Software Requirements Specification

for

CarParkGoWhere

Version 3.0 approved

Prepared by Ong Jing Jie, Seow Jing Woon, Cheng Yi Feng, Jumana Haseen, Damien Lee Han Wei, Lim Haoren

Nanyang Technological University

14/11/2024

Copyright © 1999 by Karl E. Wiegers. Permission is granted to use, modify, and distribute this document.

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Ong Jing Jie	4/9/2024	Initial Version	1.0
Lim Haoren	6/11/2024	Update SRS	2.0

1. Introduction

1.1 Purpose

With the high vehicle ownership numbers in Singapore, Singapore faces a significant parking problem, particularly in densely populated areas such as the Central Business District area. To attempt to resolve this issue, Team 46 aims to develop a website that allows users to find the nearest, cheapest or fastest car park based on their destination. This project will be considered complete when the website is tested and approved for release by the ZEA. This project supports the ZEA in eliciting innovative applications in support of the Smart Nation initiative.

1.2 Document Conventions

The body text of this document is written in standard font Arial, 11-point size for readability. All headings and subsections are written in bold and numbered hierarchically e.g. 1, 1.1, 1.1.1. Code snippets, user inputs and program outputs are written in font roboto mono, 11-point size. Prioritisation is explicitly stated for each requirement, if no priority for the detailed requirement is explicitly declared then it is assumed that they inherit the priority of the higher-level requirement. Requirements have separate parameters where each will be ranked from one to ten, ten being of higher weightage. Each functional requirement will have a unique code in the form “REQ-1” to facilitate referencing and traceability. Each sub-requirement will follow the parent requirement with an additional number in the form “REQ-1.1”. Each version of the SRS document is assigned a version number that will be incremented with each revision. The version number is listed under the project name on the first page.

1.3 Intended Audience and Reading Suggestions

This document is intended for the reading of developers, testers and project managers from ZEA.

Organization of the SRS Document:

- Introduction
- Overall Description
- External Interface Requirement
- System Features
- Other Nonfunctional Requirements
- Other Requirements

The suggested sequence for reading the document is from the start of the document to the end of the document, from section 1 to section 6 in chronological order.

1.4 Product Scope

The purpose of the software is to allow users to use an online map and GPS coordinates of the vehicle to find the most suitable parking lots in car parks near the user's final destination. We aim to reduce the hassle of users such as:

- Having to drive to the car parks to check the prices
- Arriving at car parks only to find no available spaces and facing long wait times.
- The trouble of finding the cheapest and fastest route to the car park.

The software aims to provide users with parking rates at the start of the journey, assist the user in choosing the car parks based on the user's destination, price and distance of the user's location to the car park, and provide the fastest route to the car park.

2. Overall Description

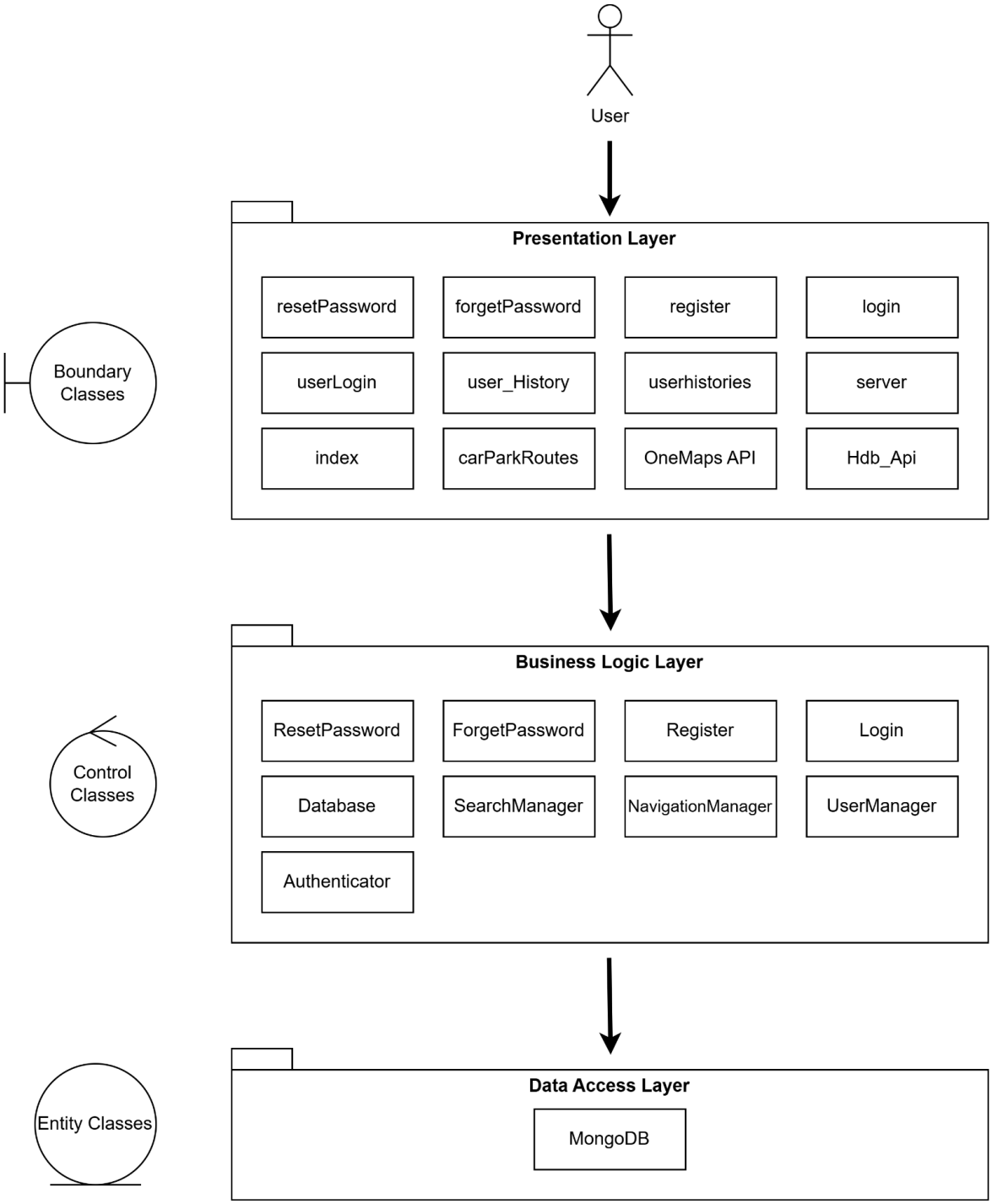
2.1 Product Perspective

This website has been developed as a response to the need for a comprehensive parking solution in Singapore since motorists face difficulty finding affordable and nearby parking spaces that are available. Although existing solutions such as Park&Go @SG, Breeze and SGCarMart's Carpark Rates App provide similar functionalities, they still have certain limitations. For example, some of these applications may lack real-time traffic data accuracy or seamless user experience.

This product is not a follow-on member of this existing product family but it could be considered an enhancement over their systems. The purpose of this system is to build an all-in-one platform, which integrates and improves their strengths such as more accurate real-time data provision and cost calculation, while addressing their weaknesses such as lack of the need for more intuitive user interface or lack of alternative suggestions.

The website is designed to function as a central hub with aggregated parking information from authoritative data sources such as the Housing Development Board (HDB) and map data providers. These data feed our system with accurate real-time data, which will then analyse and provide comprehensive information to users.

