Software Requirements Specification

for

EzPark

Version 1.4 approved

Prepared by

SIM OI LIANG,

LEE CI HUI,

HIRASHIMA SHUNYA,

LINCH LIM DE ZHI

Milky Shake, Team 04, SSP2, Nanyang Technological University

2022-11-05

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Hirashima Shunya	13 Sep 2022	Initial write up for Use Case Description	1.0
Lee Ci Hui	29 Oct 2022	Initial write up for Section 1 and 2	1.1
Hirashima Shunya	4 Nov 2022	Initial write up for Section 4 and 5	1.2
Lee Ci Hui	5 Nov 2022	Refactor Section 4 and 5	1.3
Lee Ci Hui	6 Nov 2022	Initial Write up for Section 3 and Appendix	1.4
Sim Oi Liang	6 Nov 2022	Final touch up for Section 3	1.4
Lee Ci Hui	6 Nov 2022	Finalizing SRS	1.4

1.Introduction

1.1. Purpose

The purpose of this Software Requirement Specification (SRS) is to serve as a documentation of an android based mobile application, *EzPark*, build version 1.0. To facilitate the development process among the relevant stakeholders, this SRS document describes the requirements specifications for *EzPark* in detail, including but not limited to, mobile application features, functional and non-functional requirements, interface design and limitations.

1.2. Document Conventions

The standards and typographical conventions of this document are as followed:

Software Requirement Specification Standard: IEEE 830-1998. Priorities of higher level requirements are inherited by detailed level requirements.

Font:	Times New Roman
Heading:	Size 18, Bold
Subheading:	Size 14, Bold
Content:	Size 12
Spacing in content:	1.5 line spaced

Further conventions on special terms used throughout this document are described in *Appendix A: Data Dictionary*.

1.3. Intended Audience and Reading Suggestions

This document is intended for all stakeholders including the users, developer team, project managers, testers and marketing team of *EzPark*.

All Stakeholders are recommended to begin the document by reading section 1 and Appendix A to have an overall view about the document for better understanding the purpose of our product and familiarize with conventions, standards and technical terms used throughout the documents.

Subsequently, all stakeholders are suggested to proceed reading the document in sequential order.

1.4. Product Scope

As the digital era evolves rapidly, Singapore aims to become a world-class, tech-driven city-state in which residents can benefit from technology regardless of their background. Aligning with the idea of Smart Nation, our team decides to focus on one of the three key pillars - Digital Society. In order to accommodate the face-paced life of Singapore's residents, our team tries to find solutions in aiding their planning of daily life routine.

Being a driver, one often struggles to find a suitable car park and worries about car-park lot availability that are near to their intended destinations during planning of journey and and process of driving. They might also have troubles remembering the location they parked their cars and tracking the time elapsed of their parking.

EzPark, an Android-based mobile application, is developed by our team to ease car drivers in their journey planning and allow them to focus on their jobs without worrying about car-parking matters. By using *EzPark*, users are able to set their start location and destination while our system will show the available car-park locations in a radius of 500m from the intended destination on the map and in a list. The users can further sort the listing of car-park locations results according to their preference parameters.

After selecting their preferred car-park location, *EzPark* will render the relevant details including past ratings and reviews of that car-park from our past users as reference. By confirming to start parking on the selected location, *EzPark* has a few bonus features to aid users in car-parking.

i. fare estimation calculator
ii. timer reminder
iii. notes taking section with photo uploading section

EzPark also directs users to an external application (Google Maps) for navigation purposes.

After the user ends their parking session, they are able to provide ratings and reviews for that car-park location for future users' references. Being *EzPark* users, they are able to trace their car-park history and save their favorite car-park in the application.

Overall, *EzPark* provides an all-in-one parking finding experience by integrating relevant features in our application.

In future, *EzPark* development team wishes to expand our services worldwide once the we manage to obtain relevant data and gain sponsors for access to all required API.

1.5. References

I. *IEEE 830-1998 - IEEE Recommended Practice for Software Requirements Specifications*. IEEE Standards Association. (n.d.). Retrieved 30 October, 2022 from https://standards.ieee.org/ieee/830/1222/

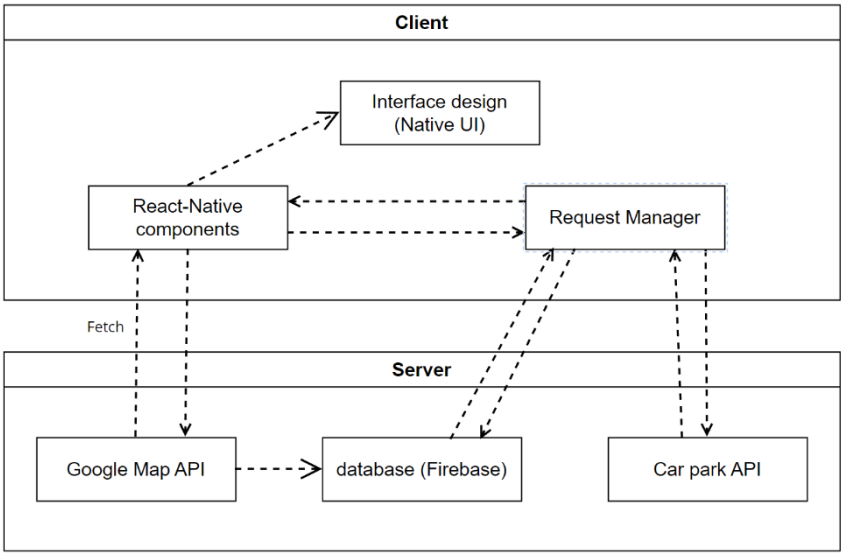
II. *React Native Documentation*. React Native. (n.d.). Retrieved 30 October, 2022 from https://reactnative.dev/docs/getting-started

III. *Firebase Documentation*. Firebase. (n.d.). Retrieved 30 October, 2022 from https://firebase.google.com/docs

2. Overall Description

2.1.Product Perspective

EzPark is a new and self-contained android-based mobile application. An overall system diagram is shown below indicating the brief operation of EzPark.



2.2.Product Functions

EzPark's features can be broken down into five main sub-categories ie. Authentication, Account, Search Car Park, Customize Helper Features Settings and Provide Feedback. This section provides the high level summary of the system features contained in the mobile application. More details regarding each feature can be found under *Section 4. System Features*.

2.2.1 Authentication

i. Login Account
ii. Register New Account
iii. Retrieve Lost Password

2.2.2 Account

i. Manage Account details
ii. Manage Car-park Favourite List
iii. Manage Car-park History List
iv. Help Page provides Information related to *EzPark*

2.2.3 Search Car-park

i. Search Start Location
ii. Search Destination
iii. Show Related Car-park Locations and Informations
iv. Sort Car-park Results based on preferred parameters
v. Select Desired Car-park
vi. Show Details of Car-park and Past Ratings and Reviews
vii. Navigation in-app

2.2.4 Customize Helper Features Settings

i. Set Parking Time Estimation
ii. Car-parking Fare Calculator
iii. Countdown Timer to Alert User Remaining Time
iv. Note-taking
v. Upload Photo in Note-taking section
vi. Navigation in-app

2.2.5 Provide Feedback

i. Write Comments
ii. Provide Ratings

2.3.User Classes and Characteristics

Ezpark anticipates its potential users as follow:

2.3.1 Owner of Small to Medium Vehicles

Aspect	**Description**
Frequency of Use	Frequency of Driving
Subset of Product Functions Used	All
Technical Expertise	- Own a smartphone - Familiar using Navigation related application
Age	18 years old and above
Characteristics	These users are predicted to use *EzPark* to search for suitable car parks during their planning of journey. Navigation function will also help them to find the route especially if they are new to the car park. They will also use the helper features to aid them during the car parking session.

2.3.2 Passenger of Small to Medium Vehicles

Aspect	**Description**
Frequency of Use	Frequency of being a Passenger
Subset of Product Functions Used	All
Technical Expertise	- Own a smartphone - Familiar using Navigation related application
Age	No Specific Requirement
Characteristics	These users are predicted to use *EzPark* to help the driver in searching suitable car parks during their planning of journey. Navigation function will also help their drivers to find the route especially if their drivers are new to the car park. They will also use the helper features to aid them during the car parking session.

2.3.3 Owner of Car Park Building

Aspect	**Description**
Frequency of Use	Low
Subset of Product Functions Used	Ratings and Comments History
Technical Expertise	- Own a smartphone
Age	No Specific Requirement
Characteristics	These users are predicted to use *EzPark* to look for the rating and comment section of their car park made by the past users of *EzPark* in order to make further improvements on their management of car park building.

2.4.Operating Environment

This section contains the Operating Environments of *EzPark*, including the accesses required from the user device, environments of product and development.

2.4.1 Access Required from the User's Device

EzPark requires access to the user device's Internet Connection, Location and Camera.

2.4.2 Environment of Production

EzPark is built on Android Studio, with API level of 31 and dp 412 x 732. Android Studio always installs the latest Android SDK by default. (*Note that the latest version of Android Version at the time of writing this documentation is Android 12.0 (Snow Cone)* .)

Hence, *EzPark* is compatible with smartphones supporting at least Android Operating System 12.0, API level of 31 and above.

2.4.3 Environment of Development

Development Environment	Description
Front-end Development: React Native (*Version 0.69.6*)	React Native is an open-source UI Software Framework developed and maintained by Meta Platforms, Inc. It allows developers to use the React Framework along with the native platform capabilities to develop mobile applications that are runnable in both Android and IOS platforms using the same codebase. *EzPark* development team uses React Native to develop all user interfaces and features' logic with reusable components.
Back-end Development: Firebase (*Version 9.12.1*)	Firebase is a set of hosting services for any type of application, including Android, iOS, JavaScript and Node.js. It is backed by Google and provides features including but not limited to NoSQL , real-time hosting of databases and notifications. EzPark development team use firebase to develop the backend database and authentication part.

2.5.Design and Implementation Constraints

This section contains the Design Standards and Constraints that have limited, or will be limiting EzPark Development Team during the development and maintenance stage.

2.5.1 Design Standards

Design Standard	Description
User Interface Standards	All User Interface designs must follow the colour scheme and layout agreed by the stakeholders.

	All Buttons must be labelled with unambiguous naming conventions and Icons used are the common standard icons.

2.5.2 Constraints

Constraint	**Description**
Google Cloud Platform (Google Map API)	The Google Cloud Platform used by EzPark endorses a free trials model with $433.42 credits and 90 days of usage. Subsequent usage Directions API, Map SDK for Android, and Places API under Google Cloud will be charged on subscription fee per-use-basis.
Car Park API	Car Park API used is obtained from api.data.gov.sg. The data rendered might face error if the API is not maintained or updated for any reason in the future.
Memory	Data including but not limited to: - Destination Details from Google Map API - Car Park Details from Car Park API - Image uploaded in Note-taking Section Are stored locally in the device's internal storage while running the application. Therefore, if the user's device internal storage is full, or the user wishes to utilize secondary memory storage, the performance of the application might be affected. Hence, user's devices are required to have at least 100 MB of free internal storage.

2.6. User Documentation

EzPark is an application that is straightforward and easy to use. The journey in the application is mostly in one-to-one direction from the searching step to select car park and navigation until configure settings in the helper features. Every screen is labeled with descriptive titles, every icon used follows the conventional design and every button created is labeled with descriptive names related to its function.

No hardcopy of the user manual will be provided as the user interface is developed user-friendly with a small learning curve upon the features that have been integrated into the application. However, if the user has a query about *EzPark*, the user is suggested to refer to the "Help" page which can be accessed from the side drawer with an information icon, by clicking the menu icon that can be found at the top left corner of every screen. If the information provided in the "Help" page does not solve the user's query, the user can use the contact information listed in the page to consult the EzPark Development Team.

2.7. Assumptions and Dependencies

This section includes the Assumptions which describes the assumed factors to ensure the requirements stated in *section 4* for *EzPark* mobile application to work smoothly.

2.7.1 Assumptions

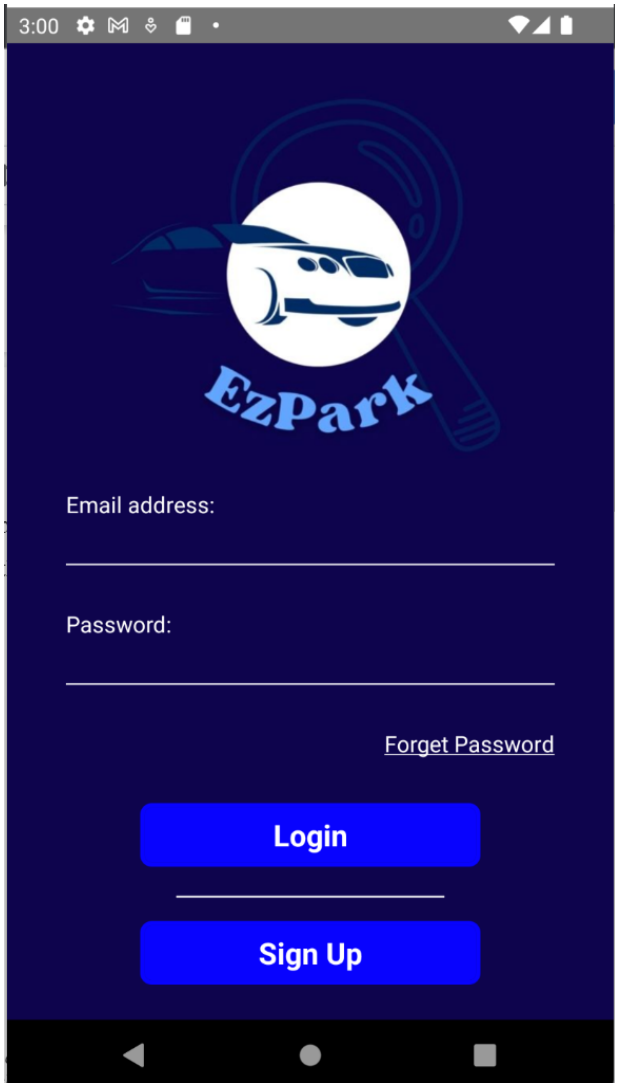
- *EzPark* highly depends on external APIs such as Car Park API and Google Map API for rendering data and *EzPark* assumes they are functioning and updated. If any of the source of the API violates the assumptions for any reason, the application will not perform its core functionality as per requirements stated.
- *EzPark* assumes its application to run on a smartphone with Android 12.0 (Snow Cone) Operating System, and designed to be the best view in a smartphone.
- *EzPark* requires a stable Internet connection.
- *EzPark* requires location access and camera access to be granted by the user to use the related features and functionalities.
- *EzPark* assumes its application to be used within Singapore only. Any attempt to search for destinations outside Singapore will not be guaranteed to provide positive

results regarding nearby car park data. Further developments are only to be conducted with dependency on the availability of similar data sources in the respective countries.

