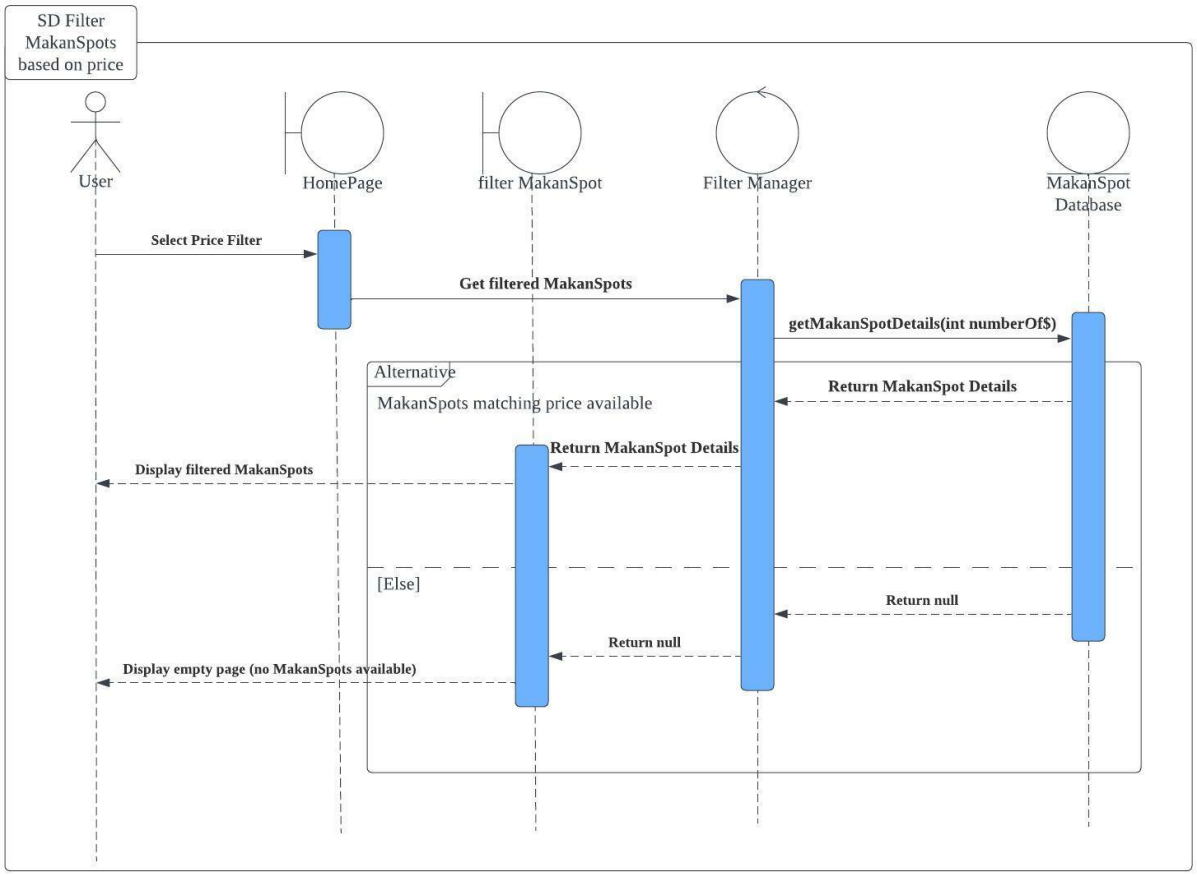
7.2.5.1 Use Case Description:

Use Case ID:	*5*		
Use Case Name:	Filter MakanSpot based on price		
Created By:	Aloysius	*Last Updated By:*	
Date Created:	*9/1/2022*	*Date Last Updated:*	

Actor:	App users, database,
Description:	At the “Filtered Makanspot” page, users are able to filter the available Makanspots according to their dietary requirements, such as halal/ non-halal, vegetarian, and their allergies
Preconditions:	Makanspots should have dietary information about them. Thus when users are adding or verifying Makanspots, they must specify the type of food each Makanspots serve
Postconditions:	The “filtered Makanspots” screen page will show the available Makanspots based on the user’s filter settings

Priority:	High
Frequency of Use:	1
Flow of Events:	1. At the “filtered Makanspots” page, users can press the filter icon at the top right of the screen 2. Users will be given a choice to select the parameters themselves, such as halal, vegetarian, allergies etc. 3. Based on the filter, the app will show recommended Makanspots again
Alternative Flows:	Nil
Exceptions:	Nil
Includes:	
Special Requirements:	Makanspots must have dietary information
Assumptions:	Nil
Notes and Issues:	*0*

7.2.5.2 Sequence Diagram:



7.2.6 Use Case 6: View MakanSpot Details

7.2.6.1 Use Case Description:

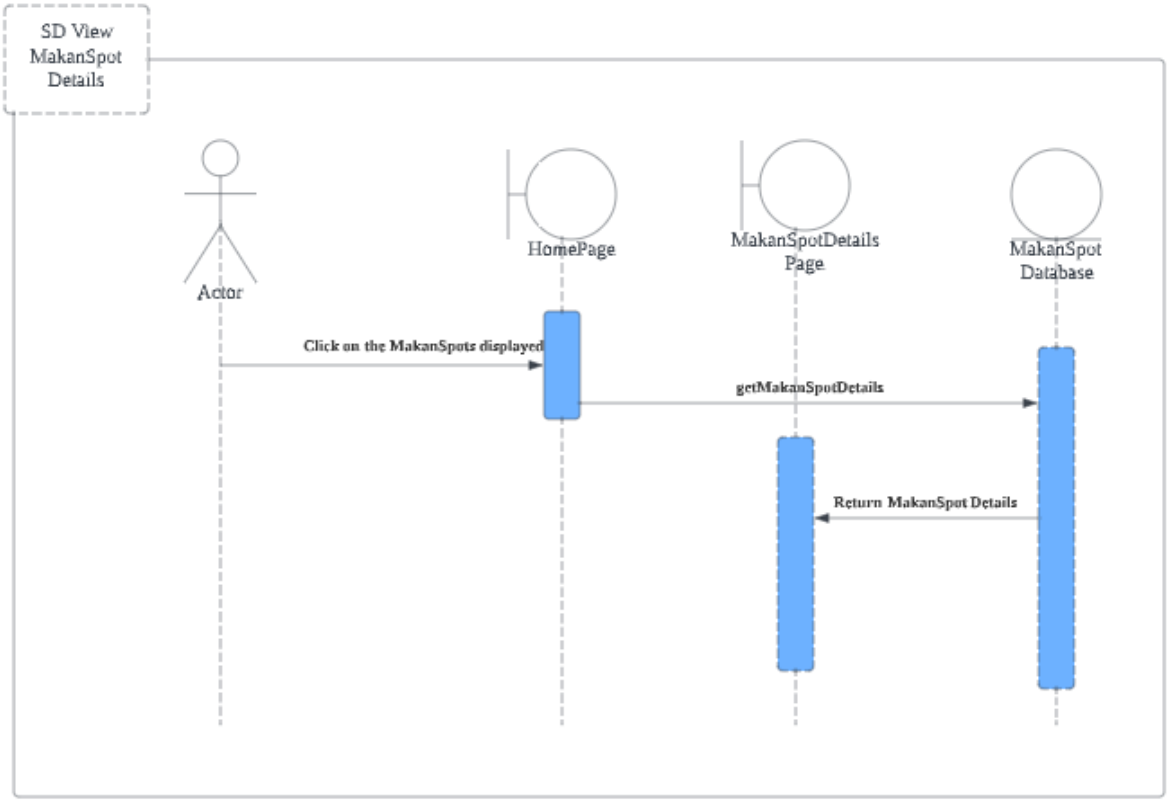
Use Case ID:	*6*		
Use Case Name:	View MakanSpot Details		
Created By:	Darren	*Last Updated By:*	

Date Created:	*9/1/2022*	*Date Last Updated:*	

Actor:	App users, database,
Description:	Displays all the detailed information of the MakanSpot such as the summary of reviews, dietary preferences and description. It also allows users to edit the makanspot and leave a review.
Preconditions:	Users would have to tap on the makanspot from the homepage
Postconditions:	The “MakanSpot Details” screen page will show the details of that particular MakanSpot
Priority:	High
Frequency of Use:	High
Flow of Events:	1. On the HomeScreen Page, users will see a list of MakanSpots in which they can scroll through. 2. Upon Finding a makanspot they would like to find out more about, users can tap on that makanspot which will bring them to the MakanSpot Details page.

Alternative Flows:	Nil
Exceptions:	Nil
Includes:	*Nil*
Special Requirements:	Nil
Assumptions:	Nil
Notes and Issues:	*0*

7.2.6.2 Sequence Diagram:



7.2.7 Use Case 7: Add MakanSpot

7.2.7.1 Use Case Description:

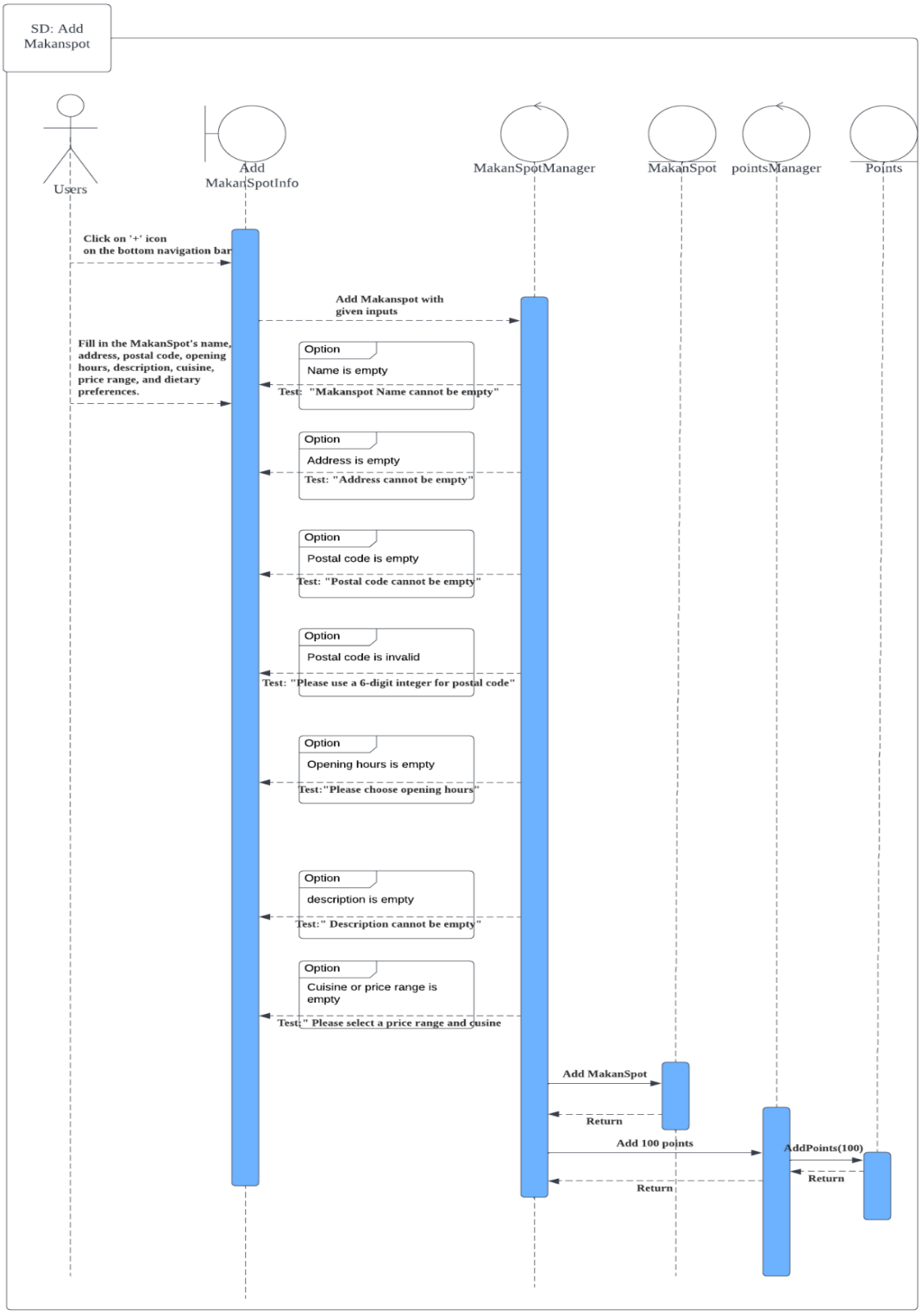
Use Case ID:	*7*		
Use Case Name:	*Add MakanSpot*		
Created By:	*Aloysius*	*Last Updated By:*	*Aloysius*

Date Created:	*8/18/2022*	*Date Last Updated:*	*8/18/2022*

Actor:	App users, database
Description:	Users are allowed to add Makanspots that are not available on the app. The added information will be stored in the Homepage where other users can also view.
Preconditions:	Users must know the correct information of the new makanspot
Postconditions:	Constant updates and community-driven updates on MakanSpot information, ensuring a smooth experience for users. Users will be rewarded for adding MakanSpots.
Priority:	High
Frequency of Use:	Medium
Flow of Events:	To add Makanspot: 1) Users click “+” 2) Users are now able to key in information of that Makan spot. 3) The List of information are as follows: ● Photo of the MakanSpot ● Name of the MakanSpot ● Location ● PostalCode

	• Operating Hours • Cuisine • Dietary Preferences • Price Range All added information will be stored in our database and displayed on the HomePage. Users will be able to see this new information in their search result. After this process, the user who added the information will be given Makanpoints, based on the number of makanspots they add.
Alternative Flows:	Nil
Exceptions:	All information must be completely filled in for the makanspot to be added.
Includes:	*Nil*
Special Requirements:	*Nil*
Assumptions:	Users key in the right and appropriate information about the Makanspot.
Notes and Issues:	nil