2.1.1. System Architecture Diagram

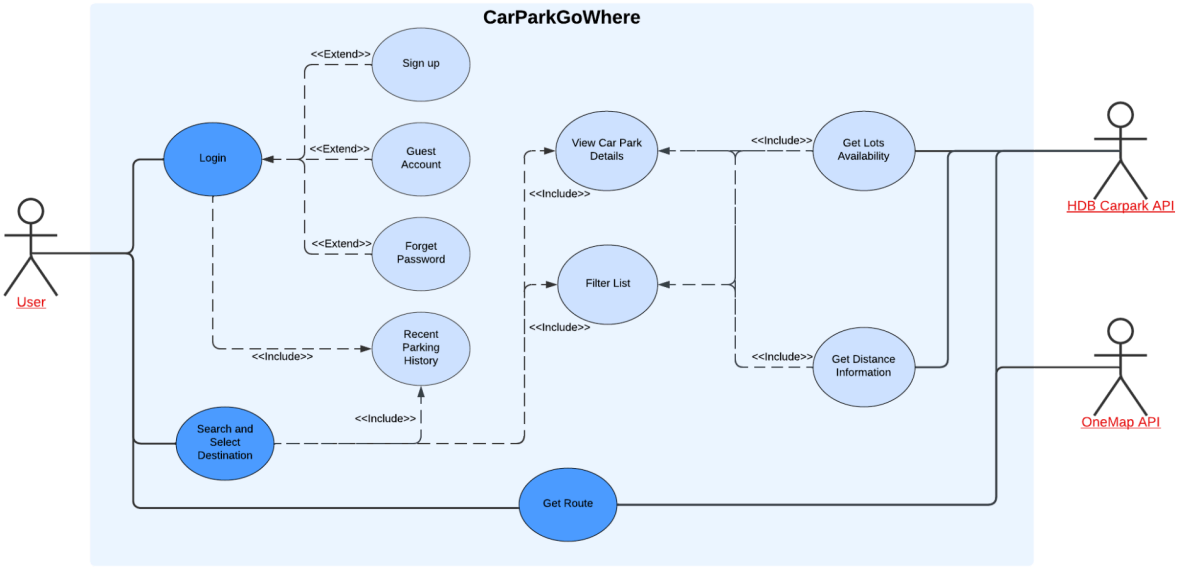


2.2 Product Functions

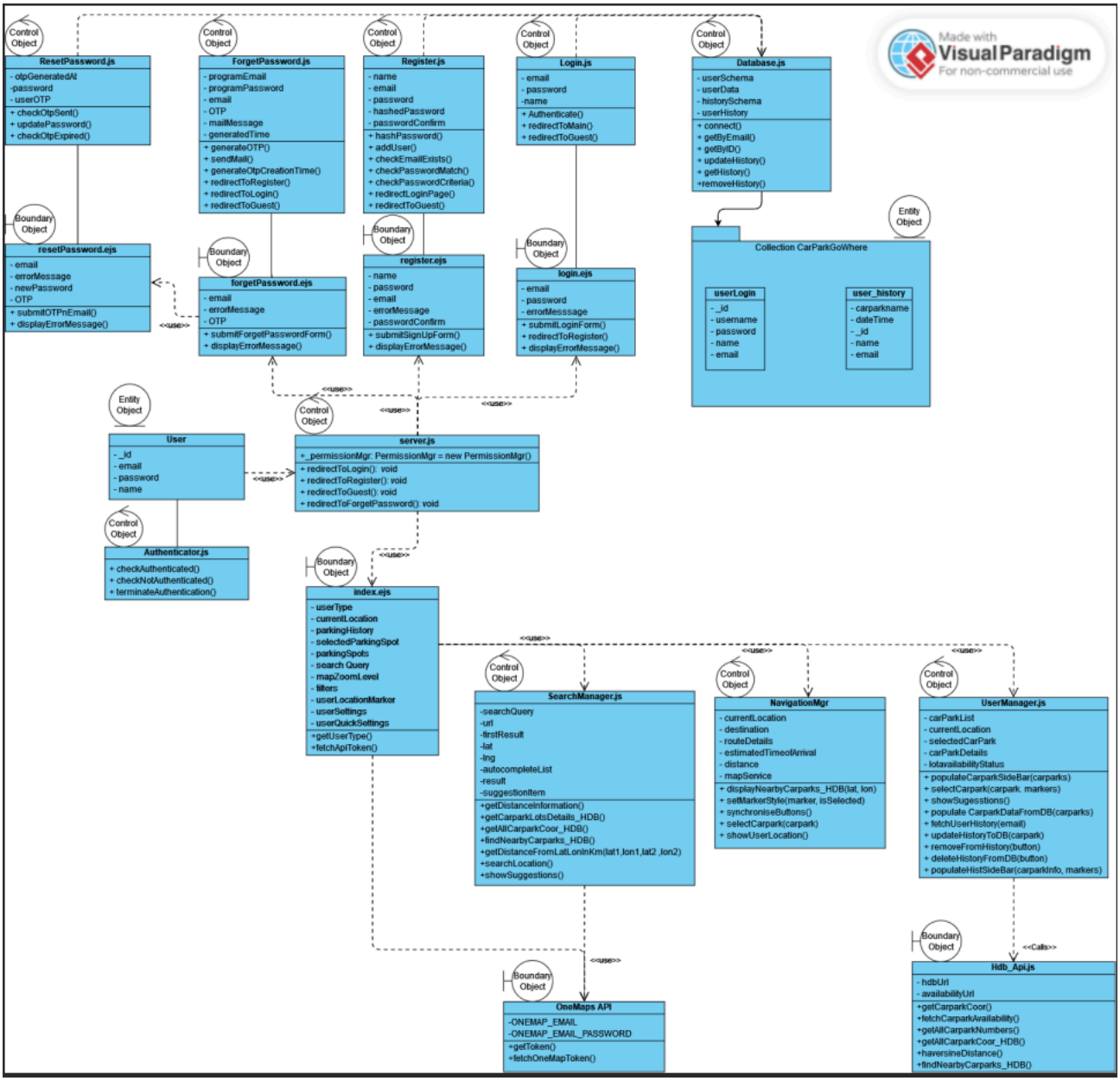
Major Functions the product must perform:

1. Location-based services
    - Display a list of car parks that are near the destination within a certain distance radius
    - Display the nearest route to the chosen car park
2. Price comparison
    - Display prices of the car parks near the destination
3. Search and filtering
    - Allow user to search for car parks near their destination
    - Allow users to filter based on price, distance and car park availability.
4. Real-time availability
    - Display real-time availability of parking spaces
    - Allow users to refresh parking availability information
5. System Navigation Support
    - Show users the best car park based on the selected filter criteria.

2.2.1. Use Case Diagram



2.2.2. Class Diagram



2.3 User Classes and Characteristics

There are several user classes that we can differentiate:

2.3.1. Daily drivers

Frequency of Use	Daily or multiple times a day
Product Functions Use	Nearest Car park, Fastest Car park
Technical Expertise	Moderate to high, familiar with navigation tools
Importance	High

2.3.2. Casual Users

Frequency of Use	Weekly or less
Product Functions Use	Nearest Car Park, cheapest Car Park
Technical Expertise	Basic to moderate
Importance	Medium as they form a significant portion of the Users but may not be using the tool frequently

2.3.3. Tourists or Visitors

Frequency of Use	High but short-term
Product Functions Use	Nearest Car Park, cheapest Car Park
Technical Expertise	Varied, but the app should be user-friendly and intuitive
Importance	Medium

2.3.4. Business Users

Frequency of Use	Regular to frequent usage depending on business needs
Product Functions Use	Cheapest Car Park
Technical Expertise	High
Importance	High, as they may represent a significant revenue stream

2.4 Operating Environment

Software Requirements:

- Google Chrome/ Firefox/ Microsoft Edge/ Safari web browser installed

Hardware Requirements:

- A device that is able to run either Chrome, Firefox, Microsoft Edge or Safari
- Internet access
- Location Service

Permission Requirements:

- Have full network access
- View network connections
- Location service
- Receive data from the internet

2.5 Design and Implementation Constraints

2.5.1 Corporate or Regulatory Policies

The application must comply with OneMap and HDB Carpark API policies, including limitations on API usage, rate limits and authentication requirements. OneMap API authentication requires an authentication key that should be refreshed every 3 days.

2.5.2 Hardware Limitations

Application should be optimised for devices with limited memory and processing power, especially considering that users may access the software on mobile devices. Data-intensive operations on the client side should be minimised to prevent performance impact on less powerful devices.

2.5.3 Technology Requirements

CarparkGoWhere is developed through JavaScript, styled using CSS and HTML.

2.5.4 Databases to be used

In providing live information on car parks, CarparkGoWhere has used APIs from data.gov.sg and mongoDB to store user data. The datasets used:

(i) https://data.gov.sg/collections/148/view

This database does not contain all the car parks in Singapore. The reliability and accuracy of live updates on the availability of the carpark are also entirely dependent on the APIs.

(ii) MongoDB
This will be used to store user credentials like, email and password, as well as user parking history.

2.5.5 Interface Requirements

The software must integrate with existing HDB carpark data APIs as well as OneMap for location service. The interface should be accessible and compatible with modern web browsers such as Google Chrome, Firefox, Safari as well as Microsoft edge.

2.5.6 Design Conventions and Standards:

Developers are required to follow good coding practices to ensure the maintainability and readability of the code. The design must also prioritise a user-friendly interface, especially for navigating map views, searching car parks and viewing parking history.

