



SC2006 Software Engineering

Lab#5 Deliverables

Lab Group: SDAA

Group Name: Team 3

Group Members

Name	Matriculation No.
Ong Sheng Da	U2220668C
Shan Yi	U2222846C
Pala Tejaswi	U2221881D
Ng Zi Xuan	U2220246D
Ong Xin Ning, Trini	U2222202K

Table of Contents

Table of Contents	2
Revision History	4
1. Requirements	5
1.1 Introduction	5
1.1.1 Purpose	5
1.1.2 Document Conventions	5
1.1.3 Intended Audience and Reading Suggestions	5
1.1.4 Product Scope	5
1.1.5 References	6
1.2. Overall Description	6
1.2.1 Product Perspective	6
1.2.2 Product Functions	6
1.2.3 User Classes and Characteristics	7
1.2.4 Operating Environment	7
1.2.5 Design and Implementation Constraints	8
1.2.6 User Documentation	8
1.2.7 Assumptions and Dependencies	8
1.3. External Interface Requirements	9
1.3.1 User Interfaces	9
1.3.2 Hardware Interfaces	9
1.3.3 Software Interfaces	9
1.3.4 Communications Interfaces	10
1.4. System Features	10
1.4.1 Create Account	10
1.4.2 User Login	11
1.4.3 GPS and Map View	11
1.4.4 Filter	11
1.4.5 Car Park Information	12
1.4.6 Favourite	13
1.4.7 Report	13
1.5. Non-Functional Requirements	15
1.5.1 Performance Requirements	15
1.5.2 Safety Requirements	15
1.5.3 Security Requirements	15
1.5.5 Business Rules	16
1.6. Use Case Diagram	16
1.7. Use Case Descriptions	17
1.8. Data Dictionary	47
1.9. UI Mockups	49
1.9.1 Homepage UI	49

1.9.2 Login UI	49
1.9.3 Create Account UI	50
1.9.4 Forget Password UI	50
1.9.5 Reset Password UI	51
1.9.6 Search Carpark UI	51
1.9.7 Choose Carpark UI	52
1.9.8 Map UI	52
1.9.9 Favourite Carpark UI	53
1.9.10 Report Fault UI	53
1.9.11 Admin UI	54
2. Design	55
2.1. Class Diagram	55
2.1.1 Class Stereotype Diagram	55
2.1.2 Class Entity Diagram	55
2.1.3 Key Public Method List	56
2.2. Sequence Diagrams	61
2.2.1 Login	61
2.2.2 Create User	62
2.2.3 Forgot Password/Get Recovery Token	63
2.2.4 Reset Password	64
2.2.5 Report Fault	65
2.2.6 Search Carparks	66
2.2.7 Navigate from Search Carpark	67
2.2.8 Add Favourites	68
2.2.9 Delete Favourite	69
2.2.10 Navigate from Favourites	70
2.2.11 Admin Resolve Report	70
2.3. Dialog Map	72
2.4. System Architecture	73
3. Testing	74
3.1. Black-Box Testing	74
3.1.1 Functionality: Create Account	74
3.1.2 Functionality: Login	80
3.1.3 Functionality: Forget Password	81
3.1.4 Functionality: Reset Password	81
3.2. White-Box Testing	86
3.2.1 Create User	86
3.2.2 Navigation	87
3.3. Demo Script	88
4. Appendix A: Glossary	90
5. Appendix B: Meeting Minutes	91

Revision History

Name	Date	Reason For Changes	Version
Sheng Da, Shan Yi, Teja, Trini, Zi Xuan	10/11/23	First Draft	1.0
Sheng Da, Shan Yi, Teja, Trini, Zi Xuan	12/11/23	Final Draft	2.0

1. Requirements

1.1 Introduction

1.1.1 Purpose

The main purpose of this Software Requirements and Specifications (SRS) document is to provide a well-defined and comprehensive understanding of Swiftpark, a web-based application where motorists in Singapore are able to use for ease of navigation to public Housing Development Board (HDB) car parks with available lots.

This SRS will offer an extensive elaboration of the web application's goals, functionalities, interfaces and constraints. In addition, this document will also specify the requirements, design principles and considerations of the web application.

1.1.2 Document Conventions

Level 1 Header - font: Calibri, size: 24, style:bold

Level 2 Header - font: Calibri, size: 18, style:bold

Level 3 Header - font: Calibri, size: 14, style:bold

Text - font: Calibri, size: 12, style: normal, align: left align

Invalid Test Case - font: Calibri, text colour: red, size: 12, style: normal, align: left align

1.1.3 Intended Audience and Reading Suggestions

This document is intended for developers, project managers, testers and general users of the Swiftpark application.

Developers and project managers are encouraged to review the entire document thoroughly to understand the intricacies of this application. Testers and general users on the other hand, are recommended to read the use case descriptions and system features to understand how they may be able to navigate this application.

The recommended sequence of reading this SRS, would be to follow the sequence of sections as listed in the content page.

1.1.4 Product Scope

Swiftpark's mission is to simplify the urban commuting experience by helping drivers quickly find available parking spaces near their desired destination. The application aims to alleviate

the frustration and time wasted in searching for parking lots. The application allows users to search for the nearest public HDB car parks from the user's location or the user's desired destination (via user input). Additionally, users may also communicate any technical and/or non-technical faults in the car parks to the system administrator, through the in-built reporting system.

With Swiftpark, urban mobility will be more efficient, convenient, and pleasant for all drivers in Singapore.

1.1.5 References

IEEE Software Requirements Specifications Template Copyright © 1999 by Karl E. Wiegers, Version 1.0

http://www.frontiernet.net/~kwiegers/process_assets/srs_template.doc

1.2. Overall Description

1.2.1 Product Perspective

With the Swiftpark web application, Singapore motorists can identify nearby public HDB car parks with available lots, and navigate to them more efficiently. This reduces the probability of drivers accessing a HDB car park, only to find out that there are no vacancies. These functionalities are underpinned with the use of the following APIs on data.gov.sg: Carpark Availability and HDB Carpark Information.

Current relevant applications available include *Parking.sg*, a government collaboration between GovTech, Urban Redevelopment Authority (URA) and HDB. *Parking.sg* currently only offers facilitation of e-payment at coupon car parks. On the other hand, Swiftpark offers novel functionalities which can be used in conjunction, or independently with existing parking apps in Singapore, given its unique value proposition.

1.2.2 Product Functions

Major Functions:

- Account Creation
- User Login
 - Forget Password
 - Reset Password
- Admin Login
- GPS Geolocation Monitoring
- Car Park Information Listing
- Filter
- Favourite

- Add Favourites
- Remove Favourites
- Navigate From Favourites
- Map View

Minor Functions:

- Report

1.2.3 User Classes and Characteristics

General users of Swiftpark are expected to be 18 years old or older, with either a valid Class 2, 2A, 2B, 3, 3C, 3CA or 3A driving license. User demographics are likely to be distributed across different genders, racial groups and religions and socio-economic status.

Furthermore, users are expected to be geographically within the boundaries of Singapore, given that the application only functions within Singapore. In general, users are minimally expected to be familiar with mobile technology and have access to the Internet.

1.2.4 Operating Environment

Swiftpark can be operated on almost all web browsers, regardless of mobile or desktops.

This web-application is developed on:

- React 18.2.0
- Vite 4.4.9
- Node 18.18.0
- Flask 3.0.0
- MSSQL 10.0.1

The frontend framework uses React with Typescript. React is a free open-source frontend library, predominantly for building User Interfaces (UI) using components. While React usually uses Javascript, Swiftpark used Typescript during the app development. React hinges on a component-based architecture which makes components easy to add, delete and maintain, which is useful for code reusability.

Vite is a development environment which supports Typescript. It also has Hot Module Replacement (HMR) which updates modifications in real time without requiring a full reload in the browser.

The backend utilises two frameworks: Node and Flask. Node is a free open-source environment which allows developers to create, delete and maintain data in a database. Node also uses asynchronous, non-blocking and does not need to await APIs to return data. Flask is a framework using Python, and supports features like URL routing, which is useful for Swiftpark's navigation functionalities.

For the database, Microsoft SQL Server was used. MSSQL is a relational database which helps with storing and retrieving data at an optimised pace. This is useful for applications which may have a large user database and requires scalability.

To test the application on localhost, the relevant frameworks need to be downloaded. Thereafter, packages can be installed for Node using “npm install”, with the package names succeeding it. Both frontend (React) and backend (Node) requires the command “nodemon app.js” to run, and for the backend (Flask), “app.py” is required.

1.2.5 Design and Implementation Constraints

Since Google Maps API is used for users’ destination search queries, varying spellings and naming conventions of destination inputs may result in different data returned by the API.

Users are also required to have an Internet connection to access the navigation and car park search query functions as they both rely on real time APIs. GPS signal also needs to be turned on for the application to display the navigation route to a selected HDB car park.

1.2.6 User Documentation

Users may contact the development team for any troubleshooting issues or concerns. Kindly refer to the [README.md](#) in a Github repository for detailed instructions to operate the web application. A demo video tutorial link is also provided.

1.2.7 Assumptions and Dependencies

Assumptions:

Swiftpark requires stable Internet connection and GPS signal to be switched on at all times. Should the user not be connected to the Internet, or have the GPS signal turned on, the navigation service will not be available.

Dependencies:

Swiftpark predominantly uses the Google Maps API to fetch the geolocation of HDB car parks, as well as any navigation routes. Without the API, users will not be able to access the map view to observe his/her geolocation, the geolocation of the destination car park, as well as the route between these two locations.

Swiftpark also uses real time API - Carpark Availability, which is provided by HDB. The API retrieves carpark information every one minute so the car park query results are dependent on the results fetched by the API. Types and categories of car parks are also retrieved using the HDB Carpark Information API, which is a static dataset. Therefore, the recency of car parks available is dependent on HDB updating the dataset. Both these APIs can be found on [data.gov.sg](#) and are licensed to be used under the Singapore Open Data License.

For easy code reusability and extension, our web application uses commonly used open-source frameworks. The frontend uses React as the primary frontend framework with Typescript as the primary programming language. The backend uses two frameworks congruently: Node and Flask, using Typescript and Python as the languages respectively.

1.3. External Interface Requirements

1.3.1 User Interfaces

User Interfaces are required as the web application will be loaded on a web browser.

When users are filtering for car parks using the search query, checkboxes must be ticked to select the corresponding filter types. Should the checkboxes be left unchecked, the filter types will be taken to be unselected by default.

In the HomeUI, CreateAccountUI and LoginUI, the navigation bar at the top offers users the choice to access the LoginUI and AboutUI pages. Upon successful login, the navigation taskbar is linked to FavouriteUI and ReportUI, with an additional button to logout and return to the LoginUI page.

Error messages are displayed using a red font in a red popup container. Success messages are displayed using a green font in a green popup container. Further details can be observed in section 1.9 (UI Mockups).

1.3.2 Hardware Interfaces

The web application is expected to run on any modern hardware devices like smartphones, tablets, laptops, desktops, as long as they have a web browser installed, with hardware components to connect to the Internet and GPS.

1.3.3 Software Interfaces

- React 18.2.0 is used to handle and display UI components for the user to navigate within the web application
- Web browsers must be installed to access the web application
- HDB Carpark Information dataset - To retrieve the types of car parks and the types of payment available in each car park
- Carpark Availability API - To retrieve real time number of lots available in each HDB car park
- Software requires Google Maps API to identify the geolocation of the user and destination car park geolocation, and the distances of HDB car parks from the user's destination input

- Software requires Google Maps Geocoding API to process geocoding of latitude and longitude coordinates.

1.3.4 Communications Interfaces

In order to load the web application on the web browser, Swiftpark requires a web protocol like HTTP. Moreover, for the Google Maps view and any relevant geolocation to be loaded, the user must be connected to the Internet in order to retrieve data using the Google Maps API.

1.4. System Features

1.4.1 Create Account

1.4.1.1 Description and Priority

- The website must allow user to create an account if user currently does not have one
- Priority: High

1.4.1.2 Stimulus/Response Sequences

- SEQ-1: User clicks on the 'Login' button
- SEQ-2: User clicks on the 'Create Account' button
- SEQ-3: User enters valid inputs for username, email and password

1.4.1.3 Functional Requirements

- REQ-1: Users must create unique individual accounts
 - 1.1 No more than 1 account shall share the same username.
 - 1.1.2 A username is initialised by the user when creating an account.
 - 1.1.3 The username cannot be changed once the account is initialised.
 - 1.2 No more than 1 account shall share the same email
 - 1.2.1 The email is initialised by the user when creating an account.
 - 1.2.2 The email cannot be changed once the account is initialised.
- REQ-2: A password must be set when an account is created
 - 2.1 The user shall be able to change his/her password.
 - 2.1.1 The user shall be able to change his/her password before login.
 - 2.1.1.1 The user must enter a valid email before changing the password.
 - 2.1.1.2 The user must receive a recovery token in the email to access the link to reset password.
 - 2.1.2 The user must enter a new password to change the password.
 - 2.1.2.1 The password must contain lowercase, uppercase letters, digits and special characters
(!@#\$%^&*()_+{}[\];<>,.?~\|.).

2.1.3 The user must re-enter the new password.

2.1.4 Once both entries of the new password match exactly, the new password must be changed successfully.

1.4.2 User Login

1.4.2.1 Description and Priority

- The website must allow user to login with their unique username
- Priority: High

1.4.2.2 Stimulus/Response Sequences

- SEQ-1: User clicks on the 'Login' button
- SEQ-2: User enters valid username and password

1.4.2.3 Functional Requirements

- REQ-1: User must be able to enter username and password
- REQ-2.1: If valid information is entered, user will be login to the main page
- REQ-2.2: If incorrect or invalid information is entered, system will notify user and user can re-enter their information

1.4.3 GPS and Map View

1.4.3.1 Description and Priority

- The website must display the google maps of the current location to the selected destination with the navigation
- Priority: High

1.4.3.2 Stimulus/Response Sequences

- SEQ-1: User clicks on the 'Navigate' button

1.4.3.3 Functional Requirements

- REQ-1: GPS system must get the current physical position of the user device.
- REQ-2: Users must be able to navigate to their chosen car park.
 - 2.1 The map must show the shortest route to the car park using the Google Map API.
 - 2.2 Map view must be displayed on an entirely new page.
 - 2.3 Map view must be in full screen.
 - 2.4 Map view has a button to go "back" to exit map view.

1.4.4 Filter

1.4.4.1 Description and Priority

- The filter must allow users to select their preferred type of parking and destination. The filter should display results that match users selection in a list.

- Priority: High

1.4.4.2 Stimulus/Response Sequences

- SEQ-1: User clicks on the search bar to enter a desired destination
- SEQ-2: User clicks on the filter button and chooses the filter options

1.4.4.3 Functional Requirements

- REQ-1: There are 2 filters for the users to set when searching for car parks:
 - 1.1 Default filter where all categories are applied
 - 1.2 Custom filter where user may select desired categories
- REQ-2: The filter shall accept user inputs from the following:
 - 2.1. Tick boxes of types of car parks with multiple selections allowed:
 - Multi-Storey Car Parks
 - Basement Car Parks
 - Surface Car Parks
 - 2.2 Tick boxes of types of parking systems with multiple selections allowed:
 - Coupon Parking
 - Electronic Parking System
 - 2.3 Tick box of Night Parking Availability:
 - Night Parking
- REQ-3: If the user did not use the filter function, the following must be the default filter:
 - All the tick boxes of types of car parks must be checked.
 - All the tick boxes of types of parking systems must be checked.
 - The tick box of "Night Parking" must NOT be checked.

1.4.5 Car Park Information

1.4.5.1 Description and Priority

- The website must show the information of the car parks after filtering.
- Priority: High

1.4.5.2 Stimulus/Response Sequences

- SEQ-1: User clicks on the 'Search' button after selecting filter options

1.4.5.3 Functional Requirements

- REQ-1: The information shall display the following:
 - Address of car park.
 - Distance from current position obtained from the GPS and the car park.
 - The number of available lots.
- REQ-2: At most 5 car park information within 2km radius from the user's current position shall be displayed to the user, selected based on the nearest

distance from the current position, if the destination location is not indicated by the user.

- REQ-3: At most 5 car park information within 2km radius from the destination location shall be displayed to the user, selected based on the distance from the destination, if the destination is indicated by the user.
- REQ-4: Distance must be calculated using Google Maps API.
- REQ-5: If no car parks are found within a 2km radius, the system must throw in an error message ““No Car Park matches Filter, Please adjust the Filter.”