7.2.7.2: Sequence Diagram



7.2.8 Use Case 8: Edit MakanSpot

7.2.8.1 Use Case Description:

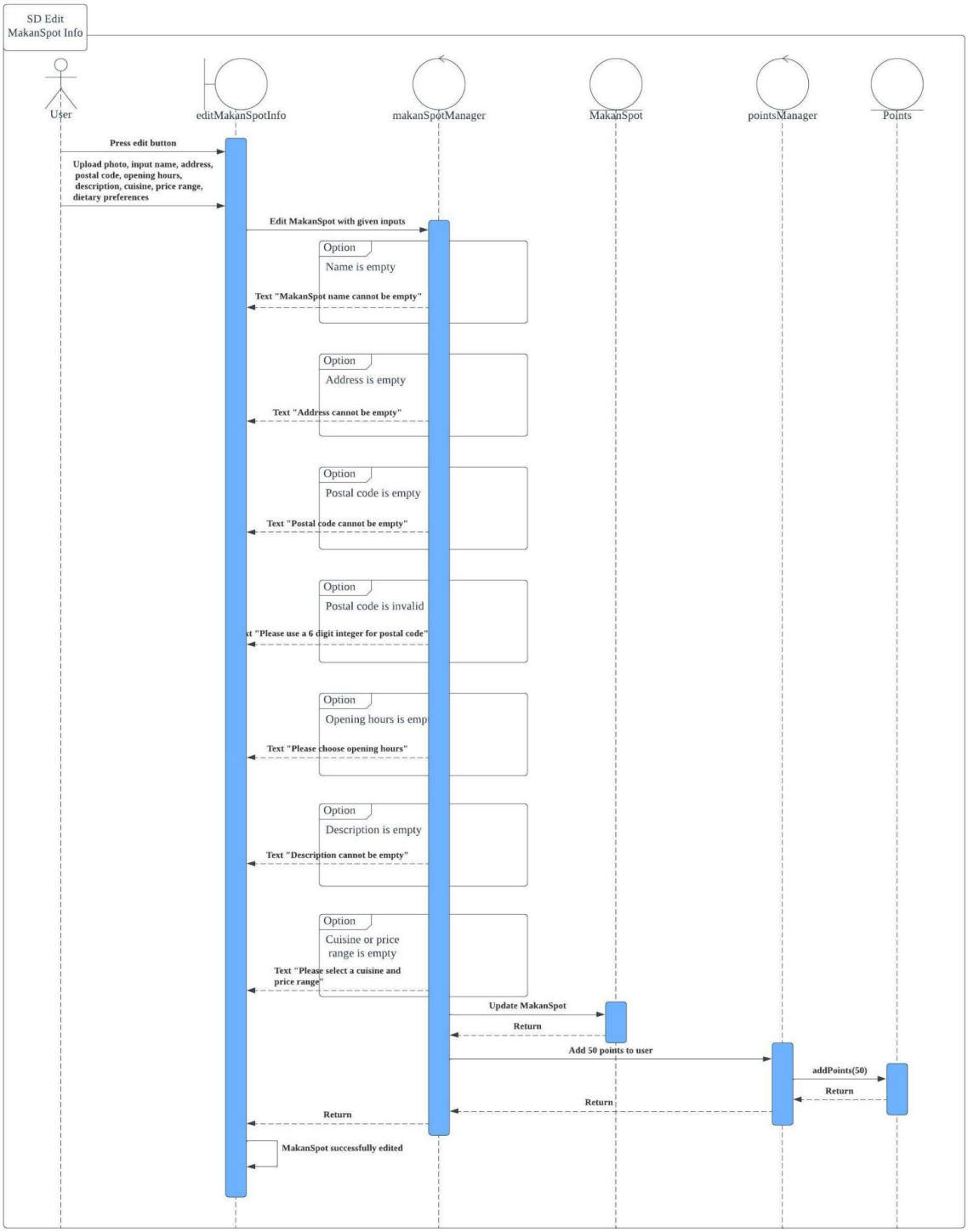
Use Case ID:	*8*		
Use Case Name:	*Edit MakanSpot*		
Created By:	*Aloysius*	*Last Updated By:*	*Davis*
Date Created:	*8/18/2022*	*Date Last Updated:*	*10/28/2022*

Actor:	App users, database
Description:	Users are now allowed to edit MakanSpot if the information of the makanspots are keyed in wrongly
Preconditions:	1. Users must know the correct updated information. 2. The makanspot to be edited exists in the database.
Postconditions:	Constant updates and community-driven updates on MakanSpot information, ensuring a smooth experience for users. Users will be rewarded for updating makanspots.
Priority:	High

Frequency of Use:	Medium
Flow of Events:	To update Makanspot: 1. Users click on the button “Edit MakanSpot”. 2. Users are now able to edit the following information: a. Photo of the MakanSpot b. Name of the MakanSpot c. Location d. PostalCode e. Operating Hours f. Cuisine g. Dietary Preferences h. Price Range 3. Users click on the button “Update” and now the information of the makanspot will be updated in the database. All updated information will be updated in our database and displayed on the HomePage. Users will be able to see this new information in their search result. After this process, the user who updated the information will be given Makanpoints, based on the number of edits they made.
Alternative Flows:	Nil
Exceptions:	There are no existing makanspots in the database.
Includes:	*Nil*
Special Requirements:	*Nil*

Assumptions:	There are existing makanspots in the database
Notes and Issues:	nil

7.2.8.2 Sequence Diagram:



7.2.9 Use Case 9: View MakanSpot Menu

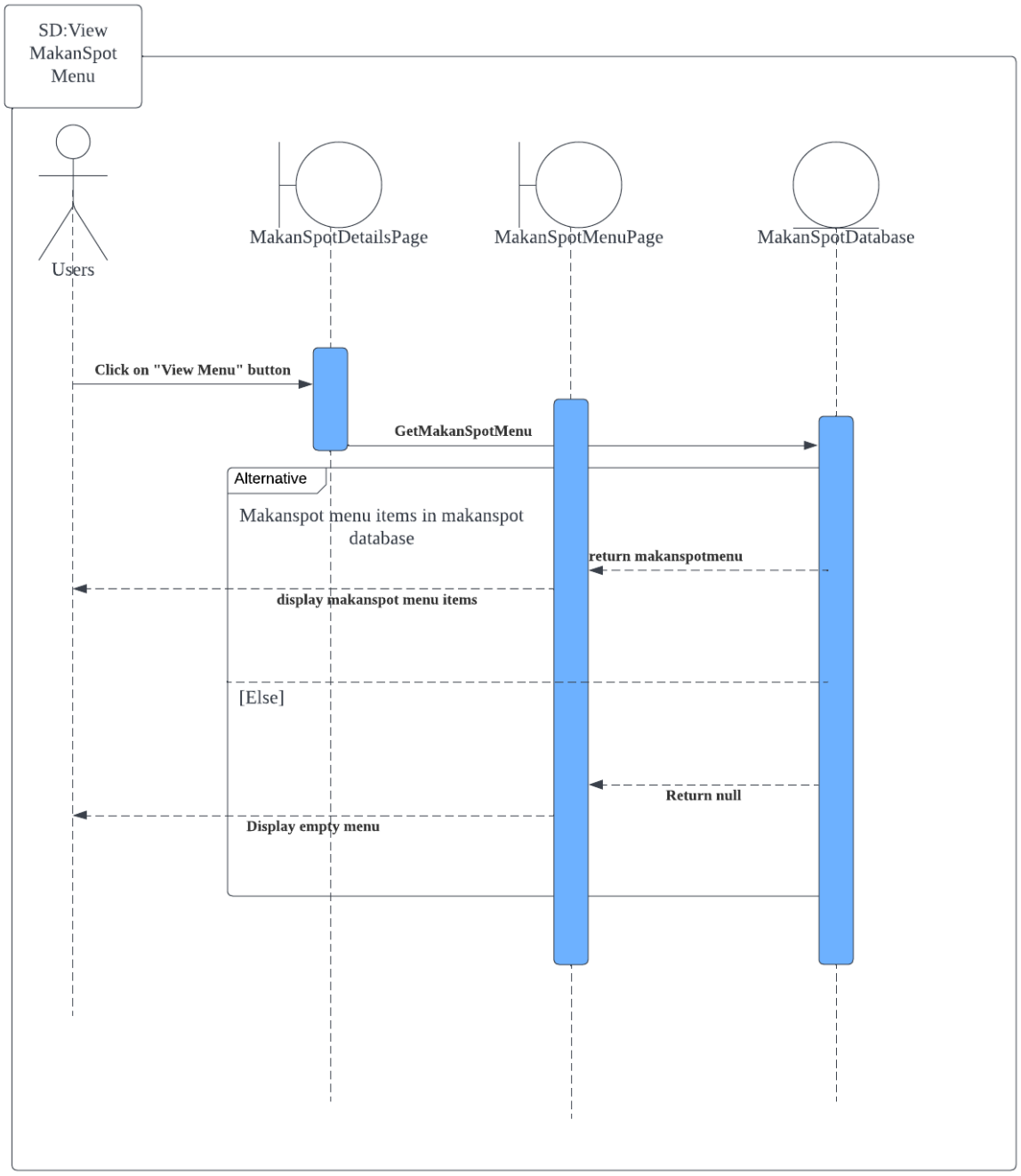
7.2.9.1 Use Case Description:

Use Case ID:	*9*		
Use Case Name:	*View MakanSpot Menu*		
Created By:	*Darren*	*Last Updated By:*	*Darren*
Date Created:	*8/18/2022*	*Date Last Updated:*	*8/18/2022*

Actor:	App users, database
Description:	Users are allowed to view the menu of the Makanspots.
Preconditions:	There must be an existing MakanSpot and a Menu list.
Postconditions:	*Users will be brought to the Menu page where they can see the list of menu items and its price.*
Priority:	High
Frequency of Use:	High

Flow of Events:	To View Makanspot Menu: 1. Users need to be in the HomePage to view the list of MakanSpots 2. Users will Tap on a MakanSpot on the HomePage to navigate to the MakanSpot Details Page. 3. Users can then tap the “View Menu” Button to view the list of menu and prices
Alternative Flows:	Nil
Exceptions:	Nil
Includes:	*Nil*
Special Requirements:	*Nil*
Assumptions:	There is already an existing MakanSpot
Notes and Issues:	nil

7.2.9.2: Sequence Diagram


SD:View MakanSpot Menu
Users
MakanSpotDetailsPage
MakanSpotMenuPage
MakanSpotDatabase
Click on "View Menu" button
GetMakanSpotMenu
Alternative
Makanspot menu items in makanspot database
return makanspotmenu
display makanspot menu items
[Else]
Return null
Display empty menu

7.2.10 Use Case 10: Add MakanSpot Menu Item

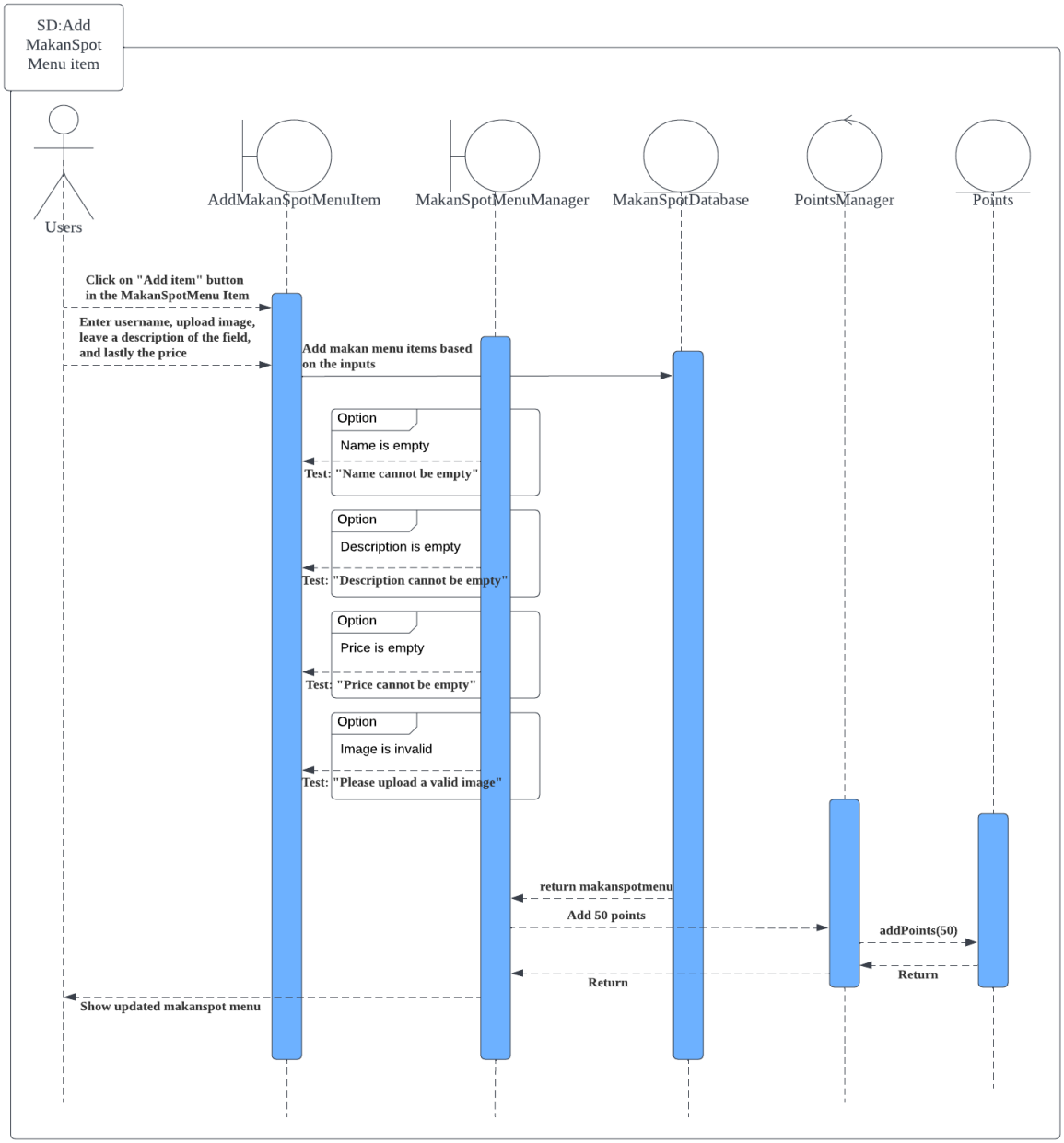
7.2.10.1 Use Case Description:

Use Case ID:	*10*		
Use Case Name:	*Add MakanSpot Menu item*		
Created By:	*Darren*	*Last Updated By:*	*Darren*
Date Created:	*8/18/2022*	*Date Last Updated:*	*8/18/2022*

Actor:	App users, database
Description:	Users are allowed to add Menu Items for Makanspots.
Preconditions:	There must be an existing MakanSpot.
Postconditions:	*MenuItem will be added to the menu of the MakanSpot*
Priority:	High
Frequency of Use:	Medium

Flow of Events:	To Add Makanspot Menu Item: 1. Users need to be in the HomePage to view the list of MakanSpots 2. Users will Tap on a MakanSpot on the HomePage to navigate to the MakanSpot Details Page. 3. Users can then tap the “View Menu” Button to view the list of menu and prices 4. Users can then Tap “Add Item” which will bring them to a page where they can key in information such as the picture of the food, description, as well as the price.
Alternative Flows:	Nil
Exceptions:	Nil
Includes:	*Nil*
Special Requirements:	*Nil*
Assumptions:	There is already an existing MakanSpot
Notes and Issues:	nil