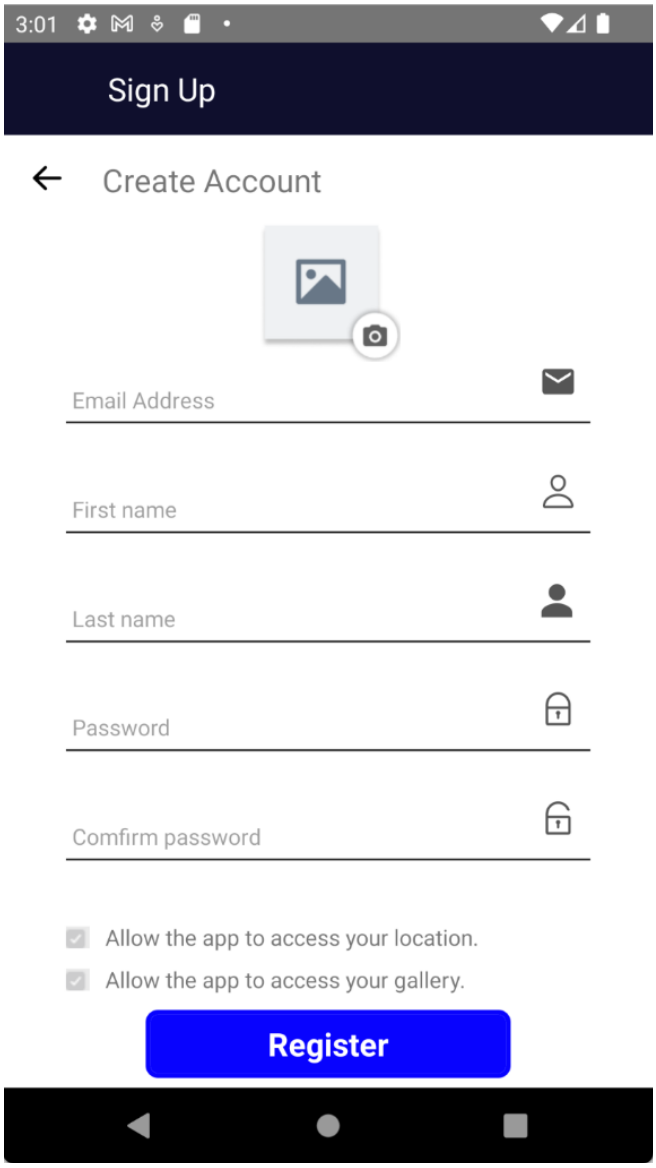
3.External Interface Requirements

3.1. User Interfaces

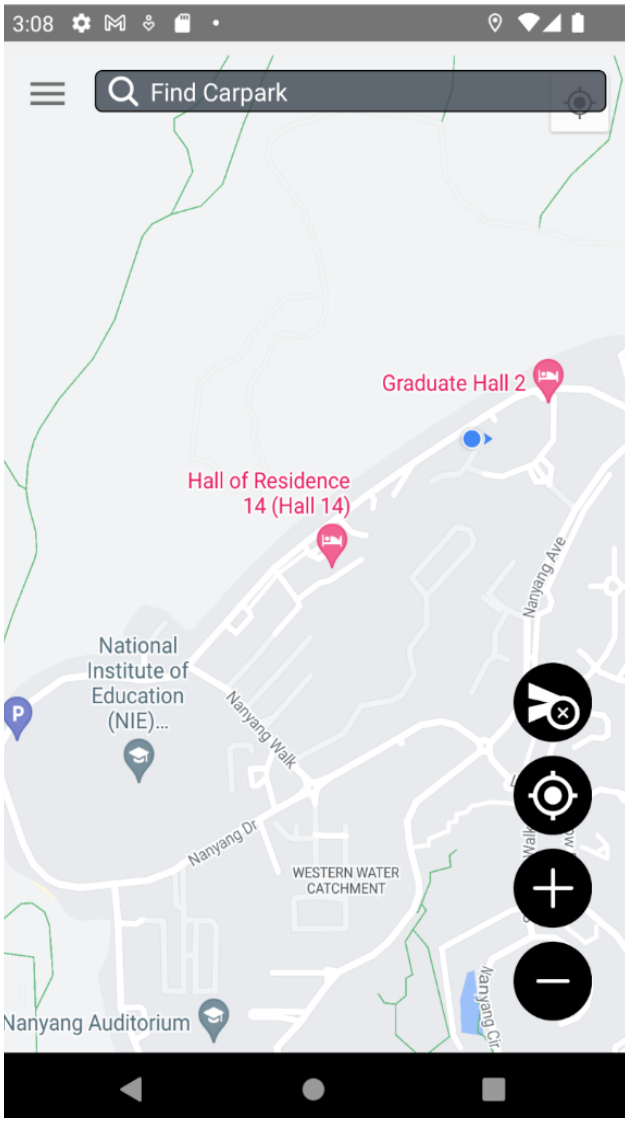
3.1.1 Login Page



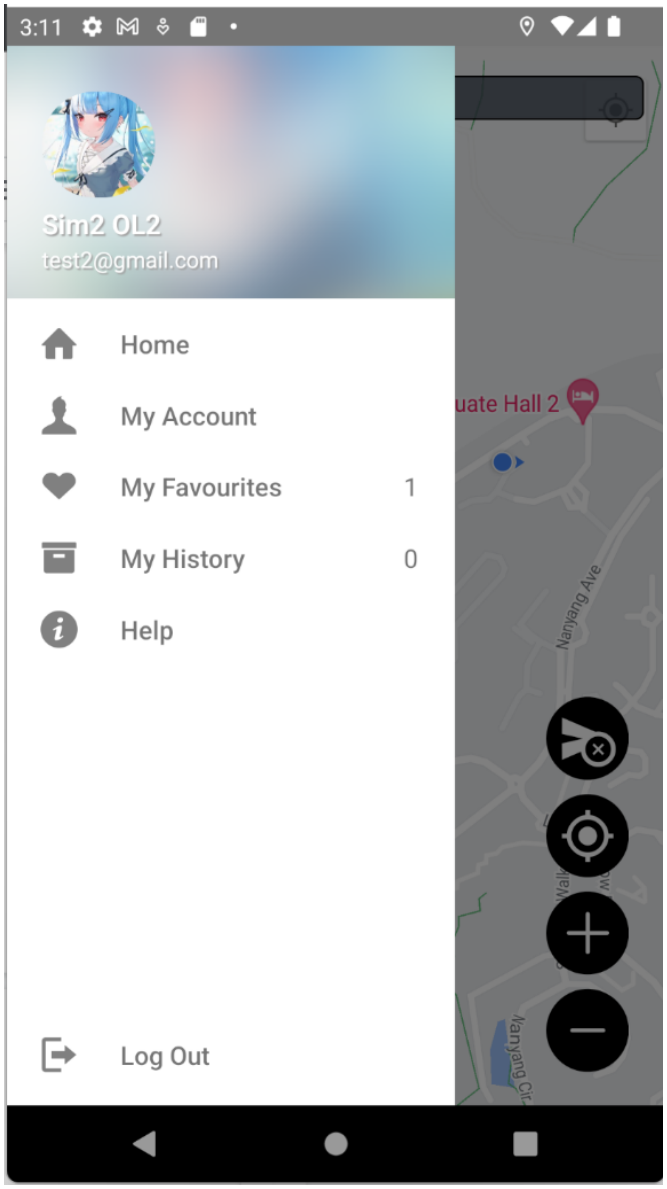
3.1.2 Register Page



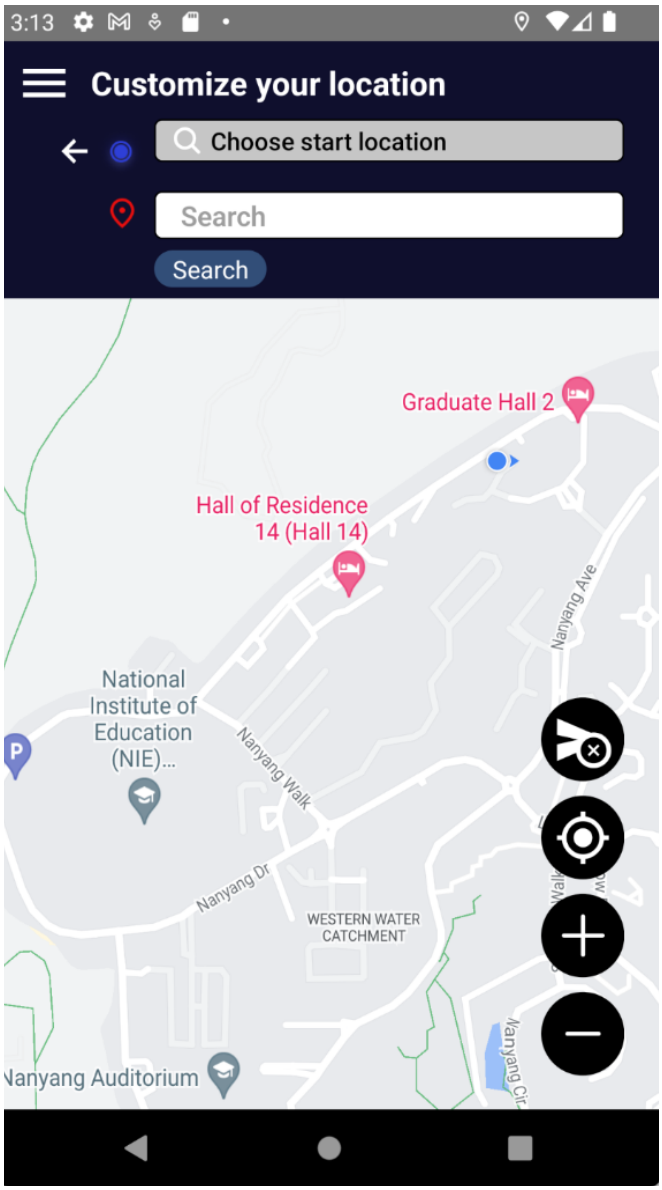
3.1.3 Home Page (with current location shown)



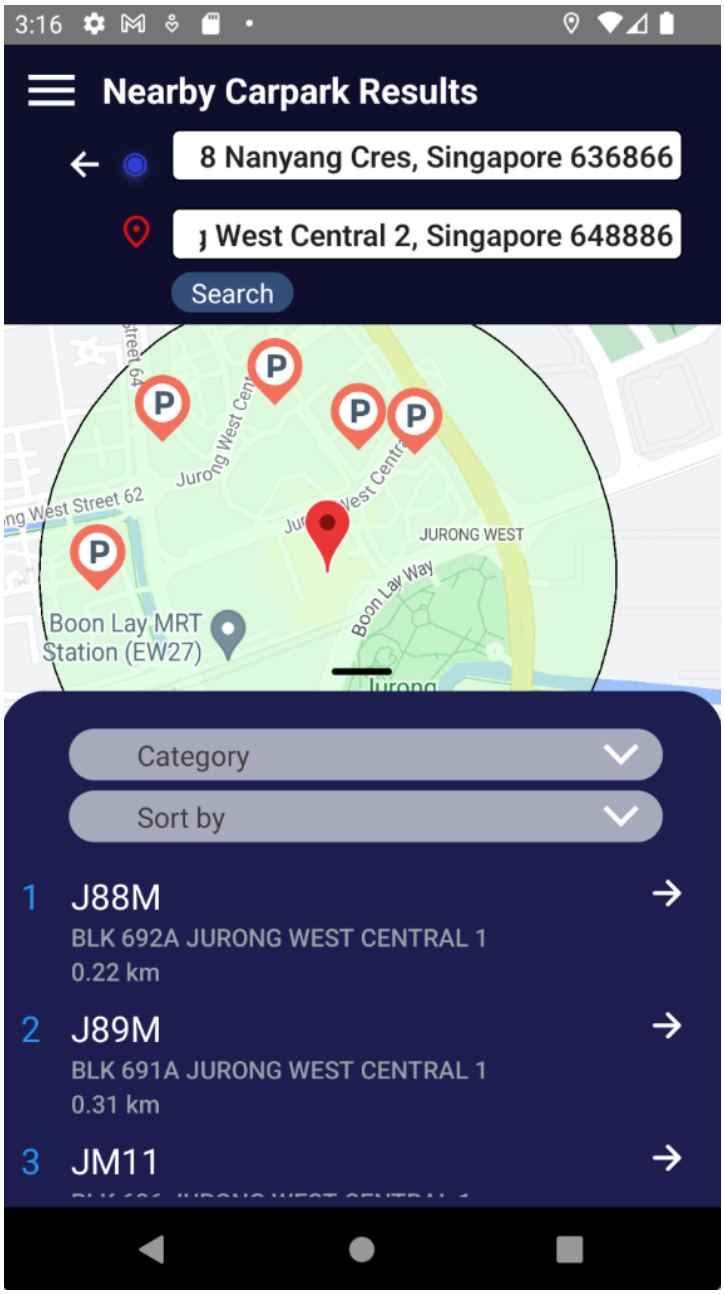
3.1.4 Menu bar (Side Drawer)



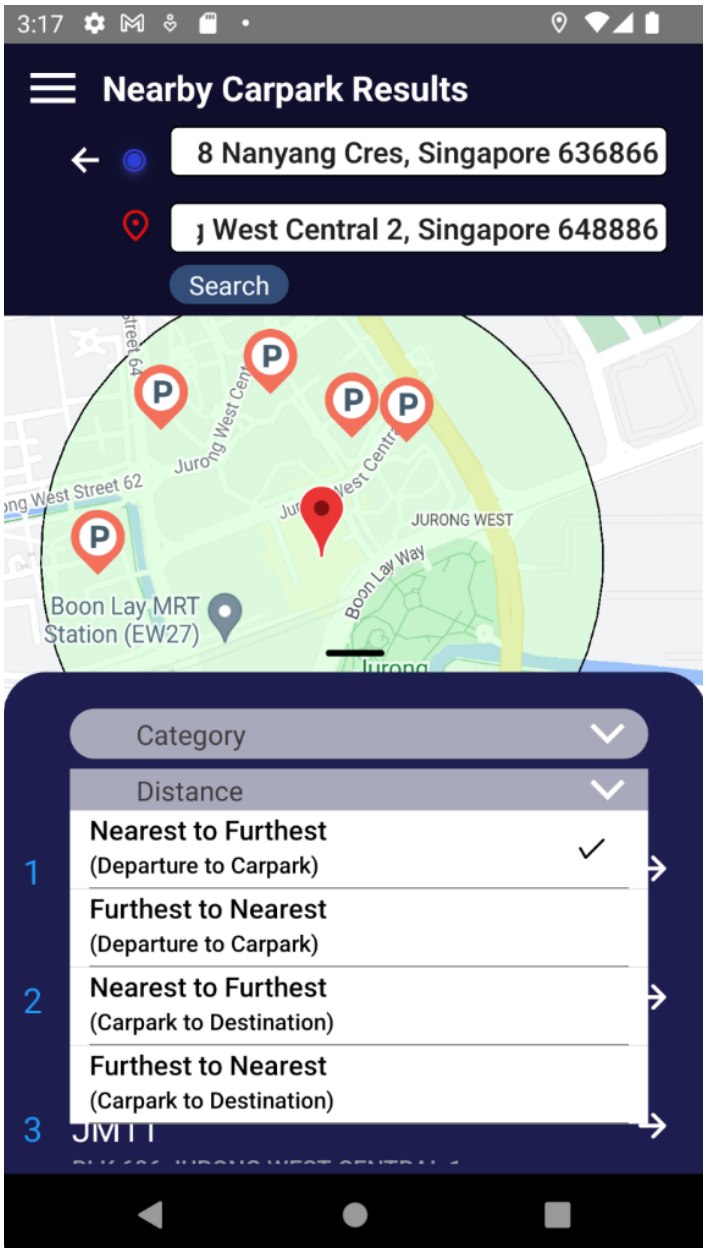
3.1.5 Search (Customize Start Location and Destination)