2.6 User Documentation

These components of user documentation should also be delivered along with the software:

2.6.1 Tutorials

Video Tutorials should demonstrate the usage of key features such as registering, searching for nearby car parks as well as selection of car parks, viewing of parking history and changing of language in the application

2.7 Assumptions and Dependencies

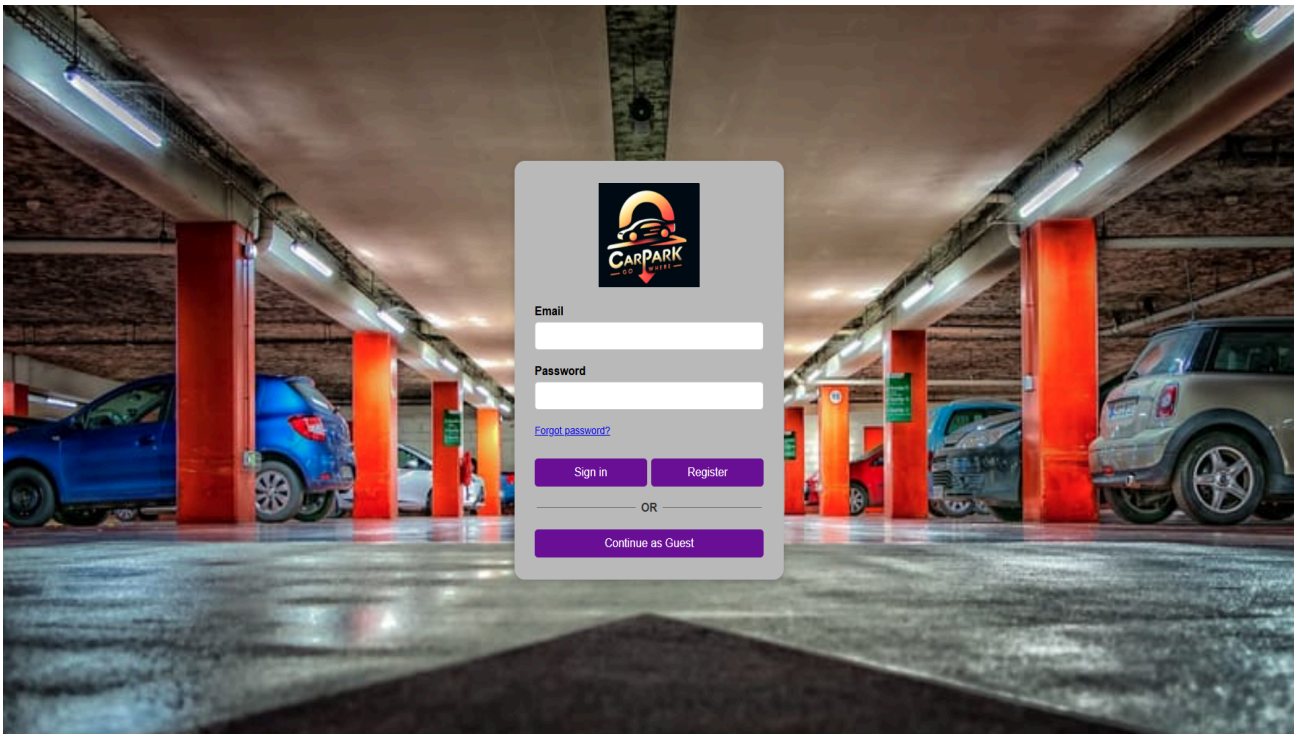
In order to run CarparkGoWhere, users need to have a reasonably capable device that has location service and access to the Internet. Moreover, users will need to have either Chrome, Firefox, Microsoft Edge or Safari installed on their device to access the website. Finally, CarparkGoWhere assumes that the API provides accurate data and that the navigation from OneMap API is precise.

3. System Features

3.1 User Interfaces

The user interface will provide an interactive map that shows car park locations based on user searches and preferences. The user will have search options such as filtering for distance, parking lots availability, and car park price.

Main Interface

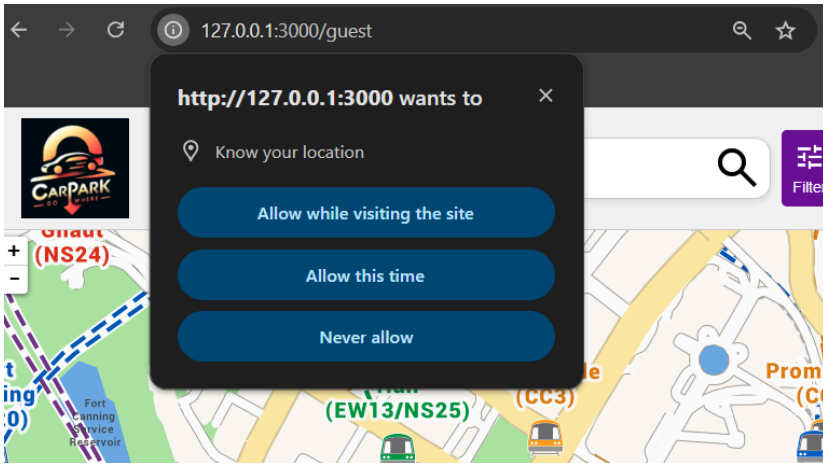


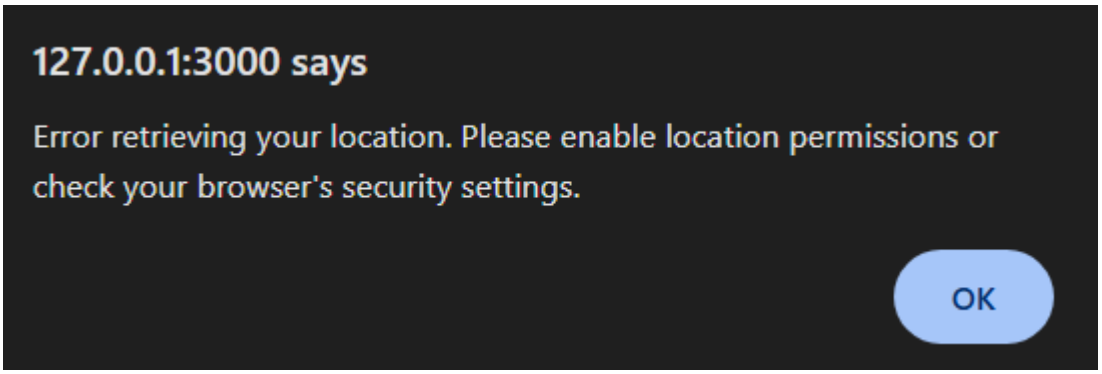
Upon accessing the website, the user will be greeted with this webpage. At this webpage the user has a couple of option:

1. Users are able to login. For registered users, users can proceed to login to their account which will give them access to information such as their location as well as their parking history.

2. For users that are not registered, they are able to either Register for an account so that they will have access to their parking history feature, or they could proceed as Guest if they want to simply quickly access the application.