1.4.6 Favourite

1.4.6.1 Description and Priority

- The website must allow user to add a destination to user’s favourites
- Priority: Medium

1.4.6.2 Stimulus/Response Sequences

- SEQ-1: User clicks on the ‘Favourite’ button

1.4.6.3 Functional Requirements

- REQ-1: Users shall be able to add their favourite car parks upon searching for the car parks.
- REQ-2: Users shall only be able to remove their favourite car parks from their list of favourite car parks.
- REQ-3: Users must be able to navigate to their favourite car parks.

1.4.7 Report

1.4.7.1 Description and Priority

- This case allows users to submit a report of any operational and infrastructural faults at HDB car parks. Users will be able to provide a short text description to explain the fault.
- Priority: Low

1.4.7.2 Stimulus/Response Sequences

- SEQ-1: User clicks on the ‘Report’ button
- SEQ-2: User enters the details and the problem

1.4.7.3 Functional Requirements

- REQ-1: Users must be able to report any faults of the car park.
 - 1.1 Users must include their car park address in the report.
 - 1.2 Users must provide a text description in the form of a string.

- REQ-2: Admin must be able to view the reports submitted by users on the admin main page.
 - 2.1 Admin shall be able to view each case in detail with a click.
 - 2.2 Admin shall resolve each case manually by contacting the town council offline.
 - 2.3 Admin shall click on the “resolved” button once they contact the town council offline.
 - 2.4 Admin shall be able to click on a back button if they have not resolved the case.

1.5. Non-Functional Requirements

1.5.1 Performance Requirements

- The system must load within 3 seconds.
- The login page must take less than 3 seconds to load.
- The page after filtering must take less than 5 seconds to load.
- The information on available car park slots must load within 5 seconds.
- The Google map should be able to render all locations within Singapore.
- The system must be able to support 10,000 concurrent user visits at the same time while maintaining optimal performance.

1.5.2 Safety Requirements

- The system must be able to ensure data integrity within the system and this can be achieved by implementing measures such as error handling, data validation and correction.

1.5.3 Security Requirements

- Users are required to log in with their username and password
- Users must protect their passwords and never share them with other people, apps, or websites.
- Users must have passwords that have a minimum of 8 characters, with at least 1 uppercase letter, 1 lowercase letter, 1 digit and 1 special character.
- The system must verify email addresses when users try to change passwords so as to prevent unauthorized access.
- The system must ensure that sensitive or personal data can only be accessed by authorised administrators.

1.5.4 Software Quality Attributes

Reliability	The application must present information that is accurate with reference to the government data sources used.
Flexibility	The application must allow users to filter through available car parks with their specific criteria such as type of parking system.
Maintainability	The structure should be well-documented and organized, making it easy for developers to understand and maintain. The program should adhere to a consistent coding style and naming conventions across the entire codebase.

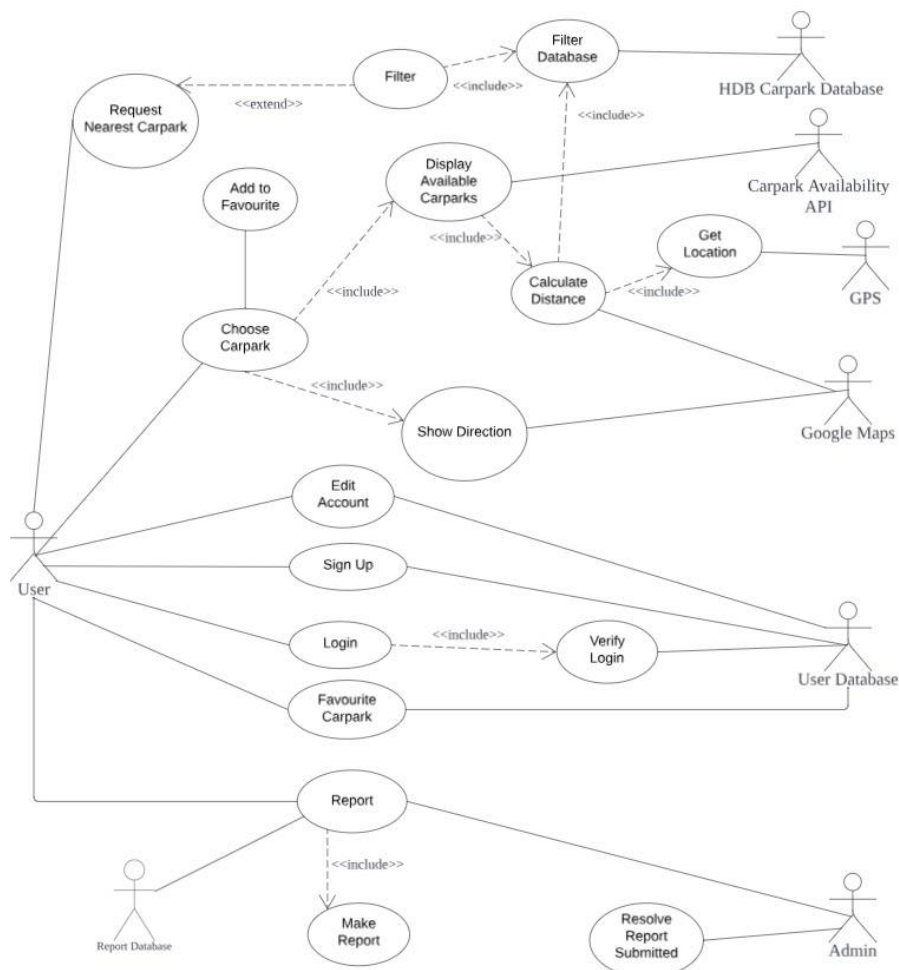
Usability

The system must be easy to use and navigate and the favourite function of the system must build a shortcut for the users to find car parks that they frequently rely on.

1.5.5 Business Rules

- Administrator
 - Administrators can remove users that do not abide by the web application guidelines.
- Users
 - Users can make adjustments to the passwords.
 - Users can decide whether or not to use the favourite and navigate functions after filtering through the car parks.

1.6. Use Case Diagram



1.7. Use Case Descriptions

Priority = High, Medium, Low

Use Case ID:	T301		
Use Case Name:	Create Account/Sign Up		
Created By:	Zi Xuan	Last Updated By:	Zi Xuan
Date Created:	02/09/2023	Date Last Updated:	23/10/2023

Actor:	User, Admin, User Database
Description:	Users who do not have a pre-existing account will be able to initialise an account with his/her personal details, in order to access the features of the web application.
Preconditions:	-
Postconditions:	<ol style="list-style-type: none">1. System will display a message "account has been created" upon successful account creation.2. System will redirect user to login page.
Priority:	High
Frequency of Use:	High

Flow of Events:	<ol style="list-style-type: none"> 1. User will click on button "Create a new account" 2. User will key in email, username and password 3. System will validate information by searching for username and email in user database 4. System will validate passwords and check if both match identically 5. System will add new entry for user information in user database 6. System redirects user to login page
Alternative Flows:	<p>AF-S3: Missing fields</p> <ol style="list-style-type: none"> 1. System will display the message "Please fill in all the fields" 2. System will return to main flow Step 2 <p>AF-S3: Email already exists</p> <ol style="list-style-type: none"> 1. System will display the message "email has been used!" 2. System will return to main flow Step 2 <p>AF-S3: Username is invalid</p> <ol style="list-style-type: none"> 1. System will display message "invalid username" 2. System will return to main flow Step 2 <p>AF-S4: Password does not match requirements</p> <ol style="list-style-type: none"> 1. System will display the message "weak password" 2. System will return to main flow Step 2 <p>AF-S4: Password and second password entry do not match identically</p> <ol style="list-style-type: none"> 1. System will display the message "passwords do not match" 2. System will return to main flow Step 2
Exceptions:	
Includes:	
Special Requirements:	

Assumptions:	<ol style="list-style-type: none">1. Device support GPS2. Device support Internet Connection
Notes and Issues:	

Use Case ID:	T302		
Use Case Name:	Login		
Created By:	Zi Xuan	Last Updated By:	Zi Xuan
Date Created:	02/09/2023	Date Last Updated:	23/10/2023

Actor:	User, Admin, User Database
Description:	This feature will allow users (and admin) to enter the main page of the web application to access its main functions. The user must input a valid username and corresponding password to successfully login.
Preconditions:	-
Postconditions:	User will be redirected to main page or admin page upon successful login
Priority:	High
Frequency of Use:	High
Flow of Events:	<ol style="list-style-type: none"> 1. User enters username and password in the login form. 2. User clicks on the "Login" button to submit the form. 3. The system will match the username and password entered in the form, with the records in User Database. 4. System will redirect user to main page.
Alternative Flows:	<p>AF-S3: Username or password not found or does not match existing records in User Database</p> <ol style="list-style-type: none"> 1. System will print an error message "Login Unsuccessful. Please use the correct username and password."

	<p>2. System returns to main flow Step 1</p> <p>AF-S3: Admin account is accessed</p> <p>1. System redirects admin to admin page</p>
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	<p>1. User already has a pre-existing account created</p> <p>2. Device support GPS</p> <p>3. Device support Internet Connection</p>
Notes and Issues:	

Use Case ID:	T303		
Use Case Name:	Reset Password		
Created By:	Zi Xuan	Last Updated By:	Zi Xuan
Date Created:	02/09/2023	Date Last Updated:	23/10/2023

Actor:	User, User Database
Description:	This case will allow users to change the password associated with their current accounts, to a different password. Login will require the new password to be entered for subsequent login attempts.
Preconditions:	-
Postconditions:	<ol style="list-style-type: none"> 1. System will send a success message "Your password has been reset successfully." 2. System will display a "back to home" button
Priority:	Low
Frequency of Use:	Low
Flow of Events:	<ol style="list-style-type: none"> 1. User enters email in Forget Password page 2. System verifies email and sends recovery token via email 3. User clicks on recovery password link in email 4. User enters email recovery token, password, confirmation password 5. System validates if password matches requirements and if passwords entered match identically 6. System sends a success message "Your password has been reset successfully." and prompts a "back to home" button
Alternative Flows:	AF-S2: Fields are missing

	<ol style="list-style-type: none"> 1. System sends an error message "Please fill in all the fields" 2. System returns to main flow Step 1 <p>AF-S2: User enters invalid email, i.e. email not found in User Database</p> <ol style="list-style-type: none"> 1. System sends an error message "account does not exist" 2. System returns to main flow Step 1 <p>AF-S5: User enters non-matching password when confirming password</p> <ol style="list-style-type: none"> 1. System sends an error message "passwords do not match" 2. System returns to main flow Step 4 <p>AF-S5: User enters password which does not match requirements</p> <ol style="list-style-type: none"> 1. System sends an error message "weak password" 2. System returns to main flow Step 4
Exceptions:	<p>Ex – Unstable Network Connection</p> <ol style="list-style-type: none"> 1. System will display a red text box of "Unstable Network Connection Detected, Please Check Your Network Connection."
Includes:	
Special Requirements:	
Assumptions:	<ol style="list-style-type: none"> 1. User will change old password which is non-identical to the new password 2. Device support GPS 3. Device support Internet Connection
Notes and Issues:	

Use Case ID:	T304		
Use Case Name:	Delete Favourite Car Park		
Created By:	Zi Xuan	Last Updated By:	Zi Xuan
Date Created:	02/09/2023	Date Last Updated:	23/10/2023

Actor:	User, User Database, HDB Carpark Database
Description:	This case will allow users to remove their most frequented car parks and delete them from their favourite lists.
Preconditions:	<ol style="list-style-type: none"> 1. User login into the system 2. There are existing carparks on the favourite list
Postconditions:	<ol style="list-style-type: none"> 1. Favourite list will be updated after carpark has been removed
Priority:	Medium
Frequency of Use:	Medium - High
Flow of Events:	<ol style="list-style-type: none"> 1. User clicks on "Favourite" link 2. System redirects user to the Favourite page with the list of favourite carparks displayed 3. User clicks on the "trash bin" button 4. System removes the corresponding favourite carpark from the favourite database 5. Webpage refreshes and the updated list following favourite removal is displayed
Alternative Flows:	
Exceptions:	

Includes:	
Special Requirements:	
Assumptions:	<ol style="list-style-type: none">1. User has a pre-existing account, and has previously added carpark to favourite list2. Device supports GPS3. Device supports Internet Connection
Notes and Issues:	

Use Case ID:	T305		
Use Case Name:	Report		
Created By:	Zi Xuan	Last Updated By:	Zi Xuan
Date Created:	02/09/2023	Date Last Updated:	23/10/2023

Actor:	User, Admin
Description:	This case allows users to submit report to report any operational and infrastructural faults at HDB car parks. Users will be able to provide a short text description to explain the fault.
Preconditions:	<ol style="list-style-type: none"> 1. User must know the address of the car park they wish to report 2. User must login into the system.
Postconditions:	<ol style="list-style-type: none"> 1. System will show a success message "Your Report has been submitted!"
Priority:	Low
Frequency of Use:	Low
Flow of Events:	<ol style="list-style-type: none"> 1. User enters car park address to report 2. System validates car park address 3. User enters text description 4. User clicks on "submit report" button 5. System prompts to "confirm" or "cancel" submission 6. User clicks "confirm" button 7. System will send this review to the admin, and display a success message "Your has been submitted!"
Alternative Flows:	<p>AF-S2: Invalid or incorrect carpark address</p> <ol style="list-style-type: none"> 1. System will display error message "No car parks found" 2. System returns to main flow Step 1

	<p>AF-S4: User does not input text</p> <ol style="list-style-type: none"> 1. System will send an error message "Please enter text!" 2. System returns to main flow Step 3 <p>AF-S5: The user clicks on "cancel button"</p> <ol style="list-style-type: none"> 1. System returns to main flow Step 4
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	<ol style="list-style-type: none"> 1. Device support GPS 2. Device support Internet Connection
Notes and Issues:	

Use Case ID:	T306		
Use Case Name:	Delete Report		
Created By:	Zi Xuan	Last Updated By:	Zi Xuan
Date Created:	02/09/2023	Date Last Updated:	02/09/2023

Actor:	User, Admin
Description:	This case allows users to delete a previously submitted report from a list of report the user has submitted.
Preconditions:	<ol style="list-style-type: none"> 1. User must login into the system. 2. User must have at least 1 or more pending report
Postconditions:	<ol style="list-style-type: none"> 1. List of report is updated, and deleted report is no longer visible 2. System displays message "Your report has been deleted"
Priority:	High
Frequency of Use:	High
Flow of Events:	<ol style="list-style-type: none"> 1. User selects report to be deleted from list of report 2. User clicks on "delete" button 3. System prompts user to confirm deletion with "confirm" or "cancel button" 4. User clicks on "confirm" button 5. System removes report from User and Admin end 6. System displays a success message "Your report has been successfully deleted!" and redirects user back to list of report
Alternative Flows:	AF-S3: User clicks on "cancel" button

	1. System returns to main flow Step 1
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	<ol style="list-style-type: none"> 1. User has at least 1 or more pending report 2. Device support GPS 3. Device support Internet Connection
Notes and Issues:	

Use Case ID:	T307		
Use Case Name:	Edit Report		
Created By:	Zi Xuan	Last Updated By:	Zi Xuan
Date Created:	02/09/2023	Date Last Updated:	23/10/2023

Actor:	User, Admin
Description:	This case allows users to edit any pre-existing report. Users may edit the text description submitted. Both the user and admin will be able to see the changes reflected on the other end.
Preconditions:	<ol style="list-style-type: none"> 1. User must login into the system. 2. User must have 1 or more pre-existing pending report 3. Admin must login into the system.
Postconditions:	<ol style="list-style-type: none"> 1. User and Admin can see edited report with the changes made
Priority:	High
Frequency of Use:	High
Flow of Events:	<ol style="list-style-type: none"> 1. User selects a report from list of report 2. User clicks on "edit button" 3. User edits text description 4. System validates form fields 5. System prompts user to "confirm" or "cancel" edits. 6. User clicks on "confirm" button 7. System reflect changes in user and admin end 8. System displays a success message "Your report has been updated!"