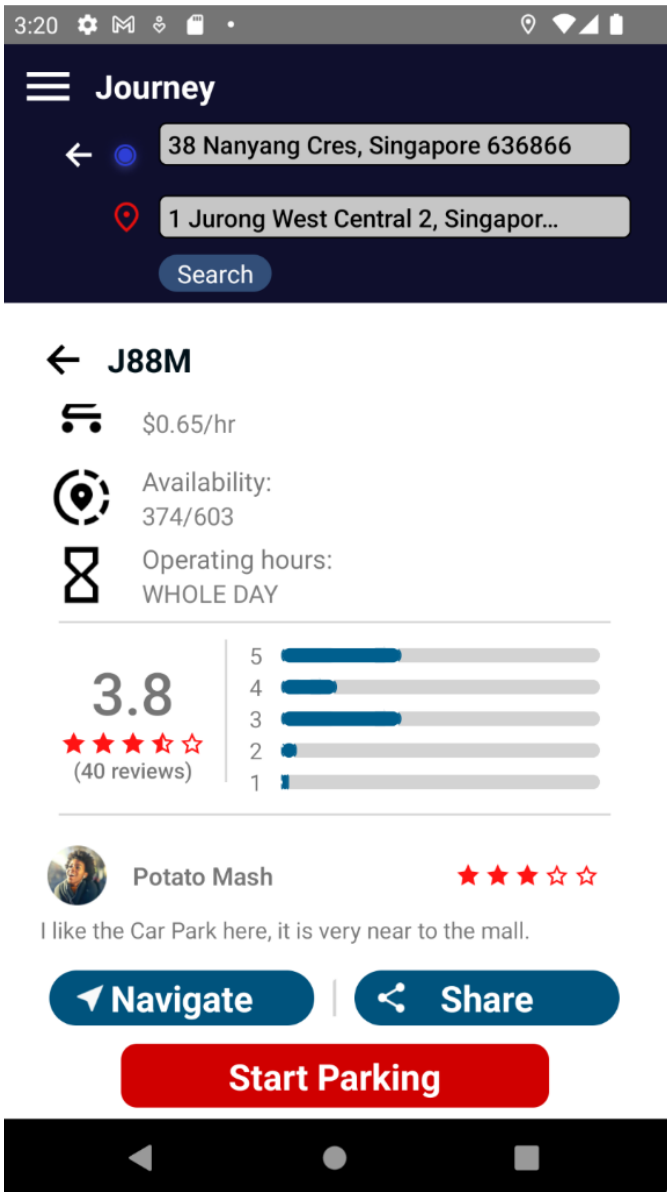
3.1.6 Show List of Car Park



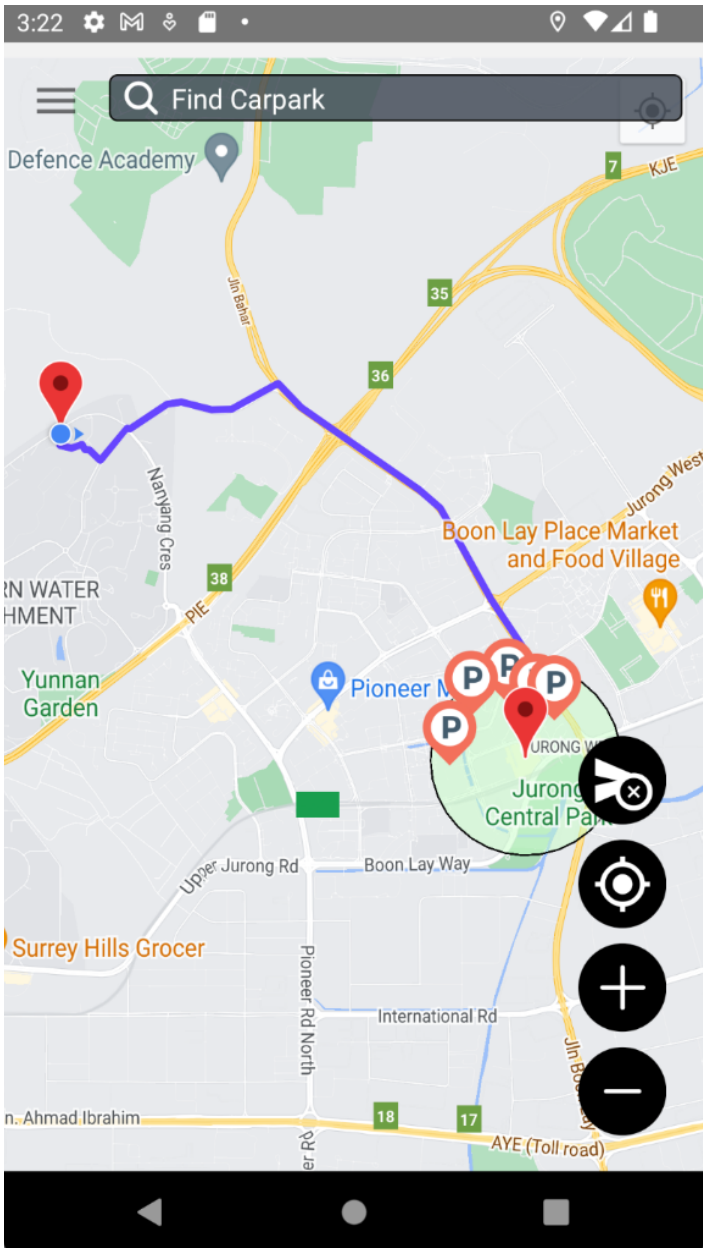
3.1.7 Sort Result



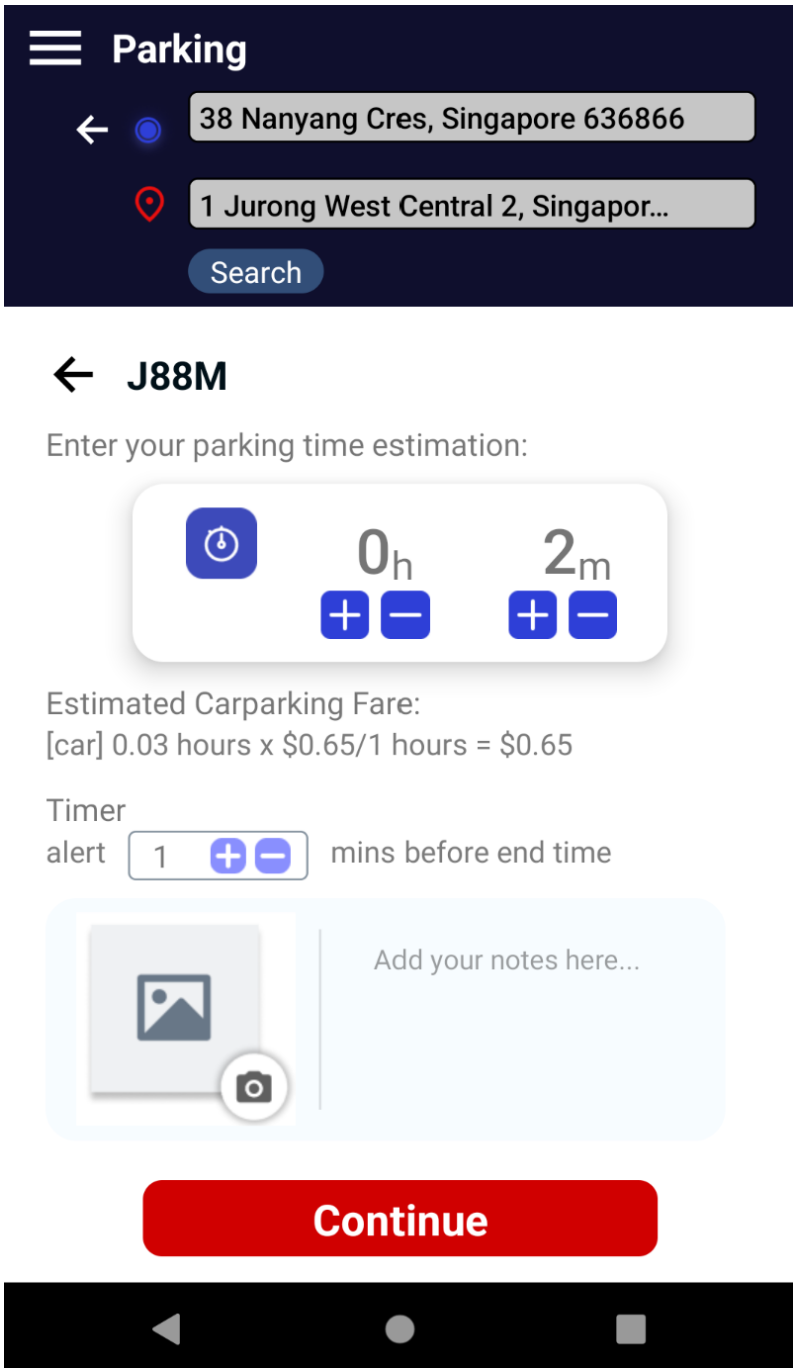
3.1.8 Car Park Information



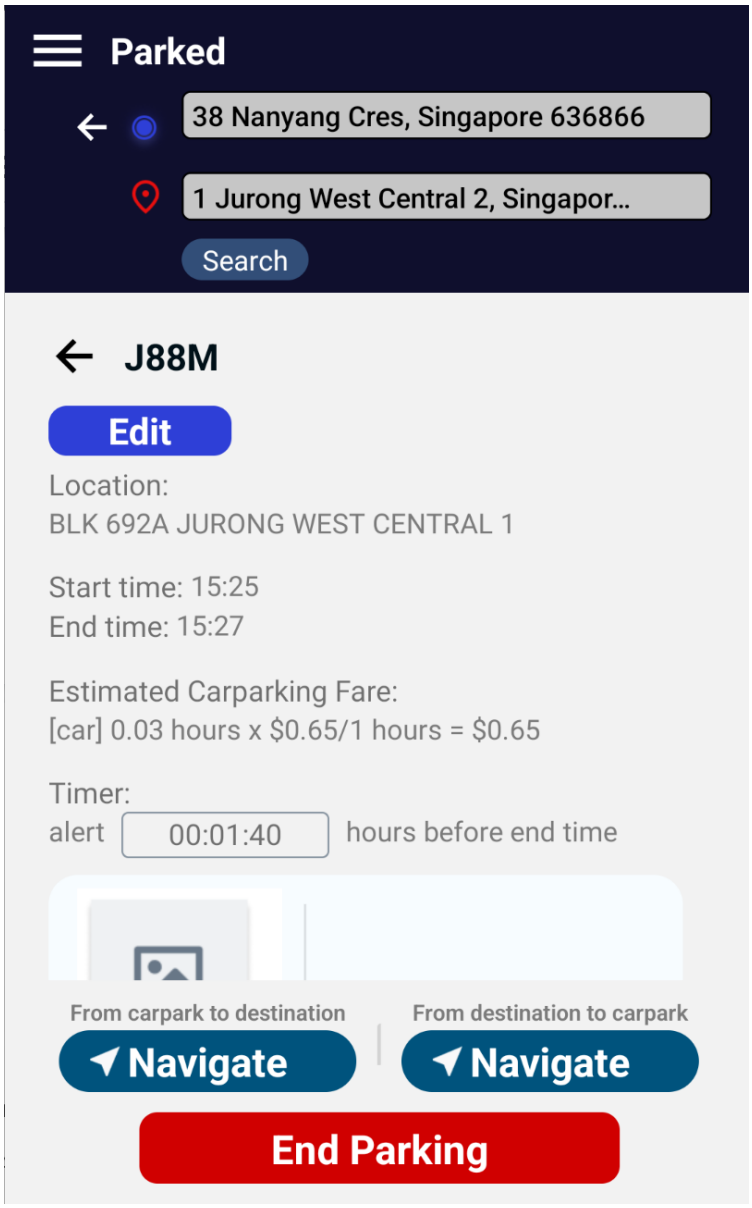
3.1.9 Navigation



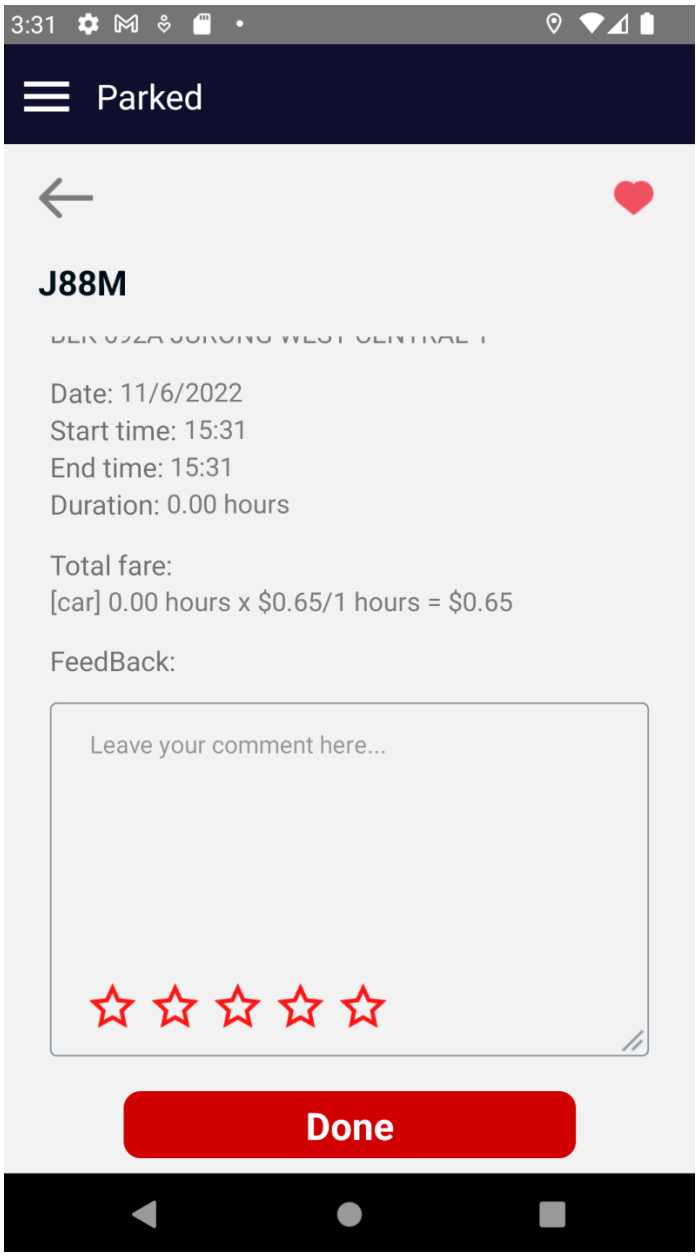
3.1.10 Parking Settings (Customize Helper Features)