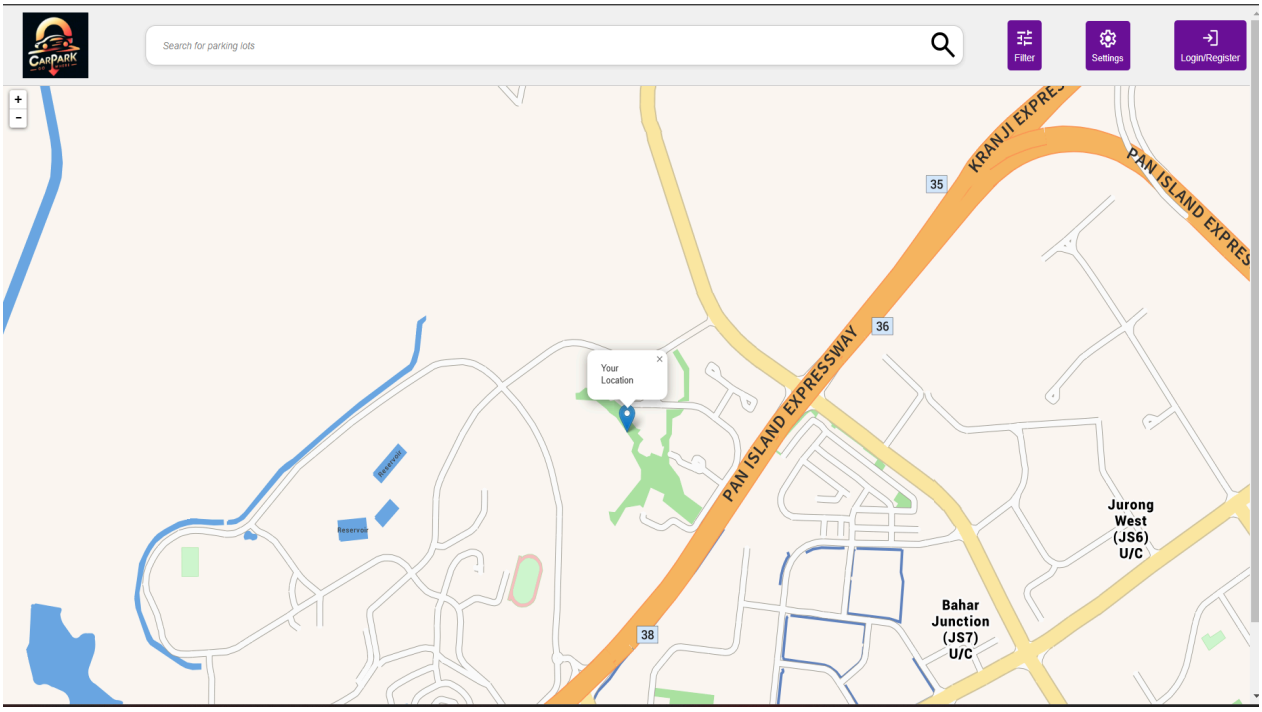
Accessing User's Location





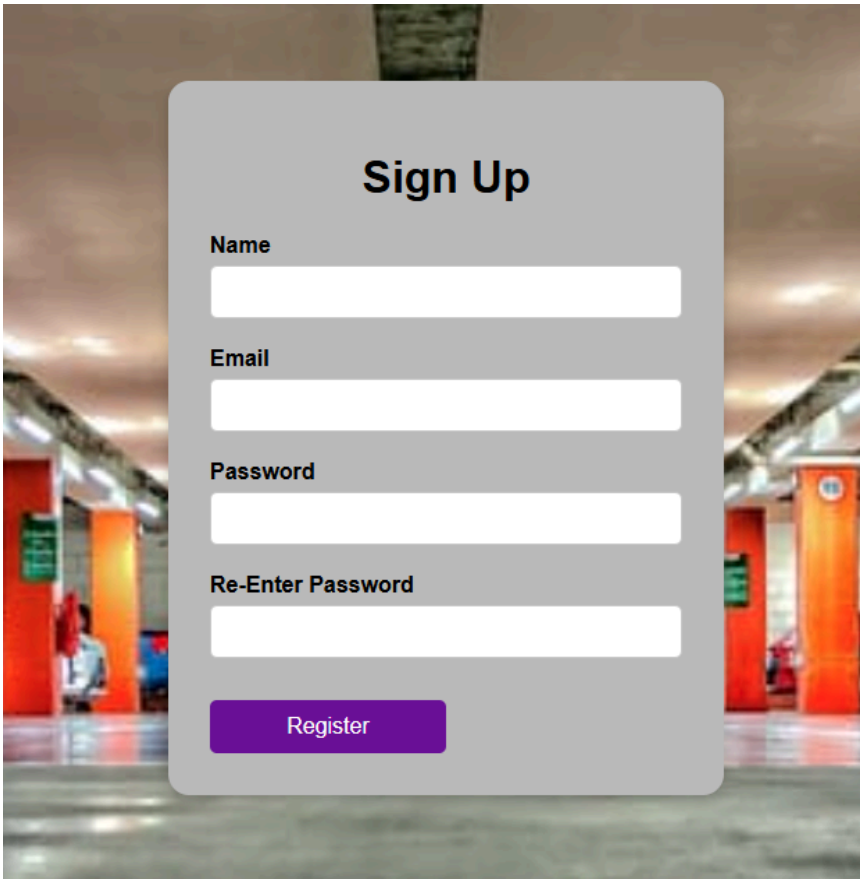
Before CarparkGoWhere is able to offer its functionality, it needs to be able to determine the user's current location. If the permission is not granted, the application will not be able to accurately determine the route to the carpark that the user would want to select. If the permission is denied the message in Figure 3.1.2 will be shown to the user and functionalities will be limited to the user. The only function that would be available to the user is the searching of the location. However, no carpark result nor routes will be displayed for the user.

Guest Interface



For the Guest interface, the user is able to interact with the application just like a normal user with the exception that the user would not be able to access their history of which carpark they chose. The user has the option to go back to the main interface to either register or login as a registered user.

Register Interface

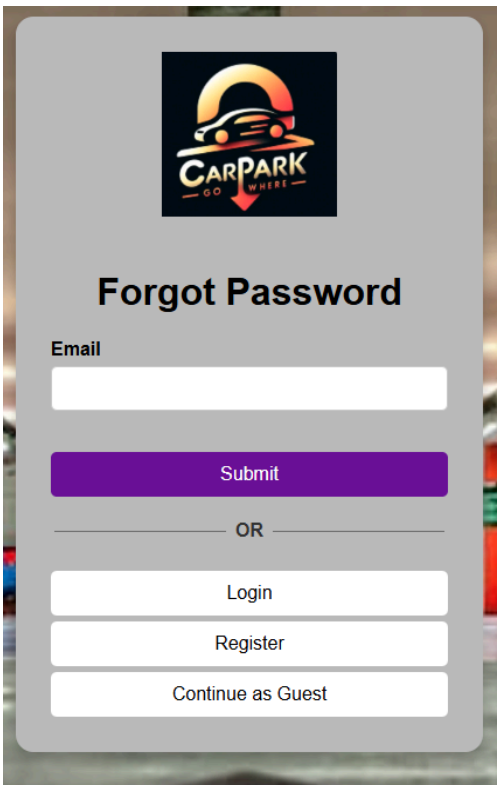


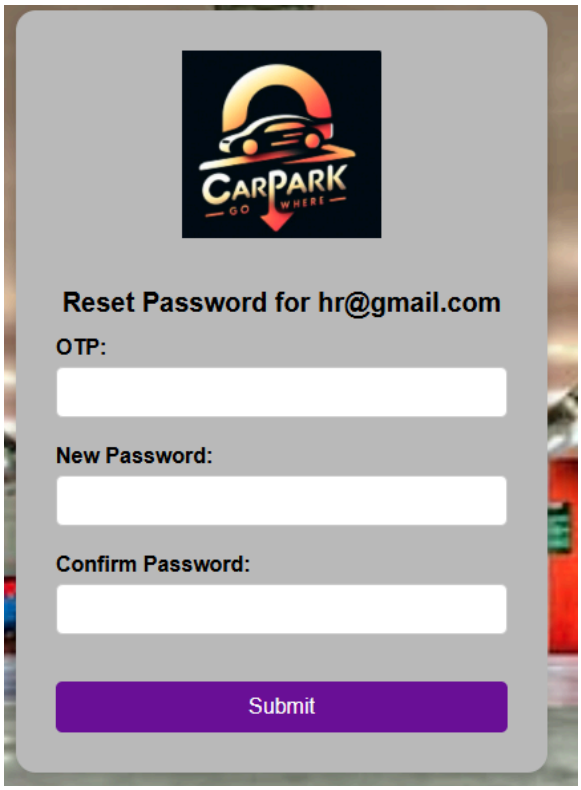

```
_id: ObjectId('672053089afbbe6b488118ce')
name : "seow"
email : "seow@gmail.com"
password : "$2b$10$uE1Na4UvQrquqq1dFS542.J1MnwdRLOKhzHw5i4LWXUQIyKl23Sa2"
__v : 0
```

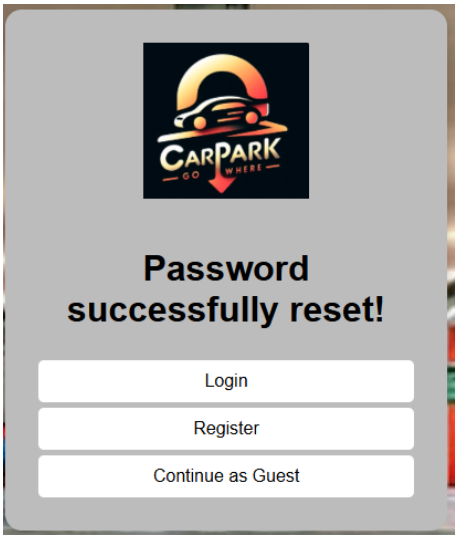
```
_id: ObjectId('67207f447395b1233269f6ed')
name : "hr"
email : "hr@gmail.com"
password : "$2b$10$B6vqGhUF.QkEDgzKQAg9GuwnS0uYzt9obhVWEArOv/2FfbL4.Fj56"
__v : 0
```

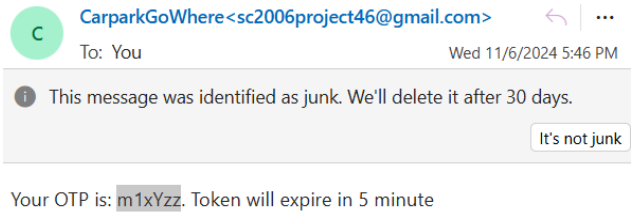

At the Register interface, users will be able to register and the required information are their Name, Email address as well as Password. After they register, their information will be stored in MongoDB. For security, their password will be stored as a hashed value instead of clear text.