Alternative Flows:	<p>AF-S4: Text field is empty</p> <ol style="list-style-type: none"> 1. System displays error message "Fill in text fields!" 2. System returns to main flow Step 3 <p>AF-S5: User clicks on "cancel" button</p> <ol style="list-style-type: none"> 1. System returns to main flow Step 3
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	<ol style="list-style-type: none"> 1. User only can edit pending report 2. Device support GPS 3. Device support Internet Connection
Notes and Issues:	

Use Case ID:	T308		
Use Case Name:	Resolve Report Submitted		
Created By:	Zi Xuan	Last Updated By:	Zi Xuan
Date Created:	02/09/2023	Date Last Updated:	02/09/2023

Actor:	Admin, User
Description:	This case allows the admin to resolve and clear pending report by users, once necessary administrative actions have been executed to remedy the faults. The report resolved will subsequently be removed from the list of report for both admin and user.
Preconditions:	<ol style="list-style-type: none"> 1. Admin must login into the system. 2. There is a pre-existing report submitted by user 3. Measures have been taken in response to user's report
Postconditions:	<ol style="list-style-type: none"> 1. System displays a success message "Report has been resolved" 2. System redirects admin back to list of report
Priority:	High
Frequency of Use:	High
Flow of Events:	<ol style="list-style-type: none"> 1. Admin selects report from the list of report submitted by user 2. Admin clicks on "resolve" button 3. System prompts user with "confirm" and "cancel" button 4. Admin clicks on "confirm" button 5. System removes report from both user and admin's list of report 6. System displays a success message "Report has been resolved" 7. System redirects admin back to list of report
Alternative Flows:	AS-S3: Admin clicks on "cancel" button

	1. System returns to main flow Step 1
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	<ol style="list-style-type: none"> 1. There is 1 or more pre-existing report submitted by users 2. Device support GPS 3. Device support Internet Connection
Notes and Issues:	

Use Case ID:	T309		
Use Case Name:	Filter		
Created By:	Sheng Da	Last Updated By:	Zi Xuan
Date Created:	02/09/2023	Date Last Updated:	23/10/2023

Actor:	User, HDB Car Park Database
Description:	User will be able to filter for types of car park they desire based on the given fields: type of car park, type of parking, night parking availability
Preconditions:	User must successfully login into the system
Postconditions:	User hit the filter button after selecting their filter
Priority:	High
Frequency of Use:	High
Flow of Events:	<ol style="list-style-type: none"> 1. User will click on the filter button 2. There will be several tick boxes for user to choose: <ol style="list-style-type: none"> a. User can choose default or custom filter b. Tick Boxes of "Multi-Storey Car Parks", "Basement Car Parks" and "Surface Car Parks" under Types of Car Parks Label c. Tick Boxes of "Coupon Parking" and "Electronic Parking System" under Types of Parking System d. Tick Box of "Night Parking" under Night Parking Availability Label 3. Once the user finishes setting their filter, they will click on the filter button again

	<ol style="list-style-type: none"> 4. The Conditions they set will be use to subset from the copy of the original database from the HDB Car Park DataBase 5. This subset of data will then be passed on to use case "Calculate Distance"(T311) for further processing
Alternative Flows:	<p>AF-S2: If the user did not choose to select any filter</p> <ol style="list-style-type: none"> 1. The following will be the default filter for the system: <ol style="list-style-type: none"> a. All the tick boxes of Type of Car Parks will be ticked. b. All the tick boxes of Type of Parking System will be ticked. c. The Tick box of "Night Parking" WILL NOT be ticked. 2. Return to Step 4 of the main flow.
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	<ol style="list-style-type: none"> 1. Device support GPS 2. Device support Internet Connection
Notes and Issues:	

Use Case ID:	T310		
Use Case Name:	Request Nearest Car Park		
Created By:	Sheng Da	Last Updated By:	Sheng Da
Date Created:	02/09/2023	Date Last Updated:	23/10/2023

Actor:	User
Description:	User will type the place that they wish to go. System will help to search available car park based on filter condition within 2km radius
Preconditions:	User must login into the system
Postconditions:	Display of 5 or less car parks will be shown that meet the user predefined filter.
Priority:	High
Frequency of Use:	High
Flow of Events:	<ol style="list-style-type: none"> 1. User will type their destination location they wish to visit 2. User can choose to apply filter (T309) to choose the type of car park they desired 3. System will search for available car parks within 2km distance that meets the user filter conditions using the coordinates of the destination they type as the centre of the circle. 4. Top 5 or less available car parks that meet condition will be displayed for user to choose in use case "Choose Car Park" (T312)
Alternative Flows:	AF-S1: User did not type their destination when searching for the car park.

	<ol style="list-style-type: none"> 1. User hit search without typing their destination address will have their current position as the centre of the circle instead of the destination they type in step 3 2. System will search for available car parks within 2 km distance that meets the user filter conditions 3. Top 5 or less available car parks that meet condition will be displayed for user to choose in use case "Choose Car Park" (T312) <p>AF-S2: User did not choose to apply any filter.</p> <ol style="list-style-type: none"> 1. User hit search without applying the filter. In this scenario, default filter described in Alternative Flow of T309 will be used. 2. Return to Step 3 of the main flow. <p>AF-S3: No carpark matches filter</p> <ol style="list-style-type: none"> 1. System displays error message "No Car Park matches Filter, Please adjust the Filter." 2. No car parks are displayed
Exceptions:	<p>Ex – GPS detects the user's current position outside of Singapore boundaries.</p> <ol style="list-style-type: none"> 1. The system will display a red text box of "Current Position Not in Singapore, Navigation Not Supported." <p>Ex - User requests for destination outside of Singapore boundaries</p> <ol style="list-style-type: none"> 1. The system will display a red text box of "Destination Location Not in Singapore, Navigation Not Supported."
Includes:	
Special Requirements:	
Assumptions:	<ol style="list-style-type: none"> 1. User type in valid Singapore location in the search box as their destination location. 2. User in within Singapore 3. Device support GPS 4. Device support Internet Connection
Notes and Issues:	

Use Case ID:	T311		
Use Case Name:	Calculate Distance		
Created By:	Sheng Da	Last Updated By:	Zi Xuan
Date Created:	02/09/2023	Date Last Updated:	23/10/2023

Actor:	GPS, Google Maps
Description:	Calculate the distances based on the given coordinates.
Preconditions:	Inputs of the address of the destination location or the current user position
Postconditions:	Return the distances of given coordinates
Priority:	High
Frequency of Use:	High
Flow of Events:	<ol style="list-style-type: none"> 1. The distance of the destination location and available car park that meets filter conditions will be calculated. 2. The top 5 nearest distance from the destination location or less will be picked out. 3. The distance of these 5 locations and the user current position will be calculated.
Alternative Flows:	<p>AF-S1: The user did not input any destination location in the search box in use case "Request Nearest Car Park" (T310).</p> <ol style="list-style-type: none"> 1. The distance of the user's current user position and the car parks that meet filter conditions will be calculated. 2. Top 5 nearest car park from the current user position will be calculated

Exceptions:	<p>Ex – GPS detects the user's current position outside of Singapore Boundaries.</p> <ol style="list-style-type: none"> 1. The system will display a red text box of “Current Position Not in Singapore, Navigation Not Supported.”
Includes:	
Special Requirements:	
Assumptions:	<ol style="list-style-type: none"> 1. Device support GPS 2. Device support Internet Connection
Notes and Issues:	

Use Case ID:	T312		
Use Case Name:	Choose Car Park		
Created By:	Sheng Da	Last Updated By:	Zi Xuan
Date Created:	02/09/2023	Date Last Updated:	23/10/2023

Actor:	User
Description:	The user will choose 1 of the car parks that he/she wishes to go from the display of available car parks
Preconditions:	At most 5 of the nearest car parks that meets the condition will be shown
Postconditions:	The user chooses the car park
Priority:	High
Frequency of Use:	High
Flow of Events:	<ol style="list-style-type: none"> 1. After the distance calculation in use case "Calculate Distance" (T311), at most 5 of car park will be shown to the user 2. Information in use case "Display Available Car Parks" (T314) will be shown to the user. 3. User will choose 1 of the car park that he/she wants to go 4. This input will feed the system with the information needed for the use case "Show Direction" (T313)
Alternative Flows:	<p>AF-S1: No car park shown in the available car parks display</p> <ol style="list-style-type: none"> 1. User will receive prompt display "No Car Park matches Filter, Please adjust the Filter", as in T310

	<p>AF-S3: User adds carpark to favourites</p> <ol style="list-style-type: none"> 1. User clicks on the “heart” button 2. System adds carpark to favourite list 3. System returns to main flow Step 3
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	<ol style="list-style-type: none"> 1. Device support GPS 2. Device support Internet Connection
Notes and Issues:	

Use Case ID:	T313		
Use Case Name:	Show Direction		
Created By:	Sheng Da	Last Updated By:	Sheng Da
Date Created:	02/09/2023	Date Last Updated:	23/10/2023

Actor:	User, Google Map, GPS
Description:	The map will show the direction from the current user position to the destination car park of choice
Preconditions:	The user select 1 of the car park from the "Choose Car Park" use case
Postconditions:	The user reaches destination
Priority:	High
Frequency of Use:	High
Flow of Events:	<ol style="list-style-type: none"> 1. The map will then show the user the direction to the destination car park 2. Once the user parked their car, they will click on the "back" button at the top right of the web app. 3. After clicking the button, it will return to the SearchCarpark page of the web app.
Alternative Flows:	
Exceptions:	Ex – GPS detects the user's current position outside of Singapore Boundaries.

	1. The system will display a red text box of "Current Position Not in Singapore, Navigation Not Supported."
Includes:	
Special Requirements:	
Assumptions:	<ol style="list-style-type: none"> 1. Device support GPS 2. Device support Internet Connectivity
Notes and Issues:	

Use Case ID:	T314		
Use Case Name:	Display Available Car Parks		
Created By:	Sheng Da	Last Updated By:	Sheng Da
Date Created:	05/09/2023	Date Last Updated:	05/09/2023

Actor:	User
Description:	The display will show top 5 available car parks sorted based on the destination location the user input
Preconditions:	The user complete use case "Request Nearest Car" Parks" (T310)
Postconditions:	The user selected the car park he/she wishes to go
Priority:	High
Frequency of Use:	High
Flow of Events:	<ol style="list-style-type: none"> 1. Information of the top 5 available car parks based on the destination location of the user input in use case "Request Nearest Car Park" (T310) 2. Other information displayed to the user include Address of car park, distance of carparks from destination input and the number of available lots
Alternative Flows:	<p>AF-S1: User did not input destination location in T310.</p> <ol style="list-style-type: none"> 1. Information of the top 5 available car parks sorted based on the current user position will be shown to user. 2. Return to main flow Step 2.

Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	<ol style="list-style-type: none">1. Device support GPS2. Device support Internet Connectivity
Notes and Issues:	

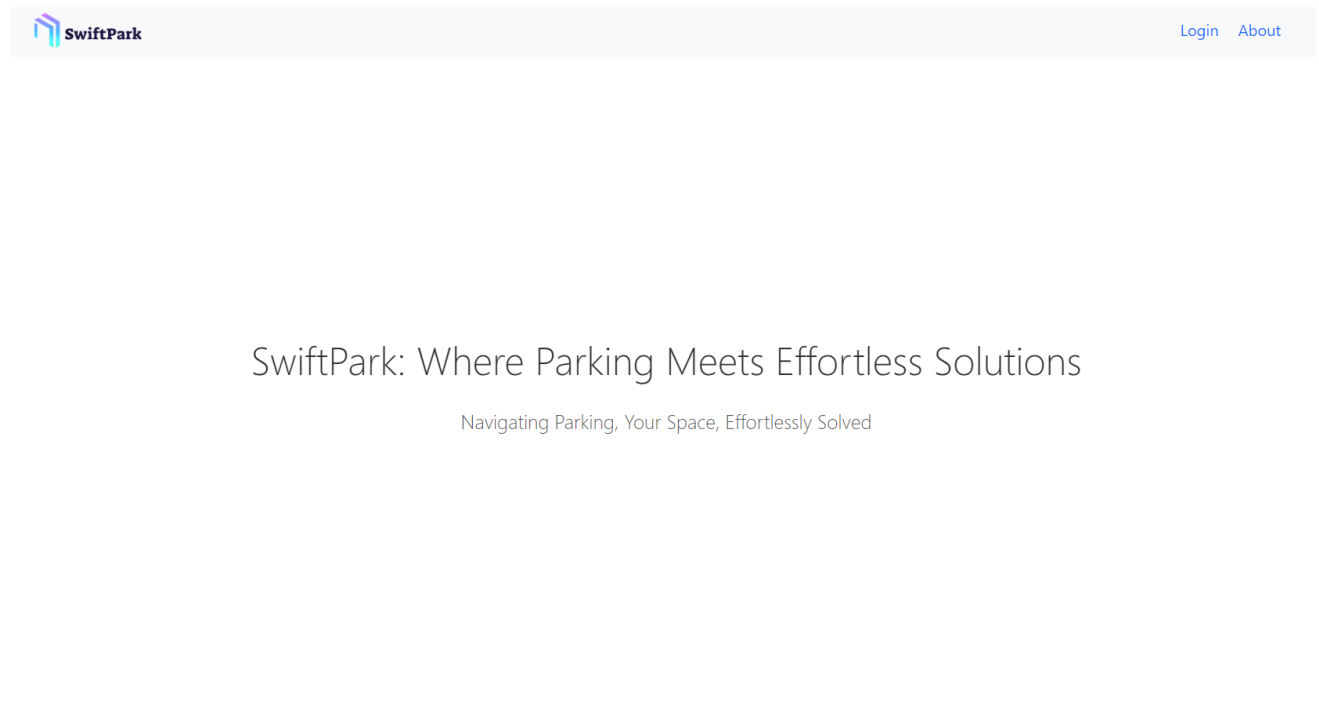
1.8. Data Dictionary

Term	Definition
User	A person who has a registered account and is using the parking app.
Carpark	Physical location where cars can be parked.
Availability	The status of a parking space, either available or occupied.
Opening hours	The hours during which a car park is open
Night Parking	The status in which the car park offers parking from 7pm - 7am
Electronic Parking System	A cashless system that uses EPS antenna to read the number of ERP In-Vehicle Unit (IU) at the entry and exit of the car park
Coupon Parking	Parking System in which drivers without coupons will have to pay through Parking.sg mobile app or display valid parking coupons if they do not have valid season parking.
Multi-Storey Car Park	Car park that has parking spaces allocated to multiple storeys
Basement Car Park	Car park that has parking spaces allocated underground
Surface Car Park	Car park that has parking spaces allocated on road level.
Destination	Location that the user will be going from his/her current position
Profile	A user's personal information, such as their name, email address, and phone number.
Username	A unique name that is associated with the user used for login.
Password	A string of characters (letters, numbers, special symbols) that allow the user to access their account
GPS	A utility system that provides users with positioning services.
Distance	The distance from the user to available car parks.
Report	A short description which allows users to talk about their experience with the car park application.

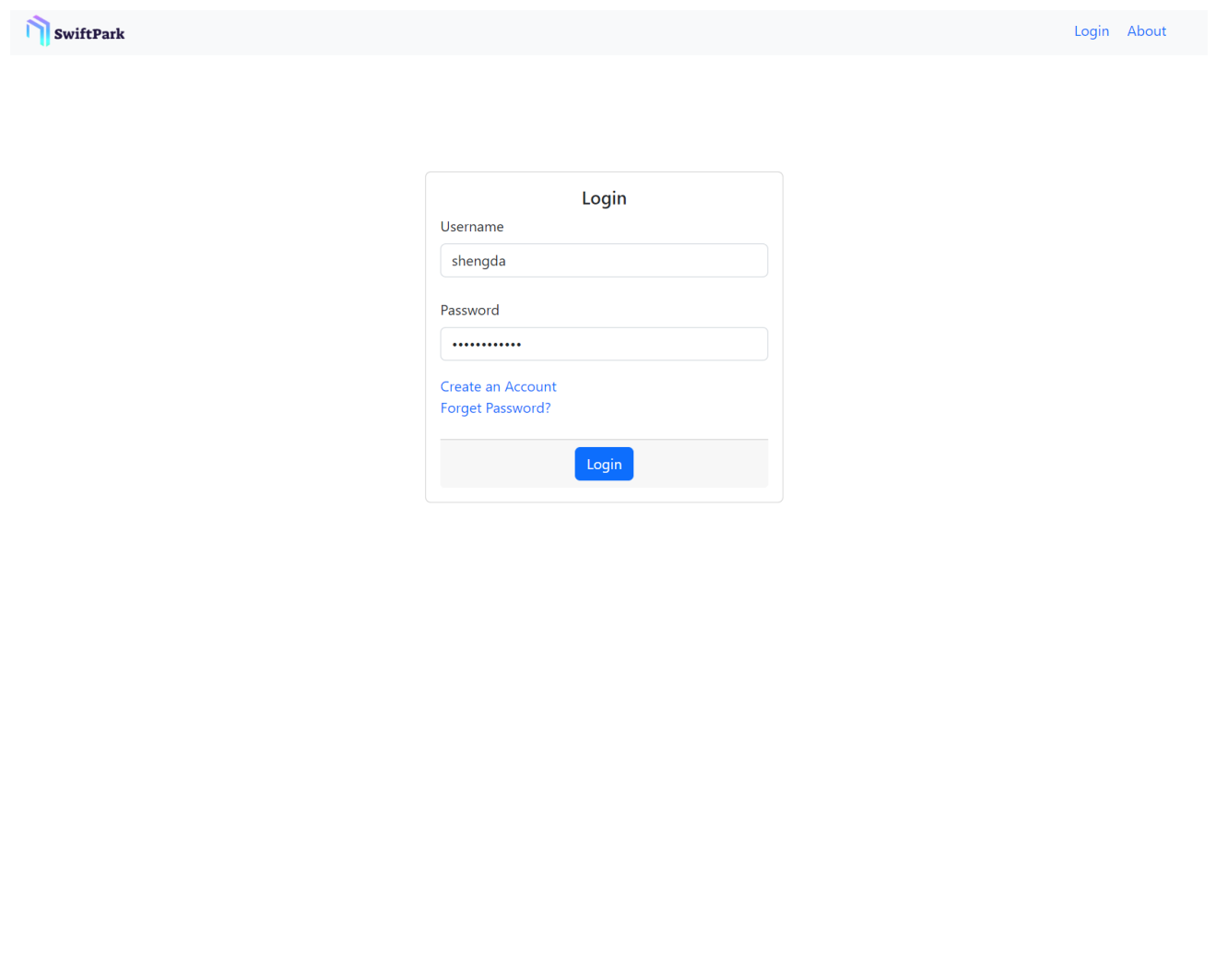
Case	Report submitted by the user which is view by the admin
Favourite	An indicator that helps the system to filter the lists of places that the user commonly goes to.
Filter	Different types and conditions of car parks can be chosen for the users for searching

1.9. UI Mockups

1.9.1 Homepage UI



1.9.2 Login UI



1.9.3 Create Account UI

Create an Account

Email Address

Username

Password

Confirm Password

1.9.4 Forget Password UI

Reset Password

To ensure it's you, we'll need to verify your email address.