7.2.10.2: Sequence Diagram



7.2.11 Use Case 11: Edit MakanSpot Menu Item

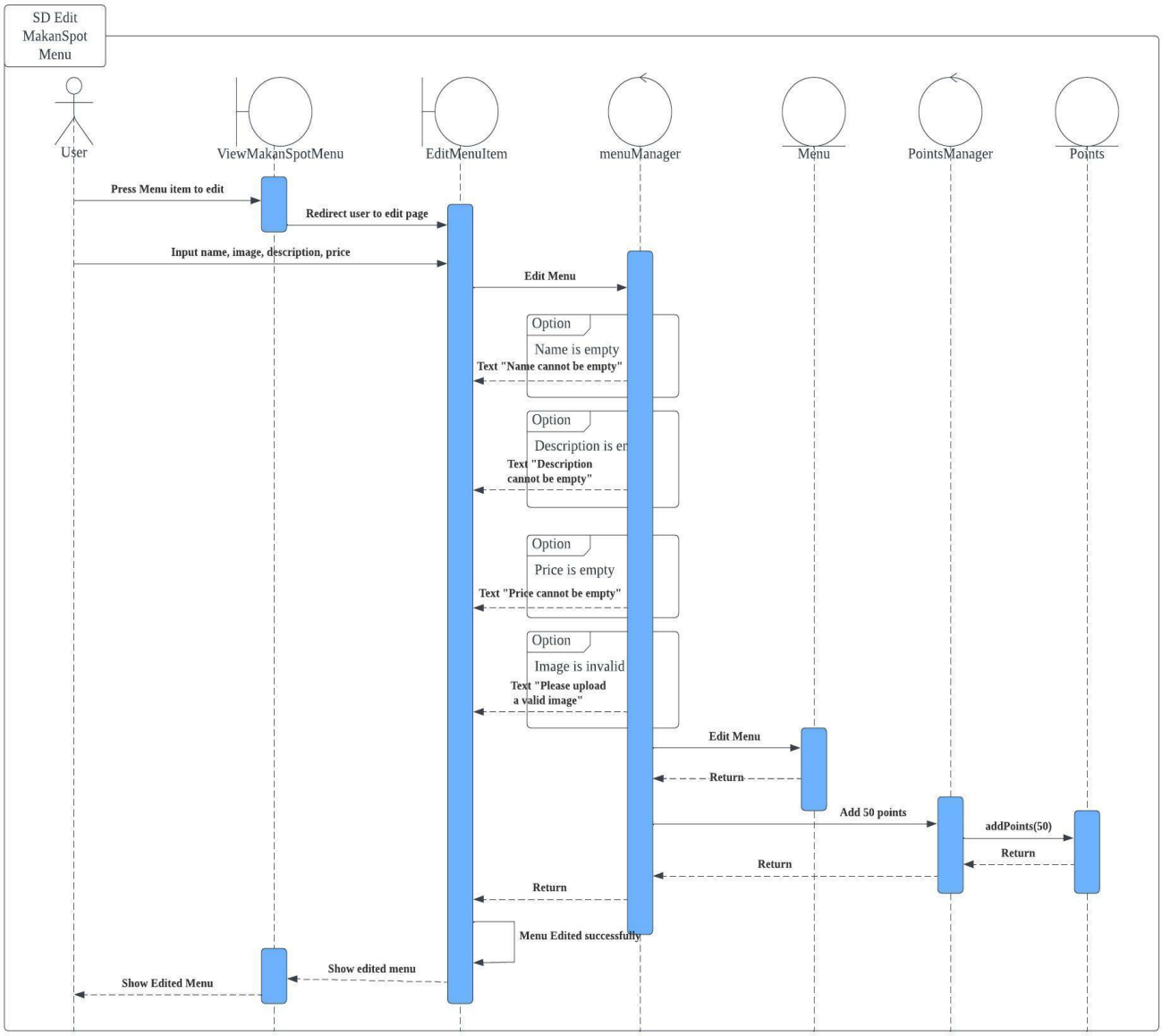
7.2.11.1 Use Case Description:

Use Case ID:	*11*		
Use Case Name:	*Edit MakanSpot Menu item*		
Created By:	*Darren*	*Last Updated By:*	*Davis*
Date Created:	*8/18/2022*	*Date Last Updated:*	*28/10/2022*

Actor:	App users, database
Description:	Users are allowed to edit Menu Items for Makanspots.
Preconditions:	There must be an existing MakanSpot.
Postconditions:	*The information of the menu items will be updated.*
Priority:	High
Frequency of Use:	Medium

Flow of Events:	To edit Makanspot Menu Item: 1. Users need to be in the HomePage to view the list of MakanSpots 2. Users will Tap on a MakanSpot on the HomePage to navigate to the MakanSpot Details Page. 3. Users can then tap the “View Menu” Button to view the list of menu and prices. 4. Users can tap on the different food items in the menu to edit Makanspot Menu Items.
Alternative Flows:	Nil
Exceptions:	Nil
Includes:	*Nil*
Special Requirements:	*Nil*
Assumptions:	There is already an existing MakanSpot and Menu
Notes and Issues:	nil

7.2.11.2 Sequence Diagram:



7.2.12 Use Case 12: View MakanSpot Reviews

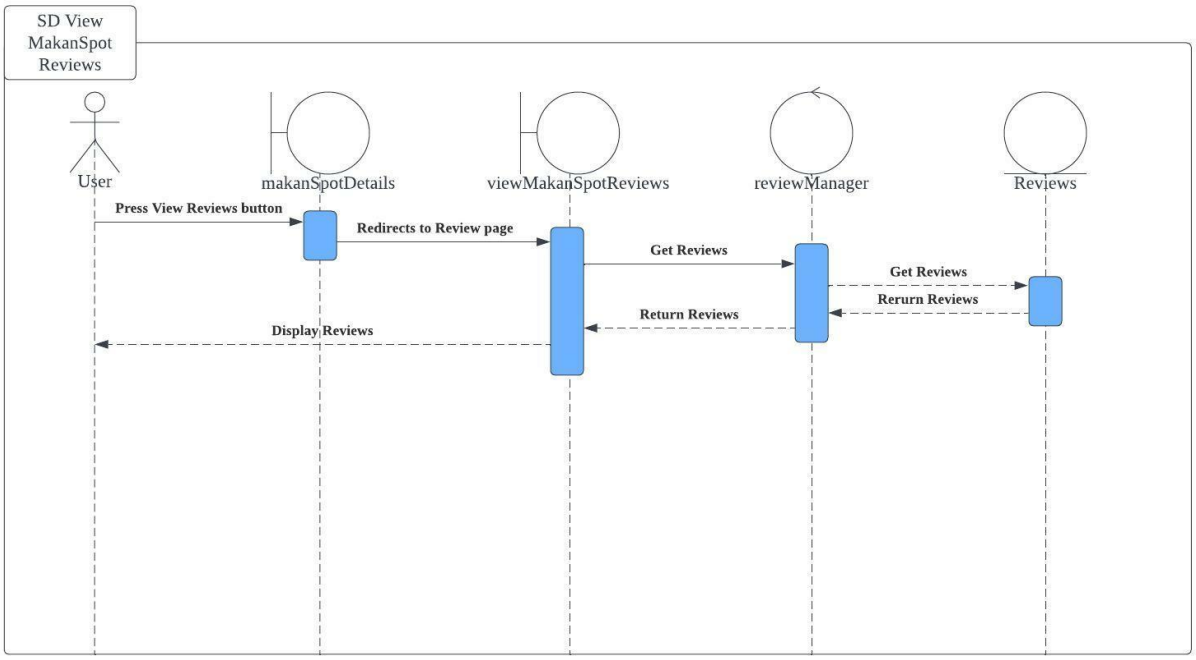
7.2.12.1 Use Case Description:

Use Case ID:	*12*		
Use Case Name:	*View MakanSpot Reviews*		
Created By:	*Darren*	*Last Updated By:*	*Davis*
Date Created:	*8/18/2022*	*Date Last Updated:*	*28/10/2022*

Actor:	App users, database
Description:	Users can view the average ratings and the list of reviews.
Preconditions:	There must be an existing MakanSpot and a list of reviews.
Postconditions:	*Ratings and reviews of that particular MakanSpot will be displayed.*
Priority:	Medium
Frequency of Use:	Medium

Flow of Events:	To view Makanspot Reviews: 1. Users need to be in the HomePage to view the list of MakanSpots 2. Users will Tap on a MakanSpot on the HomePage to navigate to the MakanSpot Details Page. 3. Users can then tap the “View Reviews” to see the list of reviews and average rating.
Alternative Flows:	Nil
Exceptions:	Nil
Includes:	*Nil*
Special Requirements:	*Nil*
Assumptions:	There is already an existing MakanSpot and users have already left a review.
Notes and Issues:	nil

7.2.12.2 Sequence Diagram:



7.2.13 Use Case 13: Review MakanSpot

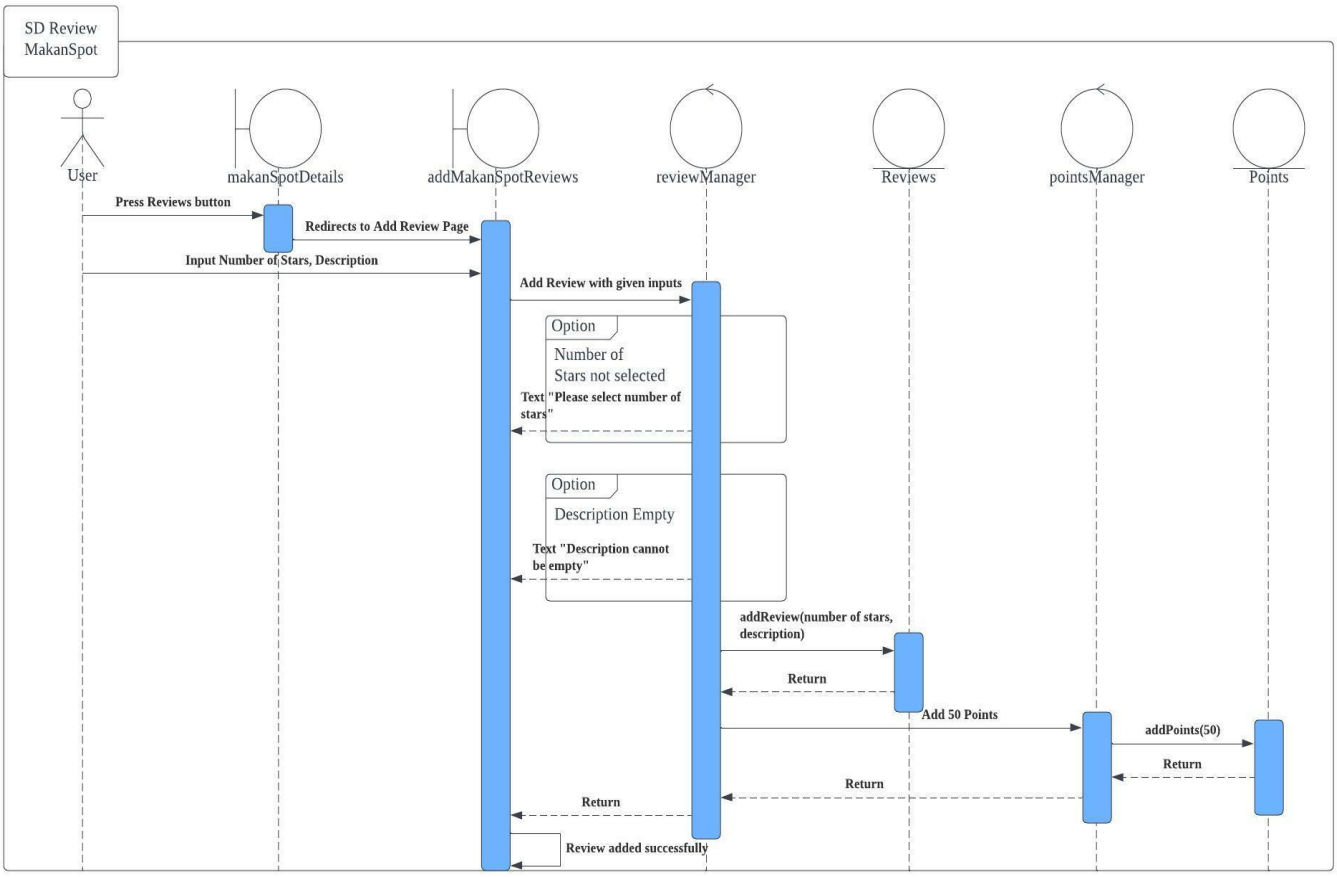
7.2.13.1 Use Case Description:

Use Case ID:	*13*		
Use Case Name:	*Review MakanSpot*		
Created By:	*Darren*	*Last Updated By:*	*Davis*
Date Created:	*8/18/2022*	*Date Last Updated:*	*28/10/2022*

Actor:	App users, database
Description:	Users can give ratings and leave comments on the makanspots.
Preconditions:	There must be an existing MakanSpot.
Postconditions:	The reviews can be viewed by the users and the average ratings of the makanspot will be updated.
Priority:	High
Frequency of Use:	Medium
Flow of Events:	To review makanspots: 1. Users need to be in the HomePage to view the list of MakanSpots 2. Users will Tap on a MakanSpot on the HomePage to navigate to the MakanSpot Details Page. 3. Users can then tap the button “Review”. 4. Users give a rating of the makanspot by clicking on the number of stars out of 5 stars. 5. Users leave some comments on the food places and click on the button “Submit Review”.
Alternative Flows:	Nil
Exceptions:	Nil

Includes:	*Nil*
Special Requirements:	*Nil*
Assumptions:	There is already an existing MakanSpot.
Notes and Issues:	nil