Forgotten Password Interface



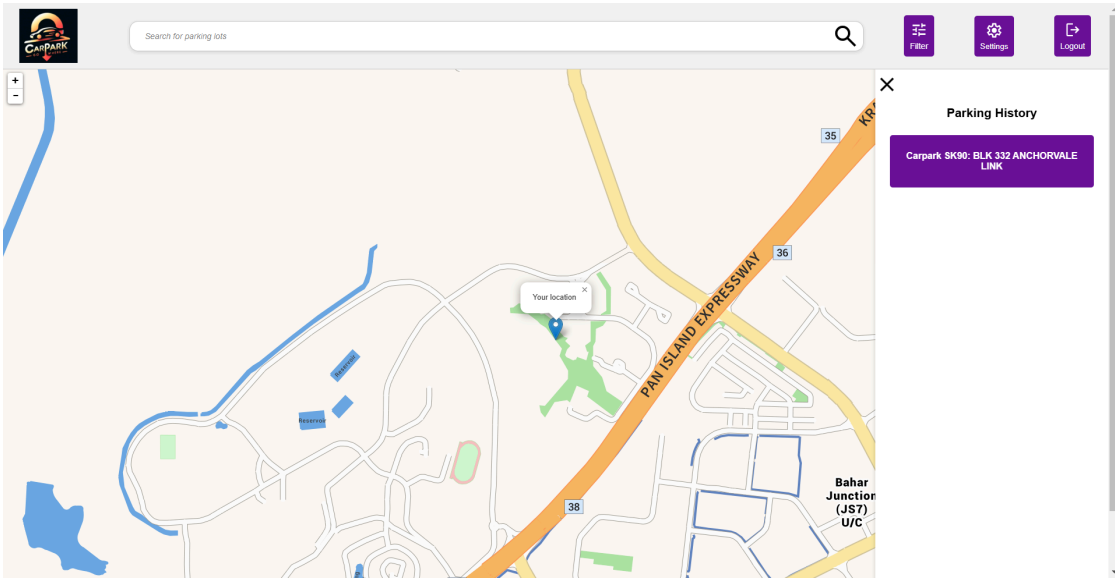


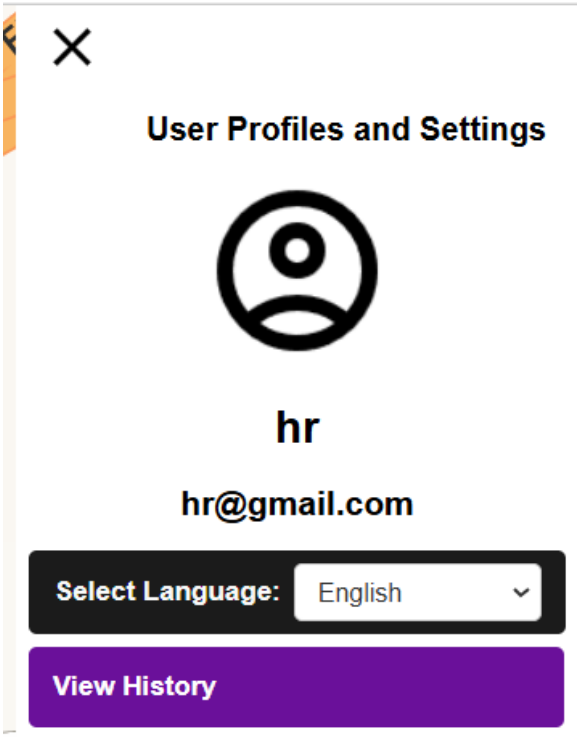




The forgotten password interface allows user to reset their password. By keying their email address as well as the one time password that would be sent to their email, they will be able to create a new password to regain access to their account.

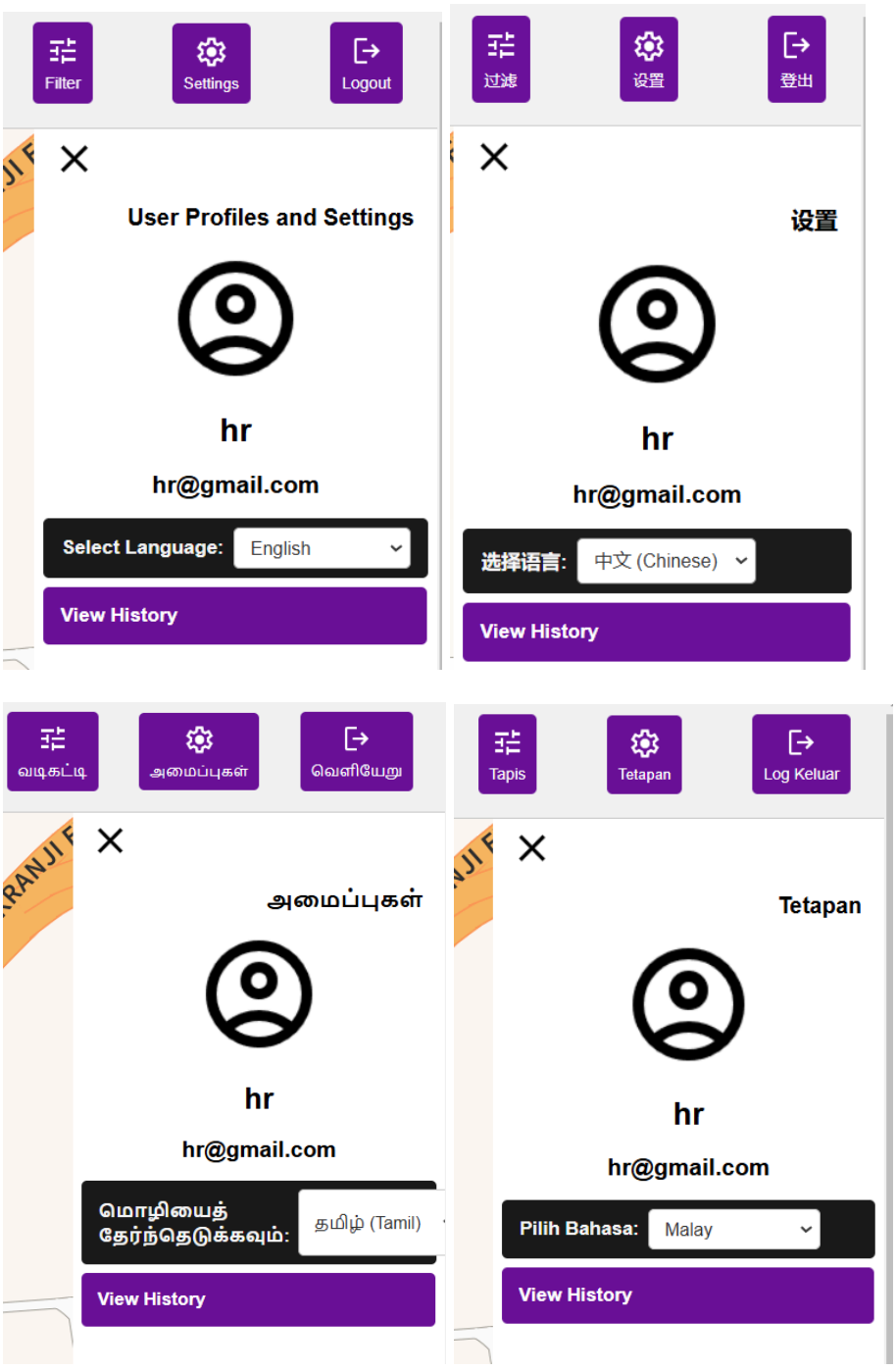
Registered User Interface





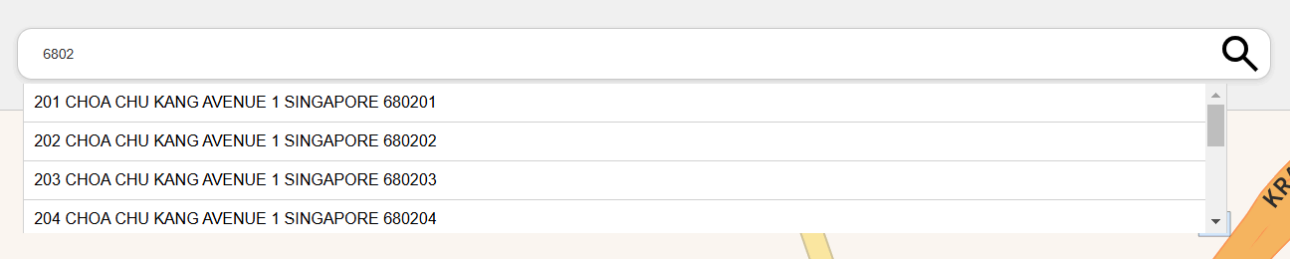
In the registered user interface, the user will be able to view his parking history based on the car parks that he had selected before. The user is also able to see his profile under the setting options. The profile will display the user's name as well as email address**.**

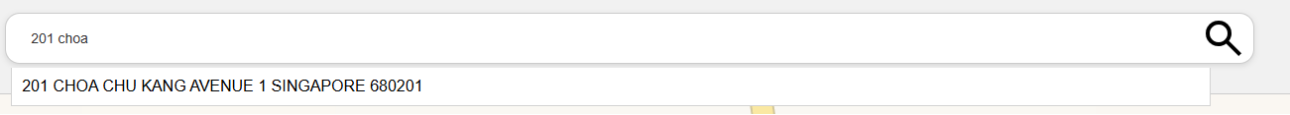
Setting interface



Under the setting interface, users are able to change the language of the application according to the 4 main languages of Singapore, namely English, Chinese, Malay and Tamil.