Email Address

1.9.5 Reset Password UI

Reset Password

New Password

Confirm Password

1.9.6 Search Carpark UI

Search The Nearest Car Park

Search

☒ Multi-Storey Car Parks


☐ Basement Car Parks

☐ Surface Car Parks











Type of Parking System

☐ Coupon Parking


1.9.7 Choose Carpark UI

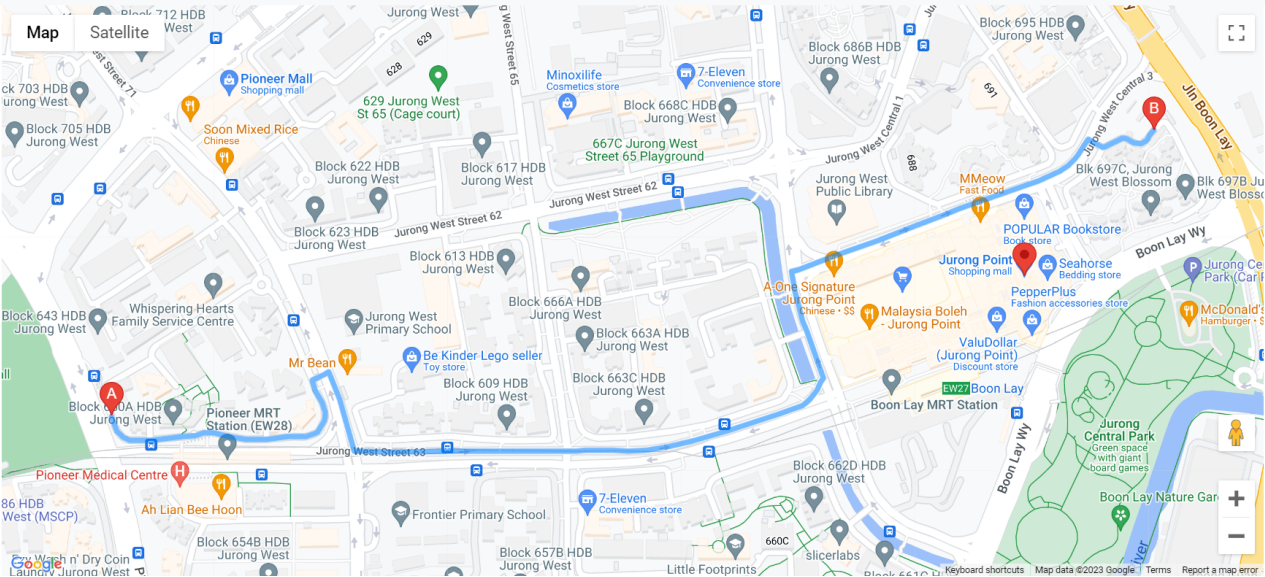
[Back](#)

Choose a Car Park

Address	Distance	Duration	Lots Vacancies	Favourite	Navigate
697 JURONG WEST CENTRAL 3	0.3 km	3 mins	248 / 410		
BLK 669 JURONG WEST STREET 64	0.8 km	6 mins	951 / 1165		
BLK 686 JURONG WEST CENTRAL 1	1.0 km	7 mins	500 / 573		
BLK 691A JURONG WEST CENTRAL 1	0.5 km	4 mins	466 / 591		
BLK 692A JURONG WEST CENTRAL 1	0.3 km	2 mins	475 / 603		











1.9.8 Map UI

[Back](#)



1.9.9 Favourite Carpark UI

Favourite Car park

Car Park	Delete	Navigate
175 YUNG KUANG ROAD		
BLK 115A HO CHING ROAD		
BLK 121A YUAN CHING ROAD		
697 JURONG WEST CENTRAL 3		
BLK 669 JURONG WEST STREET 64		

1 2

1.9.10 Report Fault UI

Report Fault

SwiftPark Details

Enter SwiftPark Details

Problem Faced

Enter Problem Faced

Submit

1.9.11 Admin UI

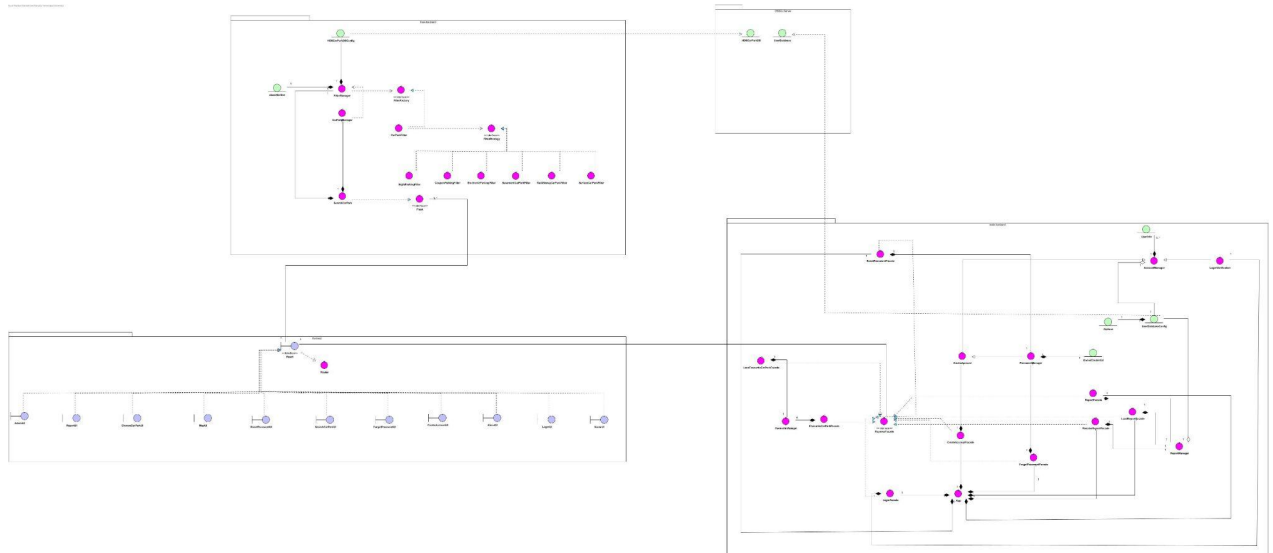
Faults Reported

Case Number	Case Details	Actions
1	Car Park A	View Details

2. Design

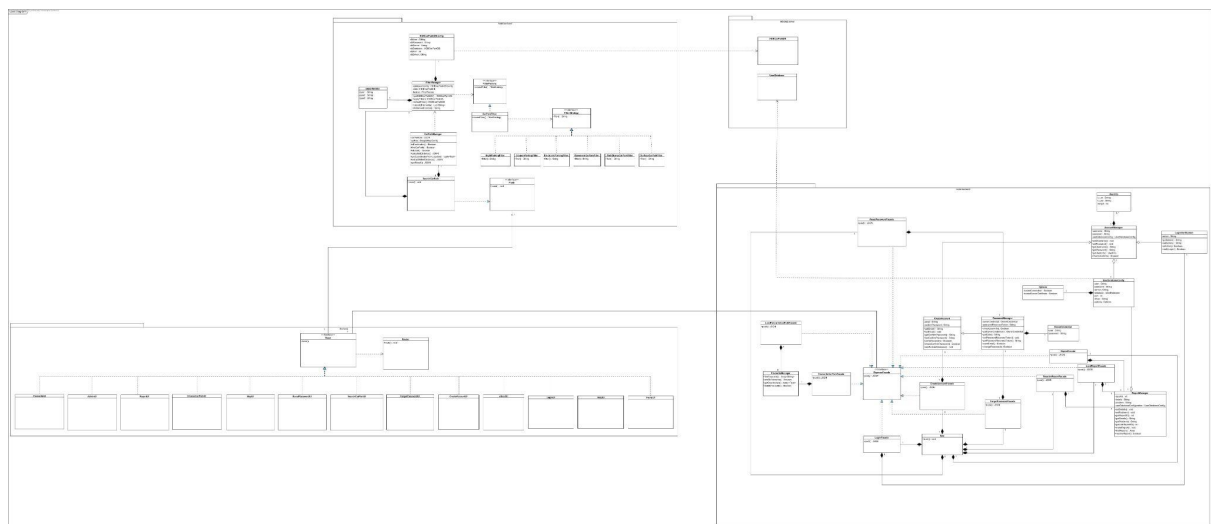
2.1. Class Diagram

2.1.1 Class Stereotype Diagram



For clearer view, please refer to Class Stereotype Diagram Final.jpg

2.1.2 Class Entity Diagram



For clearer view, please refer to Class Diagram Final.jpg

2.1.3 Key Public Method List

Flask

Class	Method Descriptions
CarparkManager	<ul style="list-style-type: none">+isDestination(searchBoxInput: String) : boolean<ul style="list-style-type: none">- To check if any search query is passed into the search field+fiveCarparks() : boolean<ul style="list-style-type: none">- To check if there are any car parks within the filter subset after filtering the (at most) 5 nearest car parks.+isLocal(userLat: float, userLong: float) : boolean<ul style="list-style-type: none">- To check whether the location coordinates are within Singapore's boundaries+calculateDistance(sourceLat: float, sourceLong: float, destinationCoordinates: float[], mode: String) : JSON<ul style="list-style-type: none">- To get the distance between source and destination of given arrays+extractWithinDistance(distances: JSON, max_distance: int, max_results: int) : JSON<ul style="list-style-type: none">- To identify all the car parks within given radius
FilterFactory	<ul style="list-style-type: none"><i>+createFilter(filter: String): boolean</i><ul style="list-style-type: none">- Abstract method for extension for factory pattern
CarParkFilter	<ul style="list-style-type: none">+createFilter(filter: String): boolean<ul style="list-style-type: none">- Factory method for instantiating car park filters, which include Night Parking, Electronic Parking System, Coupon Parking, Multi-Storey Car Park, Basement Car Park, Surface Car Park
FilterManager	<ul style="list-style-type: none">+applyFilter(choice: String[], filterOptions: String[]): JSON<ul style="list-style-type: none">- Set filter based on user's choice of car park filters+defaultFilter(): JSON<ul style="list-style-type: none">- Set default filter for car park filters
FilterStrategy	<ul style="list-style-type: none"><i>+filter(): boolean</i><ul style="list-style-type: none">- Abstract method for extension for strategy pattern
SurfaceCarparkFilter	<ul style="list-style-type: none">+filter(): string<ul style="list-style-type: none">- Method overriding for strategy pattern for filtering

MultiStoreyCarparkFilter	+filter(): string - Method overriding for strategy pattern for filtering
BasementCarparkFilter	+filter(): string - Method overriding for strategy pattern for filtering
ElectronicParkingFilter	+filter(): string - Method overriding for strategy pattern for filtering
CouponParkingFilter	+filter(): string - Method overriding for strategy pattern for filtering
NightParkingFilter	+filter(): string - Method overriding for strategy pattern for filtering
GoogleMapConfig	<No methods, just storing API key>
HDBCarpParkDBConfig	<No methods, storing configurations for MSSQL>
RetrieveCarParkLots	+retrieveData(): JSON - Get the real-time number of lots available using the Car Park Availability API from data.gov.sg
SearchCarpark	+route(): void - Method that implements app logic to hide code complexity and for routing to frontend

Node

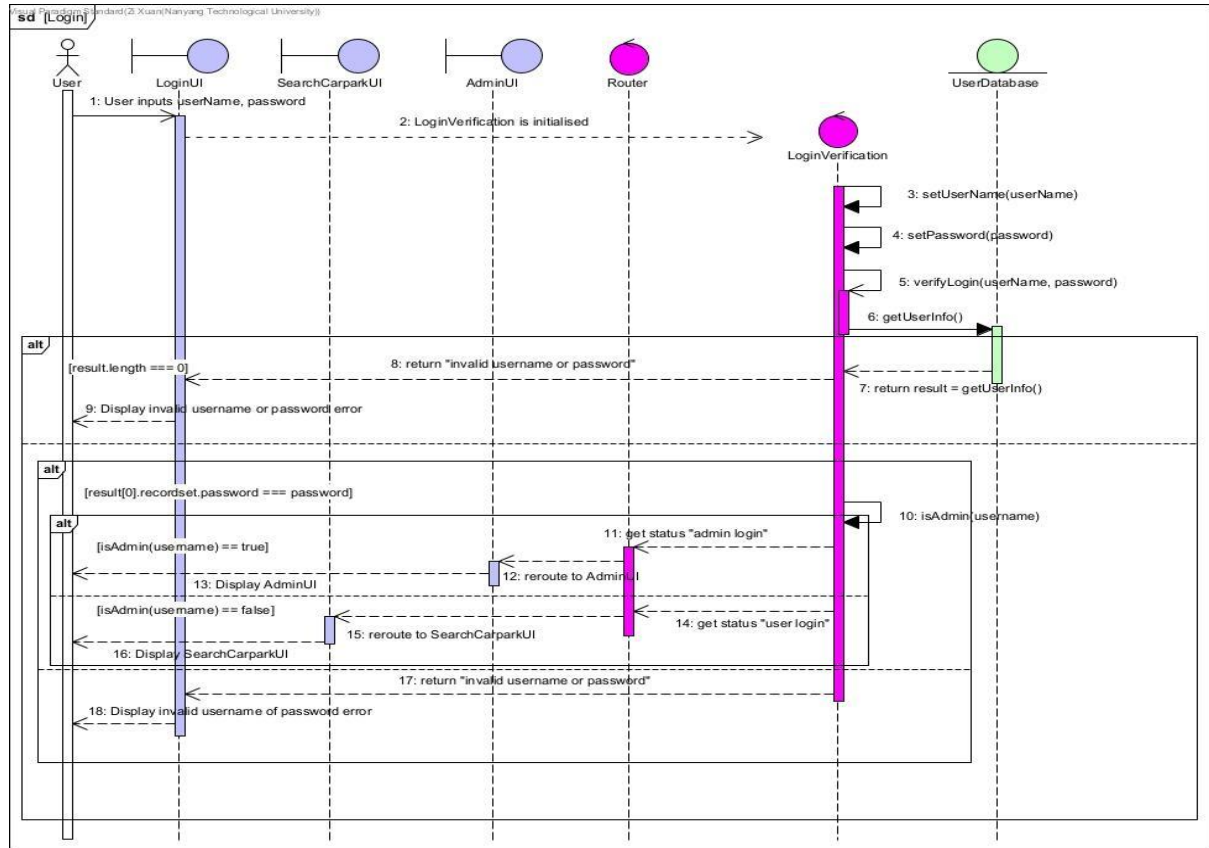
Class	Method Descriptions
AccountManager	<p>+getUserInfo(): UserInfo</p> <ul style="list-style-type: none">- To get user data from UserDatabase in MSSQL <p>+checkUserInfo(columnFields: String, target: String, targetValue: String): boolean</p> <ul style="list-style-type: none">- To check with UserDatabase in MSSQL if user record exists in database
CreateAccount	<p>+verifyPassword(password: String): boolean</p> <ul style="list-style-type: none">- Validate if password entered is strong, by ensuring if password is at least 8 characters, contains uppercase and lowercase letters, digits, special characters <p>+checkConfirmPassword(): boolean</p> <ul style="list-style-type: none">- Check whether password matches with “confirm password” <p>+addToUserDatabase(): void</p> <ul style="list-style-type: none">- Add new user record into UserDatabase in MSSQL
FavouriteManager	<p>+addToFavourite(username: String, favouriteAddress: String, latitude: float, longitude: float): boolean</p> <ul style="list-style-type: none">- Adds favourite car park destination into UserDatabase in MSSQL <p>+findFavourite(username: String): String[]</p> <ul style="list-style-type: none">- Extract all favourite car park records of a particular user from UserDatabase in MSSQL <p>+deleteFavourite(username: String, favouriteAddress: String): boolean</p> <ul style="list-style-type: none">- Deletes particular record of a particular user in UserDatabase in MSSQL
LoginVerification	<p>+verifyLogin(username: String, password: String): boolean</p> <ul style="list-style-type: none">- Verify if password of a particular user matches the record in UserDatabase in MSSQL
OwnerCredential	<No methods, store email account details of admin>
PasswordManager	+checkUserInfo(columnFields: String, target: String, targetValue: String): boolean