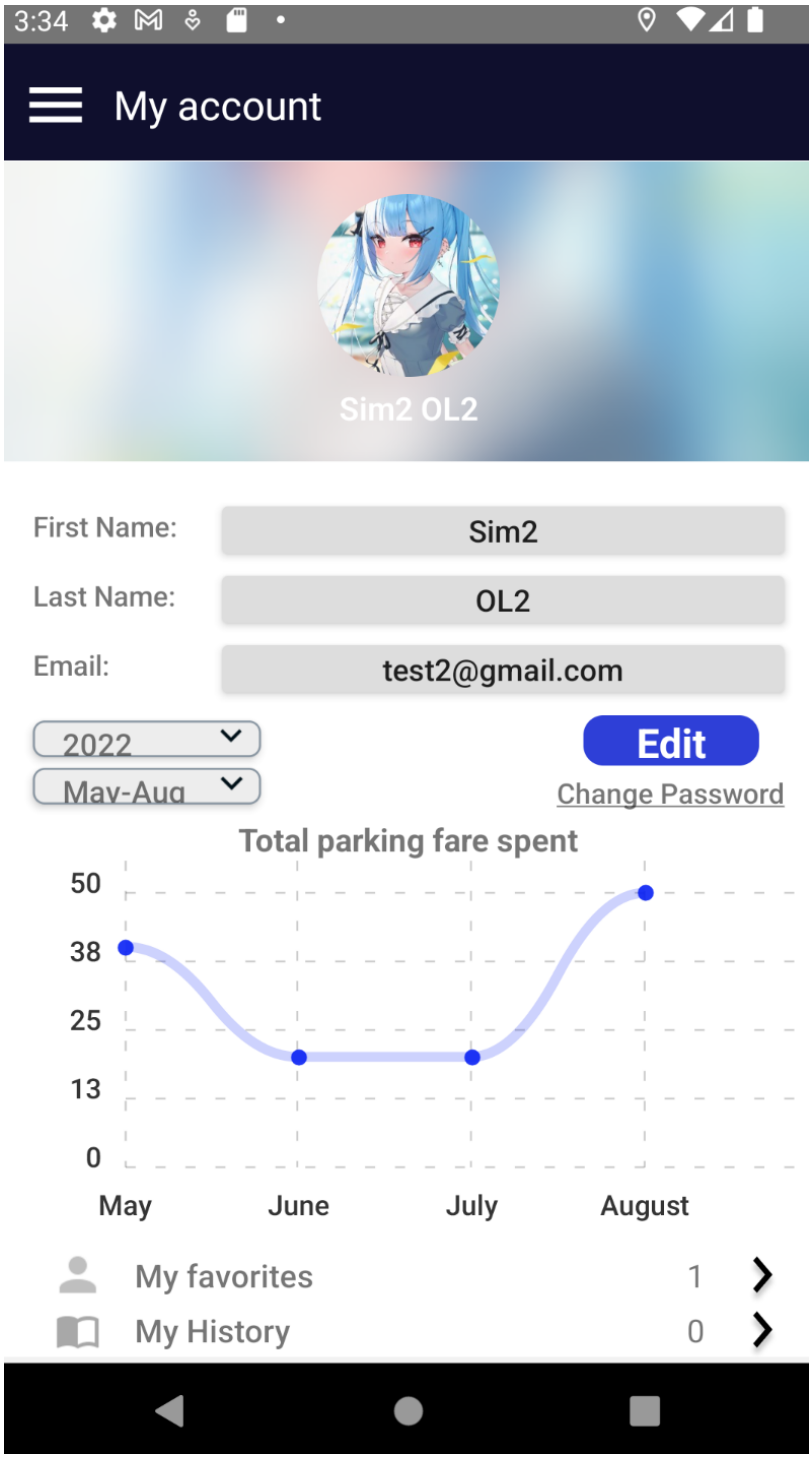
3.1.11 Parking Ongoing



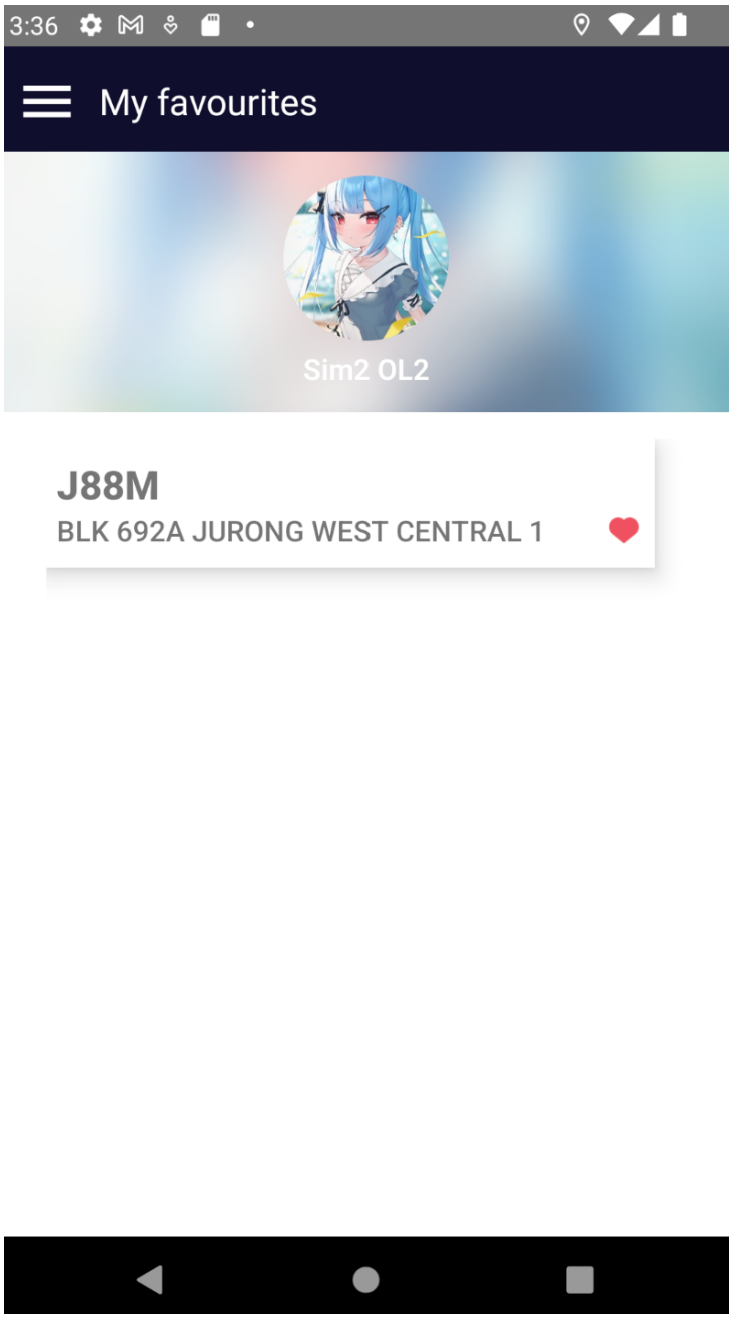
3.1.12 Comment and Rating



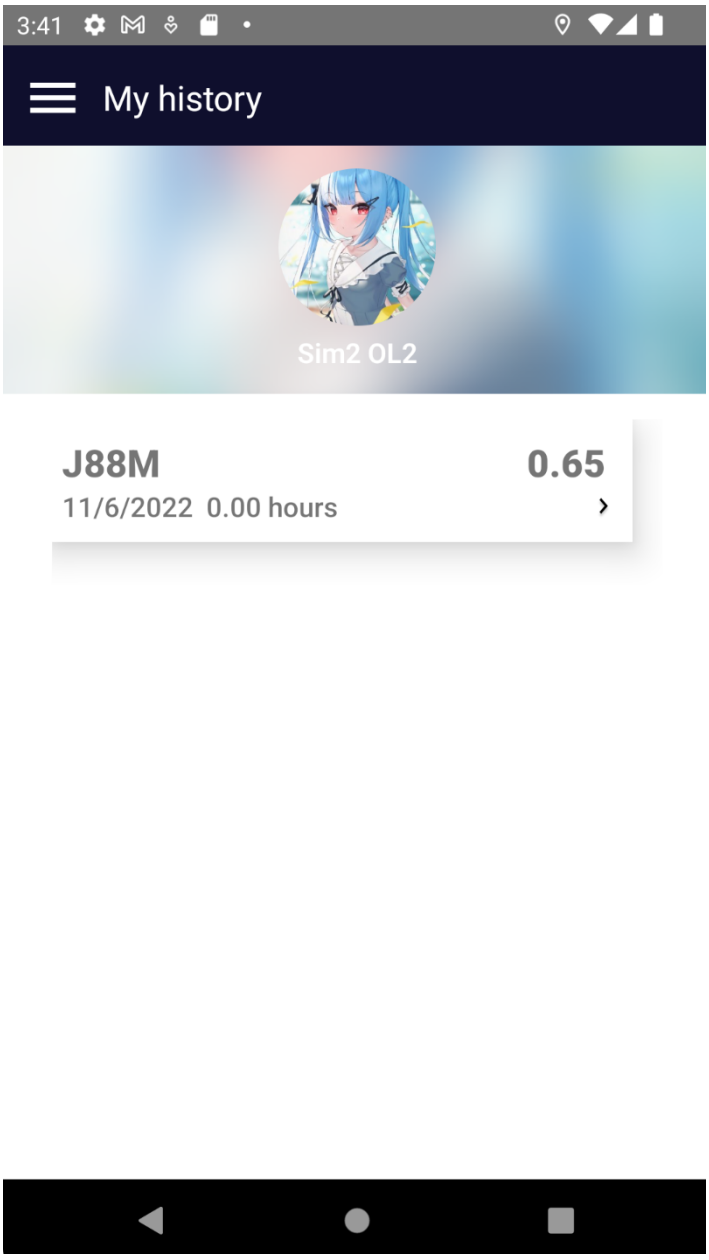
3.1.13 My Account



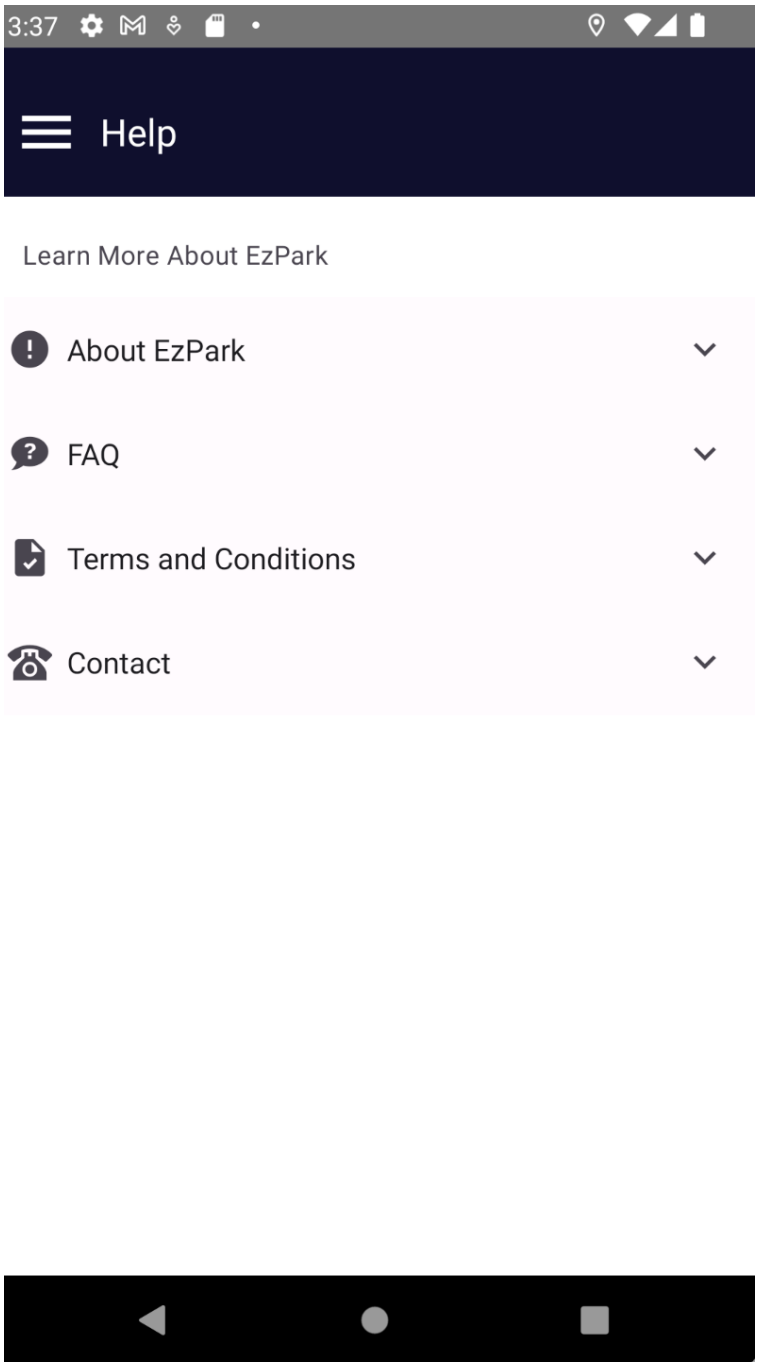
3.1.14 My Favourite



3.1.15 My History



3.1.16 Help Page



3.2.Hardware Interfaces

This section describes the hardware interface requirements for EzPark mobile application to perform its desired functionalities. All requirements in section *2.4.2 Environment of Production* have to be fulfilled at the same time.

	Description
Operating System	A device with Android 12.0 (Snow Cone) Operating System
Network Connection	Wireless Network Interface Card (WNIC) or a modem chip with cellular modem
Storage	The phone must have enough storage space for the system to store download and data.
Interaction	Touch Screen Interface that allows users to perform touch action including but not limited to swiping, scrolling, tapping and pinching.

3.3.Software Interfaces

OS: Windows 10

Tools: Android Studio Dolphin 2021.3.1 , React-Native 0.69.6, Android Emulator with Android API version 31.

Third party libraries:

1) React-natve-maps (version 1.3.2)
    - Request Data: Google Maps API key with Google Places API and Map SDK for android enabled
    - Return Data: React Component with google map embedded.
    - Purpose: To integrate google map in the system
2) React-native-google-places-autocomplete (version 2.4.1)
    - Request Data: Google Maps API key with Google Places API enabled and text information containing substring of user entered location.

- Return Data: React Component containing list of predicted matching location address and position in latitude and longitude format.
- Purpose: Allow users to select the suggested location and details of the selected location will be used by the system to determine nearby Car Parks.

3) React-native-maps-directions (version 1.9.0)
   - Request Data: Google Maps API key with Google Directions API enabled
   - Return Data: React Component with direction shown in React-native-maps component.
   - Purpose: To show direction to users during navigation.

APIs:

1. Google Maps API
   - Used by React-natve-maps, React-native-google-places-autocomplete and React-native-maps-directions to allow users to use integrated google map and services.
2. CarPark Availability API by Singapore Government
   - Get CarPark Availability information in Singapore via HTTP GET in JSON format for the purpose of showing and suggesting car parks to users using available car park lots.
3. HDB CarPark Information API by Singapore Government
   - Get CarPark information in Singapore via HTTP GET in CSV format for the purpose of showing car park details and suggesting car parks to users using distance.

The software uses Firebase Realtime Database version 20.1.0 for cloud data storage. User account-related data needs to be accessed via an object to read or write.

The other data components are stored in the local storage.

3.4.Communications Interfaces

EzCarPark main communication is using HTTP GET requests from APIs, so a stable internet connection is needed. An unstable internet connection will result in failure of fetching information from APIs and cause the search function of the app to fail.

4.System Features

4.1.Login to User Account

4.1.1. Description and Priority

When the user first runs the application, *EzPark* prompts the user to conduct a login process through inputting email address and password. The user must login before being allowed to access all functionalities provided by *EzPark*.

Overall Priority	**Description**
High	User accounts to be used must be identified before the main application process is set on.

4.1.2. Stimulus/Response Sequences

Use Case ID:	UC-LUA-001		
Use Case Name:	Login to User Account		
Created By:	Hirashima Shunya	Last Updated By:	Lee Ci Hui
Date Created:	30th August 2022	Date Last Updated:	5th November 2022

Actor:	User
Description:	Account login process through email address and password.
Preconditions:	1. User accounts must exist in the database. 2. The system is on the login page.
Postconditions:	1. A user must be able to access all functionalities. 2. A user must be able to access past records in favourite List and History List
Priority:	High

	The user must login before being allowed to access all functionalities provided by *EzPark*.
Frequency of Use:	High
Flow of Events:	1. User enters an email address and password in the login interface. 2. User selects the "Login" button. 3. The system validates the email address and password. 4. The system verifies the account by checking the user's credentials stored in a cloud database. 5. The system authenticates the user to login successfully by direct user to Home screen.
Alternative Flows:	AF-S3: If the system detects empty email address or password fields 1. The system displays an error message "Please input both email address and password." 2. Return to Step 1. AF-S4: If user enters the wrong credentials 1. The system displays an error message "Either address or password is wrong. Please try again." 2. Return to Step 1.
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	A user account must be accessed from a cloud database.
Notes and Issues:	-

4.1.3. Functional Requirements

1. The user must login to the respective user account by entering email and password to access the system.

1.1. The user must press the 'Login' button to login.

1.2. The system must display error messages if the user fails to login to the system.

1.3. The system will search the database to see if the username and password is correct.

1.4. If the data is correct, then the user is redirected to the Home Screen.

1.5. The user must stay in even after the application is closed.

4.2.Register New Account

4.2.1. Description and Priority

When the user is new to *EzPark* and does not have an account for the application, the user registers for a new user account. Once registration is successful, the details of the user will be recorded into database.

Overall Priority	**Description**
High	Creation the new user account enables to conduct the test whether the login session properly works.

4.2.2. Stimulus/Response Sequences

Use Case ID:	UC-RNA-001		
Use Case Name:	Register New Account		
Created By:	Hirashima Shunya	Last Updated By:	Lee Ci Hui
Date Created:	30th August 2022	Date Last Updated:	5th November 2022

Actor:	User
Description:	Registration for a new user account.
Preconditions:	1. The window is on the Login page.
Postconditions:	1. A new account is created and stored on a cloud database.

Priority:	High
Frequency of Use:	1 – 3 times per lifetime
Flow of Events:	1. User taps the “Sign up” button. 2. User inputs first name, last name, email address, and password. 3. User taps the “Register” button. 4. The system validates the user inputs. 5. The system verifies the account by checking the availability of the user's account in the cloud database. 6. The system creates a new account for the user in the cloud database. 7. The system sends a verification email to the user's email. 8. After the user clicks on the verification link, the system activates the new account.
Alternative Flows:	AF-S4: The user inputs an invalid email address 1. The system displays an error message, “Invalid address. Please input a valid address.” 2. Return Step 2. AF-S4: The user inputs an invalid password 1. The system displays an error message “Invalid password. Please input a valid password by referring to the conditions below.” 2. Return Step 2. AF-S5: The system detect account existed in the cloud database 1. The system displays an error message “Account already exists. Please try again.” 2. Return Step 2.

	AF-S8: The system did not receive confirmation by the verification link. 1. The system will not activate the new account until confirmation.
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	A user account must be stored in a cloud database.
Notes and Issues:	-

4.2.3. Functional Requirements

2. If the user does not have an account, the user must be directed to the Sign up screen. The user must input their email, first name, last name and their password.
   2.1. Email must be of string data type.
      2.1.1. The user must input a valid email address.
         2.1.1.1. A valid email address must include the symbol '@'
      2.1.2. If the email is empty, the user is prompted to fill up the field.
      2.1.3. If the email has already been used before and is stored in the database, then the user must be prompted that the email has already been used to create an account.
   2.2. First name must be of string data type.
      2.2.1. The user must input a valid first name.
         2.2.1.1. A valid first name cannot contain characters other than text and integer datatype and the two special characters(.,@).
      2.2.2. If the first name is left blank, the user is prompted to fill up this field.
      2.2.3. If the input value of the first name does not meet the requirements of section 2.2.1, the user is prompted to refill the field with the correct requirements.
   2.3. Last name must be of string data type.
      2.3.1. The user must input a valid last name.

2.3.1.1. A valid last name cannot contain characters other than text and integer datatype and the two special characters(.,@).

2.3.2. If the last name is left blank, the user is prompted to fill up this field.

2.3.3. If the input value of the last name does not meet the requirements of *section 2.2.1*, the user is prompted to refill the field with the correct requirements.

2.4. Password must be of string data type

2.4.1. The user must input a secure, valid password.

2.4.1.1. A valid password must have a minimum of 8 characters.

2.4.1.2. A valid password must contain at least 1 capital letter(A-Z).

2.4.1.3. A valid password must contain at least 1 number(0-9).

2.4.1.4. A valid password must contain at least 1 of the following characters-('.','_','-').

2.4.2. If the password is empty, the user is prompted to fill up the field.

2.4.3. If the password requirements of *sections 2.2.1.1, 2.2.1.2, 2.2.1.3, 2.2.1.4* are not met, the user is prompted to refill the field with the correct requirements.

2.5. The user must press the "register" button to proceed with the registration.

2.5.1. If the "register" button does not receive input from the user, the state of the application does not change.

2.5.2. If the requirements in *2.2.1 and 2.1.1* are not met, the user will not be able to register.

2.5.3. If all the requirements in *2.2.1 and 2.1.1* are met, then the user will be redirected to the Login page.

4.3.Change Password

4.3.1. Description and Priority

When the user forgets the user's password, the system prompts the user to change the password.

Overall Priority	**Description**
Medium	Without the feature, the application can still work.

4.3.2. Stimulus/Response Sequences

Use Case ID:	UC-CPA-001		
Use Case Name:	Change Password		
Created By:	Hirashima Shunya	Last Updated By:	Lee Ci Hui
Date Created:	30th August 2022	Date Last Updated:	5th November 2022

Actor:	User
Description:	Alteration of a user account password.
Preconditions:	1. A user account must exist in the database. 2. The window is on the Login page.
Postconditions:	1. A new password replaces the old one in a cloud database.
Priority:	Medium
Frequency of Use:	1 – 3 times per lifetime
Flow of Events:	1. User taps “Forget password” strings. 2. User enters an email address for an existing account. 3. The system validates the user's input. 4. The system verifies the account by checking the user’s email stored in a cloud database. 5. The system sends an email to the user's email to reset the password. 6. User inputs a new password and taps the “update” button. 7. The system checks the validity of the new password. 8. The system updates the password in the cloud database.
Alternative Flows:	AF-S3: If the system detects an empty email address field.

	1. The system displays an error message "Please input an email address." 2. Return to Step 2. AF-S4: If input address is not found on a database 3. The system lets the user input email address again, displaying an error message, "Input email address is not found. Please input an address that you register for". 4. Return to Step 2. AF-S7: If user's new password is not valid 5. The system displays an error message "Invalid password. Please input a valid password by referring to the conditions below." 6. The system lets the user input another password to be valid. 7. Return to Step 6.
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	A user account can be accessed from cloud databases.
Notes and Issues:	-

4.3.3. Functional Requirements

3. If the user has forgotten their password, the system must allow the user to reset their password.
   - 3.1. The user will be led to the forgot password page.
   - 3.2. The user will be required to enter their email.
     - 3.2.1. If the email requirements in *2.1.1* are not fulfilled, then the user will be prompted to enter a valid email.
   - 3.3. An email must be sent out to the user, using the email address that they has inputted.

3.4. The email must be sent out a link which allows the user to reset their password.

3.5. The new password must fulfill the requirements in *2.2.1*

3.5.1. If the password requirements in *2.2.1* are not fulfilled, then the user will be prompted to enter a valid password.

4.4.Retrieve Current Location

4.4.1. Description and Priority

The system retrieves the current location of the user and displays on the Google Map rendered if the user allows the permission for application to access the device's location.

Overall Priority	**Description**
Medium	The search functions can still be used without retrieving the current location of the user.

4.4.2. Stimulus/Response Sequences

Use Case ID:	UC-RCL-001		
Use Case Name:	Retrieve Current Location		
Created By:	Hirashima Shunya	Last Updated By:	Lee Ci Hui
Date Created:	30th August 2022	Date Last Updated:	5th November 2022

Actor:	System
Description:	Retrieval of user's current location.
Preconditions:	1. User gives a permission for the application to access the current location.
Postconditions:	1. The retrieved current location can be used as a departure location in search functionality.
Priority:	Medium

Frequency of Use:	0 - 10 times per day
Flow of Events:	1. The system displays a pop-up window. The uppermost bar of the window displays a bar with the message "use the current location." 2. User taps the "use the current location" bar. 3. The current location is retrieved from the Google Map API and GPS.
Alternative Flows:	AF-S2: If a user does not give a permission to use the current location 1. The uppermost message changes to be "no permission to use the current location."
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

4.4.3. Functional Requirements

4. The Home Screen has the following components:
   - 4.1. The system must display the map of Singapore.
     - 4.1.1. The map must cover the screen.
     - 4.1.2. The system must show the current location of the user by using a circle as a marker.
   - 4.2. The system must display the Search function as described in section 4.
   - 4.3. The system must display the sidebar.
     - 4.3.1. The following tabs must be included as part of the sidebar of the home page.
       - 4.3.1.1. HomePage
       - 4.3.1.2. My Account
       - 4.3.1.3. My Favourite

4.3.1.4. My History

4.3.1.5. Help

4.3.1.6. Log out

4.5.Search Car Park Location

4.5.1. Description and Priority

The user is allowed to search and set their desired start location and destination using the search bar and Google Map Display. The system will then render the top few car parks that are within the radius of 0.5km from the destination for the user to choose.

Overall Priority	**Description**
High	The case is the main functionality of the application. The case includes the searching algorithm and communication with APIs.