7.2.13.2 Sequence Diagram:



7.2.14 Use Case 14: Bring Me There!

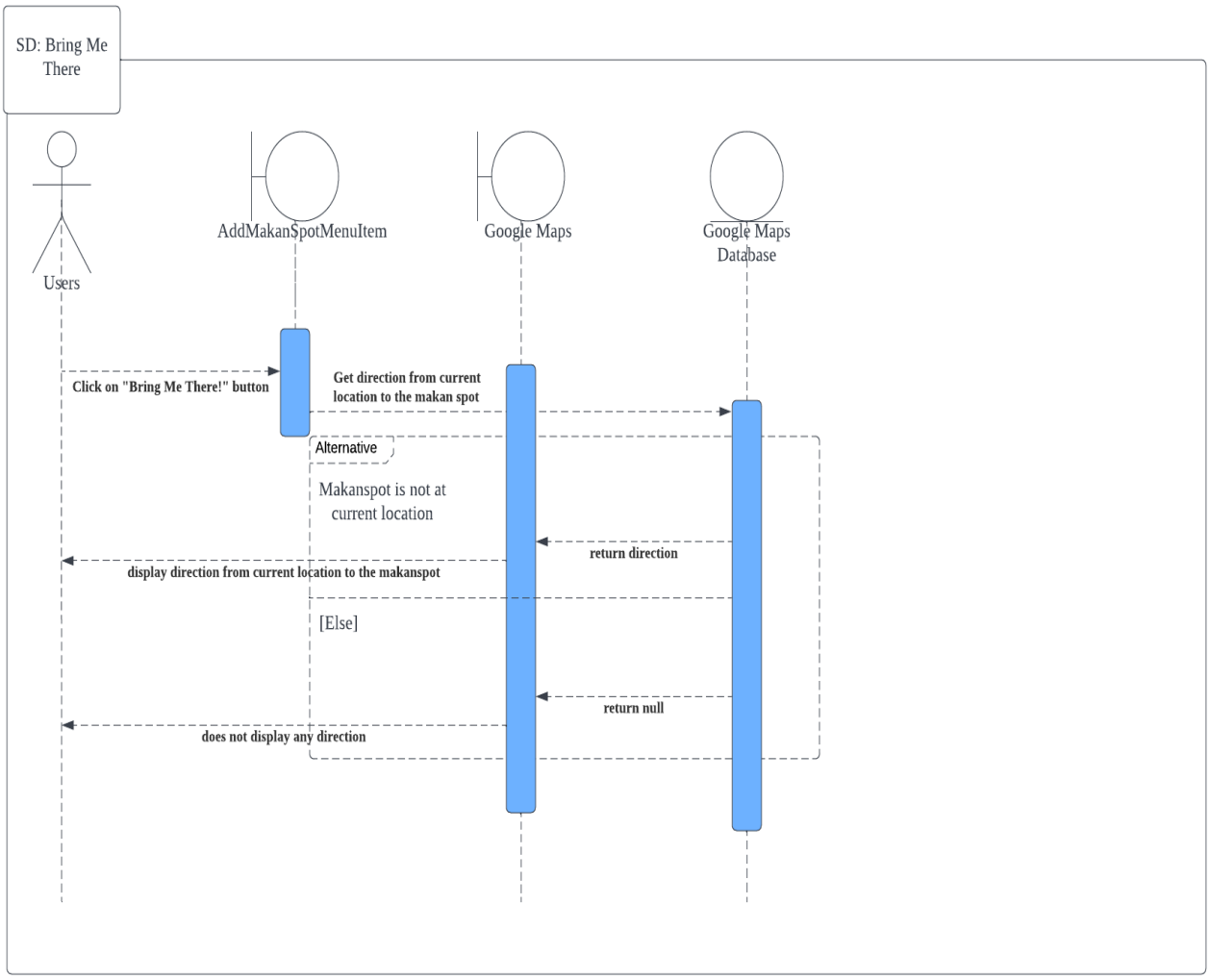
7.2.14.1 Use Case Description:

Use Case ID:	*14*		
Use Case Name:	Bring me There!		
Created By:	*Darren*	*Last Updated By:*	*Davis*
Date Created:	*8/18/2022*	*Date Last Updated:*	*28/10/2022*

Actor:	App users, database
Description:	Users can get the directions from their current location to the makanspot
Preconditions:	1. The makanspot must exist in the database.
Postconditions:	Users can find the direction from their current location to the makanspot.
Priority:	High

Frequency of Use:	High
Flow of Events:	1. Users need to be in the HomePage to view the list of MakanSpots 2. Users will Tap on a MakanSpot on the HomePage to navigate to the MakanSpot Details Page. 3. Users can tap on the button “Bring me There!” 4. Users are now directed to the google maps on their phone. 5. The direction of their current location to the makanspot is displayed.
Alternative Flows:	Nil
Exceptions:	If the user’s current location is at the makanspot, the direction to the makanspot will not be shown on the google map.
Includes:	Nil
Special Requirements:	Nil
Assumptions:	1. Users have google map installed on their phone 2. Users have enabled google maps to use their current location.
Notes and Issues:	nil

7.2.14.2: Sequence Diagram



7.2.15 Use Case 15: Start Chat

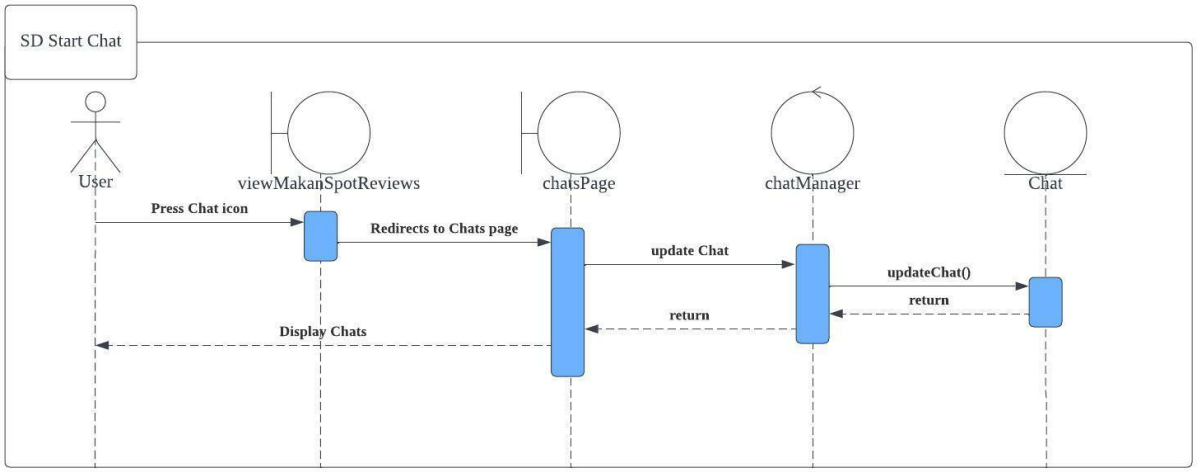
7.2.15.1 Use Case Description:

Use Case ID:	*15*		
Use Case Name:	*Start Chat*		
Created By:	*Darren*	*Last Updated By:*	*Darren*
Date Created:	*8/18/2022*	*Date Last Updated:*	*8/18/2022*

Actor:	App users
Description:	Users are able to connect with other foodies using the start chat function. Fromthe reviews page, users can see reviews left by others and can start a chat with them to find out more about their experiences at that makanspot
Preconditions:	Users must have reviewed a MakanSpot
Postconditions:	Users will be able to communicate with others using the Start Chat function
Priority:	*Med*

Frequency of Use:	Med
Flow of Events:	1. From the MakanSpots Details page, users will tap on “view reviews” to see the list of reviews left by other users. 2. Users can then tap on the chat icon to communicate with the user who left that review.
Alternative Flows:	Nil
Exceptions:	Nil
Includes:	*Nil*
Special Requirements:	Nil
Assumptions:	Other users have already left a review on that MakanSpot
Notes and Issues:	nil

7.2.15.2 Sequence Diagram:



7.2.16 Use Case 16: View User Profile (Light/ Dark Mode)

7.2.16.1 Use Case Description:

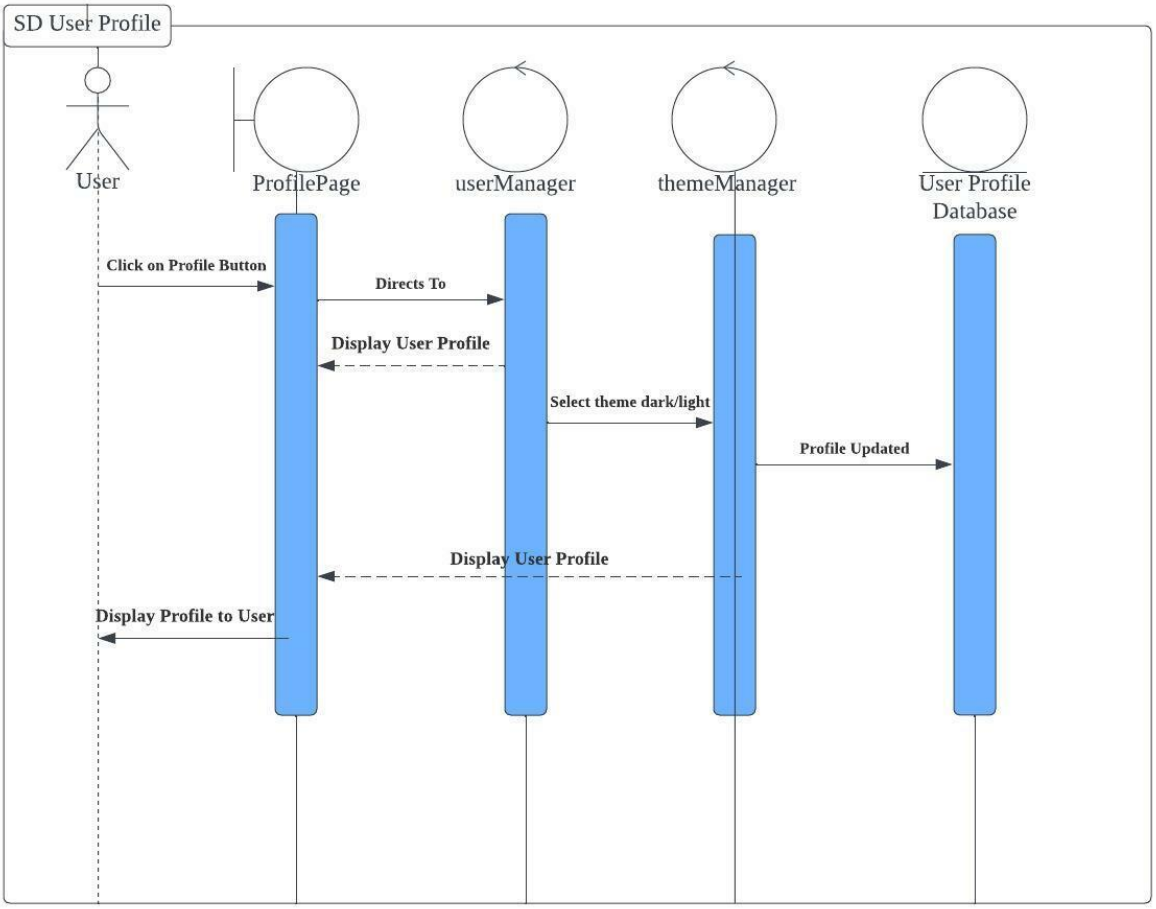
Use Case ID:	*16*		
Use Case Name:	*View User Profile(Light/Dark Mode)*		
Created By:	*Darren*	*Last Updated By:*	*Darren*
Date Created:	*8/18/2022*	*Date Last Updated:*	*8/18/2022*

Actor:	App users

Description:	Users are able to view their own profile in the profile page. They will be able to see their profile picture,name,email as well as the number of points they have. A dark and light mode also allows users to switch between different light settings to suit their desired preferences.
Preconditions:	Users will need to have an existing account.
Postconditions:	Nil
Priority:	*Med*
Frequency of Use:	Med
Flow of Events:	1. On the navigation bar, users can select the profile icon on the bottom right of their screen. 2. They can then proceed to their profile page where they can view their details.
Alternative Flows:	Nil
Exceptions:	Nil
Includes:	*Nil*
Special Requirements:	Nil

Assumptions:	*Users must be registered on MakanGoWhere*
Notes and Issues:	nil

7.2.16.2: Sequence Diagram



7.2.17 Use Case 17: Edit User Profile

7.2.17.1 Use Case Description:

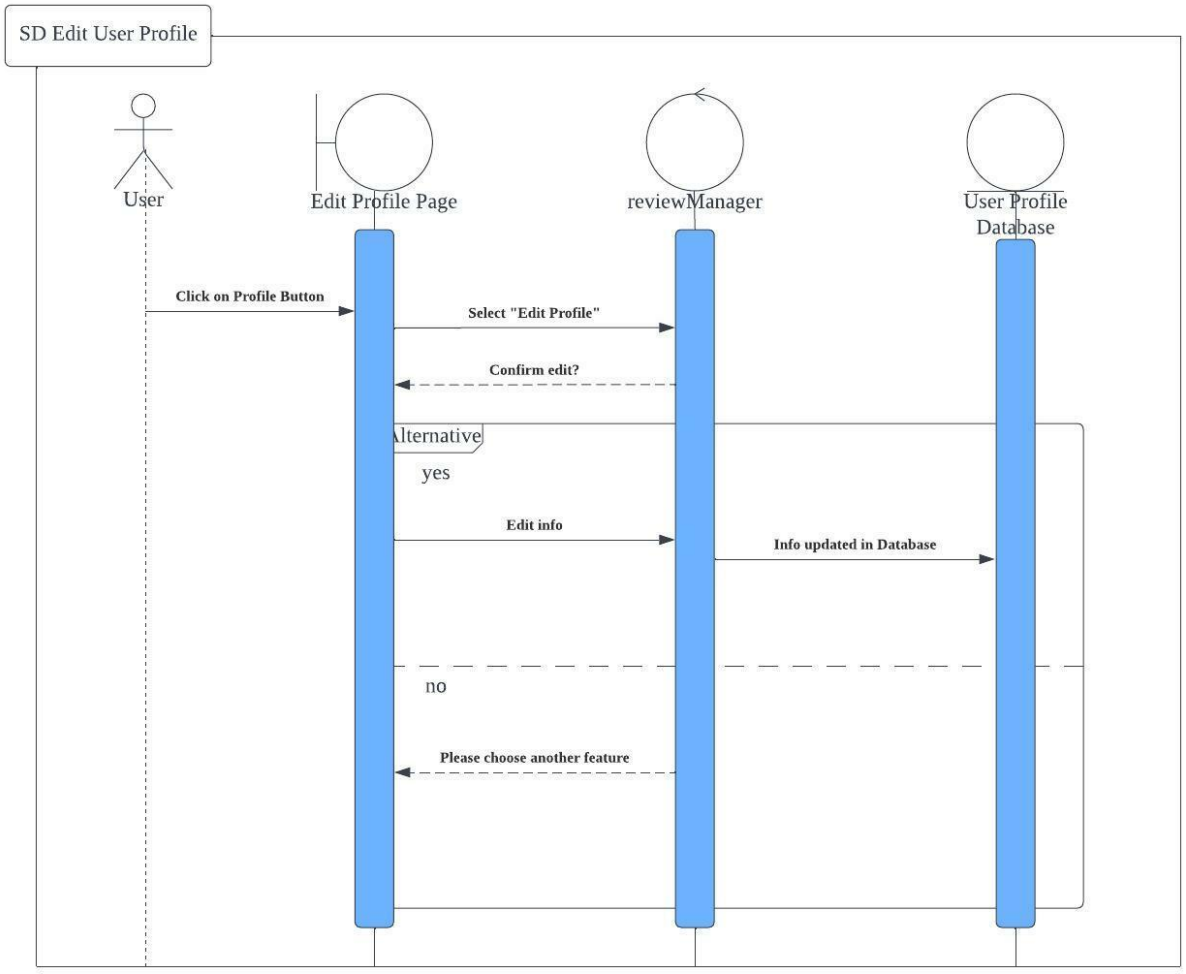
Use Case ID:	*17*

Use Case Name:	*Edit user profile*		
Created By:	*Darren*	*Last Updated By:*	*Davis*
Date Created:	*8/18/2022*	*Date Last Updated:*	*28/10/2022*