	<ul style="list-style-type: none"> - Check if user email input exists in UserDatabase in MSSQL +sendEmail(value: String): boolean <ul style="list-style-type: none"> - Send reset password link to user via email +changePassword(): boolean <ul style="list-style-type: none"> - Updates password in existing record in UserDatabase in MSSQL
ReportManager	+makeReport(): void <ul style="list-style-type: none"> - Add a report record into UserDatabase in MSSQL +findReport(): String[] <ul style="list-style-type: none"> - Locate all report records in UserDatabase in MSSQL +resolveReport(reportID: int): boolean <ul style="list-style-type: none"> - Delete a particular report record in UserDatabase in MSSQL
UserDatabaseConfiguration	<No methods, storing configurations for MSSQL>
ExpressFacade	+post(): JSON <ul style="list-style-type: none"> - Abstract method for routing to frontend, for extension for Facade pattern
LoadFavouriteCarParkFacade	+post(): JSON <ul style="list-style-type: none"> - Method that implements app logic to hide code complexity and for routing to frontend
FavouriteCarParkFacade	+post(): JSON <ul style="list-style-type: none"> - Method that implements app logic to hide code complexity and for routing to frontend
CreateAccountFacade	+post(): JSON <ul style="list-style-type: none"> - Method that implements app logic to hide code complexity and for routing to frontend
LoginFacade	+post(): JSON <ul style="list-style-type: none"> - Method that implements app logic to hide code complexity and for routing to frontend
ForgetPasswordFacade	+post(): JSON <ul style="list-style-type: none"> - Method that implements app logic to hide code complexity and for routing to frontend
ResetPasswordFacade	+post(): JSON

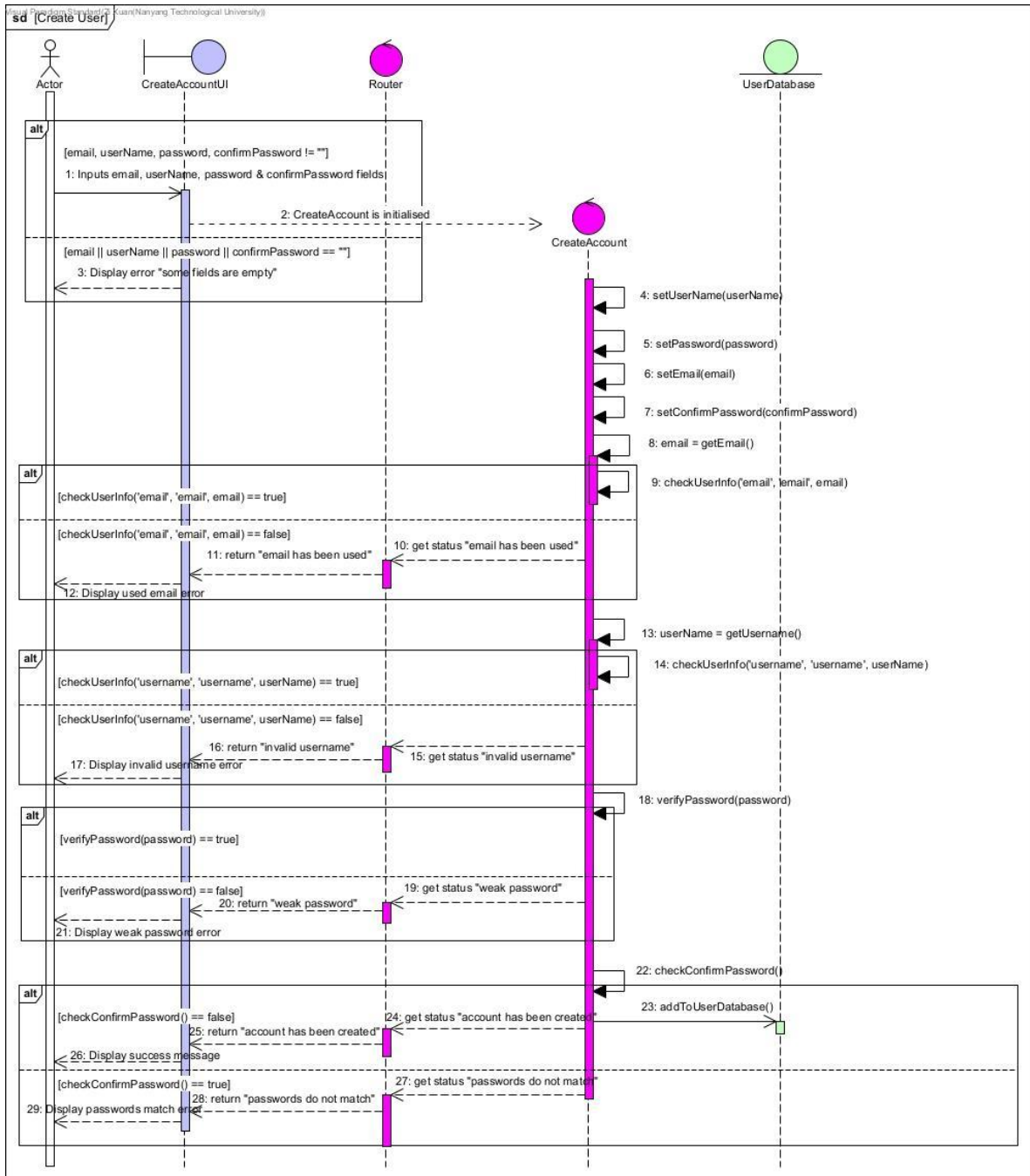
	<ul style="list-style-type: none"> - Method that implements app logic to hide code complexity and for routing to frontend
ResolveReportFacade	+post(): JSON <ul style="list-style-type: none"> - Method that implements app logic to hide code complexity and for routing to frontend
LoadReportFacade	+post(): JSON <ul style="list-style-type: none"> - Method that implements app logic to hide code complexity and for routing to frontend
ReportFacade	+post(): JSON <ul style="list-style-type: none"> - Method that implements app logic to hide code complexity and for routing to frontend

2.2. Sequence Diagrams

2.2.1 Login

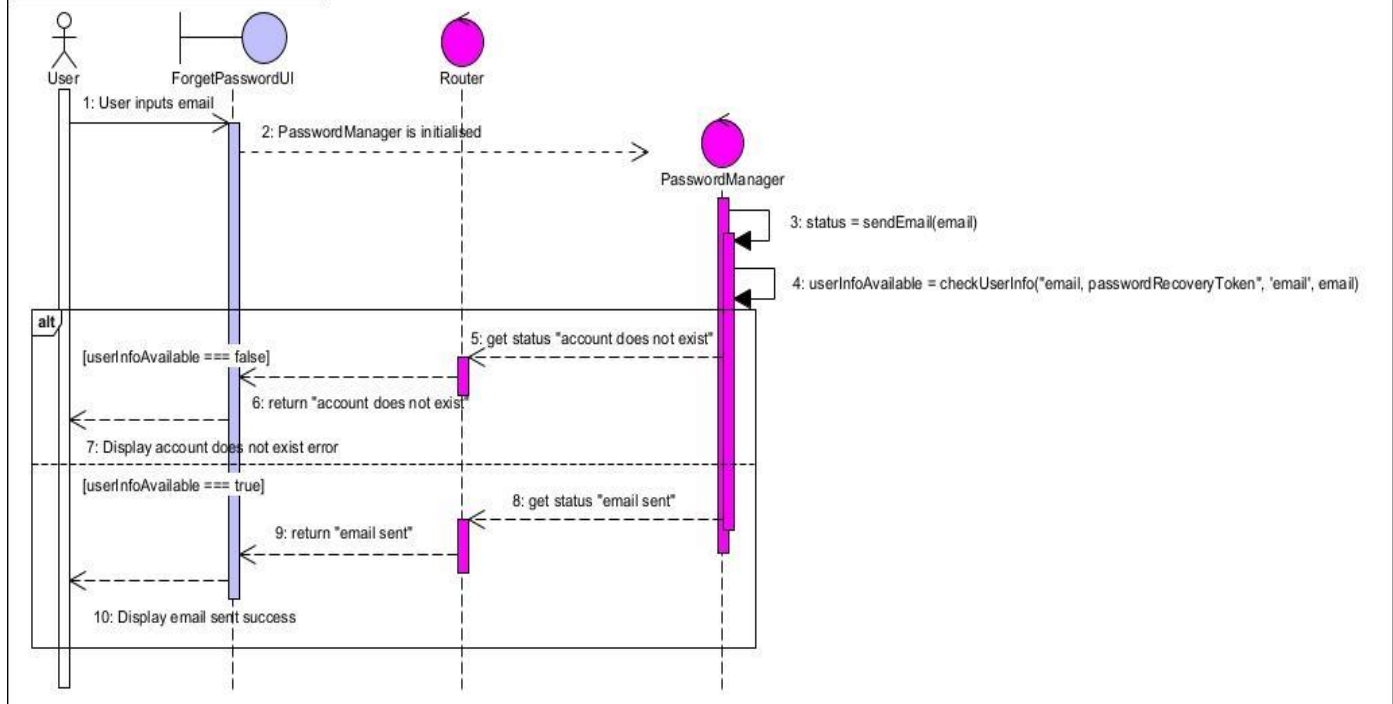


2.2.2 Create User



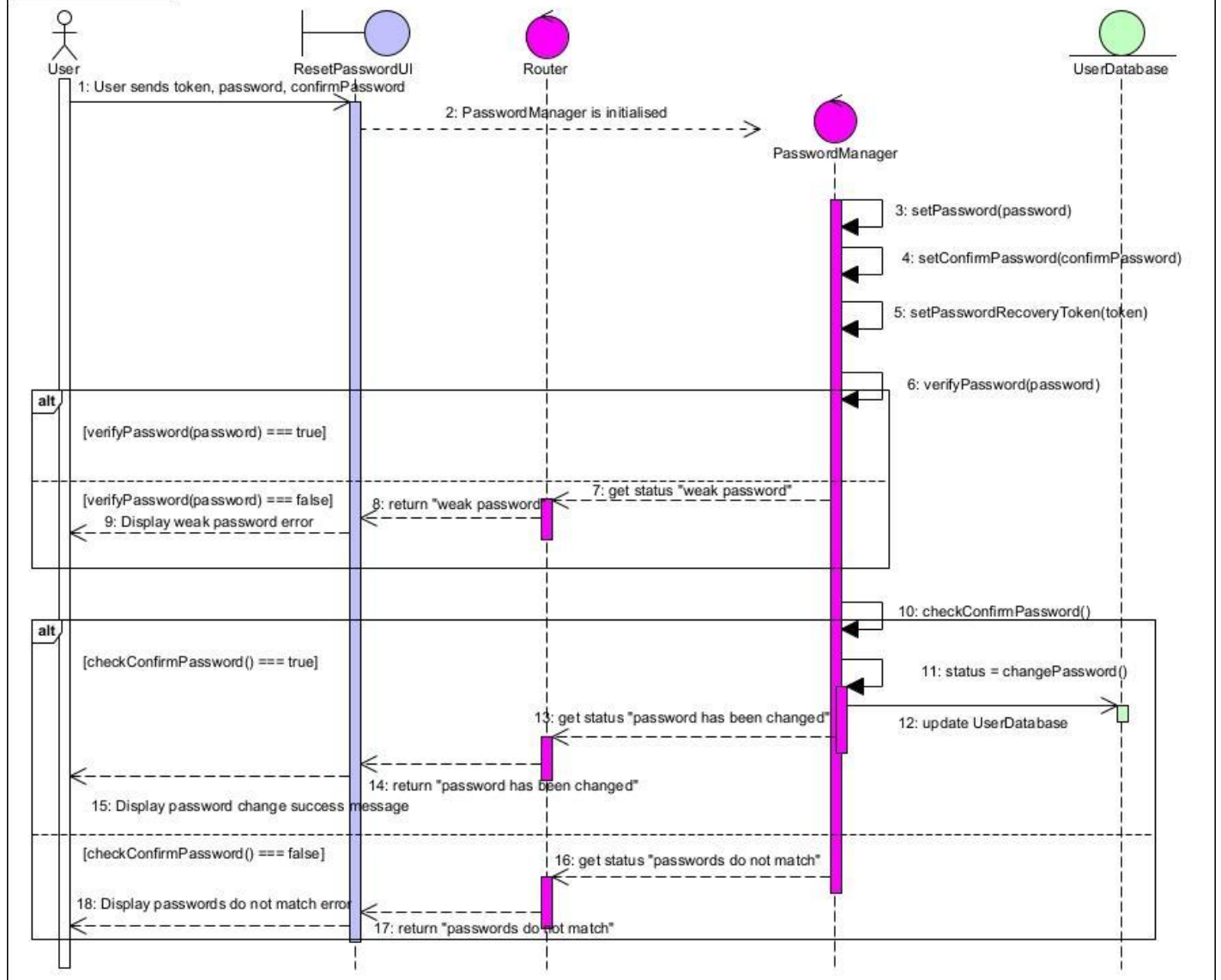
2.2.3 Forgot Password/Get Recovery Token

sd [Forgot Password/Get Recovery Token]



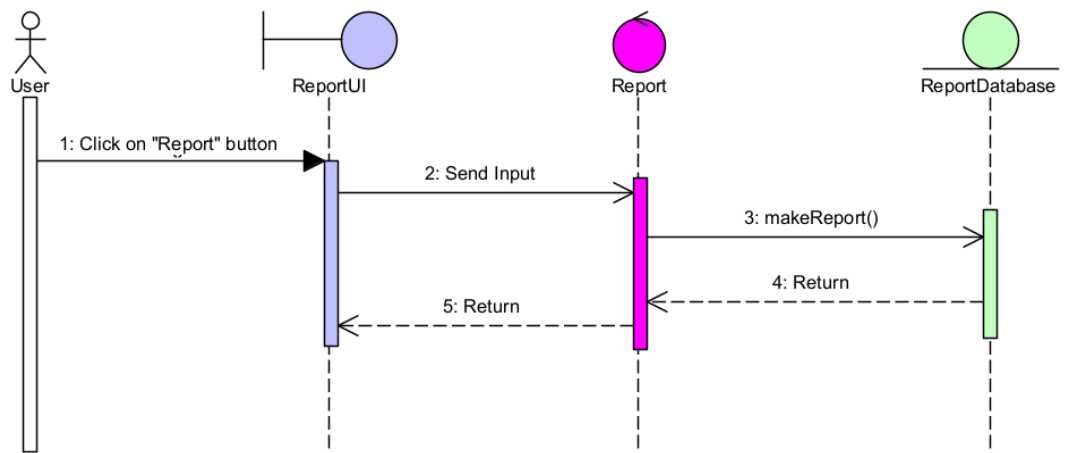
2.2.4 Reset Password

sd [Reset Password]