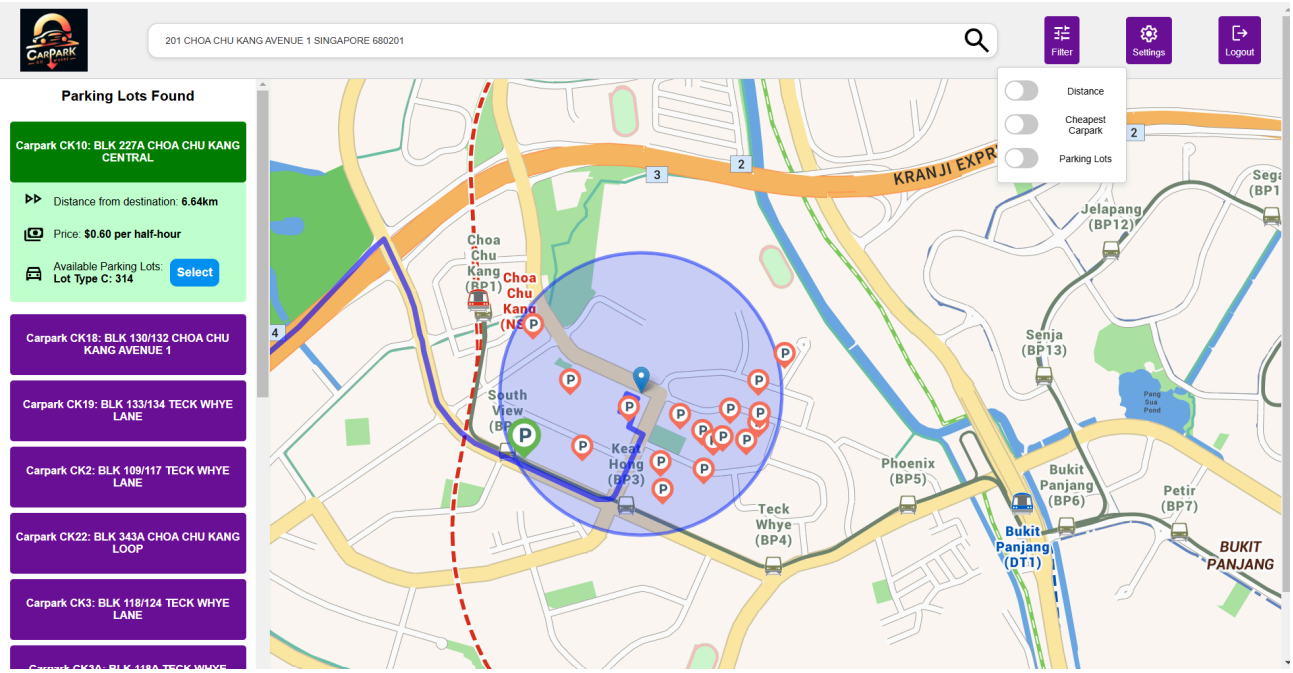
Searching Interface





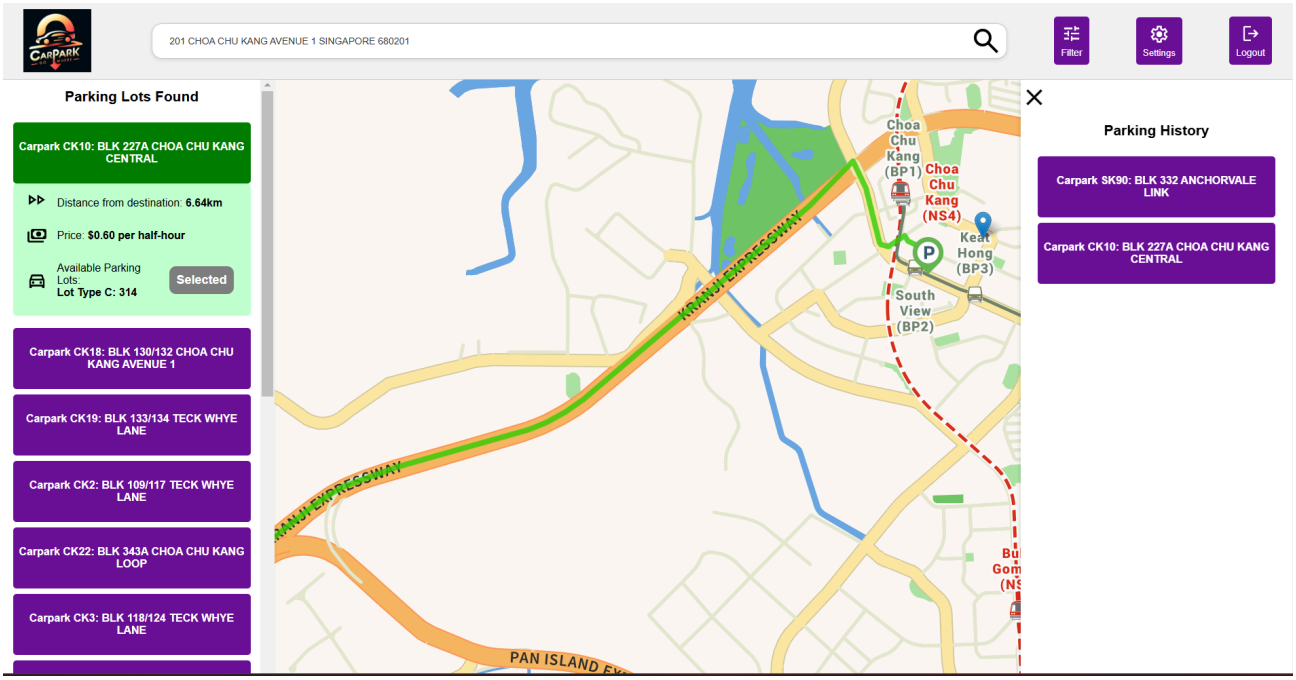
For both registered and non-registered users, they are both able to search for the location and put typing either the postal code or the full address, a list of suggested items will show up.

Carpark Interface



By searching for a location, the different car parks within 500 metres radius of the searched location will be displayed along with the route to the searched location from the current location. Users are also able to filter based on distance (From user current location to the different car parks), Cheapest car parks based on the parking rates as well as the number of available parking lots.

Routing interface



By clicking on the Select button on the left bar, the application will automatically display the route to the carpark that the user selected and the carpark will be added to the parking history of the user.

3.2 Hardware Interfaces

CarparkGoWhere is a web application that can be used on any device that can run either google chrome/Firefox/Microsoft Edge or Safari

3.3 Software Interfaces

CarparkGoWhere requires the user to install any of the web Browser stated below:

- Google Chrome
- Firefox
- Microsoft Edge
- Safari

3.4 Communications Interfaces

HTTPS communication protocol is used for requesting the API

4. System Features

4.1 Car Park Search & Routing

4.1.1 Description and Priority

This allows users to input their current/starting location and destination to search for available nearby car parks. The system will then display the route from the starting location to the car park. This feature is of High priority.

Components	Rating (out of 10)	Description
Benefit	10	Core functionality allows users to search and visualise the route of their journey.
Penalty	10	Without this feature, users would not be able to find suitable car parks at their destination.
Cost	7	It requires integration with OneMap and HDB car park APIs to update results dynamically.
Risk	8	High technical complexity due to the need for integration with current and destination locations and live car park data.

4.1.2 Stimulus/Response Sequences

Stimulus 1:

User: Input their current and destination locations (address or postal code) manually or automatic detection of users' GPS coordinates

System: Retrieves the route from the current location/ starting location to the suitable car park and displays it as a highlighted route on the map.

4.1.3 Functional Requirements

REQ-1: Users must be able to query the car park system. The query can either be of address line or postal code.

4.2 Car Park Map Display & Sidebar Sorting

4.2.1 Description and Priority

This will display the available car parks on the map and mark available car parks with a marker. This will also sort and rank the car parks based on set users' filter parameters in the sidebar. The first car park shown at the top of the sidebar will be based on nearest distance if no filter is selected. This feature is of High priority.

Components	Rating (out of 10)	Description
Benefit	10	Helps users easily find the most suitable car park at a centralised sidebar.
Penalty	10	Without this feature, users would not be able to visualise where the car parks are.
Cost	6	Moderate complexity, mainly requiring sorting algorithms and integration with the filtering mechanism.
Risk	3	Low risk since the sorting logic is straightforward, though usability must be carefully considered.

4.2.2 Stimulus/Response Sequences

Stimulus 1:

User: Car park search & Routing (Stimulus 1)

System: Displays an interactive map that marks each available car park with a marker on map. The sidebar will display all detailed information of the car park. The sidebar also displays only car parks that are within a radius of destination and is sorted by default settings, in ascending order.

Stimulus 2:

User: Clicks on a specific car park entry in the sidebar

System: Highlights the marker on map by changing to a different colour and the respective sidebar dropdown of the car park will be expanded to display the detailed information.

4.2.3 Functional Requirements

REQ-1: Maps displayed must be interactive.

REQ-1.1: The user must be able to zoom in on the map

REQ-1.2: The user must be able to zoom out on the map

REQ-1.3: User must be able to drag the map in order to move in all directions

REQ-2: Available car parks populate the side bar and show as markers on the interactive map.

REQ-3: Selecting a car park in the sidebar will highlight the marker on the map and expand the respective car park dropdown to show hourly rate, driving distance to the car park.

REQ-3.1: Selecting a marker will highlight the marker and expand the car park details in the drop-down at the sidebar.

REQ-3.2: Expanding the car park drop-down in the sidebar will highlight the respective car park marker.

4.3 Car Park Filtering

4.3.1 Description and Priority

This allows users to filter car park search results. The filtering ensures that users can quickly narrow down to the most suitable car park. This is a Medium-priority feature.

Components	Rating (out of 10)	Description
Benefit	8	Ensures users find suitable car parks efficiently.
Penalty	6	Without this feature the system can still function, but may result in users manually sifting through too many options, increasing the time to find the most suitable car park.
Cost	4	Low development complexity.
Risk	4	Low risk, as the feature involves straightforward input filtering.