4.5.2. Stimulus/Response Sequences

Use Case ID:	UC-SCP-001		
Use Case Name:	Search Car Park Location		
Created By:	Hirashima Shunya	Last Updated By:	Lee Ci Hui
Date Created:	30th August 2022	Date Last Updated:	5th November 2022

Actor:	User
Description:	Search for Car Park Location based on the start location and destination set by the user.
Preconditions:	1. The window is on the Home page.
Postconditions:	1. The system accepts the departure location and the destination.

	2. The system displays a map interface showing pointers on car parks near to the selected destination.
Priority:	High
Frequency of Use:	High
Flow of Events:	1. User taps a searching blank displaying "Search Car Park". 2. The system pops up the "Search Start Location" and "Destination" drawer which contains search bars. 3. User inputs a departure location in the "start location" search bar, which is set as a departure location by the system. 4. User inputs a destination in the "destination" search bar, which is set as a destination by the system. 5. User taps the "Search" button. 6. The system checks for car park availability near the destination. 7. The system displays car parks which are within 0.5km from the destination set.
Alternative Flows:	AF-S7: If a user inputs nothing on the departure location blank 1. The system sets the departure location as the user's current location 2. The system retrieves the user's current location. 3. Return to Step 7.
Exceptions:	Ex1: If a user does not give a permission of using the current location and the system attempts to set the departure location as the current location 1. The system displays an error message: "Please give permission for the usage of the current location or input the departure location."

Includes:	1. Retrieve User's current location 2. Check for Car Park Availability 3. Choose car park
Special Requirements:	-
Assumptions:	1. At least one option can be found.
Notes and Issues:	-

4.5.3. Functional Requirements

5. The system must allow the user to use the search function to search a location.
   5.1. The search function must allow the user to input the following locations:
      5.1.1. Destination
      5.1.2. Departure
         5.1.2.1. By default, the departure location must be the current location.
   5.2. Users must be able to search the location using the following criteria:
      5.2.1. The system must allow the user to search for a location using the name of the place.
      5.2.2. The system must allow the user to search for a location using the address of the place.
      5.2.3. The system must allow the user to search for a location using a postal code.
   5.3. When using the search function, the suggested search results must pop up.
      5.3.1. A message will be displayed if no results are found.

4.6.Check Car Park Availability

4.6.1. Description and Priority

The system will check the availability of car parks and render the information accordingly.

Overall Priority	**Description**
High	The case is the main functionality of the application. The

	case includes the searching algorithm and communication with APIs.

4.6.2. Stimulus/Response Sequences

Use Case ID:	UC-CCA-001		
Use Case Name:	Check Car Park Availability		
Created By:	Hirashima Shunya	Last Updated By:	Lee Ci Hui
Date Created:	30th August 2022	Date Last Updated:	5th November 2022

Actor:	System
Description:	Check whether the respective car park is available.
Preconditions:	1. At least one car park is fetched from Google Maps API. 2. The user had chosen a destination location.
Postconditions:	1. The checked availability is sent to the user interface for searching functionality.
Priority:	High
Frequency of Use:	0 - 10 times per day
Flow of Events:	1. The system fetches car parks that are within a radius of 0.5km from the selected destination from Google Maps API. 2. The system fetches information about availability of each car park. 3. The availability data is tied to the car park location. The availability data is shown on the “Carpark Selected” page.

	4. The system repeats updating the car park availability as a background process after a car park is selected.
Alternative Flows:	-
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

4.6.3. Functional Requirements

6. The system must display available car parks on the map near the car park.
   6.1. The car parks must be in a 0.5km radius from the destination.
   6.2. The system must list out the available car parks in the drawer box pop out from the bottom half of the screen.

4.7.Choose Car Park

4.7.1. Description and Priority

Users choose their desired car park based on the rendered list. They can sort the car parks using the parameters based on their preferences.

Overall Priority	**Description**
High	The case determines which car park to be used for the next functionalities.

4.7.2. Stimulus/Response Sequences

Use Case ID:	UC-CCP-001
Use Case Name:	Choose Car Park

Use Case ID:	UC-CCP-001		
Created By:	Hirashima Shunya	Last Updated By:	Lee Ci Hui
Date Created:	30th August 2022	Date Last Updated:	5th November 2022

Actor:	User
Description:	To select a car park the user wishes to visit and to look for more information.
Preconditions:	1. The system is on the Home page or the Search page. 2. User taps the “Search” button on the Home page.
Postconditions:	1. The selected car park is displayed on the Search page as a destination. 2. The system calculates a route for driving.
Priority:	High
Frequency of Use:	0 - 10 times per day
Flow of Events:	1. The system displays 2 dropdown menus: the one determines the category of sorting and the other determines the order of the result display. 2. The system displays 5 of car park candidates in a pop up drawer on the bottom side of the user interface with trivial information including the distance and the time to be taken. 3. The system displays candidate car parks near the destination in Google Maps using point markers in a circle marker with radius of 0.5km from the destination set by the user. 4. User taps the right-handed arrow symbol with respect to one of the candidate car parks.

	5. The system displays detailed information about the car park including the ratings and comments from past users. 6. The user pushes the navigate button. 7. The system determines trails to car park location using Google Map API. 8. The system displays the trail to the car park location on the map until the user reaches the destination or presses the "Close Navigation" button.
Alternative Flows:	AF-S4: If user desires to choose the other car park 1. User taps a left hand-sided arrow symbol displayed on the "Carpark Selected" page. 2. Return to Step 2.
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

4.7.3. Functional Requirements

7. The caparks must be grouped under the header 'category' as a dropdown box with the following options:
    - 7.1.1. Distance
    - 7.1.2. Fare
    - 7.2. The car parks must be able to be sorted by under the header 'sort by' based on the category selected:
        - 7.2.1. If the category is "distance", the available sort options are the followings:
            - 7.2.1.1. Nearest to Furthest(Departure to Carpark)
            - 7.2.1.2. Furthest to Nearest(Departure to Carpark)
            - 7.2.1.3. Nearest to Farthest(Carpark to Departure)

7.2.1.4. Farthest to Nearest(Carpark to Departure)

7.2.2. If the category is “fare”, the available sort options are the followings:

7.2.2.1. Cheapest to Most Expensive

7.2.2.2. Most Expensive to Cheapest

8. The system must allow the user to select the desired carpark.

8.1. The system must display information related to the selected carpark

8.1.1. The following information must be included

8.1.1.1. Carpark name

8.1.1.2. Carpark location address

8.1.1.3. Driving time/distance (from departure to carpark)

8.1.1.4. Walking time/distance (from carpark to destination)

8.1.1.5. Carpark parking rate

8.1.1.5.1. There must be carpark calculator

8.1.1.5.1.1. User will input the number of hours they intend to stay for

8.1.1.5.1.2. Calculator will calculate the parking fare in dollars according to what the user has inputted.

8.1.1.6. Carpark availability

8.1.1.7. Carpark operating hours

8.1.1.8. Comments and ratings by past users

8.2. There must be a “Navigate” button that directs the user to Google Map rendered in the HomePage.

8.2.1. The departure location in Google Maps must be the departure location selected by the user.

8.2.2. The destination location in Google Maps must be the Carpark selected by the user.

8.3. There must be a “Start Parking” button for the user to press once the user reaches the car park.

4.8.Parking Settings (Customize Helper Features)

4.8.1. Description and Priority

Once the user reaches the desired car park lot, the application allows the user to utilize the helper features which includes fares estimation calculator, set timer, and add images and notes in the note-taking section.

Overall Priority	**Description**
High	The case involves the determination of timer, fare estimation, and adding an image and a note to a selected carpark. The case ends when the user parks.

4.1.2. Stimulus/Response Sequences

Use Case ID:	UC-PSC-001		
Use Case Name:	Parking Settings (Customize Helper Features)		
Created By:	Hirashima Shunya	Last Updated By:	Lee Ci Hui
Date Created:	30th August 2022	Date Last Updated:	5th November 2022

Actor:	User
Description:	Allow user to estimate fares, set timer, add images and add notes
Preconditions:	1. User pressed the “Start Parking” button and it is in the Parking Setting page.
Postconditions:	1. The system displays the navigation for destination or the selected car park.
Priority:	High
Frequency of Use:	High
Flow of Events:	1. The system displays Estimation End Time, Timer, Add Image, Add Note buttons in the parking setting page.

	2. If the user selects the Add Image box, then the system directs the user to select a photo from the device's gallery or Google Drive and then displays the image on the Add Image box. 3. If a user enters text in the Add Note box, then the system saves the user's notes in the note-taking section. 4. If the user selects Estimation End Time, then the user uses the included use case Estimation Fare. 5. If the user sets a timer by selecting the Timer buttons, then the user uses the included use case Start Timer. 6. User presses the Continue button. 7. The system displays the Parking Ongoing Screen.
Alternative Flows:	-
Exceptions:	-
Includes:	1. Estimation Fare 2. Start Timer
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

4.8.3. Functional Requirements

9. After the user press "start parking" button, the system must show the followings:
    9.1. The car park address.
    9.2. The system must allow the user to enter user's parking time estimation
        9.2.1. The user must be able to customize the hour.
            9.2.1.1. The hour must be in integer data type
            9.2.1.2. The hour must have the range from 0 to 23.
            9.2.1.3. The default number is 0.
        9.2.2. The user must be able to customize the minute.

9.2.2.1. The minute must be in integer data type

9.2.2.2. The minute must have a range from 0 to 59.

9.2.2.3. The default number is 0.

9.3. The system must calculate and display the estimated car parking fare.

9.3.1. The format of calculation is the following:

9.3.1.1. The minute set by the user must convert into an hour.

9.3.1.2. The converted minute must be added to the hour.

9.3.1.3. The formula is [total hour] x [car parking rate] = [total amount in Singapore dollar]

9.4. The system must allow the user to set a timer.

9.4.1. The details are covered under *section 12*.

9.5. The system must allow the user to input note as following:

9.5.1. The system must allow users to upload a photo.

9.5.1.1. The size of the photo must be less than 500MB.

9.5.2. The system must allow the user to input word notes.

9.5.3. The word notes must be in string data type.

9.6. There must be a “continue” button.

10. When the user has parked the car, in which after “continue” button is pressed

10.1. The system must display the followings:

10.1.1. Car Park location address

10.1.2. Start time

10.1.2.1. The start time is taken at the moment when the user presses the “continue” button.

10.1.3. Estimated end time

10.1.3.1. The estimated end time is calculated by adding the estimated hour and minute set by the user to the start time.

10.1.4. Timer

10.1.5. Photo uploaded and note written.

10.2. There must be a “Navigate” button at the bottom left corner of the screen with a “from carpark to destination” label that directs the user to Google Map rendered at Homepage.

10.2.1. The departure location in Google Maps must be the carpark location selected by the user.

10.2.2. The destination location in Google Maps must be the destination selected by the user.

10.3. There must be a “Navigate” button at the bottom right corner of the screen with “from destination to car park” label that directs the user to Google Map rendered at Homepage.

10.3.1. The departure location in Google Maps must be the destination selected by the user.

10.3.2. The destination location in Google Map must be the carpark selected by the user.

10.4. There must be an “end parking” button at the bottom of the screen.

4.9.Fare Estimator

4.9.1. Description and Priority

The user set the estimated end time of their car parking session. The system will then help to calculate the car parking fare incurred.

Overall Priority	**Description**
Medium	The case involves the fare estimation. The case ends when the user parks.

4.9.2. Stimulus/Response Sequences

Use Case ID:	UC-FER-001		
Use Case Name:	Fare Estimator		
Created By:	Hirashima Shunya	Last Updated By:	Lee Ci Hui
Date Created:	30th August 2022	Date Last Updated:	5th November 2022

Actor:	User
Description:	Estimate fare needed based on user desired duration of parking.
Preconditions:	1. The user selects Estimation buttons in the Parking Setting page. 2. The system is able to retrieve the user's parking location's parking fare.
Postconditions:	-
Priority:	Medium
Frequency of Use:	0-10 times per day
Flow of Events:	1. User input his estimation duration of car parking. 2. The system calculates the estimated fare needed for the user estimated duration. 3. The system saves the estimated fare and displays it in the Parking Setting page.
Alternative Flows:	-
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

4.9.3. Functional Requirements

11. The user must be allowed to input the estimated end time.
    11.1. The system calculates the duration based on the start time and the inputted end time.
    11.2. The system calculates the fare incurred based on the duration and the car park rate.
    11.3. The system displayed the calculated fare.

4.10.Timer

4.10.1. Description and Priority

The user sets the time he wishes the system to alert him before the reaches the estimated end time set. The system alerts the user by sending a notification when the countdown timer reaches the time set.

Overall Priority	**Description**
Medium	The case involves the Timer. The case ends when the timer ends.

4.10.2. Stimulus/Response Sequences

Use Case ID:	UC-TMR-001		
Use Case Name:	Timer		
Created By:	Hirashima Shunya	Last Updated By:	Lee Ci Hui
Date Created:	30th August 2022	Date Last Updated:	5th November 2022

Actor:	System
Description:	Activation of the set timer.
Preconditions:	1. The system is on the “Parking Ongoing” page.
Postconditions:	1. The timer system activates the set timer.
Priority:	High
Frequency of Use:	0 - 10 times per day

Flow of Events:	1. User taps the “Continue” button on the “Parking Settings” page. 2. The system makes the window display the “Parking Ongoing” page. 3. When the user taps the “Continue” button on the “Parking Settings” page, the system starts the timer set by the user. 4. The system makes the window display the “Parking Ongoing” page. 5. In the middle of the window, the system makes the window display “alert { set time } mins before end times.” by calculating from a set timer. 6. The system starts the timer set by the user. 7. The system send a notification to alert user when the timer ends.
Alternative Flows:	AF-S5: If no timer is set 1. The system does display the set time as 0 minutes. 2. Return to Step 6.
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

4.10.3. Functional Requirements

12. The system must allow the user to set a timer.

12.1. The system must allow the user to customize the minute.

12.1.1. The minute must be integer data type.

12.1.2. The minute must be in the range from 0 to 59.

12.1.3. The default number is 0.

12.2. The system must send notification to the user based on the selected minute before the end time estimated by the user.

12.3. The system must calculate the end time by adding the hours and minutes set by the user to the start time, which is the current time once the user presses the “continue” button.

4.11.Navigation

4.11.1. Description and Priority

The system displays the routes for start location to car park, car park to destination and destination to car park based on the button selected by the user.

Overall Priority	**Description**
Medium	The case involves the Navigation. The case ends when the user parks.

4.11.2. Stimulus/Response Sequences

Use Case ID:	UC-NAV-001		
Use Case Name:	Navigation		
Created By:	Hirashima Shunya	Last Updated By:	Lee Ci Hui
Date Created:	30th August 2022	Date Last Updated:	5th November 2022

Actor:	User
Description:	Show trail from car park to destination and vice versa.
Preconditions:	1. The system is on the Parking Ongoing page. 2. User requests to determine the trail route by tapping the “Navigation” button.
Postconditions:	1. The system calculates and finds the trail route.

Priority:	Medium
Frequency of Use:	0 - 10 times per day
Flow of Events:	1. User taps the “Navigation” button on the bottom. 2. The system directs users to Google Maps rendered in application. 3. The system concurrently displays the route and current location of the user. 4. The user current location marker moves accordingly.
Alternative Flows:	AF-S1: User taps the “Navigation” button on the right side. 1. The system directs the user to Google Maps rendered. 2. The system concurrently sends information about the location of the car park as a destination and the destination that a user has already input as a departure location to the Google Map.
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

4.11.3. Functional Requirements

13. The user selects the “Navigation” button.

13.1. The system changes to “HomeScreen” and shows the route requested by the user on the Google Map rendered.

13.2. The current location is shown on the map, and moves according to the user's current location.

4.12.Ratings and Comments

4.12.1. Description and Priority

When the car park session finishes, the user can rate and comment on the car park visited.

Overall Priority	**Description**
Medium	The case is for the better user experience: users can enjoy rate and comment on a car park they used.

4.12.2. Stimulus/Response Sequences

Use Case ID:	UC-RAC-001		
Use Case Name:	Ratings and Comments		
Created By:	Hirashima Shunya	Last Updated By:	Lee Ci Hui
Date Created:	30th August 2022	Date Last Updated:	5th November 2022

Actor:	User
Description:	A space for rating and commenting for a car park.
Preconditions:	1. The system is on the Comment and Ratings page.
Postconditions:	User's rate and comment are stored on a cloud database.
Priority:	Medium
Frequency of Use:	0 - 5 times per day
Flow of Events:	User fills comments on the Comment box. User taps on a number of stars upon the five white stars to rate the car park out of 5 points. User taps the "Done" button. The system sends the content of the rating and commenting to the cloud database. The ratings and comments section of the information page of this car park is updated with the latest comment and rating. The system displays the Home page UI.

Alternative Flows:	-
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

4.12.3. Functional Requirements

14. The system must allow the user to comment and rate the carpark.
15. The rating must be a 1-5 star rating system.
16. The system must allow the user to type out the comment in the box.
    16.1. The comment must be in string data type
17. There must be a “done” button
    17.1. If the user presses this button, it represents the whole parking assistance service is ended and must direct the user back to Homepage

4.13.Menu Bar (Side Drawer)

4.13.1. Description and Priority

The system displays the Menu Bar - a sliding drawer which includes touchable containers of Home Page, My Account, My Favourite, My History, Help Page and Log out.