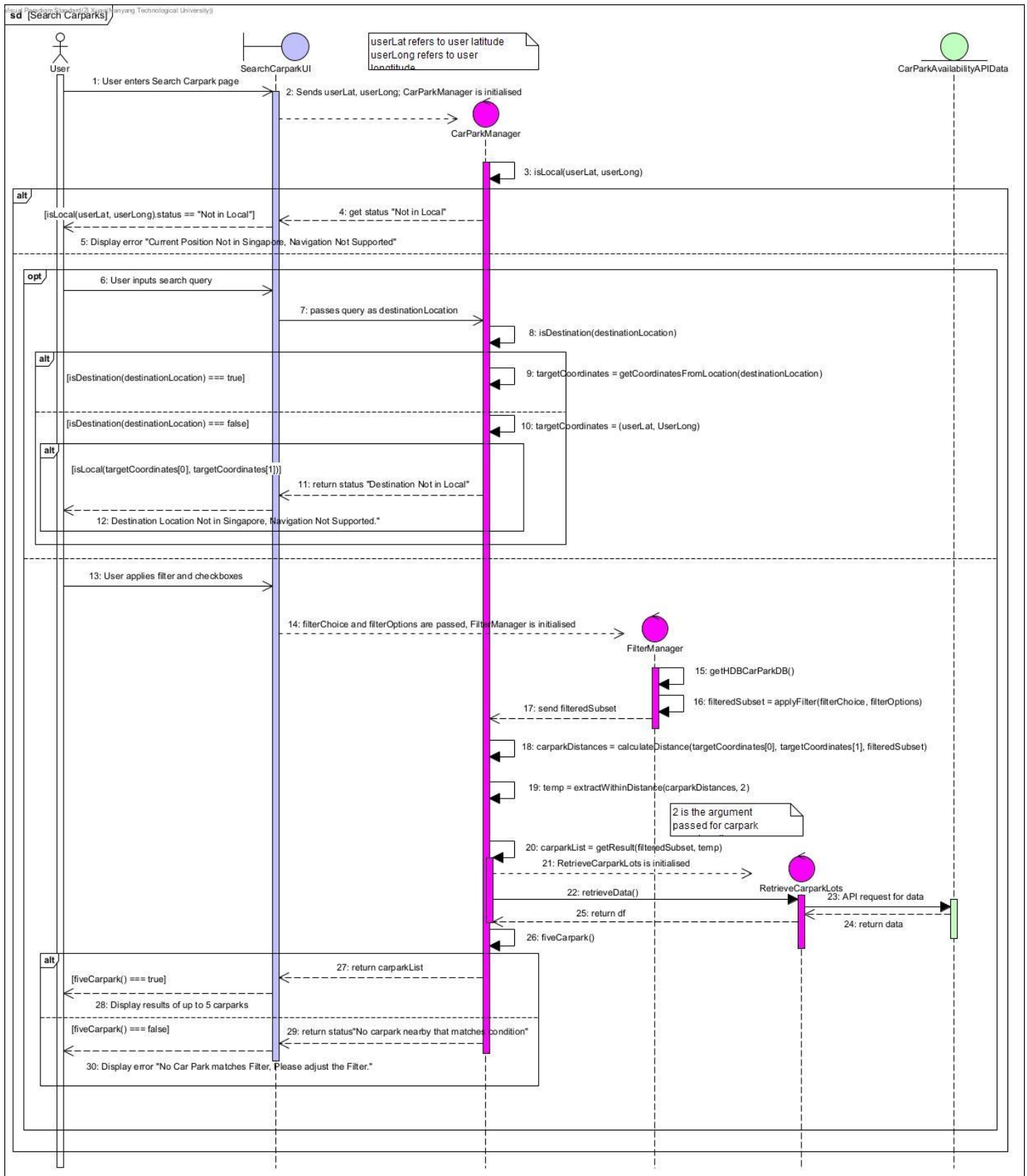


2.2.5 Report Fault

sd [Report]



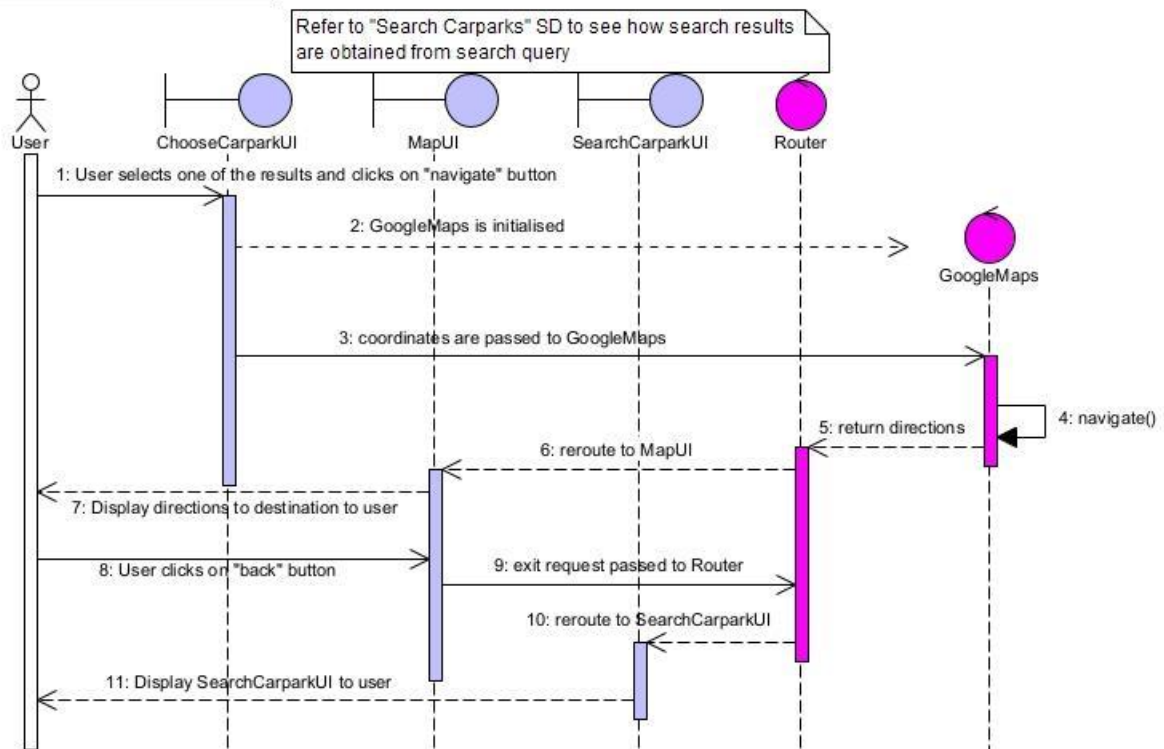
2.2.6 Search Carparks



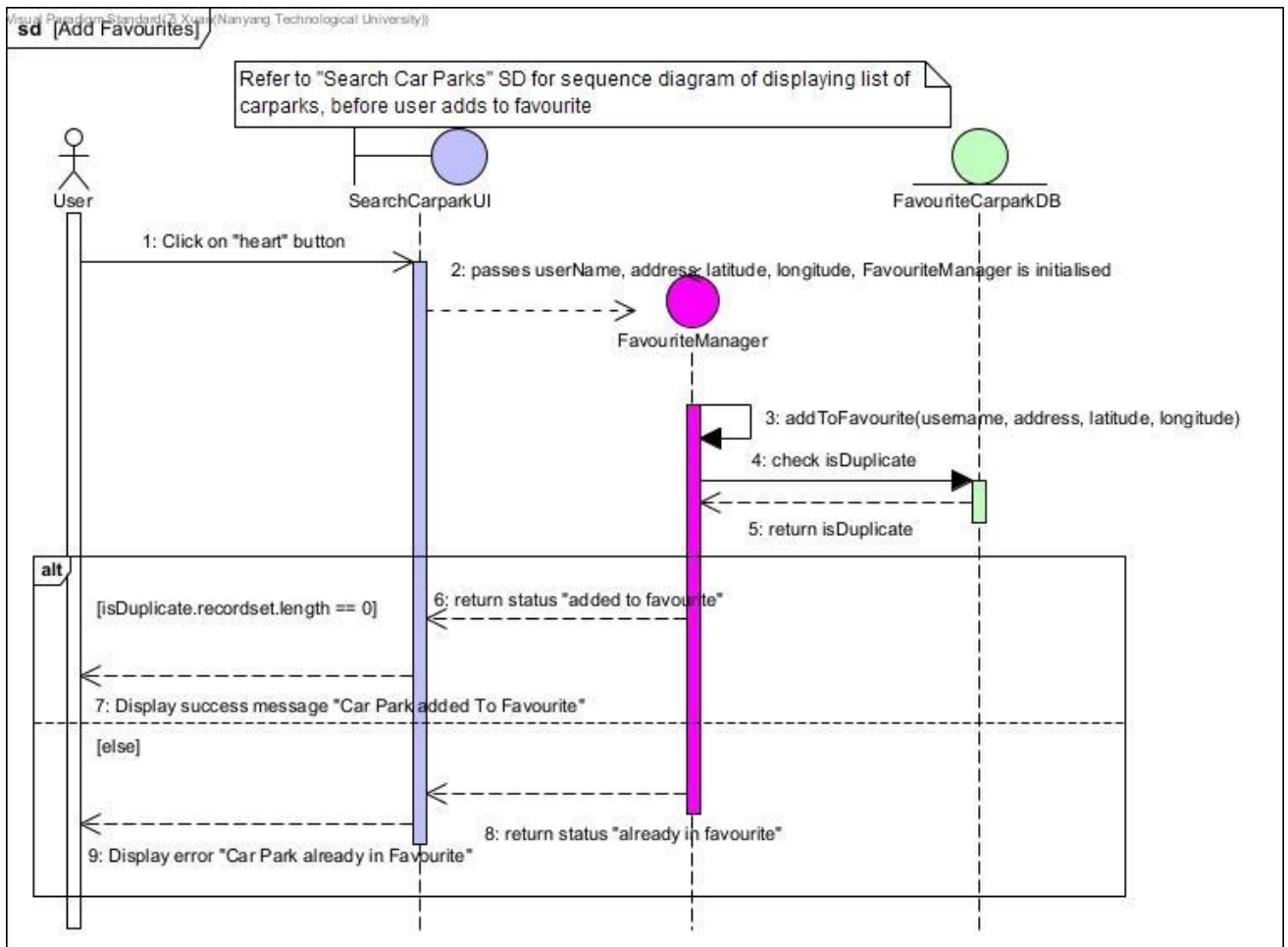
2.2.7 Navigate from Search Carpark

Visual Programming Standard (2020) (Keele University)

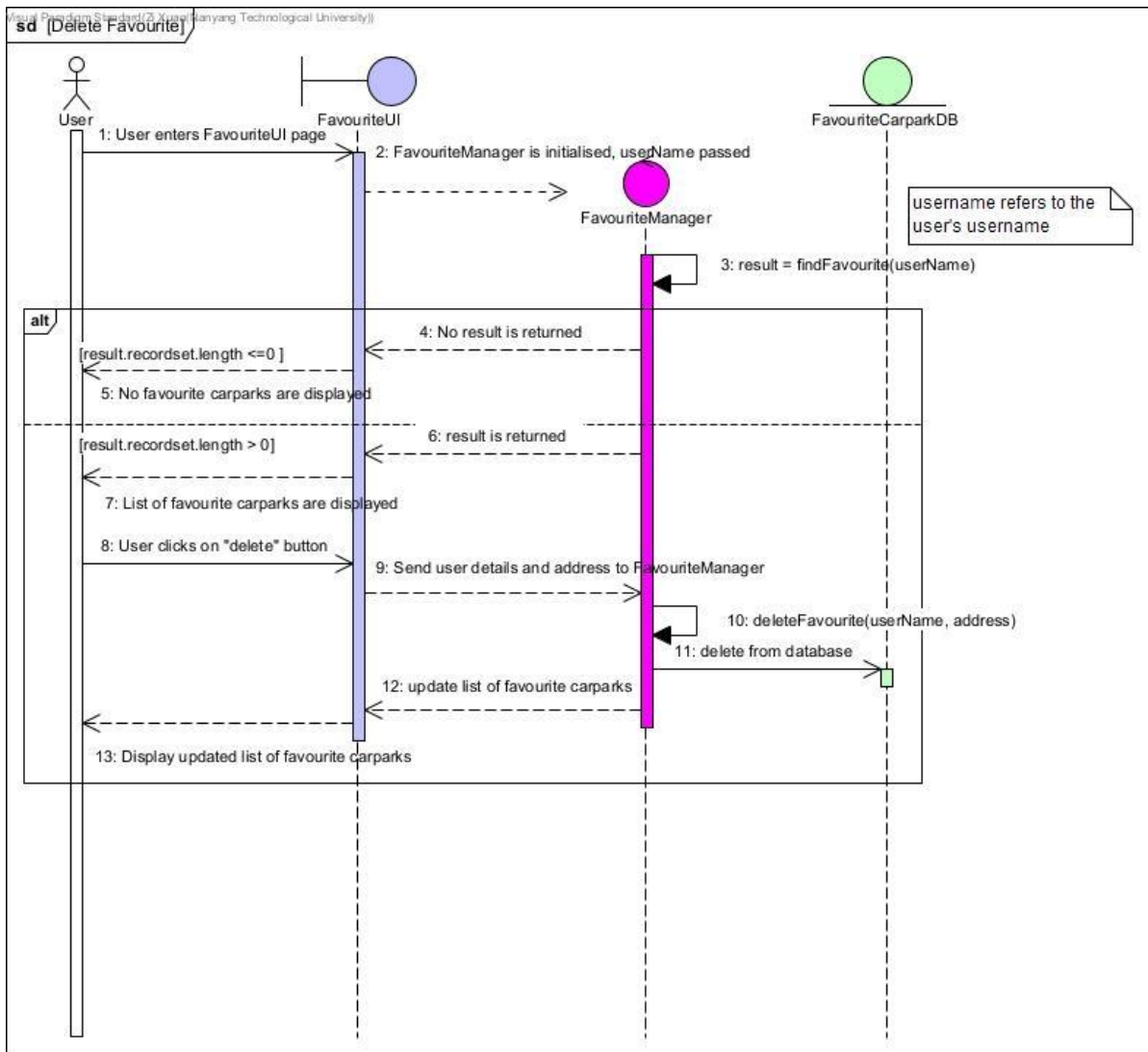
sd (Navigate from Search Carpark)