Actor:	App users
Description:	Users can edit their profile
Preconditions:	Users must have existing profiles
Postconditions:	Users can now change their user profile picture, email address, and their full name.
Priority:	*Medium*
Frequency of Use:	Med
Flow of Events:	1. Users click on the profile icon on the navigation bar. 2. Users click on “Edit Profile” 3. Users can now edit the following information: a. Email address b. GenjiSucks c. Profile picture

	4. Users click on the button “Save Changes”.
Alternative Flows:	Nil
Exceptions:	Nil
Includes:	*Nil*
Special Requirements:	Nil
Assumptions:	nil
Notes and Issues:	nil

7.2.17.2: Sequence Diagram



7.2.18 Use Case 18: Forget Password

7.2.18.1 Use Case Description:

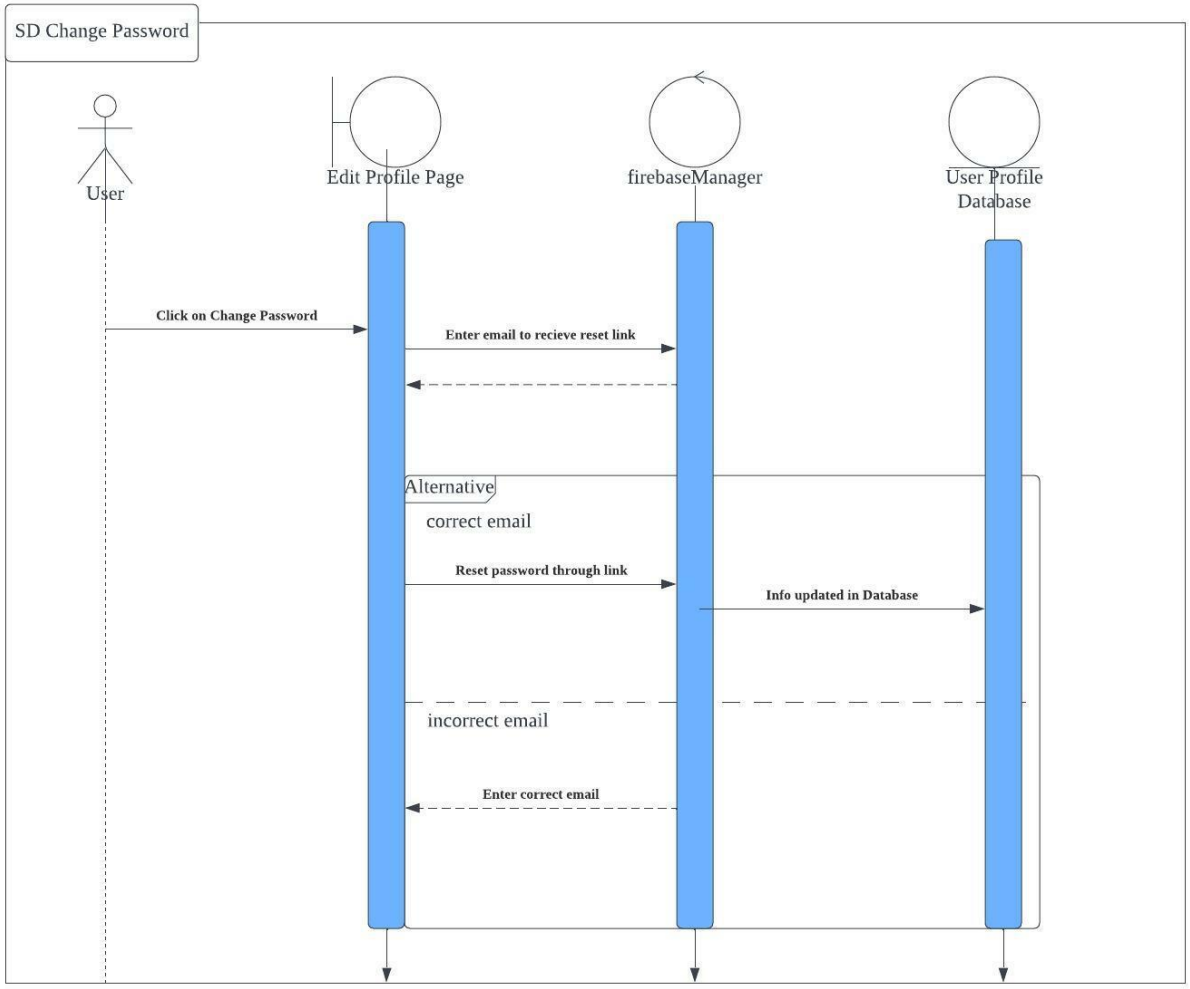
Use Case ID:	*18*
Use Case Name:	*Forget Password*

Created By:	*Darren*	*Last Updated By:*	*Davis*
Date Created:	*8/18/2022*	*Date Last Updated:*	*28/10/2022*

Actor:	App users,Database
Description:	Users can change their password
Preconditions:	Users must have existing profiles
Postconditions:	Users password will be changed
Priority:	*Medium*
Frequency of Use:	Med
Flow of Events:	1. From the profile page, users can tap on the “change password” button to change their password. 2. Users can then key in their email and press the “send reset link” 3. Users will then receive an email from our DataBase Server which would then allow them to change their password
Alternative Flows:	Nil

Exceptions:	Nil
Includes:	*Nil*
Special Requirements:	Nil
Assumptions:	*Users must already be registered on MakanGoWhere*
Notes and Issues:	nil

7.2.18.2 Sequence Diagram:



7.2.19 Use Case 19: View MakanPoints

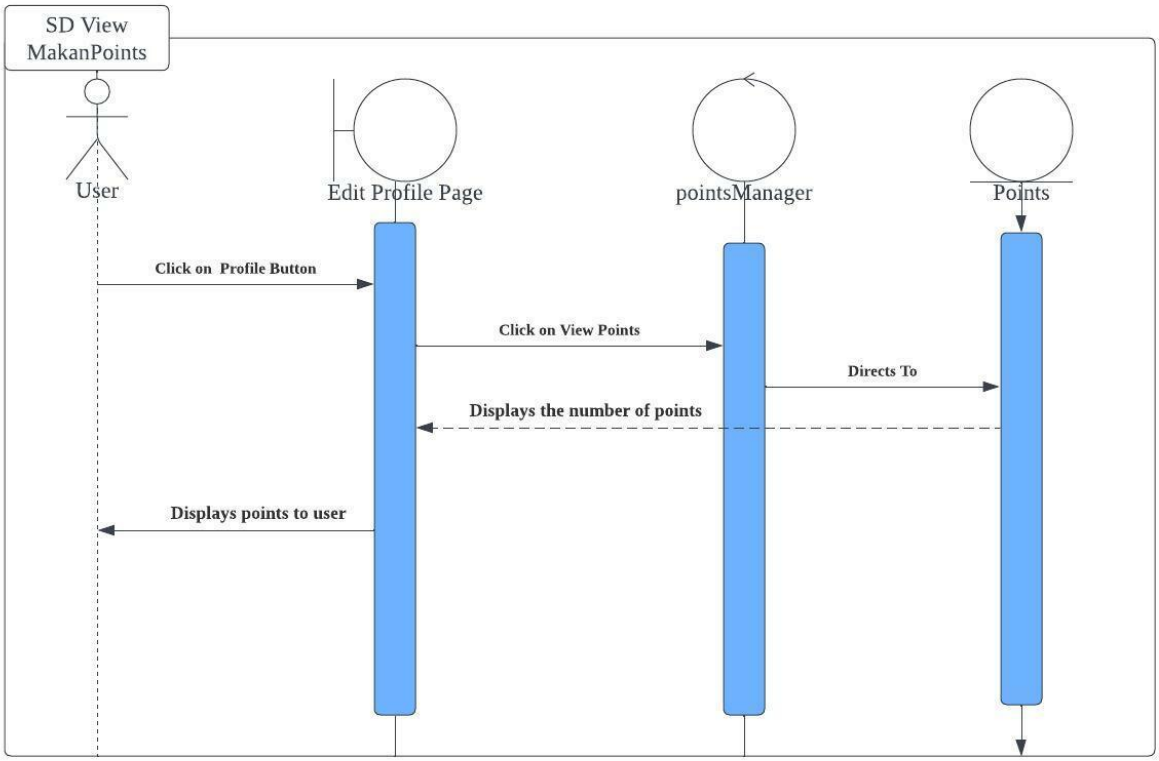
7.2.19.1 Use Case Description:

Use Case ID:	*19*		
Use Case Name:	View Makanpoints		
Created By:	*Aloysius*	*Last Updated By:*	*Davis*
Date Created:	*8/18/2022*	*Date Last Updated:*	*28/10/2022*

Actor:	App users, points system
Description:	Users are able to view how many points they have.
Preconditions:	Users must have an account.
Postconditions:	Users can see exactly the number of points they have
Priority:	Low
Frequency of Use:	Medium

Flow of Events:	1. Users click on the profile icon on the navigation bar. 2. Users click on “View Points” 3. A message “Hello {account name}, you have XXX points” will be displayed.
Alternative Flows:	Nil
Exceptions:	Nil
Includes:	Nil
Special Requirements:	Nil
Assumptions:	Nil
Notes and Issues:	nil

7.2.19.2 Sequence Diagram:



7.2.20 Use Case 20: Claim Rewards Using Makan Points

7.2.20.1 Use Case Description:

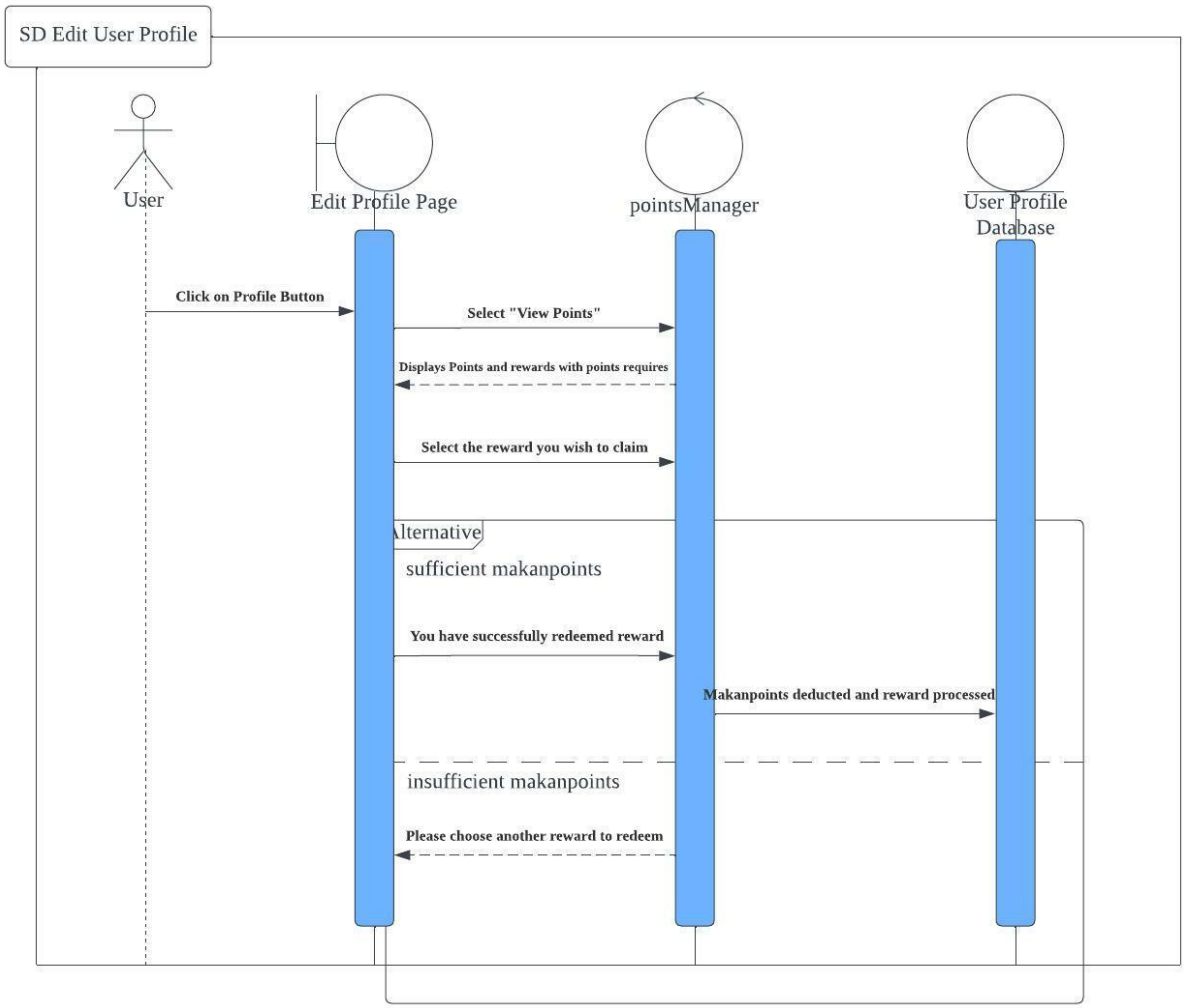
Use Case ID:	*20*		
Use Case Name:	Claim Rewards Using Makan Points		
Created By:	*Darren*	*Last Updated By:*	*Darren*