Overall Priority	**Description**
High	The case involves the choices for Home Page, My Account, My Favourite, My History, Help Page and Log out

4.13.2. Stimulus/Response Sequences

Use Case ID:	UC-MNB-001		
Use Case Name:	Menu Bar (Side Drawer)		
Created By:	Hirashima Shunya	Last Updated By:	Lee Ci Hui
Date Created:	30th August 2022	Date Last Updated:	5th November 2022

Actor:	User
Description:	A user interface to switch pages on the application.
Preconditions:	-
Postconditions:	1. The window moves to a selected page.
Priority:	High
Frequency of Use:	0 - 30 times per day
Flow of Events:	1. User taps the Menu Bar icon. 2. The system displays the list of options: ‘HomePage’, ‘My account’, ‘My favorites’, ‘My history’, ‘Help’ and ‘Log out’. 3. The user can choose to click any of the following of the above options. 4. The system makes the window display the selected page.
Alternative Flows:	AF-S2: If a user taps the sidebar again 1. The system closes the list of options.
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

4.13.3. Functional Requirements

17. The following tabs must be included as part of the sidebar of the home page
    17.1. My account
    17.2. My favorites
    17.3. My history

4.14.Add Favourites List

4.14.1. Description and Priority

The system displays the My Favourite page.

Overall Priority	**Description**
Medium	The case displays the My Favourite page

4.14.2. Stimulus/Response Sequences

Use Case ID:	UC-FAV-001		
Use Case Name:	My Favourite Page		
Created By:	Hirashima Shunya	Last Updated By:	Lee Ci Hui
Date Created:	30th August 2022	Date Last Updated:	5th November 2022

Actor:	User
Description:	Addition of Car Park to Favorite List.
Preconditions:	1. The window is on either the user profile or the end of the car parking page.
Postconditions:	1. A selected car park is added to a favorite list.
Priority:	Medium
Frequency of Use:	0 - 5 times per day
Flow of Events:	1. User taps an unfilled heart-shaped diagram shown beside the Car Park description.

	2. The system retrieves the selected car park information from the CarparkAPI. 3. The system verifies the existence of the selected car park in the user's favorite list in the cloud database. 4. The car park is added to the user's favorite list. The heart-shaped diagram is filled with red.
Alternative Flows:	AF-S3: If the selected car park already existed in the user's favorite list. 1. The system displays a pop-up notification "This car park has already been in your favorite list. Would you like to delete it?" with "Yes" and "No" tabs. 2. If the user taps "Yes", the system will remove the selected car park from the user's favorite list in the cloud database.
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

4.14.3. Functional Requirements

18. My favorites

18.1. Shows a list of car parks saved by the user by clicking the heart shape.

4.15. Upload Photo

4.15.1. Description and Priority

The system allows user to upload his profile picture

Overall Priority	**Description**
Medium	The case displays the Photo uploading function.

4.15.2. Stimulus/Response Sequences

Use Case ID:	UC-FAV-001		
Use Case Name:	Upload photo (Profile Picture)		
Created By:	Hirashima Shunya	Last Updated By:	Lee Ci Hui
Date Created:	30th August 2022	Date Last Updated:	5th November 2022

Actor:	User
Description:	Uploading a photo used for an icon of a user profile.
Preconditions:	1. The window is on the user profile page.
Postconditions:	1. An icon photo of a user profile is updated. 2. The photo is stored on a cloud database.
Priority:	Medium
Frequency of Use:	1 - 3 times per lifetime
Flow of Events:	1. User taps the profile icon. 2. The system accesses and displays the user's photo gallery or Google Drive. 3. User chooses a photo from the user's photo gallery or Google Drive. 4. User confirms a photo. 5. The system returns the window to the user profile page and the icon photo is updated.
Alternative Flows:	AF-S2: If a user taps the "not now" block 1. The system returns the window to the user profile page. AF-S4: The system unable to access user's photo album 1. The system displays an error message "Unable to access photo gallery. Please allow permission"

Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	jpg and png files are supported.
Notes and Issues:	-

4.15.3. Functional Requirements

19. The system must allow users to upload a photo.
    19.1. The size of the photo must be less than 500MB

4.16. Update User Profile

4.16.1. Description and Priority

The system displays the MyAccount Page. The user is allowed to edit the information of their account.

Overall Priority	**Description**
Medium	The case displays the MyAccount Page. The user is allowed to edit the information of their account.

4.16.2. Stimulus/Response Sequences

Use Case ID:	UC-FAV-001		
Use Case Name:	My Favourite Page		
Created By:	Hirashima Shunya	Last Updated By:	Lee Ci Hui
Date Created:	30th August 2022	Date Last Updated:	5th November 2022

Actor:	System
Description:	An interface for updating the user profile.

Preconditions:	1. The request of adding a new car park to the favorite list or updating an icon photo is sent from user.
Postconditions:	1. The information related to the user profile on a database is updated.
Priority:	High
Frequency of Use:	0 - 10 times per day
Flow of Events:	1. When a user taps the 'my account' on a submenu, the system displays the user interface including first name, last name, and email of the user. 2. User clicks the 'Edit' button. 3. User inputs particulars to each blank. 4. After filling particulars, the user taps the 'Edit' button again. 5. If a user updates the email address, the system checks whether it is unique and satisfies the requirements. 6. If a user updates the username, the system checks whether the username is unique. 7. The system updates user information on a cloud database.
Alternative Flows:	AF-S6: If the email does not fulfill the requirements 1. The system prompts the user to enter email again.
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

4.16.3. Functional Requirements

20. My account

20.1. Shows user account profile which includes the following:

20.1.1. First Name

20.1.2. Last Name

20.1.3. Email

20.1.4. Profile photo

20.1.4.1. The default photo is that of a car.

20.2. User must be able to edit the information in 20.1.

20.3. User must be able to change their password by clicking the “Change password”.

20.4. There must be a graph showing the total parking fare spent per month.

21. User must be allowed to customise the year.

22. User must be allowed to customise the month.

22.1. The months must have the following categories:

22.1.1. January - April

22.1.2. May - August

22.1.3. September - December

5.Other Nonfunctional Requirements

5.1.Performance Requirements

5.1.1 System Specification

1. *EzPark* requires a device with at least 2 GB of RAM, a quad core CPU at 1.4 GHz and a stable network connection.

1.1. The software’s act depends on the device’s HTTP communication speed and the amount of data communicated with the cloud database.

5.1.2 System Response Time

2. System must be able to respond to the user's input within 5 seconds.

2.1. The system must load data within 5 seconds upon launching.

2.2. The system must not crash when the user opens the application.

2.3. When a user launches the route search functionality, The system must display the result within 30 seconds.

2.4. The system must detect the difference in the number of car parks every 5 minutes.

2.5. When the system displays a notification, the system must show a pop-up window in less than 3 seconds.

2.5.1. The pop-up must be with a button for closing the pop-up window.

2.6. When the system migrates to another display without access to a database, the system must display the consequent display in up to 8 seconds.

5.2.Safety Requirements

5.2.1 Safety of Database

3. Since the software utilizes Firebase Realtime Database managed by Google, all risk is caused by Firebase server; therefore, the safety is managed by Google.

5.2.2 Accuracy of Fare Shown

4. System must be able to show the car park rate (fare per hour) accurately rendered from the car park API.

4.1. System must present the fare of car park in the correct currency, ie. Singapore Dollar (SGD).

4.2. System must present the fare of the car park up to 2 decimal points behind a period (.) symbol.

5.3.Security Requirements

5.3.1 Safety of Database

5. Since the software utilizes Firebase Realtime Database managed by Google, all risk is caused by Firebase server; therefore, the safety is managed by Google.

5.3.2 Required for Login

6. System must make sure the user is logged in before access to any functionalities provided within the application.
    6.1. System must ensure the user is logged in with their registered email and password.
    6.2. System must verify the credentials of the user.
    6.3. System must hash all passwords using the SHA-256 algorithm before storing them into the database.

5.4.Software Quality Attributes

5.4.1 Reliability

7. The system must be reliable.
    7.1. The fetched website must be secured.
    7.2. The system must fetch websites protected by utilizing SSL.
    7.3. The system must show users up-to-date information fetched from APIs.
        7.3.1. The system allows up to 15 seconds of delay.
    7.4. The system must be maintained with little or no downtime occurring.
    7.5. The system must support the internal locus of control.
        7.5.1. The system must respond without any lag or latency.

5.4.2 Usability

8. System must ensure the design is user friendly.
    8.1. The system must reduce short-term memory load.
        8.1.1. To keep the display simple.
    8.2. The system must display all contents in English.
    8.3. The system must offer informative feedback.
        8.3.1. When the system detects invalid inputs, the system must display an error message.
        8.3.2. When certain processes fail, the system must display an error message according to each error.
            8.3.2.1. After showing the error message, the system returns to the initial display for each functionality.

8.4. The system must strive for consistency.

8.4.1. The system must retain a consistent visual layout.

8.4.1.1. For example, fonts, colors, placements of buttons.

5.5.Business Rules

5.5.1 Operating Principle

9. Under certain circumstances, respective individuals need to perform their roles respectively.

9.1. If the software encounters a misbehavior in manipulation of the database, a data engineer needs to respond to it.

9.2. If the software encounters a misbehavior in access to cloud services, a cloud engineer needs to respond to it.

9.3. If the software encounters a misbehavior in User Interface, a front-end UI engineer needs to respond to it.

Appendix A: Data Dictionary

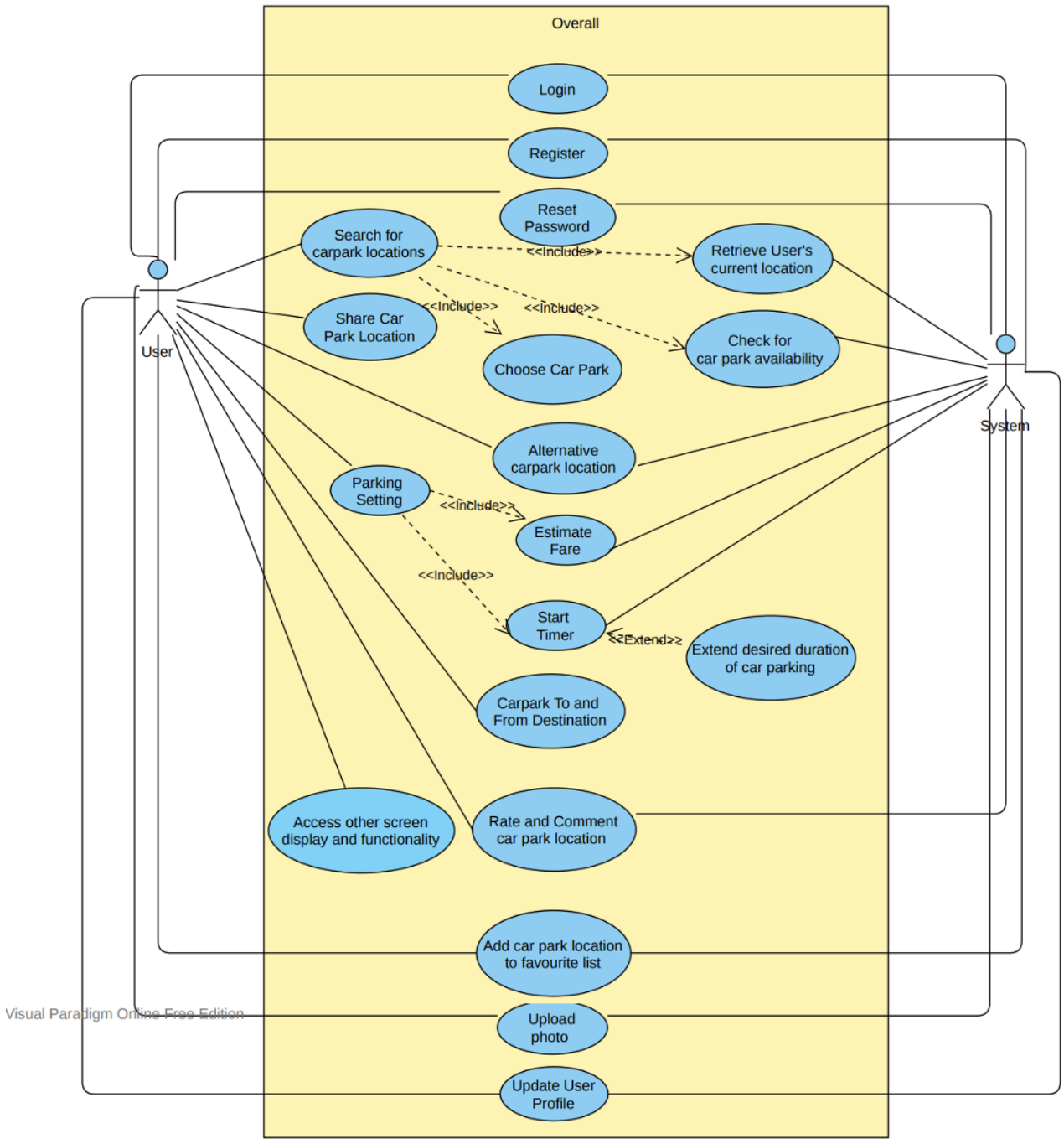
User	A user is a person who registered an account in the *EzPark* mobile application. This individual is entitled to access all functionality including but not limited to search desired car park and utilize the helper features.
System	A system refers to the EzPark Android-based Mobile Application.
Login	An act of starting to use the application by inputting username and password.
Account	An Account holds user's personal information with historical data tied to the application.
Shut down	An act of exiting from the application.
Carpark	A carpark is a garage for parking used by visitors.
Departure	The departure is the place that the user set as their desired starting point.
Destination	The destination is the place that the user set to which the user desires to go.
Current Location	The current location refers to the place in which the user is.
My History	My history is a list of Car Parks that the user visited in the past and contains the corresponding details that are being saved in the database.
My favourites	My favourites is a list that the user can add items into. The user can save their favourite car parks into this list and be allowed to directly start their journey to this car park in *EzPark* by selecting it.
Travel	Travel refers to the one consecutive flow from a departure to arriving at a destination.
Map	A map indicates the Google Map rendered and displayed on the application.

Display	To display is an act of showing user interfaces of the application on the user's android smartphone.
Search	To search is an act to collect information related to the locations inputted and rendered the car parks available within 0.5km radius from the destination inputted.
Search Function	The search function is a function of searching places using Google Cloud Places API with Google Autocomplete API.
Menubar	The Menu Bar is a UI displayed on the top-left of the application window to open the side drawer.
Available	The car park is available when a car park has at least one vacancy.
Availability	Availability refers to whether a chosen carpark is available.
Button	A button is a displayed block that acts something when being pushed.
Algorithm	An algorithm is an advanced feature that calculates the best route by combining Car park fee rate, Opening time, Availability and Distance from location.
Notification	A notification is a small window that notifies a user in prompt. It popped up on the window with a message.
Note	A note is a place where a user can write text-based descriptions on the carpark as a memo and upload an image.
Comment	A comment is a user's description that indicates how the user thinks about the car park.
Rating	A rating displays a user's satisfaction of visiting the car park. This consists of 5 stars; user's satisfaction increases in ascending order.

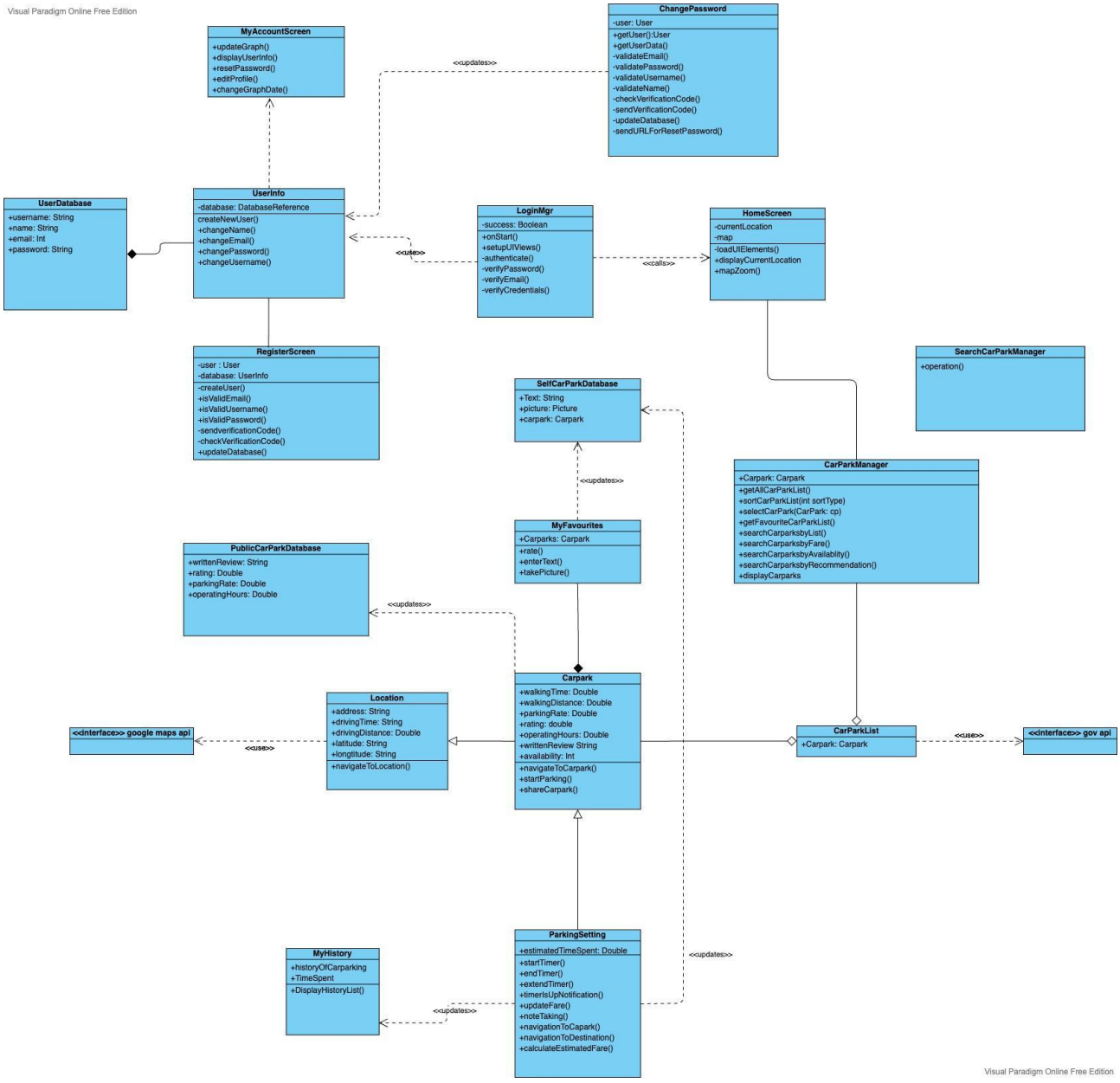
Fare	Total cost incurred when parking at that respective car park. The fare is displayed in Singapore Dollar (SGD).