4.3.2 Stimulus/Response Sequences

Stimulus 1:

User: Car park Search and Routing (Stimulus 1)

System: Presents a list of car parks based on the default nearest distance

Stimulus 2:

User: Select and set filter options; hourly rate, driving distance to car park or number of available lots

System: Updates the car park results list in real time based on filter preference

Stimulus 3:

User: Modifies one filter

System: Dynamically refreshes the available car parks to match the new selected filter.

4.3.3 Functional Requirements

REQ-1: Given the car parks at the current level of detail, only those that fulfil the filter parameters will be displayed on UI.

REQ-2: Filter parameters must be able to be toggled.

REQ-2.1: The hourly rate must be accurate to double the nearest $0.10.

REQ-2.2: The driving distance to the car park must be accurate to the nearest 0.1km.

4.4 Account System

4.4.1 Description and Priority

This feature allows users to create personal accounts to save preferences, and store recent parking and search history. It is of Low priority, providing added convenience for frequent users.

Components	Rating (out of 10)	Description
Benefit	7	Enhances user experience by personalising the system and reducing the need to re-enter information.
Penalty	3	Without this feature, users may have to input preferences and locations repeatedly, leading to frustration.
Cost	5	Standard complexity, requiring user authentication and basic account management functions.
Risk	4	Low risk, but protection of sensitive data such as passwords must be handled securely.

4.4.2 Stimulus/Response Sequences

Stimulus 1:

User: Creates a new account.

System: collects user details and creates a profile.

Stimulus 2:

User: Log in to their account

System: retrieves recent parking history.

Stimulus 3:

User: Guest Account

System: Does not load parking history

4.4.3 Functional Requirements

REQ-1: The system must allow users to create new accounts.

REQ-1.1: Users must input a valid email address.

REQ-1.2: The user must input a string password from 8 to 20 characters.

REQ-1.3: Passwords must include one lower case, one upper case, one number and one special character.

REQ-2: The system must allow users to log in to their accounts.

REQ-3: The system must allow users to use the application with a guest account.

REQ-4: The system must not allow identical usernames

REQ-5: The system must be able to validate corresponding username and password.

5. Nonfunctional Requirements

Non-functional requirements describe the properties the system must have but are not directly related to the functional behaviour of the system. It serves as constraints or restrictions on the design of the system. These may include useability, reliability, performance and supportability of the system.

5.1 Performance Requirements

5.1.1 The product should provide search results within a certain timeframe. The search for the nearest/cheapest/fastest car park should take less than 10 seconds and less than 15 seconds for detailed information such as the price of the car park, estimated time of arrival to the carpark as well as the availability of the carpark

5.1.2 The page load time should not exceed 2 seconds under normal load conditions and should not exceed 3 seconds under high load conditions.

5.1.3 The system should also have an uptime of 99.9% or higher, ensuring that it is available most of the time. The system should handle unexpected errors gracefully and provide reliable access to parking information.

5.1.4 After a system reboot, full system functionality should be achieved within 10 seconds.

5.2 Safety Requirements

5.2.1 The user interface of the system should require minimal actions from the user. E.g. the map should have the location of the user centred at all times so that the user does not need to adjust the map manually.

5.3 Security Requirements

5.3.1 The system must protect user data such as names, contact details and location data through various means such as encryption and comply with data protection regulations such as the Personal Data Protection Act (PDPA). The application should only collect data that is necessary for the functionality of the service to ensure that data retention policies are adhered to.

5.3.2 The system should be regularly updated to address security vulnerabilities and improve safety features.

5.3.3 The system should obtain relevant safety certification such as the ISO/IEC 27001 for information security management as well as comply with industry standards for app development and data security, such as Open Web Application Security Project (OWASP) guidelines

5.4 Software Quality Attributes

5.4.1 The user interface should have an intuitive and user-friendly interface. 80% of first-time users should be able to use the application effectively on their first try.

5.4.2 The system must be able to display information using four of Singapore's most common languages: English, Malay, Tamil and Chinese.

a. The default language of the system is English

b. Users must be able to change the system language on the settings page.

5.5 Business Rules

5.5.1 Users

Normal users should be able to search and view car park information

5.5.2 System Administrators

System administrators configure system settings, manage API integration and oversee the general system performance.

6. Other Requirements

Appendix A: Glossary

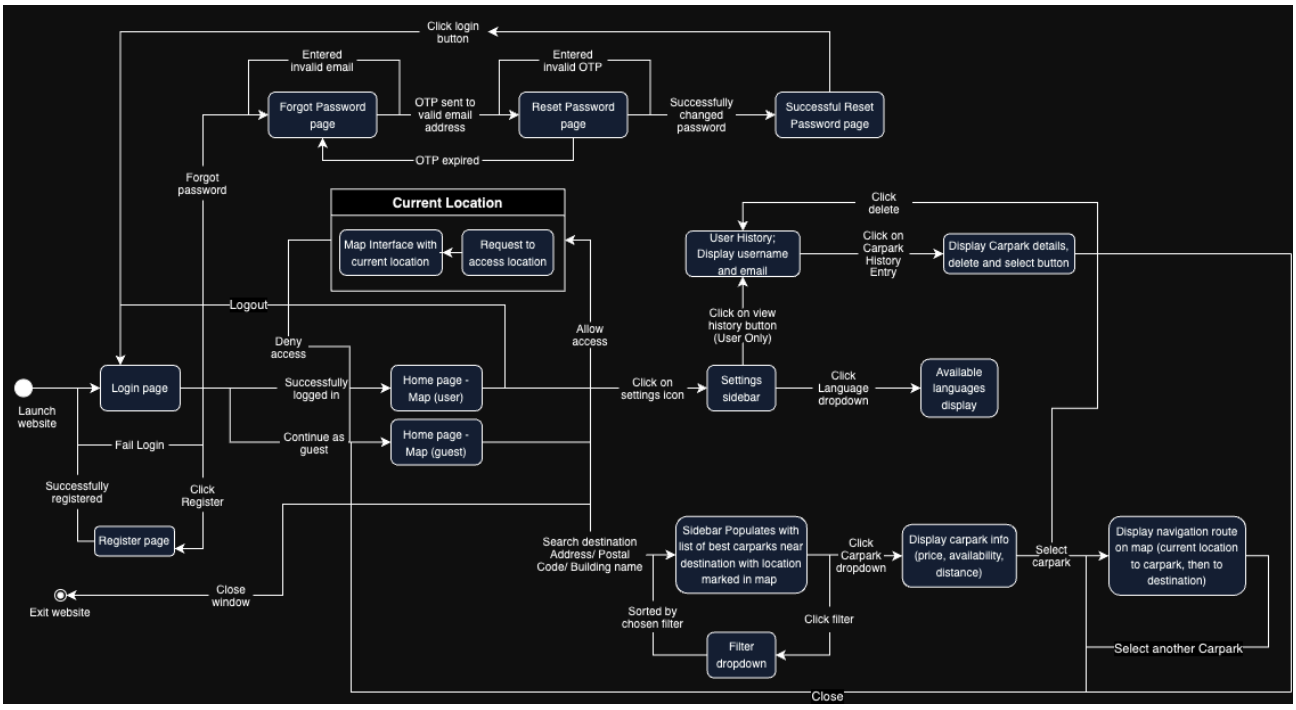
<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organisation, and just include terms specific to a single project in each SRS.>

Term	Definition
System	This refers to the web application.
Aggregated Parking Information	Collected information on car park prices, availability of parking lots, location of the car park that is available for users to read through.
Identifier	An easily seen marker/point for users.
UI	User Interface
GPS	Global Positioning System
HDB	Housing & Development Board
Location services	Services which provide GPS coordinates such as current coordinates and destination coordinates.

Query	Sending a data packet requesting information from various databases.
Filter	The process of selectively extracting information that meets predefined criteria
API	Application Programming Interface is the software middle man that allows two programs to talk to each other.
Algorithm	A process or set of rules to be followed in calculations or other problem-solving operations, especially by a computer.