Date Created:	*8/18/2022*	*Date Last Updated:*	*28/10/2022*

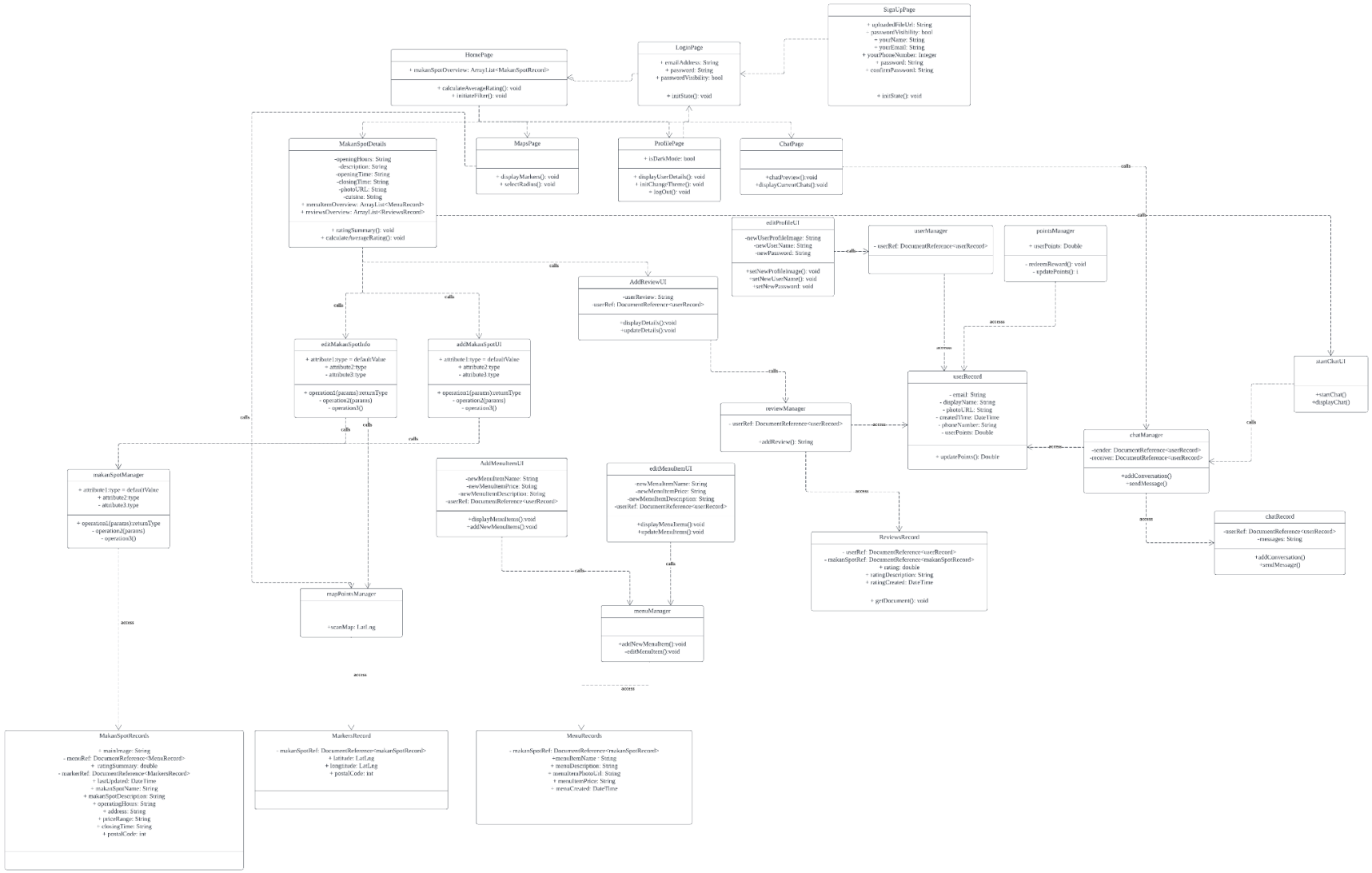
Actor:	App users, points system
Description:	Users are able to exchange available Makanpoints for rewards and deals
Preconditions:	Users have to acquire sufficient Makanpoints to redeem the rewards
Postconditions:	Users successfully redeem the rewards and they can use them for their next meal
Priority:	Low
Frequency of Use:	High
Flow of Events:	1. Users have to go to their profile page. There, they are able to see the number of Makanpoints they currently have 2. Select “view points”. All available rewards will be shown, along with the number of Makanpoints needed to claim them 3. Users will select the rewards they want. If they have sufficient Makanpoints, they will successfully claim the reward. If they do not have sufficient Makanpoints, no rewards would be shown

	4. Makanpoints will be deducted upon successful redemption
Alternative Flows:	If users have insufficient Makanpoints,no rewards will be shown.
Exceptions:	Nil
Includes:	
Special Requirements:	Users must have an account
Assumptions:	Users have points to redeem rewards
Notes and Issues:	nil

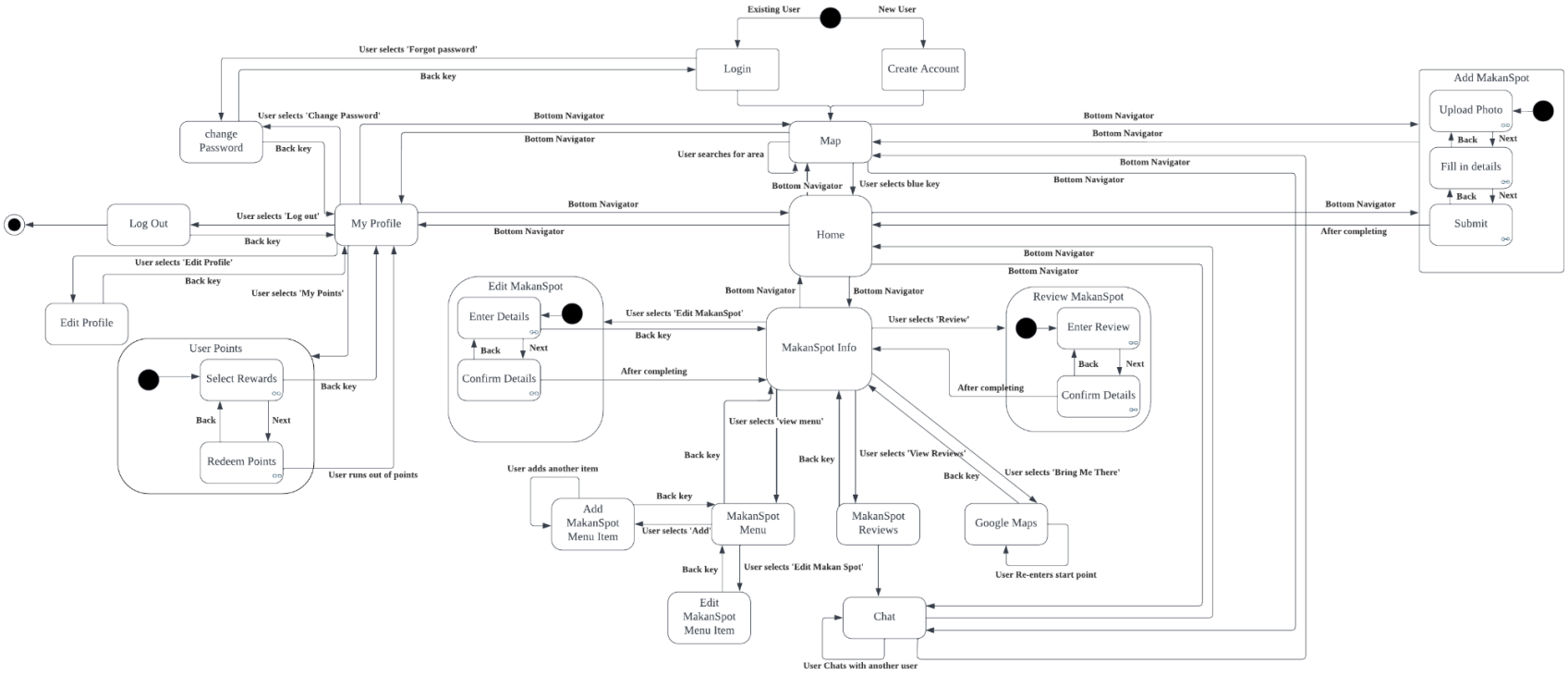
7.2.20.2 Sequence Diagram:



8. Class Diagram



9. Dialog Map


Existing User
New User
Login
Create Account
User selects 'Forgot password'
Back key
change Password
User selects 'Change Password'
Back key
Map
Bottom Navigator
User searches for area
User selects blue key
Add MakanSpot
Upload Photo
Back
Next
Fill in details
Back
Next
Submit
After completing
Log Out
User selects 'Log out'
Back key
My Profile
Home
User selects 'Edit Profile'
Back key
User selects 'My Points'
Edit Profile
User Points
Select Rewards
Back
Next
Redeem Points
Back key
User runs out of points
Edit MakanSpot
Enter Details
Back
Next
Confirm Details
User selects 'Edit MakanSpot'
Back key
After completing
MakanSpot Info
Review MakanSpot
Enter Review
Back
Next
Confirm Details
User selects 'Review'
After completing
User selects 'view menu'
Back key
User selects 'View Reviews'
Back key
User selects 'Bring Me There'
Back key
User adds another item
Add MakanSpot Menu Item
User selects 'Add'
MakanSpot Menu
MakanSpot Reviews
Google Maps
User Re-enters start point
Back key
User selects 'Edit Makan Spot'
Edit MakanSpot Menu Item
Chat
User Chats with another user

10. Software Engineering Principles

10.1 Software Engineering Practices

10.1.1 KISS PRINCIPLE

- Kept the code as simple and as readable as possible
- High maintainability of code, could be easily changed whenever a bug occurred
- Code can be extended for implementation of additional functionalities.

10.1.2 AGILE PRINCIPLE

- Weekly meetings where group leader acts as the client
- Client will specify additional requirements to be included in each update
- We will then work on only implementing the functionalities that meets the client's requirements. (YAGNI principle)
- Started on simple and basic features before adding more complicated features:
    - Chat System
    - Bring me there (navigation)

10.1.3 YAGNI PRINCIPLE

- You Aren't Gonna Need It is a practice in software development which states that features should only be added when required.
- First started with basic functions such as AddMakanSpot, EditMakanSpot and ProfilePage that achieved the Minimal Viable Product (MVP) standards
- Added on functionalities that we feel will benefit the usability of the app only in later stages

10.1.4 DRY PRINCIPLE

- Don't Repeat Yourself tells us that duplicate code leads to more code, and more code is harder and more costly to maintain
- The main purpose is to reduce the repetition of patterns and code duplication in favour of abstractions and avoiding redundancy
- In our context, we divided our code into smaller dart files and we can use the smaller code by importing those files and calling them when necessary

10.2 Software Engineering Design Principles

10.2.1 Model-View-Controller(MVC)

- Model represents an object or JAVA POJO carrying data. It can also have logic to update controllers if its data changes.
- View represents the visualisation of the data that model contains.
- Controller acts on both model and view. It controls the data flow into the model object and updates the view whenever data changes. It keeps view and model separate.
- In our context, for example, we implemented a Menu object acting as a model.
- We have a View Menu function in the app that allows the users to view the list of food/dishes of a MakanSpot
- We have a MenuManager class which acts as a controller to demonstrate MVC design pattern usage.

11. Testing

11.1 Black Box Testing

11.1.1 Login Function

According to our functional requirements, for the users to login successfully, users have to key in a valid email address and the correct password. To do black box testing on this function, we tested with multiple inputs and noted the results. According to our functional requirements, the application must reject any login attempts with badly formatted email, invalid email, or incorrect passwords.

Equivalence classes:

Input data	Valid	Invalid
email	Any valid email	Empty or badly formatted email, such as inputs without '@'
password	The correct password of that account	Missing characters, wrong capitalisation on letters

Test cases:

Test Cases	Inputs	Expected results	Actual results
Case 1: Correct Login details	"davidlee12@gmail.com" "Password123!"	Login successful	Login successful
Case 2: Email does not exist	"jojoba@gmail.com" "Password123!"	Login unsuccessful	Login unsuccessful "There is no user record corresponding to this identifier. The user may have been deleted"
Case 3: Badly formatted email without '@'	"davidlee12gmail.com" "Password123!"	Login unsuccessful	Login unsuccessful "The email address is badly formatted"
Case 4: Badly formatted email without domain	"davidlee@" "Password123!"	Login unsuccessful	Login unsuccessful "The email address is badly formatted"

Case 5: Wrong password with missing characters	“davidlee12@gmail.com” “Password12”	Login unsuccessful	Login unsuccessful “The password is invalid or the user does not have a password”
Case 6: Wrong password with wrong capitalisation on some characters	“davidlee12@gmail.com” “password123!”	Login unsuccessful	Login unsuccessful “The password is invalid or the user does not have a password”
Case 7: Empty email or password domain	“” “”	Login unsuccessful	Login unsuccessful “Error: Given String is empty or null”

11.1.2 Create Account

According to 1.1 of our functional requirements, to create an account, the application must accept a valid email, username and password when a user registers. The email should not be in use by other users as well. To do black box testing on this function, we tested with multiple inputs and noted the results. According to our functional requirements, the application must reject any registration with missing field information, badly formatted email, emails that are already in use and invalid password inputs. To test whether the email is already in use, we will use the email used in 11.1.1 black box testing.

Equivalence classes:

Input data	Valid	Invalid
Username	Any string input	Leaving the field empty
Email	Any valid email	Empty or badly formatted email, such as inputs without ‘@’
Phone Number	Any 8 digit number	Numbers less than 8 digits, numbers less than 60,000,000 and more than 99,999,999
Password	Any string input more than 6	Input less than 6 characters

	characters	
Confirm Password	The same as the password the user chose	Input different from the ‘password’ input

Test cases:

Test Cases	Inputs	Expected results	Actual results
Case 1: Correct input details	“David Lee” “davidlee12@gmail.com” “98327878” “Password123!” “Password123!”	Create account successful	Create account successful
Case 2: Badly formatted email without ‘@’	“David Lee” “davidlee12gmail.com” “98327878” “Password123!” “Password123!”	Create account unsuccessful	Create account unsuccessful “The email address is badly formatted”
Case 3: Badly formatted email without domain	“David Lee” “davidlee12@” “98327878” “Password123!” “Password123!”	Create account unsuccessful	Create account unsuccessful “The email address is badly formatted”
Case 4: Password less than 6 characters	“David Lee” “davidlee12@gmail.com” “98327878” “Pas” “Pas”	Create account unsuccessful	Create account unsuccessful “Error: Password should be at least 6 characters”
Case 5: Password and Confirm Password are not the same	“David Lee” “davidlee12@gmail.com” “98327878” “Password12!” “Passssssssss”	Create account unsuccessful	Create account unsuccessful “Passwords don’t match!”
Case 6: Missing information	“” “” “98327878” “Password12!” “”	Create account unsuccessful	Create account unsuccessful “Error: given string is empty or null”

Case 7: Phone number invalid (numbers less than 60,000,000 and more than 99,999,999	“59,999,999” “22” “100,000,000”	Create account unsuccessful	Create account unsuccessful “Error: phone number is invalid”

11.1.3 Edit Profile

According to our functional requirements, users are able to edit their user name, profile photo or email.The application must accept a valid email and the username domain must not be empty. The email should not be in use by other users as well. Users are allowed not to have a profile picture, however, their profile picture will be a cross if that happens. To do black box testing on this function, we tested with multiple inputs and noted the results. According to our functional requirements, the application must reject any change attempts with badly formatted email, emails that are already in use and empty username domain. To test whether the email is already in use, we will use the email used in 11.1.1 black box testing.