Appendix B: Analysis Models

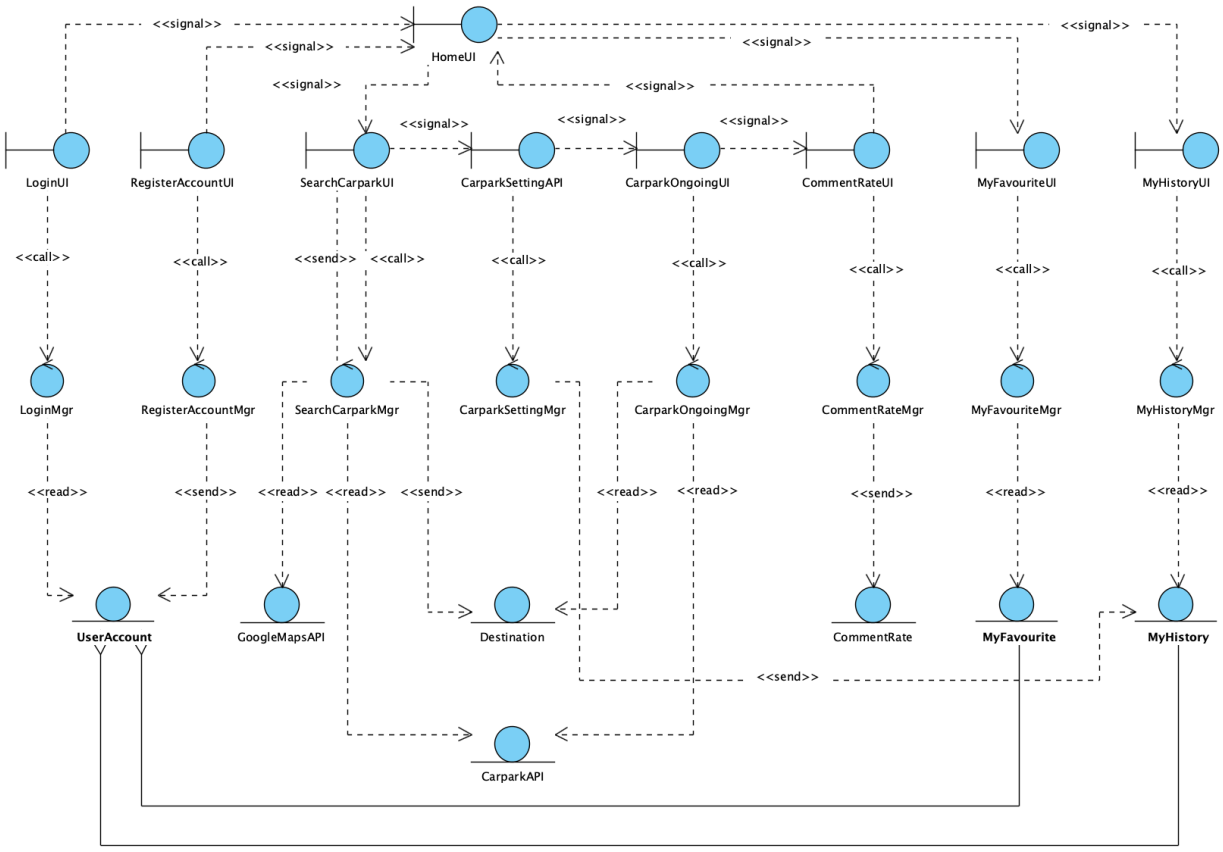
B1. Use Case Model



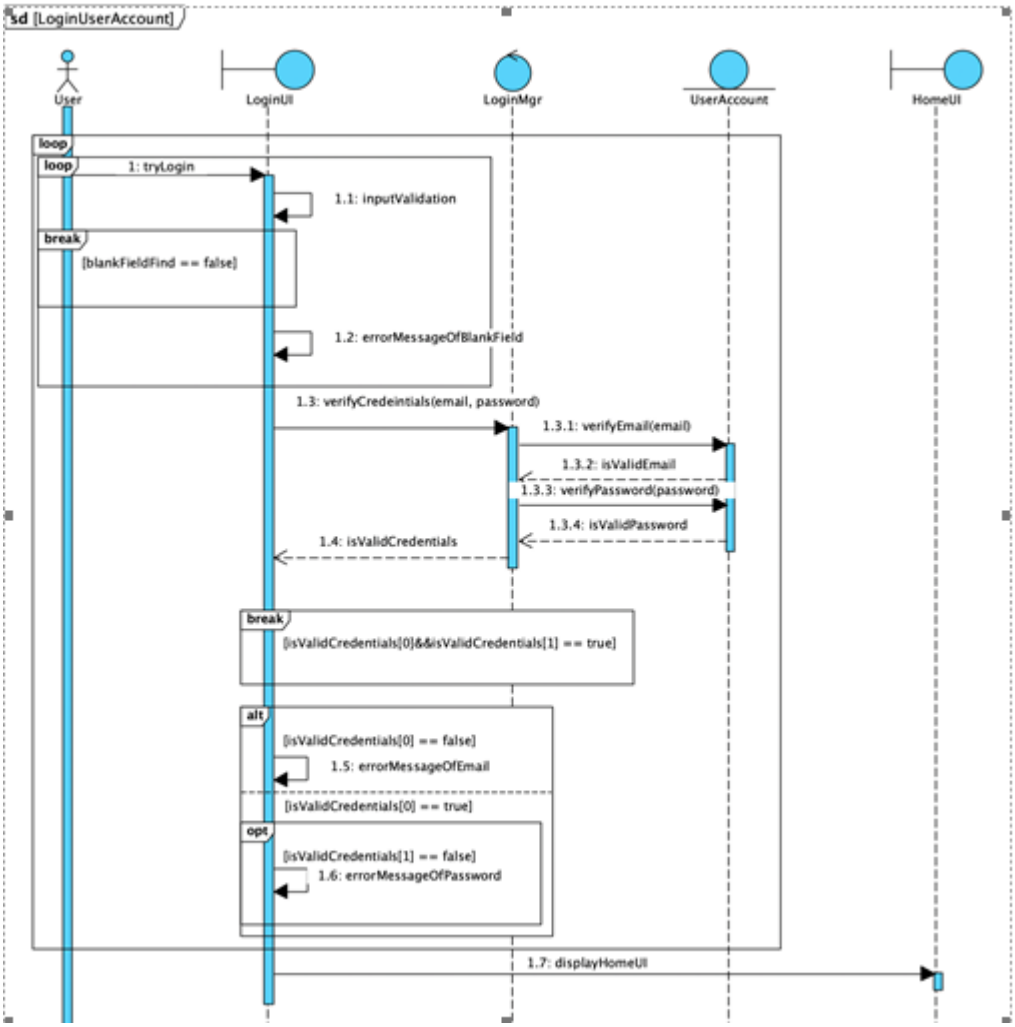
B2. Entity Class Diagram



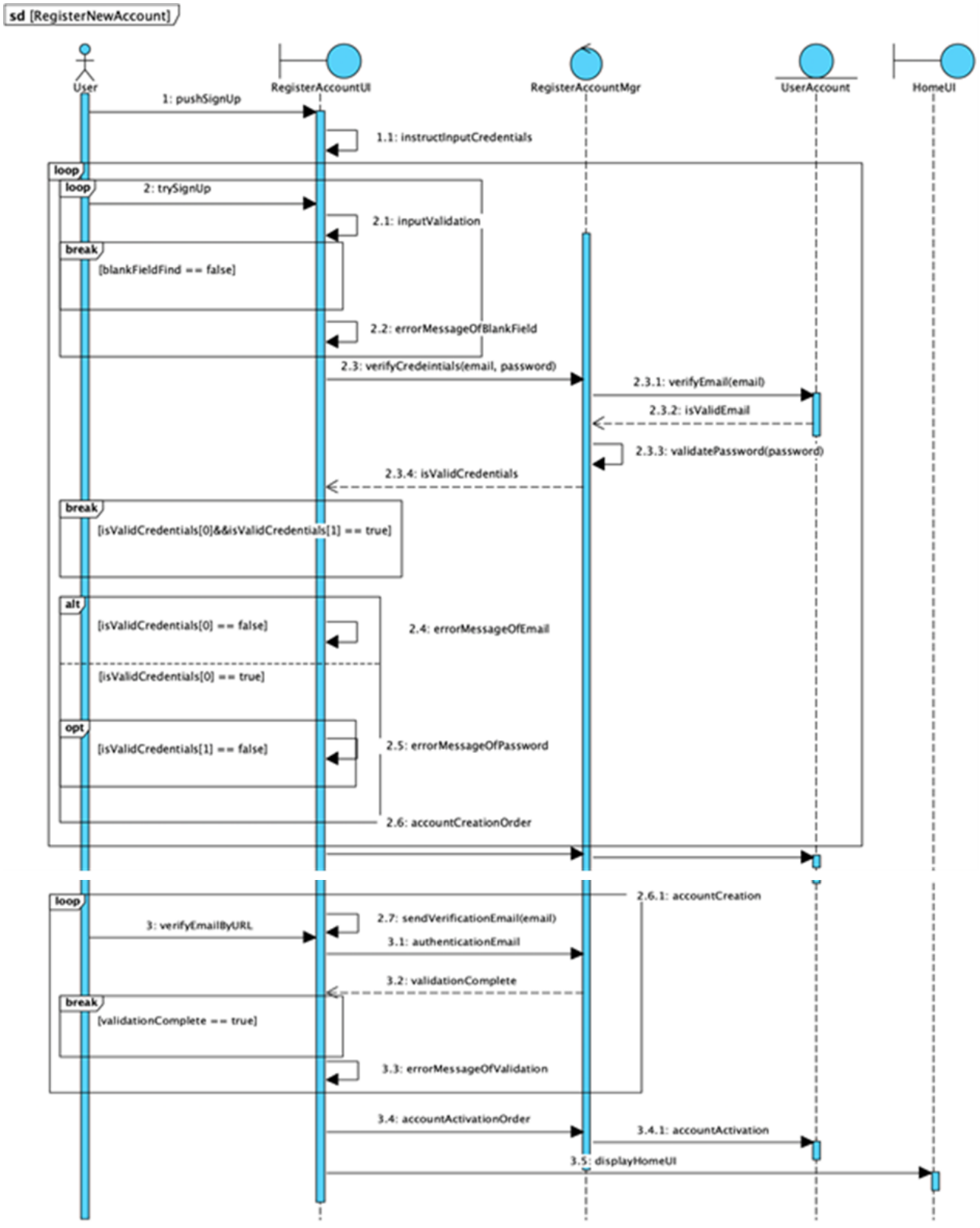
B3. Control and Boundary Class Diagram



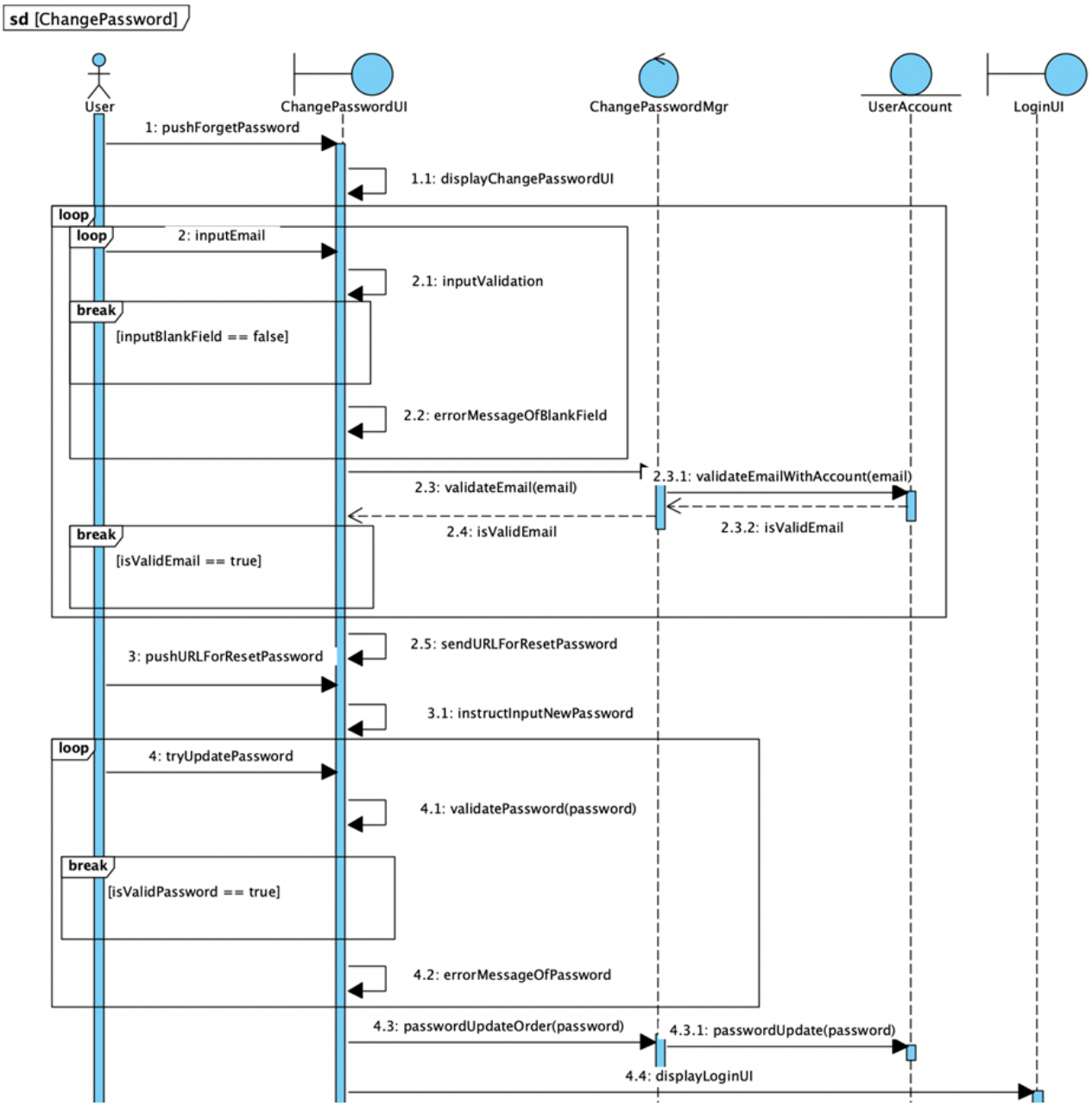
B4. Sequence Diagram



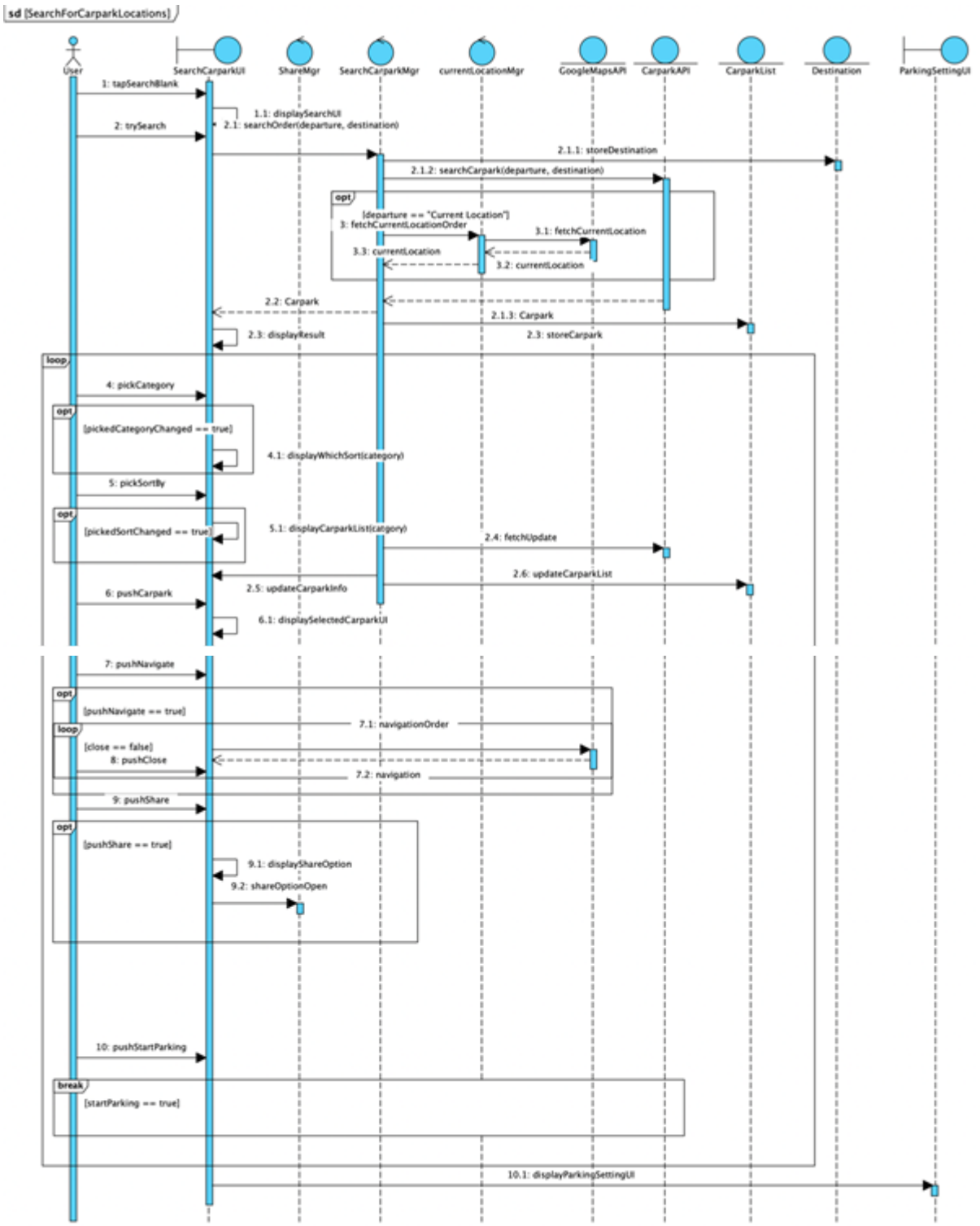
SEQUENCE DIAGRAM – LOGIN



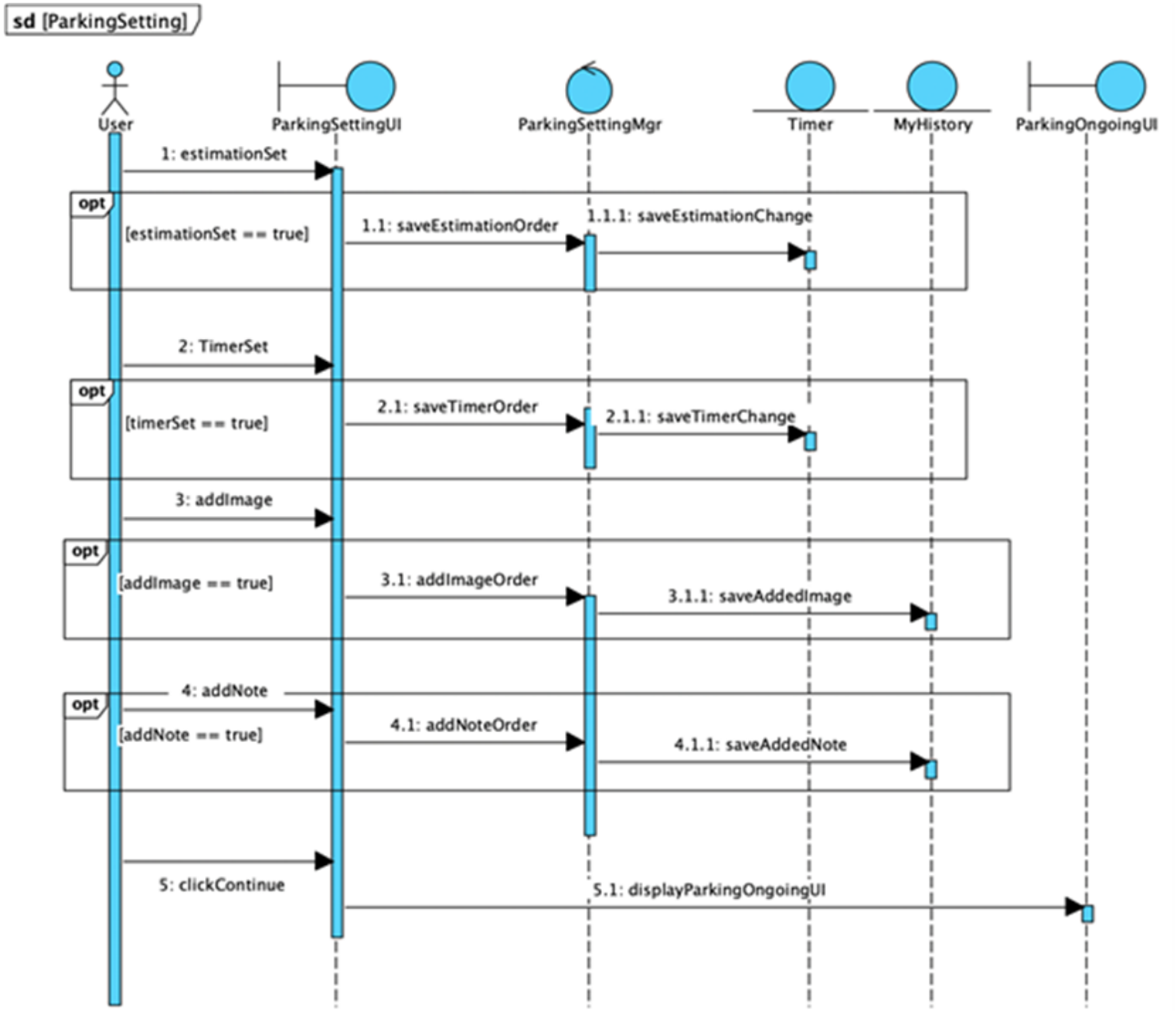
SEQUENCE DIAGRAM – REGISTER ACCOUNT



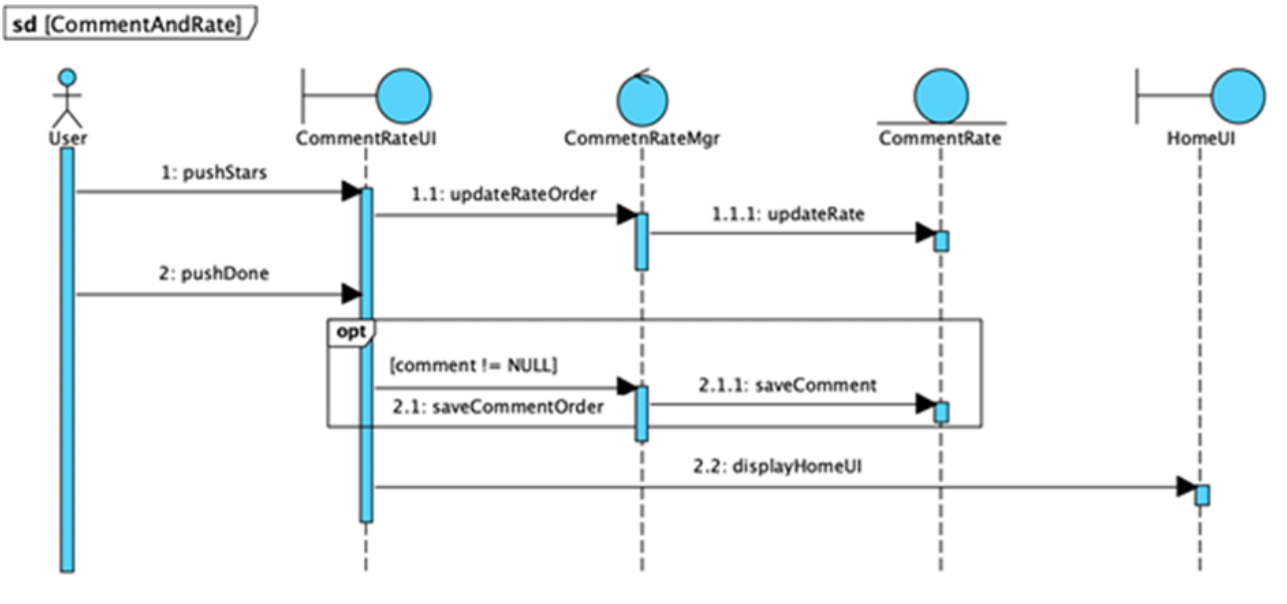