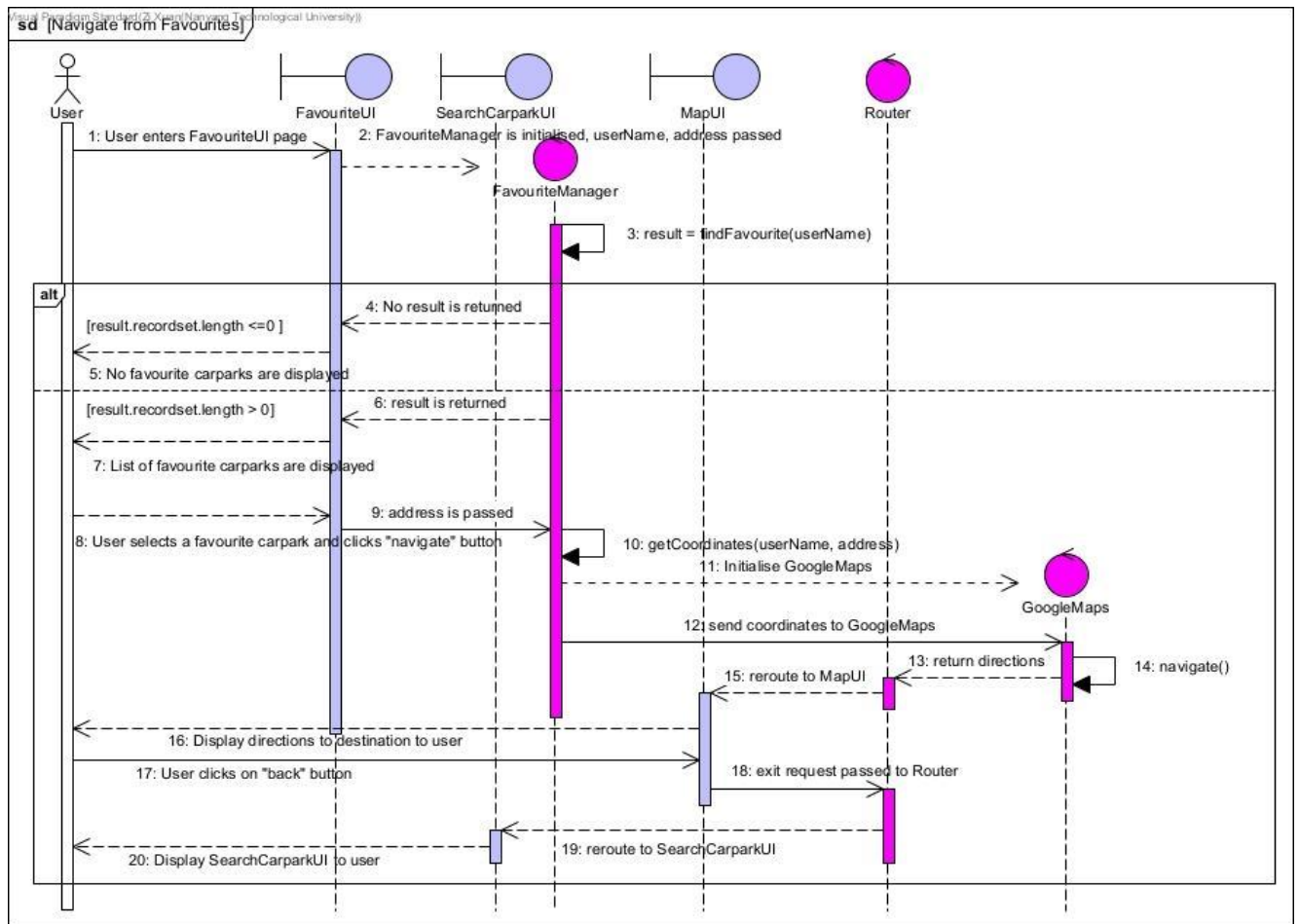
2.2.8 Add Favourites



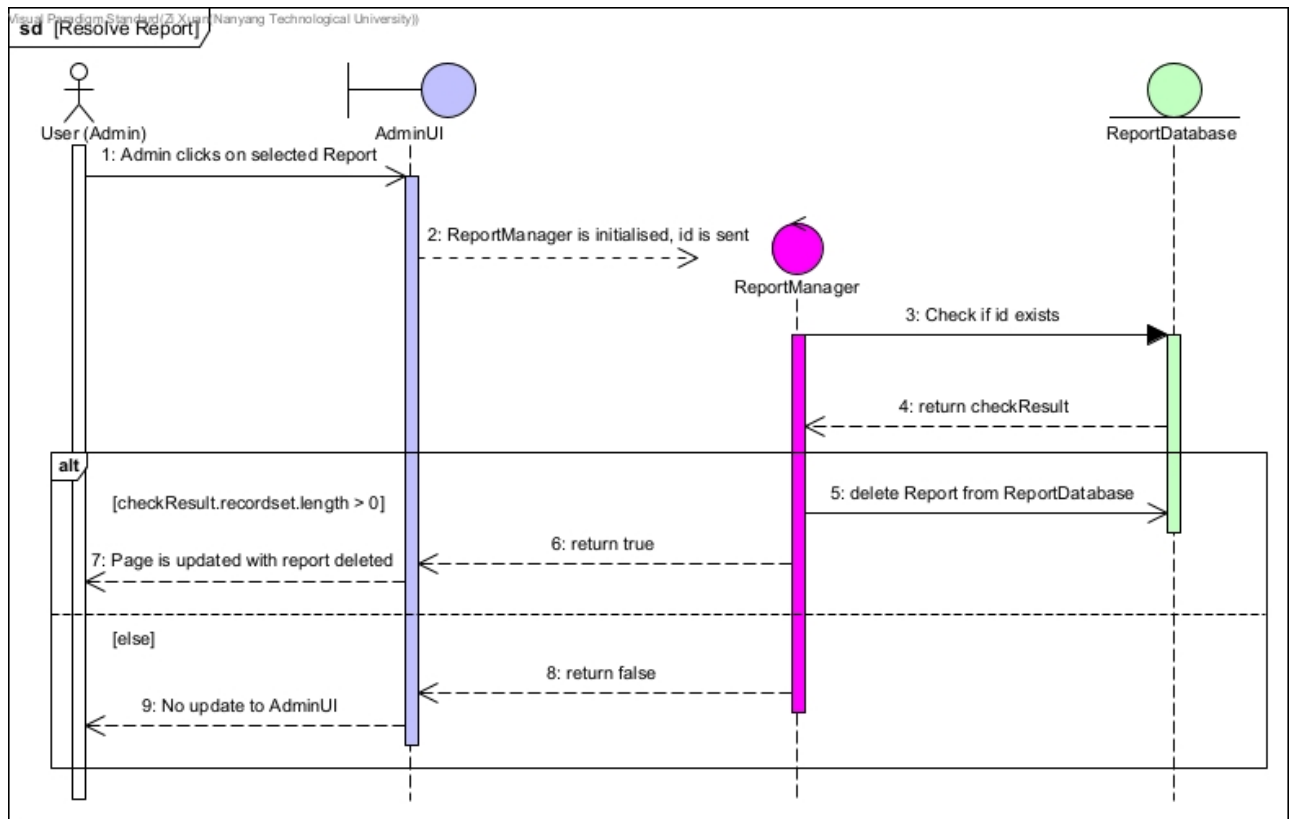
2.2.9 Delete Favourite



2.2.10 Navigate from Favourites

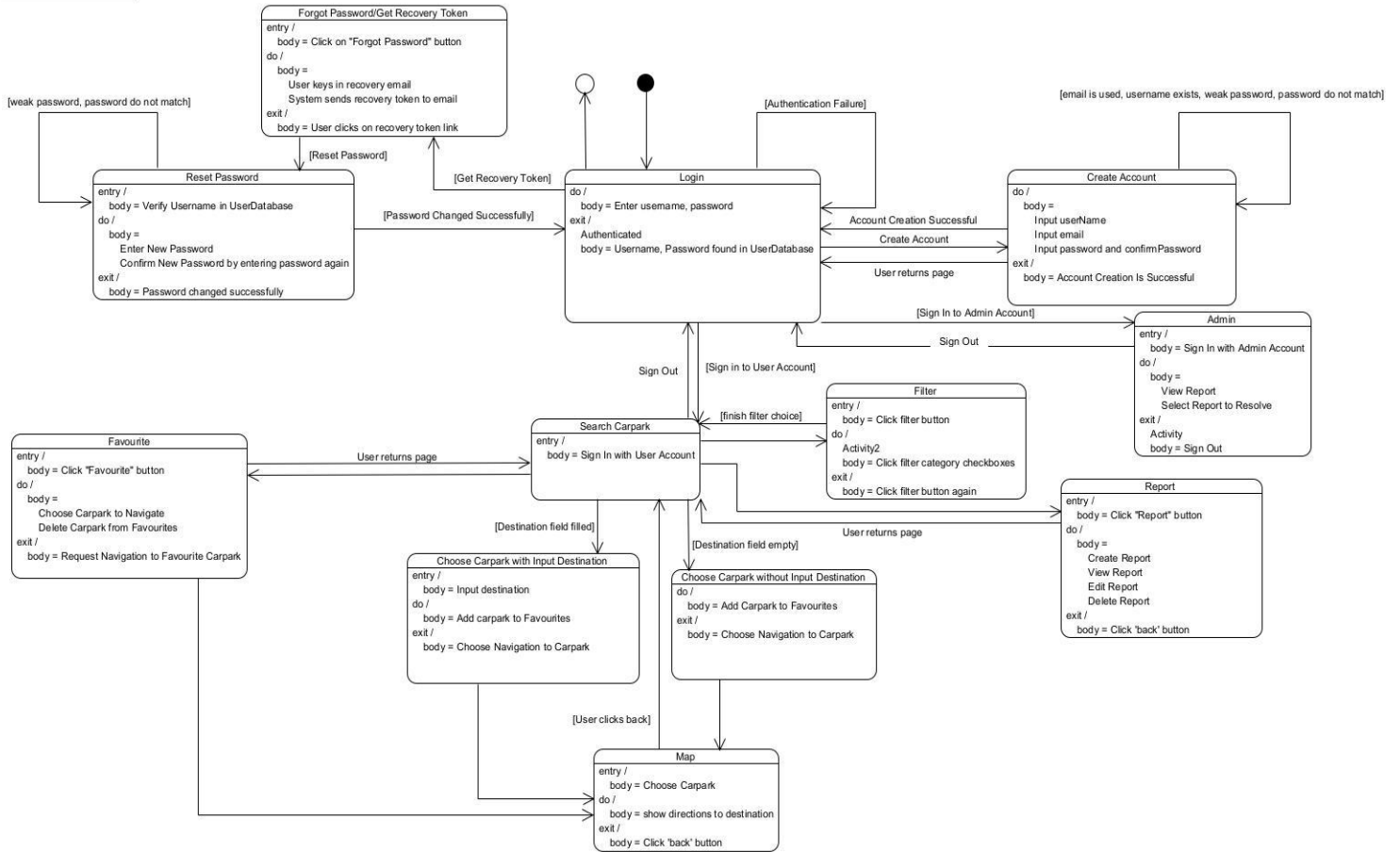


2.2.11 Admin Resolve Report

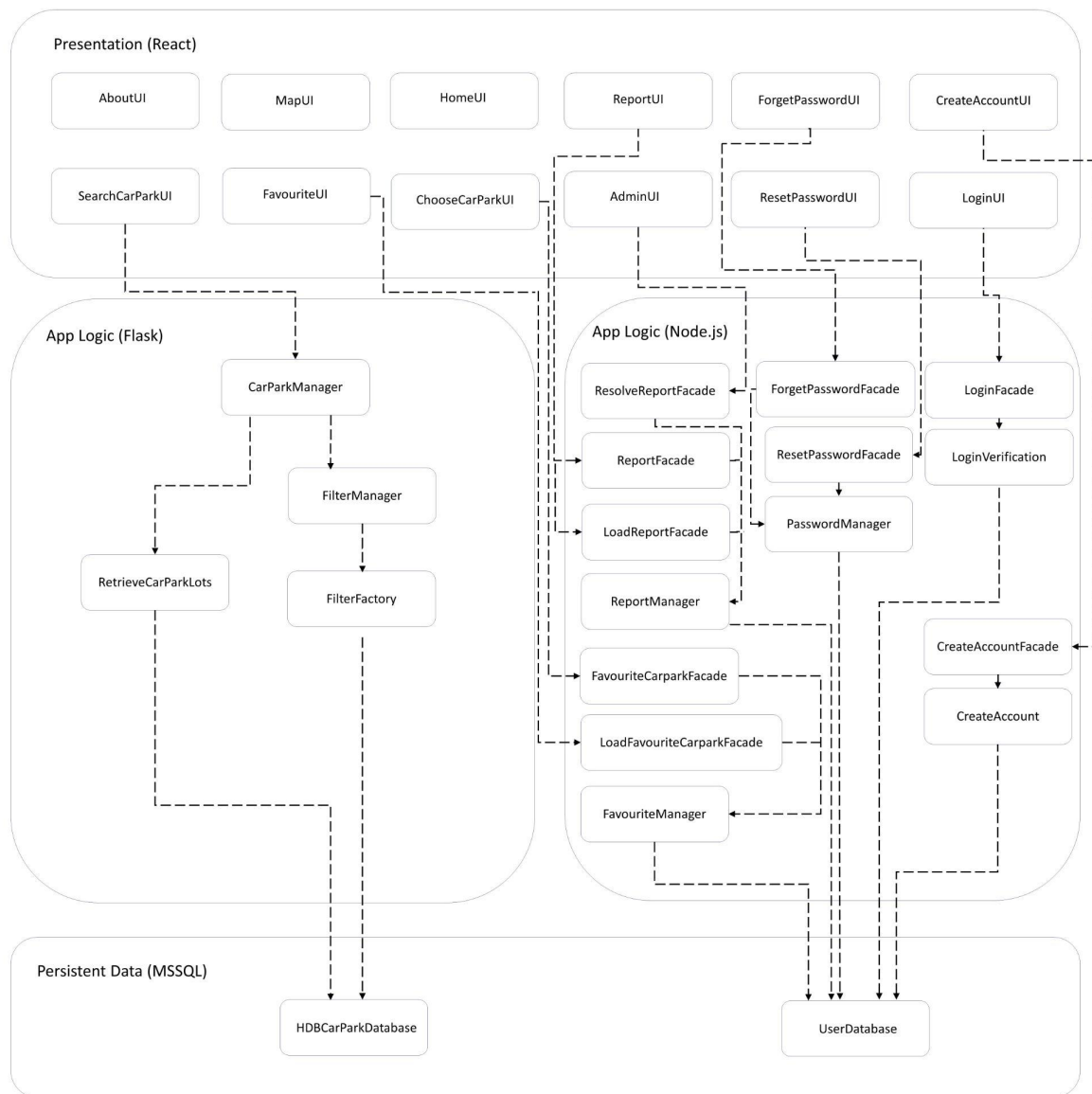


2.3. Dialog Map

tm [State Machine Diagram]



2.4. System Architecture



3. Testing

3.1. Black-Box Testing

3.1.1 Functionality: Create Account

Test Case: Fields Are Left Empty

Location: Create Account

Input	Oracle	Output
Email: ongs0144@e.ntu.edu.sg Password: Abcd@123 Confirm Password: Abcd@123	Successful account creation, redirected to LoginUI	Successful account creation, redirected to LoginUI
Email: ongs0144@e.ntu.edu.sg Password: Confirm Password:	Return error message to prompt user to fill in fields	Return error message to prompt user to fill in fields

Test Case: Password Length is at least 8

Location: Create Account

Input	Oracle	Output
Email: ongs0144@e.ntu.edu.sg Password: Abcd@123 Confirm Password: Abcd@123	Successful account creation, redirected to LoginUI	Successful account creation, redirected to LoginUI
Email: ongs0144@e.ntu.edu.sg Password: Abcd@12 Confirm Password: Abcd@12	Return error message that password is weak	Return error message that password is weak

Test Case: Password Contains Uppercase Letter

Location: Create Account

Input	Oracle	Output
-------	--------	--------

Email: ongs0144@e.ntu.edu.sg Password: Abcd123@ Confirm Password: Abcd123@	Successful account creation, redirected to LoginUI	Successful account creation, redirected to LoginUI
Email: ongs0144@e.ntu.edu.sg Password: abcd123@ Confirm Password: abcd123@	Return error message that password is weak	Return error message that password is weak

Test Case: Password Contains Lowercase Letter

Location: Create Account

Input	Oracle	Output
Email: ongs0144@e.ntu.edu.sg Password: aBCD123@ Confirm Password: aBCD123@	Successful account creation, redirected to LoginUI	Successful account creation, redirected to LoginUI
Email: ongs0144@e.ntu.edu.sg Password: ABCD123@ Confirm Password: ABCD123@	Return error message that password is weak	Return error message that password is weak

Test Case: Password Contains Special Character

Location: Create Account

Input	Oracle	Output
Email: ongs0144@e.ntu.edu.sg Password: Abcd123@ Confirm Password: Abcd123@	Successful account creation, redirected to LoginUI	Successful account creation, redirected to LoginUI
Email: ongs0144@e.ntu.edu.sg Password: Abcd123! Confirm Password: Abcd123!	Successful account creation, redirected to LoginUI	Successful account creation, redirected to LoginUI
Email:	Successful account creation,	Successful account creation,

ongs0144@e.ntu.edu.sg Password: Abcd123# Confirm Password: Abcd123#	redirected to LoginUI	redirected to LoginUI
Email: ongs0144@e.ntu.edu.sg Password: Abcd123\$ Confirm Password: Abcd123\$	Successful account creation, redirected to LoginUI	Successful account creation, redirected to LoginUI
Email: ongs0144@e.ntu.edu.sg Password: Abcd123% Confirm Password: Abcd123%	Successful account creation, redirected to LoginUI	Successful account creation, redirected to LoginUI
Email: ongs0144@e.ntu.edu.sg Password: Abcd123^ Confirm Password: Abcd123^	Successful account creation, redirected to LoginUI	Successful account creation, redirected to LoginUI
Email: ongs0144@e.ntu.edu.sg Password: Abcd123& Confirm Password: Abcd123&	Successful account creation, redirected to LoginUI	Successful account creation, redirected to LoginUI
Email: ongs0144@e.ntu.edu.sg Password: Abcd123* Confirm Password: Abcd123*	Successful account creation, redirected to LoginUI	Successful account creation, redirected to LoginUI
Email: ongs0144@e.ntu.edu.sg Password: Abcd123(Confirm Password: Abcd123(Successful account creation, redirected to LoginUI	Successful account creation, redirected to LoginUI
Email: ongs0144@e.ntu.edu.sg Password: Abcd123) Confirm Password: Abcd123)	Successful account creation, redirected to LoginUI	Successful account creation, redirected to LoginUI
Email:	Successful account creation,	Successful account creation,

ongs0144@e.ntu.edu.sg Password: Abcd123_ Confirm Password: Abcd123_	redirected to LoginUI	redirected to LoginUI
Email: ongs0144@e.ntu.edu.sg Password: Abcd123+ Confirm Password: Abcd123+	Successful account creation, redirected to LoginUI	Successful account creation, redirected to LoginUI
Email: ongs0144@e.ntu.edu.sg Password: Abcd123{ Confirm Password: Abcd123{	Successful account creation, redirected to LoginUI	Successful account creation, redirected to LoginUI
Email: ongs0144@e.ntu.edu.sg Password: Abcd123} Confirm Password: Abcd123}	Successful account creation, redirected to LoginUI	Successful account creation, redirected to LoginUI
Email: ongs0144@e.ntu.edu.sg Password: Abcd123\ Confirm Password: Abcd123\ 	Successful account creation, redirected to LoginUI	Successful account creation, redirected to LoginUI
Email: ongs0144@e.ntu.edu.sg Password: Abcd123[Confirm Password: Abcd123[Successful account creation, redirected to LoginUI	Successful account creation, redirected to LoginUI
Email: ongs0144@e.ntu.edu.sg Password: Abcd123] Confirm Password: Abcd123]	Successful account creation, redirected to LoginUI	Successful account creation, redirected to LoginUI
Email: ongs0144@e.ntu.edu.sg Password: Abcd123: Confirm Password: Abcd123:	Successful account creation, redirected to LoginUI	Successful account creation, redirected to LoginUI
Email:	Successful account creation,	Successful account creation,

ongs0144@e.ntu.edu.sg Password: Abcd123; Confirm Password: Abcd123;	redirected to LoginUI	redirected to LoginUI
Email: ongs0144@e.ntu.edu.sg Password: Abcd123< Confirm Password: Abcd123<	Successful account creation, redirected to LoginUI	Successful account creation, redirected to LoginUI
Email: ongs0144@e.ntu.edu.sg Password: Abcd123> Confirm Password: Abcd123>	Successful account creation, redirected to LoginUI	Successful account creation, redirected to LoginUI
Email: ongs0144@e.ntu.edu.sg Password: Abcd123, Confirm Password: Abcd123,	Successful account creation, redirected to LoginUI	Successful account creation, redirected to LoginUI
Email: ongs0144@e.ntu.edu.sg Password: Abcd123. Confirm Password: Abcd123.	Successful account creation, redirected to LoginUI	Successful account creation, redirected to LoginUI
Email: ongs0144@e.ntu.edu.sg Password: Abcd123? Confirm Password: Abcd123?	Successful account creation, redirected to LoginUI	Successful account creation, redirected to LoginUI
Email: ongs0144@e.ntu.edu.sg Password: Abcd123~ Confirm Password: Abcd123~	Successful account creation, redirected to LoginUI	Successful account creation, redirected to LoginUI
Email: ongs0144@e.ntu.edu.sg Password: Abcd123\ Confirm Password: Abcd123\	Successful account creation, redirected to LoginUI	Successful account creation, redirected to LoginUI
Email:	Successful account creation,	Successful account creation,

ongs0144@e.ntu.edu.sg Password: Abcd123- Confirm Password: Abcd123-	redirected to LoginUI	redirected to LoginUI
Email: ongs0144@e.ntu.edu.sg Password: Abcd123a Confirm Password: Abcd123a	Return error message that password is weak	Return error message that password is weak