Equivalence classes:

Input data	Valid	Invalid
Username	Any string input	Leaving the field empty
email	Any valid email	Empty or badly formatted email, such as inputs without ‘@’
Profile Picture	Any pictures selected from gallery/ camera/ no selection	Invalid file type

Test cases:

Test Cases	Inputs	Expected results	Actual results
Case 1: Correct input details Image uploaded	“David Lee” “davidlee12@gmail.com” ‘Picture’	Account successfully edited	Account successfully edited
Case 2: Badly formatted email	“David Lee” “Davidlee12gmail.co	Account cannot be edited successfully	Account cannot be edited successfully

without '@' Image uploaded	m" 'Picture'		"The email address is badly formatted"
Case 3: Badly formatted email without domain Image uploaded	"David Lee" "davidlee12@" 'Picture'	Account cannot be edited successfully	Account cannot be edited successfully "The email address is badly formatted"
Case 4: Username or email domain left blank Image uploaded	"" "" 'Picture'	Account cannot be edited successfully	Account cannot be edited successfully "Error: Given string is empty or null"
Case 5: File type is invalid	User tries to submit a pdf document instead of a photo	Account cannot be edited successfully	Account cannot be edited successfully "File type is unsupported"
Case 6: Only profile photo not provided No image uploaded	"David Lee" "davidlee12@gmail.com"	Account successfully edited	Account successfully edited. Profile picture replaced with a cross telling users they did not submit a profile picture

11.1.4 Add MakanSpot Menu Item

According to our functional requirements, users are able to add MakanSpot menu items if some menu items are missing, and edit existing menu items should they contain any errors. To do black box testing on this function, we tested with multiple inputs and noted the results. According to our functional requirements, the application must reject the change if there are any fields that are missing information.

Equivalence classes:

Input data	Valid	Invalid
Name of item	Any string input	Leaving the field empty
Image	Any valid email	Invalid file type
Item description	Any string input	Leaving the field empty
Item price	Any integer value	Leaving the field empty or non-integer input given

Test cases:

Test Cases	Inputs	Expected results	Actual results
Case 1: Correct input details Image uploaded	“Chicken Burger” Image of item “Juicy chicken burger” “9”	Menu item successfully added	Menu item successfully added
Case 2: Any field left empty	“” Image “Juicy chicken burger” “”	Menu item cannot be added	Menu item cannot be added. “Given string is empty or null”
Case 3: Invalid image file	“Chicken Burger” Invalid Image file of item “Juicy chicken burger” “9”	Menu item cannot be added	Menu item cannot be added “File type is unsupported”
Case 4: Price not an integer	“Chicken Burger” Image of item “Juicy chicken burger” “Hello”	Menu item cannot be added	Menu item cannot be added “Invalid input data type”

11.1.5 Edit MakanSpot Menu Item

According to our functional requirements, users are able to add MakanSpot menu items if some menu items are missing, and edit existing menu items should they contain any errors. To do black box testing on this function, we tested with multiple inputs and noted the results. According to our functional requirements, the application must reject the change if there are any fields that are missing information.

Equivalence classes:

Input data	Valid	Invalid
Name of item	Any string input	Leaving the field empty
Image	Any valid email	Invalid file type
Item description	Any string input	Leaving the field empty

Item price	Any integer value	Leaving the field empty or non-integer input given

Test cases:

Test Cases	Inputs	Expected results	Actual results
Case 1: Correct input details Image uploaded	“Chicken Burger” Image of item “Juicy chicken burger” “9”	Menu item edited successfully	Menu item edited successfully
Case 2: Any field left empty	“” Image “Juicy chicken burger” “”	Menu item cannot be edited	Menu item cannot be edited. “Given string is empty or null”
Case 3: Invalid image file	“Chicken Burger” Invalid Image file of item “Juicy chicken burger” “9”	Menu item cannot be edited	Menu item cannot be edited “File type is unsupported”
Case 4: Price not an integer	“Chicken Burger” Image of item “Juicy chicken burger” “Hello”	Menu item cannot be edited	Menu item cannot be edited “Invalid input data type”

11.1.6 Add MakanSpot Review

According to our functional requirements, users are able to add reviews for MakanSpots. To do black box testing on this function, we tested with multiple inputs and noted the results. According to our functional requirements, the application must reject the reviews with missing pieces of information or if the description is too long.

Equivalence classes:

Input data	Valid	Invalid
Number of stars	Select the number of stars,	Leaving the field empty

	from 1 up to 5 stars	
Description	Any string input	String with more than 50 characters

Test cases:

Test Cases	Inputs	Expected results	Actual results
Case 1: Correct input details	5 Stars selected “Service and food was good!”	Review successfully added	Review successfully added
Case 2: Users did not select the number of stars	No stars selected “Service and food was good!”	Review cannot be added	Review cannot be added. Please select the number of stars
Case 3: No description provided	3 Stars selected “”	Review cannot be added	Review cannot be added “Given string is empty or null”

11.1.7 Add MakanSpot

According to our functional requirements, users are able to add MakanSpot to the database if the MakanSpot is currently not in the database. To do black box testing on this function, we tested with multiple inputs and noted the results. According to our functional requirements, the application must reject the change if there are any fields that are missing information, or if the information provided is an invalid type.

Equivalence classes:

Input data	Valid	Invalid
Image	Any valid email	Invalid file type
Name of MakanSpot	Any string input	Leaving the field empty
Address	Any string input	Leaving the field empty
Postal code	Any integer input with 6 digits	Leaving the field empty or giving a non-integer value or giving a number with less than 6 digits

Opening time and Closing time	Any choice displayed by the drop down button	Leaving the field empty
MakanSpot description	Any string input	Leaving the field empty
Cuisine	Any choice displayed by the drop down button (Japanese, Korean, Chinese, Malay, Indian, Western)	Leaving the field empty
Price range	Any choice displayed by the drop down button ($, $$, $$$)	Leaving the field empty
Dietary preference	If the MakanSpot has a particular dietary preference, then select the option. Default value is false, thus this input is optional	-

Test cases:

Test Cases	Inputs	Expected results	Actual results
Case 1: Correct input details Image uploaded	Image Uploaded “KFC” “NTU North Spine” “637331” 12 00, 20 00 “KFC Fried Chicken” Western $$ Halal	MakanSpot successfully added	MakanSpot successfully added
Case 2: Dietary Preference left empty Image uploaded	Image Uploaded “KFC” “NTU North Spine” “637331” 12 00, 20 00 “KFC Fried Chicken” Western $$	MakanSpot successfully added	MakanSpot successfully added
Case 3: Any other field left empty Image uploaded	Image Uploaded “” “NTU North Spine” “637331” 12 00, “KFC Fried Chicken”	MakanSpot cannot be added	MakanSpot cannot be added. “Some fields required are empty”

	$$ Halal		
Case 3: Invalid image file	Invalid Image file “KFC” “NTU North Spine” “637331” 12 00, 20 00 “KFC Fried Chicken” Western $$ Halal	MakanSpot cannot be added	MakanSpot cannot be added “File type is unsupported”
Case 4: Non-integer given for postal code	Image Uploaded “KFC” “NTU North Spine” “Hello” 12 00, 20 00 “KFC Fried Chicken” Western $$ Halal	MakanSpot cannot be added	MakanSpot cannot be added “Invalid input data type for postal code”
Case 5: Not a 6 digit number given for postal code	Image Uploaded “KFC” “NTU North Spine” “600” 12 00, 20 00 “KFC Fried Chicken” Western $$ Halal	MakanSpot cannot be added	MakanSpot cannot be added “Postal code must be 6 digits”

11.1.8 Edit MakanSpot

According to our functional requirements, users are able to edit MakanSpot if they find any information incorrect. To do black box testing on this function, we tested with multiple inputs and noted the results. According to our functional requirements, the application must reject the change if there are any fields that are missing information, or if the information provided is an invalid type.

Equivalence classes:

Input data	Valid	Invalid

Image	Any valid email	Invalid file type
Name of MakanSpot	Any string input	Leaving the field empty
Address	Any string input	Leaving the field empty
Postal code	Any integer input with 6 digits	Leaving the field empty or giving a non-integer value or giving a number with less than 6 digits
Opening time and Closing time	Any choice displayed by the drop down button	Leaving the field empty
MakanSpot description	Any string input	Leaving the field empty
Cuisine	Any choice displayed by the drop down button (Japanese, Korean, Chinese, Malay, Indian, Western)	Leaving the field empty
Price range	Any choice displayed by the drop down button ($, $$, $$$)	Leaving the field empty
Dietary preference	If the MakanSpot has a particular dietary preference, then select the option. Default value is false, thus this input is optional	-

Test cases:

Test Cases	Inputs	Expected results	Actual results
Case 1: Correct input details Image uploaded	Image Uploaded “KFC” “NTU North Spine” “637331” 12 00, 20 00 “KFC Fried Chicken” Western $$ Halal	MakanSpot successfully edited	MakanSpot successfully edited
Case 2: Dietary Preference left empty Image uploaded	Image Uploaded “KFC” “NTU North Spine” “637331”	MakanSpot successfully edited	MakanSpot successfully edited

	12 00, 20 00 “KFC Fried Chicken” Western $$		
Case 3: Any other field left empty Image uploaded	Image Uploaded “” “NTU North Spine” “637331” 12 00, “KFC Fried Chicken” $$ Halal	MakanSpot cannot be edited	MakanSpot cannot be edited. “Some fields required are empty”
Case 3: Invalid image file	Invalid Image file “KFC” “NTU North Spine” “637331” 12 00, 20 00 “KFC Fried Chicken” Western $$ Halal	MakanSpot cannot be edited	MakanSpot cannot be edited “File type is unsupported”
Case 4: Non-integer given for postal code	Image Uploaded “KFC” “NTU North Spine” “Hello” 12 00, 20 00 “KFC Fried Chicken” Western $$ Halal	MakanSpot cannot be edited	MakanSpot cannot be edited “Invalid input data type for postal code”
Case 5: Not a 6 digit number given for postal code	Image Uploaded “KFC” “NTU North Spine” “600” 12 00, 20 00 “KFC Fried Chicken” Western $$ Halal	MakanSpot cannot be added	MakanSpot cannot be added “Postal code must be 6 digits”

11.1.9 Filter MakanSpots

According to our functional requirements, users are able to filter MakanSpots based on their budget. To do black box testing on this function, we tested with multiple inputs and noted the results. According to our functional requirements, the application must be able to filter MakanSpots based on price using the number of '$'. '$' means price less than $10, '$$' means price between $10 and $50, and '$$$' means price more than $50. If the user did not select anything, that means he did not use the filter.

Equivalence classes:

Input data	Valid	Invalid
Number of $	Select the number of $, from 1 up to 3 $	-

Test cases:

Test Cases	Inputs	Expected results	Actual results
Case 1: 1 $	$	Only MakanSpots with 1 $ will show	Only MakanSpots with 1 $ shown
Case 2: 2 $	$$	Only MakanSpots with 2 $ will show	Only MakanSpots with 2 $ shown
Case 3: 3 $	$$$	Only MakanSpots with 3 $ will show	Only MakanSpots with 3 $ shown
Case 4: All $ selected	$ $$ $$$	All MakanSpots will show	All MakanSpots shown

11.2 Boundary Value Testing

11.2.1 Boundary Value Testing - Create Account

Test Case	Lower Boundary (Invalid)	Lower Boundary (Valid)	Upper Boundary (Valid)	Upper Boundary (InValid)
Number of Images (x≦1)	NA	0	1	2

UserName (1≤ x≤ 100)	0	1	100	101
Phone Number (60,000,000≤ x≤ 99,999,999)	59,999,999	60,000,000	99,999,999	100,000,000

11.2.2 Boundary Value Testing - Add/ edit MakanSpot

Test Case	Lower Boundary (Invalid)	Lower Boundary (Valid)	Upper Boundary (Valid)	Upper Boundary (InValid)
Number of Images (x≦1)	NA	0	1	2
MakanSpot Name (1≤ x≤ 100)	0	1	100	101
Postal Code (60,000,000≤ x≤ 99,999,999)	59,999,999	60,000,000	99,999,999	100,000,000
Description (1≤ x ≤ 500	0	1	500	501

11.2.3 Boundary Value Testing - Add/ Edit Menu Item

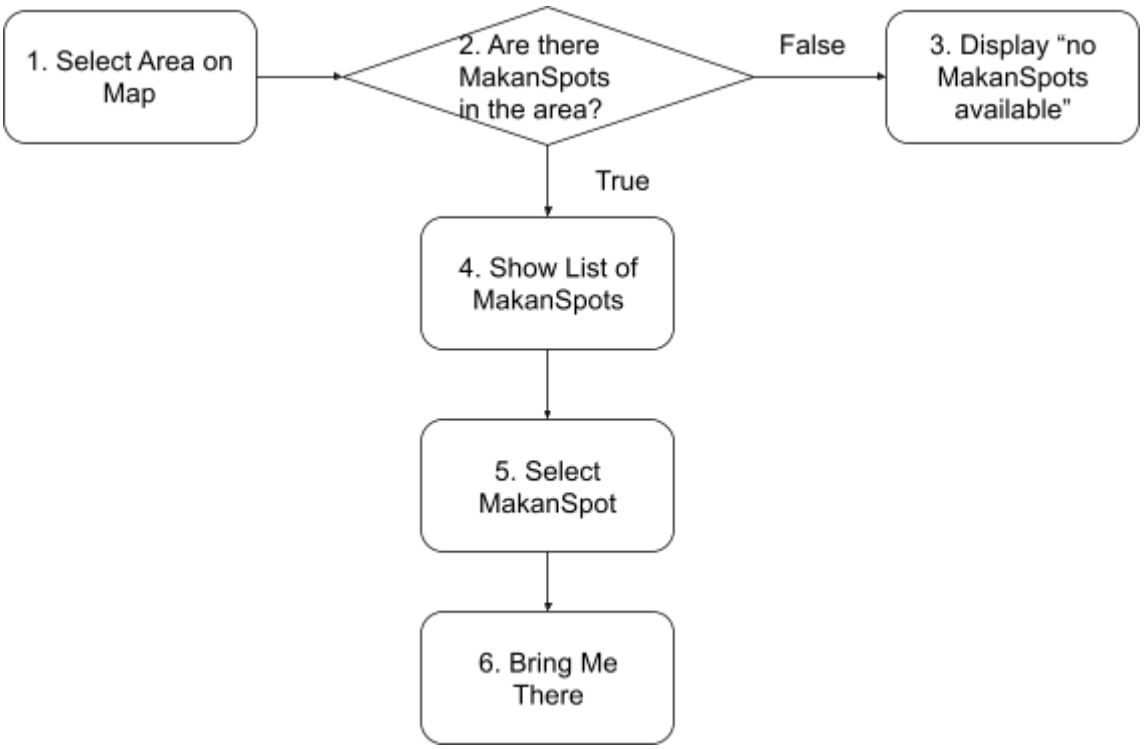
Test Case	Lower Boundary (Invalid)	Lower Boundary (Valid)	Upper Boundary (Valid)	Upper Boundary (InValid)
Number of Images (x≦1)	NA	0	1	2
Menu Item Name (1≤ x≤ 100)	0	1	100	101
Description (1≤ x ≤ 500	0	1	500	501
Price of item (x≥ 0)	-1	0,1	NA	NA