SEQUENCE DIAGRAM - CHANGE PASSWORD



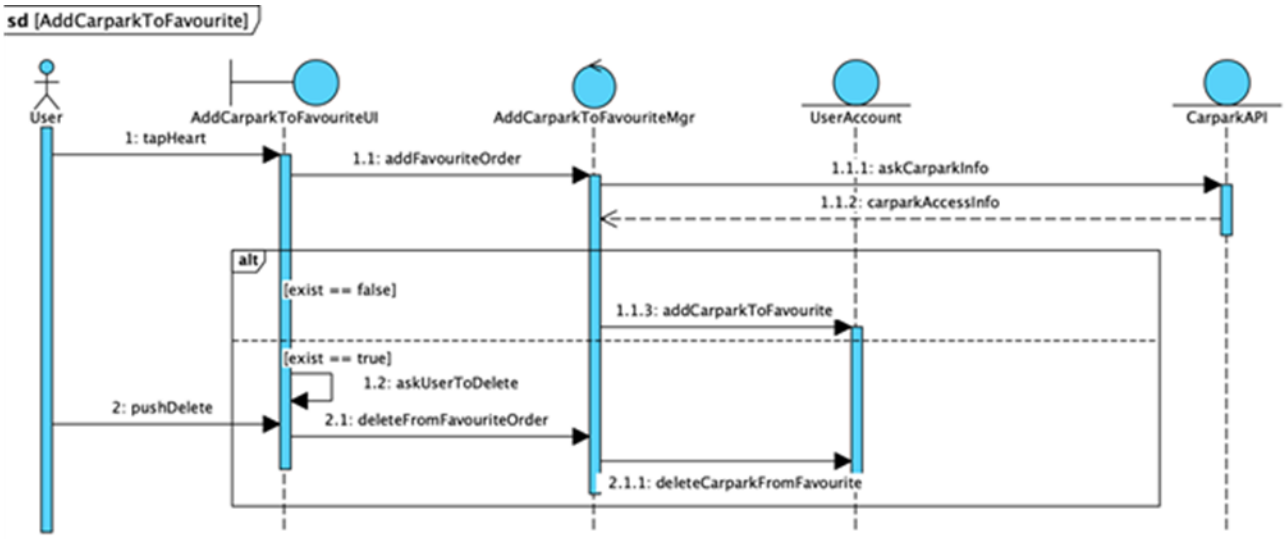
SEQUENCE DIAGRAM – SEARCH



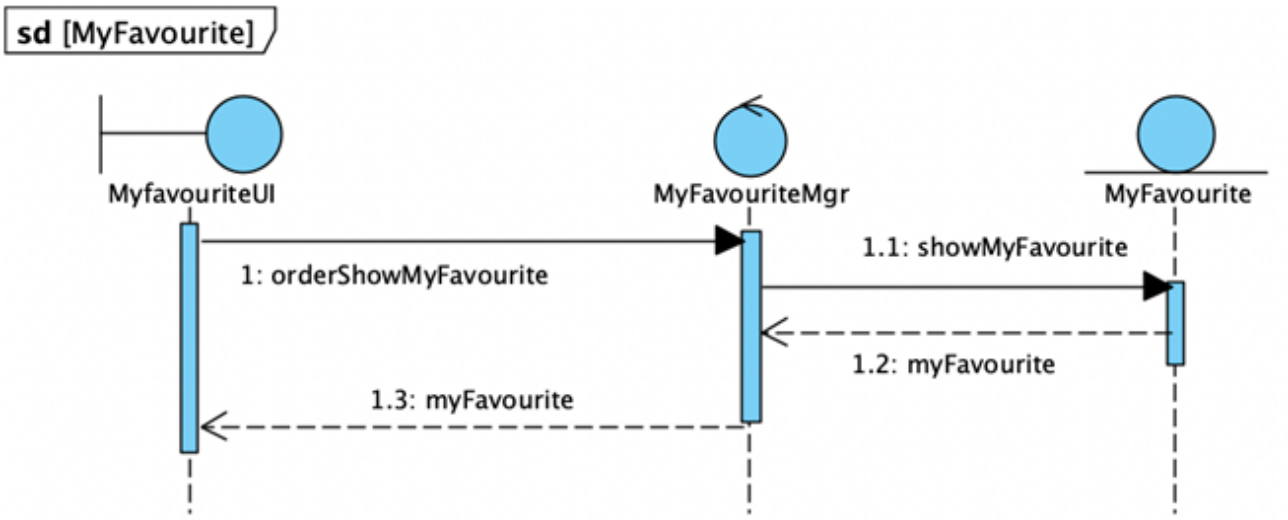
SEQUENCE DIAGRAM – PARKING SETTING



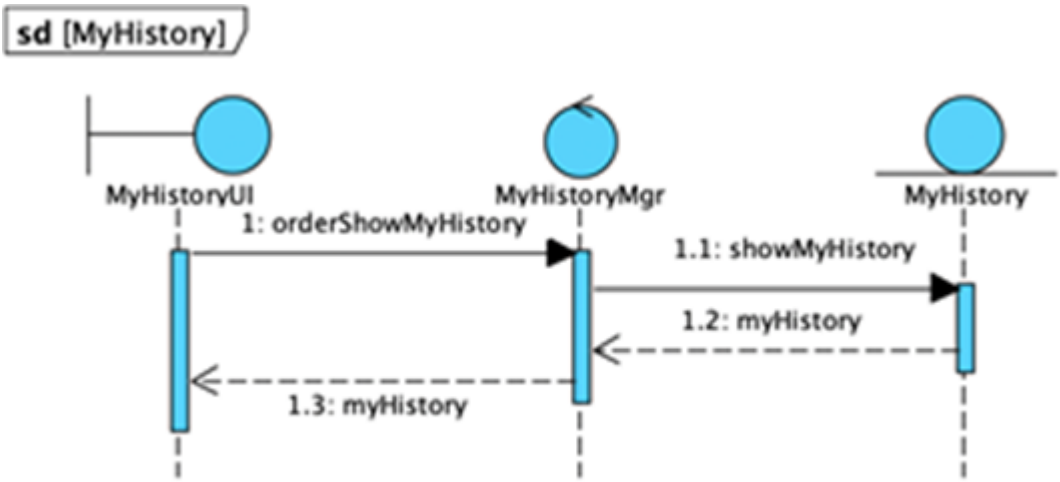
SEQUENCE DIAGRAM – COMMENT AND RATE



SEQUENCE DIAGRAM – ADD CARPARK TO FAVORITE

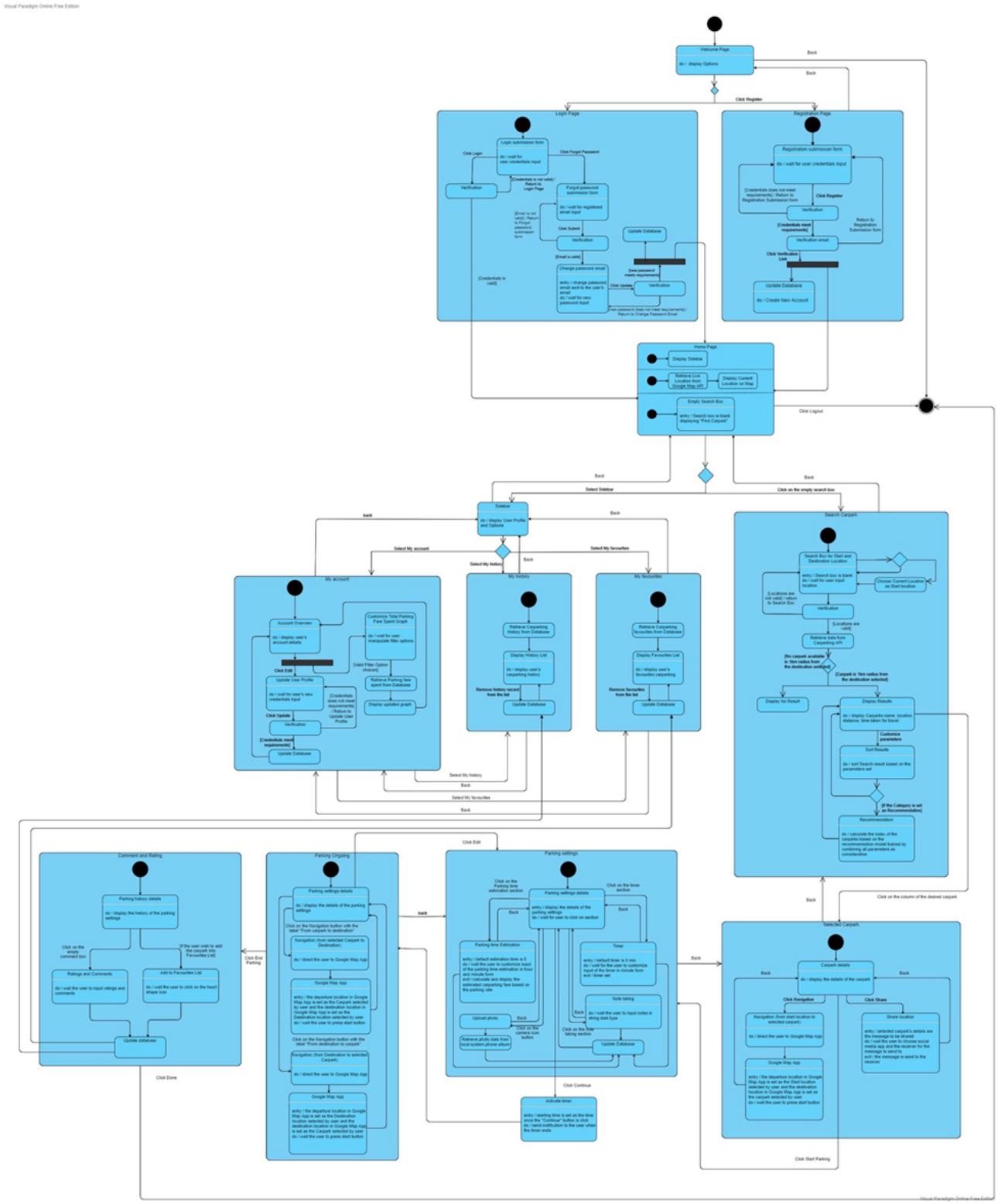


SEQUENCE DIAGRAM – MY FAVORITE



SEQUENCE DIAGRAM – MY HISTORY

B5. State Machine Diagram



B6. System Architecture Model