Test Case: Password Contains Digit

Location: Create Account

Input	Oracle	Output
Email: ongs0144@e.ntu.edu.sg Password: Abcdab@1 Confirm Password: Abcdab@1	Successful account creation, redirected to LoginUI	Successful account creation, redirected to LoginUI
Email: ongs0144@e.ntu.edu.sg Password: Abcdab@a Confirm Password: Abcdab@a	Return error message that password is weak	Return error message that password is weak

Test Case: Passwords Do Not Match

Location: Create Account

Input	Oracle	Output
Email: ongs0144@e.ntu.edu.sg Password: Abcdab@1 Confirm Password: Abcdab@1	Successful account creation, redirected to LoginUI	Successful account creation, redirected to LoginUI
Email: ongs0144@e.ntu.edu.sg Password: Abcdab@1 Confirm Password: Abcdab@2	Return error message that passwords do not match	Return error message that passwords do not match

Test Case: Invalid username (i.e ongs0144@e.ntu.edu.sg is already used)

Location: Create Account

Input	Oracle	Output
Email: hello@e.ntu.edu.sg Password: Abcdab@1 Confirm Password: Abcdab@1	Successful account creation, redirected to LoginUI	Successful account creation, redirected to LoginUI
Email: ongs0144@e.ntu.edu.sg Password: Abcdab@1 Confirm Password: Abcdab@1	Return error message that email has already been used	Return error message that email has already been used

3.1.2 Functionality: Login

Test Case: Invalid username (i.e. no record of ongs0144@e.ntu.edu.sg)

Location: Login

Input	Oracle	Output
*email record exists Email: ongs0144@e.ntu.edu.sg Password: Abcdab@1	Successful login, redirected to SearchCarparkUI	Successful login, redirected to SearchCarparkUI
*email record does not exist Email: ongs0144@e.ntu.edu.sg Password: Abcdab@1	Return error message that username or password is wrong	Return error message that username or password is wrong

Test Case: Invalid password

Location: Login

Input	Oracle	Output
*password in database is "Abcdab@1" Email: ongs0144@e.ntu.edu.sg Password: Abcdab@1	Successful login, redirected to SearchCarparkUI	Successful login, redirected to SearchCarparkUI
*password in database is	Return error message that	Return error message that

not "Abcdab@1" Email: ongs0144@e.ntu.edu.sg Password: Abcdab@1	username or password is wrong	username or password is wrong
-----------------------------------------------------------------------------	-------------------------------	-------------------------------

3.1.3 Functionality: Forget Password

Test Case: Fields are Left Empty

Location: Forget Password

Input	Oracle	Output
Email: ongs0144@e.ntu.edu.sg	Email sent to email in input field, displays success message to indicate verification email has been sent	Email sent to email in input field, displays success message to indicate verification email has been sent
Email:	Return error message to prompt user to fill in fields	Return error message to prompt user to fill in fields

Test Case: Email Is Not In Database

Location: Forget Password

Input	Oracle	Output
*Email exists Email: ongs0144@e.ntu.edu.sg	Email sent to email in input field, displays success message to indicate verification email has been sent	Email sent to email in input field, displays success message to indicate verification email has been sent
*Email does not exist Email: ongs0144@e.ntu.edu.sg	Return error message that account is not found	Return error message that account is not found

3.1.4 Functionality: Reset Password

Test Case: Fields are Left Empty

Location: Reset Password

Input	Oracle	Output
-------	--------	--------

Password: Abcd@123 Confirm Password: Abcd@123	Successful password reset, success message displayed	Successful password reset, success message displayed
Password: Confirm Password:	Return error message to prompt user to fill in fields	Return error message to prompt user to fill in fields

Test Case: Password Length is at least 8

Location: Reset Password

Input	Oracle	Output
Password: Abcd@123 Confirm Password: Abcd@123	Successful password reset, success message displayed	Successful password reset, success message displayed
Password: Abcd@12 Confirm Password: Abcd@12	Return error message that password is weak	Return error message that password is weak

Test Case: Password Contains Uppercase Letter

Location: Reset Password

Input	Oracle	Output
Password: Abcd123@ Confirm Password: Abcd123@	Successful password reset, success message displayed	Successful password reset, success message displayed
Password: abcd123@ Confirm Password: abcd123@	Return error message that password is weak	Return error message that password is weak

Test Case: Password Contains Lowercase Letter

Location: Reset Password

Input	Oracle	Output
Password: aBCD123@ Confirm Password: aBCD123@	Successful password reset, success message displayed	Successful password reset, success message displayed
Password: ABCD123@ Confirm Password: ABCD123@	Return error message that password is weak	Return error message that password is weak

Test Case: Password Contains Special Character

Location: Reset Password

Input	Oracle	Output
Password: Abcd123@ Confirm Password: Abcd123@	Successful password reset, success message displayed	Successful password reset, success message displayed
Password: Abcd123! Confirm Password: Abcd123!	Successful password reset, success message displayed	Successful password reset, success message displayed
Password: Abcd123# Confirm Password: Abcd123#	Successful password reset, success message displayed	Successful password reset, success message displayed
Password: Abcd123\$ Confirm Password: Abcd123\$	Successful password reset, success message displayed	Successful password reset, success message displayed
Password: Abcd123% Confirm Password: Abcd123%	Successful password reset, success message displayed	Successful password reset, success message displayed
Password: Abcd123^ Confirm Password: Abcd123^	Successful password reset, success message displayed	Successful password reset, success message displayed
Password: Abcd123& Confirm Password: Abcd123&	Successful password reset, success message displayed	Successful password reset, success message displayed
Password: Abcd123* Confirm Password: Abcd123*	Successful password reset, success message displayed	Successful password reset, success message displayed
Password: Abcd123(Confirm Password: Abcd123(Successful password reset, success message displayed	Successful password reset, success message displayed
Password: Abcd123) Confirm Password: Abcd123)	Successful password reset, success message displayed	Successful password reset, success message displayed
Password: Abcd123_ Confirm Password: Abcd123_	Successful password reset, success message displayed	Successful password reset, success message displayed
Password: Abcd123+ Confirm Password:	Successful password reset, success message displayed	Successful password reset, success message displayed

Abcd123+		
Password: Abcd123{ Confirm Password: Abcd123{	Successful password reset, success message displayed	Successful password reset, success message displayed
Password: Abcd123} Confirm Password: Abcd123}	Successful password reset, success message displayed	Successful password reset, success message displayed
Password: Abcd123\ Confirm Password: Abcd123\ 	Successful password reset, success message displayed	Successful password reset, success message displayed
Password: Abcd123[Confirm Password: Abcd123[Successful password reset, success message displayed	Successful password reset, success message displayed
Password: Abcd123] Confirm Password: Abcd123]	Successful password reset, success message displayed	Successful password reset, success message displayed
Password: Abcd123: Confirm Password: Abcd123:	Successful password reset, success message displayed	Successful password reset, success message displayed
Password: Abcd123; Confirm Password: Abcd123;	Successful password reset, success message displayed	Successful password reset, success message displayed
Password: Abcd123< Confirm Password: Abcd123<	Successful password reset, success message displayed	Successful password reset, success message displayed
Password: Abcd123> Confirm Password: Abcd123>	Successful password reset, success message displayed	Successful password reset, success message displayed
Password: Abcd123, Confirm Password: Abcd123,	Successful password reset, success message displayed	Successful password reset, success message displayed
Password: Abcd123. Confirm Password: Abcd123.	Successful password reset, success message displayed	Successful password reset, success message displayed
Password: Abcd123? Confirm Password: Abcd123?	Successful password reset, success message displayed	Successful password reset, success message displayed

Password: Abcd123~ Confirm Password: Abcd123~	Successful password reset, success message displayed	Successful password reset, success message displayed
Password: Abcd123\ Confirm Password: Abcd123\ 	Successful password reset, success message displayed	Successful password reset, success message displayed
Password: Abcd123- Confirm Password: Abcd123-	Successful password reset, success message displayed	Successful password reset, success message displayed
Password: Abcd123a Confirm Password: Abcd123a	Return error message that password is weak	Return error message that password is weak

3.2. White-Box Testing

3.2.1 Create User



3.2.2 Navigation



3.3. Demo Script

Create Account

- 1) Create an account with:
 - a. email "ongs0144e.ntu.edu.sg"
 - b. username: "shengda"
 - c. password: "QWERTY1234\$a"
 - d. confirm password: "QWERTY1234\$a"
- 2) Show that the account cannot be created with the same email address and password.

Login

- 1) Login with the credentials of the created account

Search Car Park

- 1) Show that the user cannot search a place not within Singapore, for example, "Eiffel Tower".
- 2) Show that the user can search for a place within Singapore.
- 3) Show that data shows the details of car park availability, distance from destination inputted, and ability to add to favourite.
- 4) Once added to the favourite, show cannot be added to the favourite again.
- 5) Navigate to the car park of choice.
- 6) Exit Navigation

Favourite Car Park

- 1) Show the ability to delete the car park previously added to the favourite.
- 2) Show the ability to navigate to the car park of choice.

Report Fault

- 1) Show that users are able to report any faults, regarding app bugs, car park faults etc.
- 2) Explain that the report is only visible to the admin in charge of the app. **No warnings will be issued to other users should there be any car park faults reported to prevent abuse usage of the app by the users.**

Admin

- 1) Show that on the admin page, admins are able to view the faults reported.
- 2) Explain that the admins will resolve the issues reported via human intervention such as reporting to relevant authorities manually, or contacting the programming team in case of app bugs.
- 3) Show that the case reported will be deleted once the admins click the “resolve” button.

Forget Password

- 1) Show that email credentials are needed to send users a recovery link to reset their password

Reset Password

- 1) Enter a password to reset the password.
- 2) Show that the user will be able to login with their reset password.

4. Appendix A: Glossary

Acronym/Abbreviation	Full Name
API	Application Programming Interface - A software service which is in charge of interaction between two or more software. It is usually used by the developers of a software product.
GPS	Global Positioning System - A satellite radio-based navigation system which transmits receivers geolocation and time information
HDB	Housing Development Board - A statutory board in Singapore which is responsible for matters related to public housing
JSON	Javascript Object Notation - A data format written in natural language which contains value-pairs and/or arrays
MSSQL	Microsoft Structured Query Language Server - A relational database management system developed by Microsoft
UI	User Interface - The external hardware of a device which allows user to interact with an application

5. Appendix B: Meeting Minutes

Week/Date	Tasks
Week 2 25 Aug 2023	<ul style="list-style-type: none">● Appoint Group Leader● Create Telegram chat group for communication and exchange contacts● Created document for Lab 1 Deliverables● Project Ideation
Week 3 29 Aug 2023	<ul style="list-style-type: none">● Start on requirement specifications on Lab 1 document:<ul style="list-style-type: none">○ Data Dictionary○ Functional Requirements○ Non-Functional Requirements● Share individual use case diagrams agree on the functionalities of the application● Begin preliminary drafts of UI prototypes
Week 4 4 Sep 2023	<ul style="list-style-type: none">● Finalise drafts of:<ul style="list-style-type: none">○ Functional Requirements○ Non-Functional Requirements○ Use Case Diagram○ Use Case Descriptions○ UI Mockups
Week 4 6 Sep 2023	<ul style="list-style-type: none">● Look through the Lab 1 Deliverables for any inconsistencies● Standardise the Use Case Descriptions for the Lab 1 Deliverables document
Week 4 8 Sep 2023	<ul style="list-style-type: none">● Attend Lab 2 to review Lab 1 Deliverables● Reword phrasing of Use Case Descriptions to make them more concrete● Include exceptions, assumptions into Use Case Descriptions
Week 4 9 Sep 2023	<ul style="list-style-type: none">● Discuss tasks for next meeting (Class Stereotype Diagram)
Week 5 10 Sep 2023	<ul style="list-style-type: none">● Combine ideas to abstract Class Stereotype Diagram
Week 5 11 Sep 2023	<ul style="list-style-type: none">● Finalise the Class Stereotype Diagram
Week 5 15 Sep 2023	<ul style="list-style-type: none">● Discuss the sequence flow for Sequence Diagrams● Work on individual Sequence Diagrams

Week 5 17 Sep 2023	<ul style="list-style-type: none"> ● Finalise Sequence Diagrams for Lab 2 Deliverables ● Complete Class Entity Diagram
Week 6 18 Sep 2023	<ul style="list-style-type: none"> ● Complete State Machine Diagram ● Review UI Mockups after edits of State Machine Diagram
Week 6 20 Sep 2023	<ul style="list-style-type: none"> ● Review Sequence Diagrams, Class Entity Diagram and State Machine Diagram for inconsistencies ● Familiarise with Node, React individually
Week 7 25 Sep 2023	<ul style="list-style-type: none"> ● Delegate programming tasks for frontend and backend ● Review methods and classes needed for different use cases
Recess Week 3 Oct 2023	<ul style="list-style-type: none"> ● Discuss source codes for frontend and backend
Week 8 11 Oct 2023	<ul style="list-style-type: none"> ● Discuss source codes for frontend and backend ● Review Class Entity Diagrams after referencing to source codes ● Start on System Architecture
Week 9 20 Oct 2023	<ul style="list-style-type: none"> ● Revisit Sequence Diagrams, State Machine Diagram ● Finalise System Architecture Diagram ● Finalise Use Case Diagram
Week 10 24 Oct 2023	<ul style="list-style-type: none"> ● Synthesise all the Lab 3 Deliverables into a document
Week 11 4 Nov 2023	<ul style="list-style-type: none"> ● Discuss source codes for frontend and backend
Week 12 7 Nov 2023	<ul style="list-style-type: none"> ● Create Demo Script for Live Demo ● Create Powerpoint slides for Lab 4/5 presentation ● Start on Black-Box Testing Cases ● Complete Key Public Methods Description
Week 12 9 Nov 2023	<ul style="list-style-type: none"> ● Finalise Demo Script, presentation slides for Live Demo ● Rehearse Demo Script ● Troubleshoot and test source code for any bugs ● Create SRS document
Week 12 10 Nov	<ul style="list-style-type: none"> ● Transfer all relevant sections and diagrams into SRS document ● Review SRS document for any inconsistencies and complete SRS documentation ● Film Demo Video ● Prepare for Lab 4/5 submission on 12 Nov 2023