11.2.4 Boundary Value Testing - Add Review

Test Case	Lower Boundary (Invalid)	Lower Boundary (Valid)	Upper Boundary (Valid)	Upper Boundary (InValid)
Number of Stars (1 ≤ x ≤ 5)	0	1	5	6
Description (1≤ x ≤ 500	0	1	500	501

11.3 White Box Testing

11.3.1 Control Flow Test: BringMeThere to MakanSpot



11.3.1.1 Basis paths:

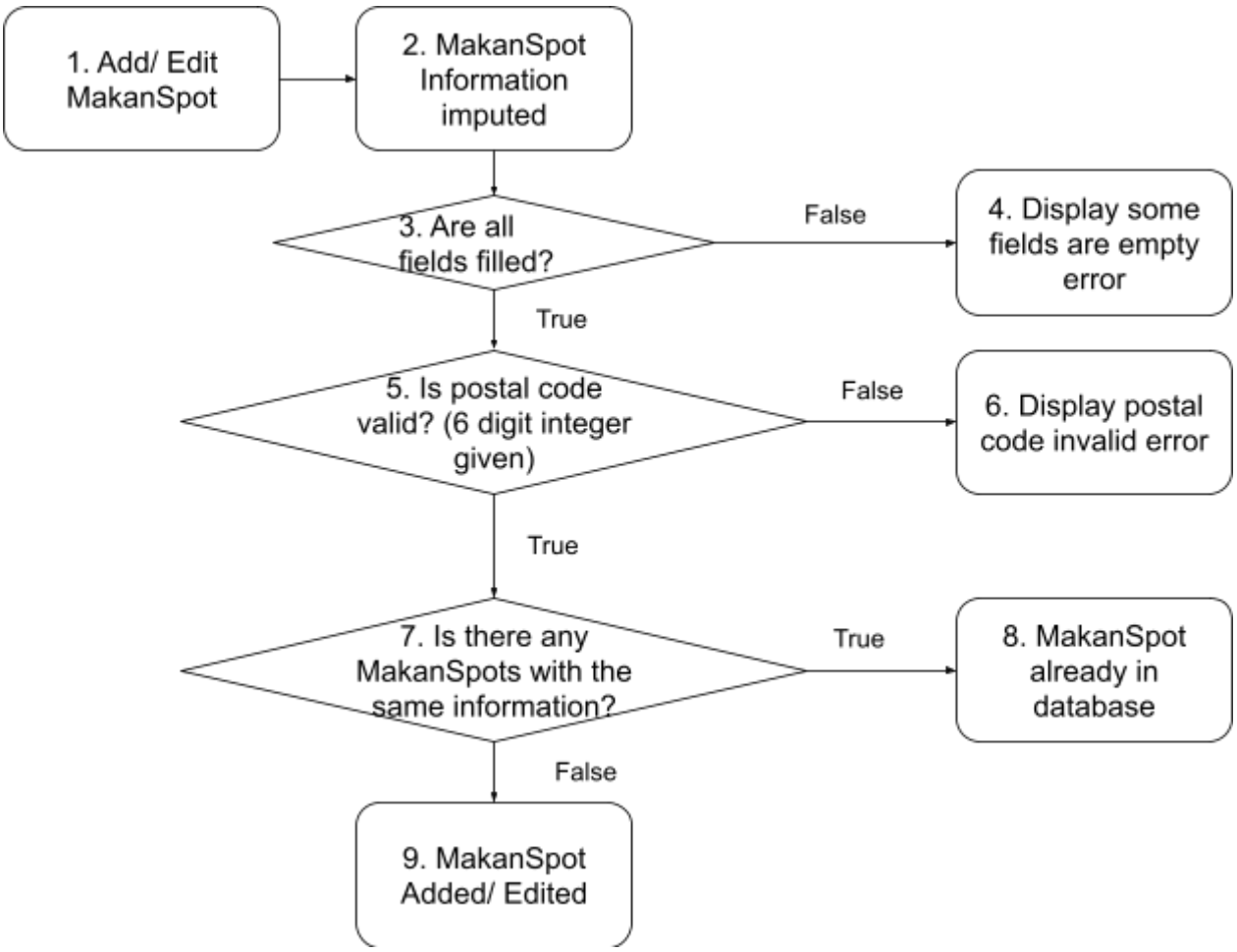
a) {1,2,4,5,6}
b) {1,2,3}

Basis path (a):
This scenario occurs as the main flow when the user wants to use the Bring Me There Function. The user first has to select the area on the map and select his desired MakanSpot. Then click the Bring Me There function

Basis path (b):
This scenario occurs when there are no MakanSpot available in the area selected by the user

11.2.1.2 Cyclomatic complexity:
Since all decision points are binary, CC =|1| + 1 = 2

11.3.2 Control Flow Test: Add/ Edit MakanSpot



11.3.2.1 Basis paths:

c) {1,2,3,5,7,9}
d) {1,2,3,4}
e) {1,2,3,5,6}
f) {1,2,3,5,7,8}

Basis path (a):

This scenario occurs as the main flow when the user adds or edits a MakanSpot successfully. This happens when all fields are filled, the postal code is valid and when there are no MakanSpots with the same exact information.

Basis path (b):
This scenario occurs when the user left some fields blank

Basis path (c):
This scenario occurs when the user enters an invalid postal code. Postal codes in Singapore must be 6 digits long and must be an integer

Basis path (d):
This scenario occurs when the user enters details which have the same exact information found in another existing MakanSpot. To avoid confusion to other users, each MakanSpot should have their own unique information

11.3.2.2 Cyclomatic complexity:

Since all decision points are binary, CC =|3| + 1 = 4

12. Revision History

Name	Date	Reason for Change	Version
ZiHan	2 Nov 2022	Added in Test Cases	1.1
Aloysius	5 Nov 2022	Added in Functional Requirements	1.2
Chien Her	6 Nov 2022	Added in black box testing	1.3
Aloysius	14 Nov 2022	Edited non-Functional Requirements	1.4
Davis	18 Nov 2022	Added Use Case 1-10	1.5
Darren	20 Nov 2022	Added Use Case 11-20	1.6
Nirvik	30 Nov 2022	Edited introductions	1.7

13. Meeting Minutes

The following table documents meeting keynotes and objectives as well outline of individual deliverables and tasks.

Date of meeting	Objectives/Tasks	Details
23/08/22	- Decide on initial idea - Brainstorming and allocation of tasks	Decided on the MakanGoWhere App idea. Started work on Software requirements. Started work on initial Use Case models Decided on UIzard for creating UI Mockup for user interface
01/09/22	- Refine Use case for submission 1 - Refine UI Mockup on UIzard	Splitting of work for complete use case diagram (Aloysius), use case description (Davis), Class diagram and key boundary classes (Zihan and Darren), Sequence diagrams (Nirvik) and initial dialog map (Chien her)
08/09/22	- Progress check and refinement	On task for submission, refinement and clarification of details on specific use cases and functionalities. Decision to dial back on functionalities and not be over-ambitious
15/09/22	- Submitted lab 2 deliverables	Submitted lab 2 deliverables, begin work on skeleton code, refinement of

		use case diagrams with sequence diagrams for each use case. Members to familiarise themselves with flutter and firebase for working on the working product. Splitting of workload.
22/09/22	- Flutter overview	Adding and creating base classes for user accounts, MakanSpot, map pages and reviews. Each of us took one functionality for sign up (Darren), sign in (Nirvik), add makanSpot (Zihan), map (David), edit makanSpot (Aloysius) and review makanSpot (Chien Her).
06/10/22	- Compilation of code	Compiled the code and connected to firebase. Ensure working authentication for users as well as ability to add makanSpot locations. Functionality review: decision to work on implementing googleMapsAPI and googlePlacesAPI with a search radius function. Review of Use Case diagrams and sequence diagrams (Aloysius, Nirvik and Chien Her). More detailed dialog map and class diagram (Zihan, Darren and Davis).
13/10/22	- Review of work products	With each team member roleplaying as customer, review work products and functionality. Minimal functionality goals are achieved.
20/10/22	- Lab 4 Submission 3	Submission of deliverables. Preparation of working application prototype. Planned out additional features to be added: - Add menu Item (Darren)

		- Edit menu Item (Aloysius) - BringMeThere (Davis) - CalculateAverageRating (Zihan) - Improvise MakanSpot details and touch up on rough looking interface of application (Chien Her)

27/10/22	- Working Product review
	- Work on detailed use case diagrams
		Acting as a client, we reviewed the current working application. Decided that we need more work on the user profile, point system, displaying of makanSpotDetails and filtering of makanSpots in the homepage. This will make the user experience much more seamless and provide them with a proper outlet to share their foodie experiences and discover options. This will be split by:
		- User profile details (Chien Her)
		  - Light and Dark mode implementation
		  - Edit profile
		  - Upload photos
		  - Forget password
		- Point system (Zihan)
		  - Everytime we interact with makanSpot and menuItems or leave review, users will get points
		  - Reward system where users can only see rewards they have enough points for
		- Filter MakanSpot (Darren)
		  - Filter MakanSpot by price points

		- Can select multiple options
		- Display makanSpotDetails (Aloysius and Davis)
		  - Show average ratings
		  - Show preview of some reviews
		  - Show preview of some menuItems
		  - BringMeThere button as well as review button and edit button to be implemented

31/10/22	- Work product review	Combined all the functionalities and did another client review. Look into adding a chat functionality and having BringMeThere open googleMaps and provide users with directions. (Done by Zihan, Aloysius, Darren, Davis, Chien Her) Split workload for final documentations for use case diagrams, sequence diagrams, class diagram dialog map, presentation script and slides as well as demo video
02/11/22	- Presentation rehearsal	Reviewed and ensured working final product. Creation of script for live-demo and slides to display use cases and software engineering practices. (Done by Zihan, Aloysius, Darren, Davis, Chien Her)
03/11/22	Live-demo and presentation	After a successful presentation, we conducted a meeting to finalise remaining documentation details together. (All group members present)

14. Appendix

14.1 Live-Demo Script

Green Text: Data Used
Blue Text: Actions
Black Text: Voice over

Upon first downloading the app, we will first need to create a new account.
Create new user -> exception handling for user:
Name: char char
Picture: charmander

1. Invalid email address (without @) aloysiusang2004gmail.com
2. Existing email available aloysius2000@live.com
3. Password and confirm password are different password, passwor1234

Upon signing up, the user will be greeted with the map. Pressing a point on the map will create an area on the map which scans the area for available makanspots. The app then redirects users to the list of makanspots available in that area in the home page
(Create an area on the map)

Users can filter the makanspots based on the price. 1 $ mean less than $10, 2 $ means price range between 10 and 50, and 3 $ means price more than $50
(set the filter) (in order: 1, 2, 3, 12, 123)

Clicking on the makanspot, users will be shown the details of the makanspot
(press a random makanspot) Press sandwich guys

From here, we can look at the description, address and dietary preference of the makanspot. We can also look at the menu of the makanspot and available reviews

Lets quickly go to the profile page to show how many points the user has. Currently, it is 0
(go to profile page, view points)

Now, lets go back to the makanspot details. By clicking on the menu, we can see the list of menu items and their prices.
(go back to sandwich guys, click on the view menu)

We can add more menu options by clicking add item at the bottom of the page. We will then be prompted to key in the information of the menu item
(press the add item button) Pork burger, $10

If we find that some menu options are wrong, we can edit the menu options by clicking on the menu item
(click on the menu item to change its information) Pasta change to beef burger

We can see the list of reviews showing the average rating of the makanspot. Notice that it is currently 3.9
(click on the list of reviews button beside view menu button)

We can add more reviews about the makanspot by clicking on the review button
(click the review button at the bottom of the page) 5 star review

Going back to the list of reviews, we can see the stars have changed and our review has been added
(go back to the list of reviews)

We can start a chat with fellow foodies by clicking on the reviews that they posted
(click on the review and start chatting. Have a simple conversation. Another device is needed)
Start chat with aloysius

If we find that some information about the makanspot is incorrect, we can edit the details of the makanspot by clicking on the edit makanspot button. We will be prompted to key in the new details, if available. We will edit the price and description
(click on the edit makanspot button) $ change to 3 $,
To prove that changes we made are saved, we will go back to the home page and try to use the filter again. As you can see the change has been reflected. If we look at the makanspot details again, we can see the details have changed
(use the filter on the home page and click on the details again and show the description)

Our app is able to direct the users to the makanspot by clicking the bring me there button, if the users are unsure on how to get there
(click on bring me there button)

Next, we can add a new makanspot. We will be prompted to key in the details of the makanspot
(
(click on + icon
KFC North Spine
76 Nanyang Dr, #01-04 NTU North Spine Plaza
637331
12
18
KFC Fried Chicken
Western

$
Vegan, vegetarian, halal
)

We can see the changes reflected by going back to the home page. As you can see, KFC has been added
(go back to the home page)

Next, navigating to the users profile page will show the details of the user.
(Go to profile page)

The number of points the users have is now… Recall that initially, it was 0. Users can gain points by adding or editing makanspots information and reviews

We have a light and dark mode theme to allow the users to suit their preferences
Show that the app can do dark theme.

From here, users can edit their profile by clicking the edit profile button. We will then be prompted to key in our details
(press the edit profile button) name change to dio, picture change

As you can see, the changes are updated
(go back to the profile page and show that the name and profile photo has been changed)

Users can see the points they have by pressing view points button
(click on the view points button)

Users can exchange their points for rewards if they have enough. The changes in their points will be reflected. If the user do not have enough points, the rewards will not be shown. At the end, no rewards will be shown due to the lack of points. For demonstration purpose, we will change the user's points to 30,000 through the database
(change the points on firebase. Select the rewards to show the points are deducted. Show if the points are not enough, the rewards will not be shown)

The users are able to change password by clicking change password. Users have to key in their email and press the button. The app will send an email to the user to change their password
(click change password. Key in the email. Click on the button. Check email address (junk))

We have come to the end of our presentation. We can now log out of the app
(click on the logout button)

14.2 Live-Demo Video

Link:
https://youtu.be/u9BuqYJuGPE