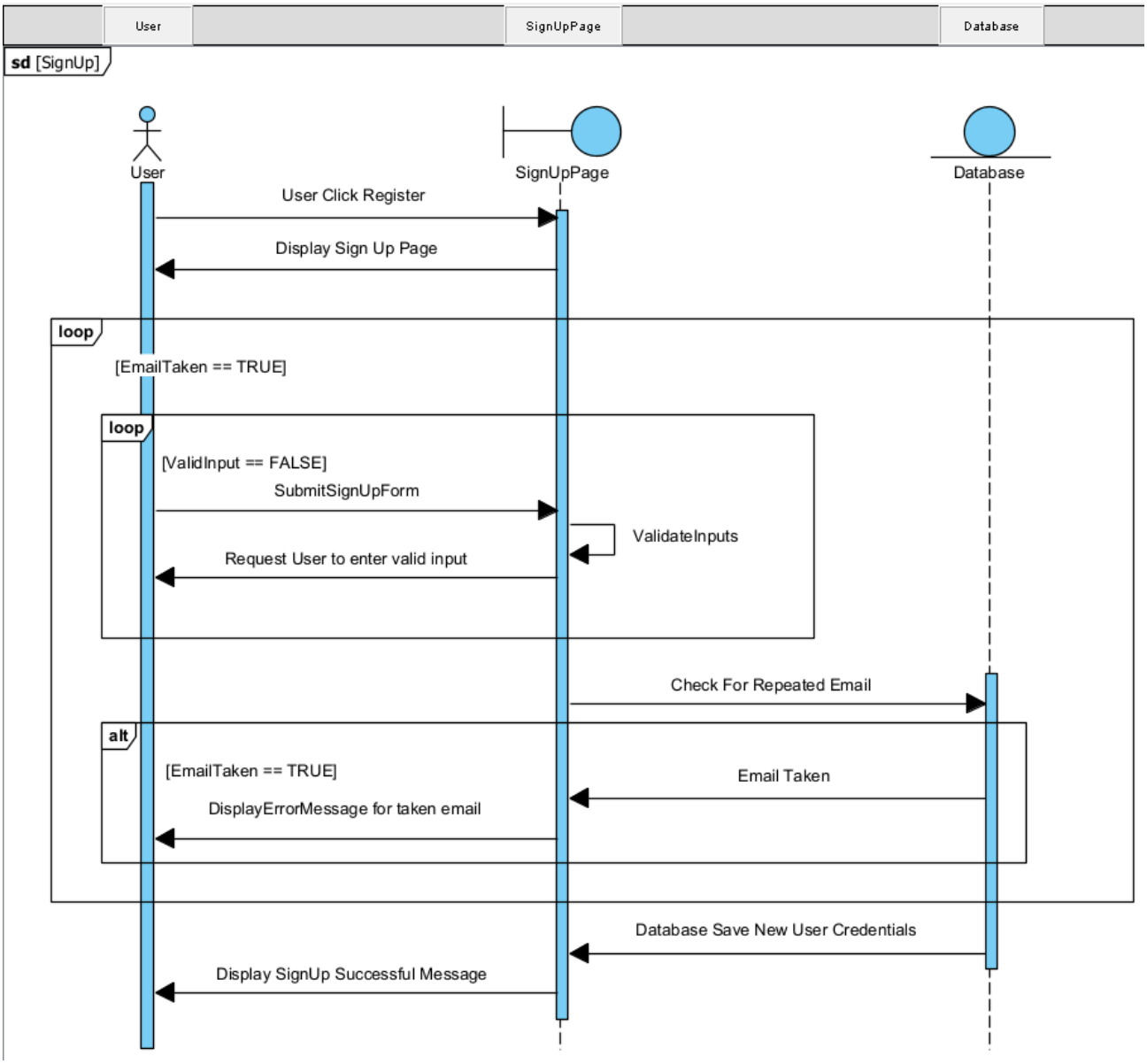
Appendix B: Analysis Models

Dialog Map

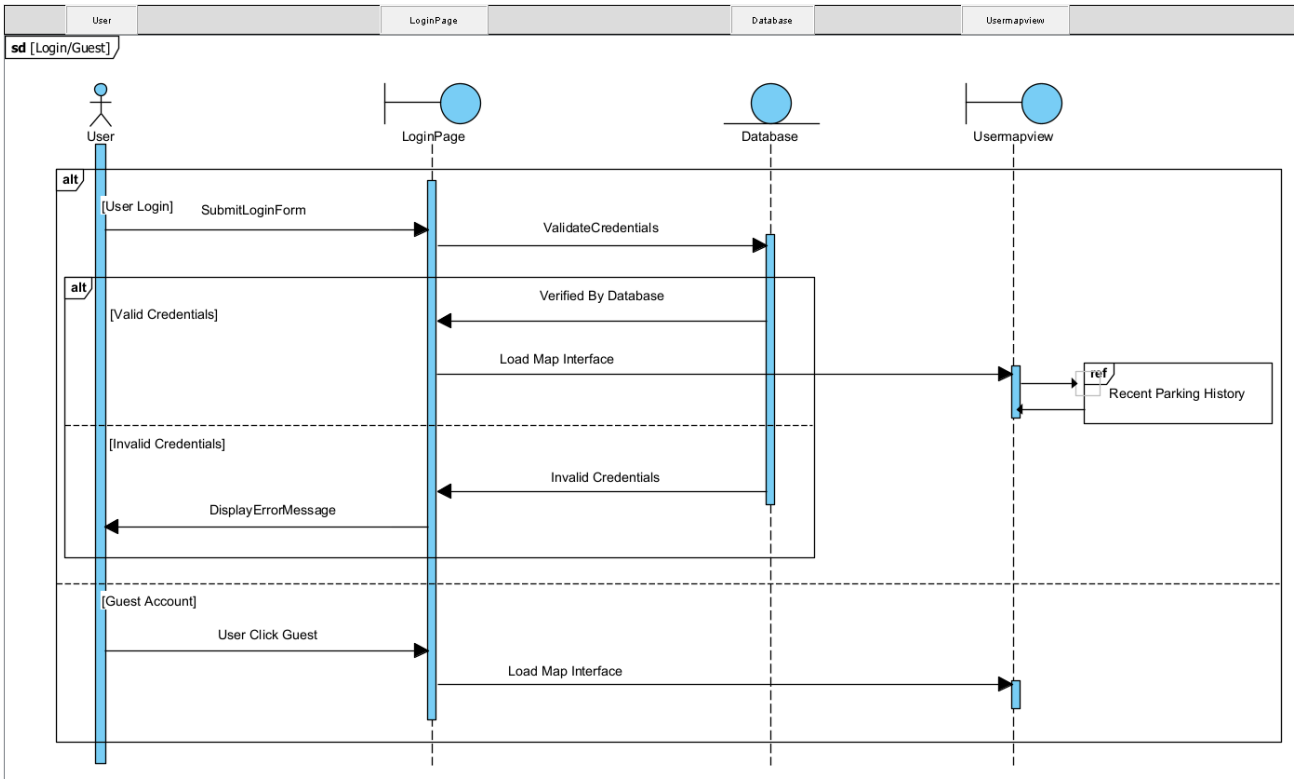


Sequence Diagrams

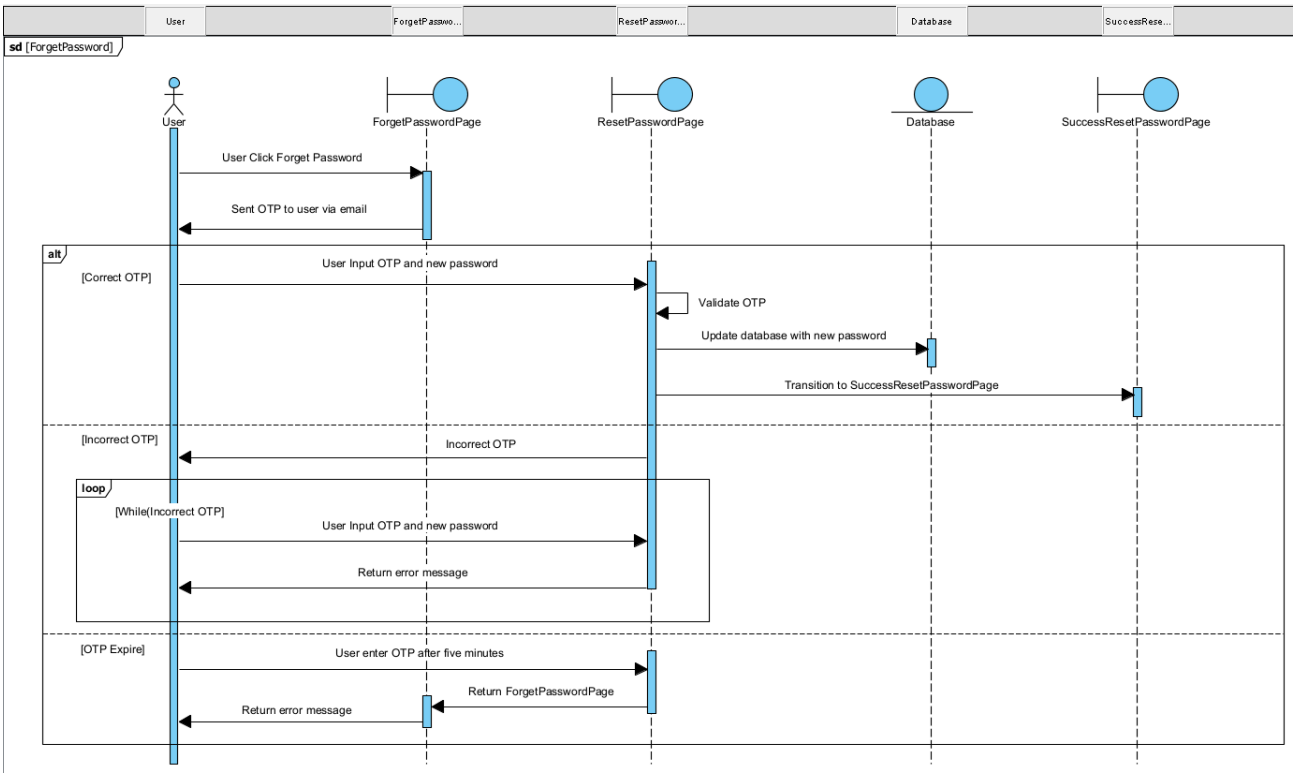
SignUp



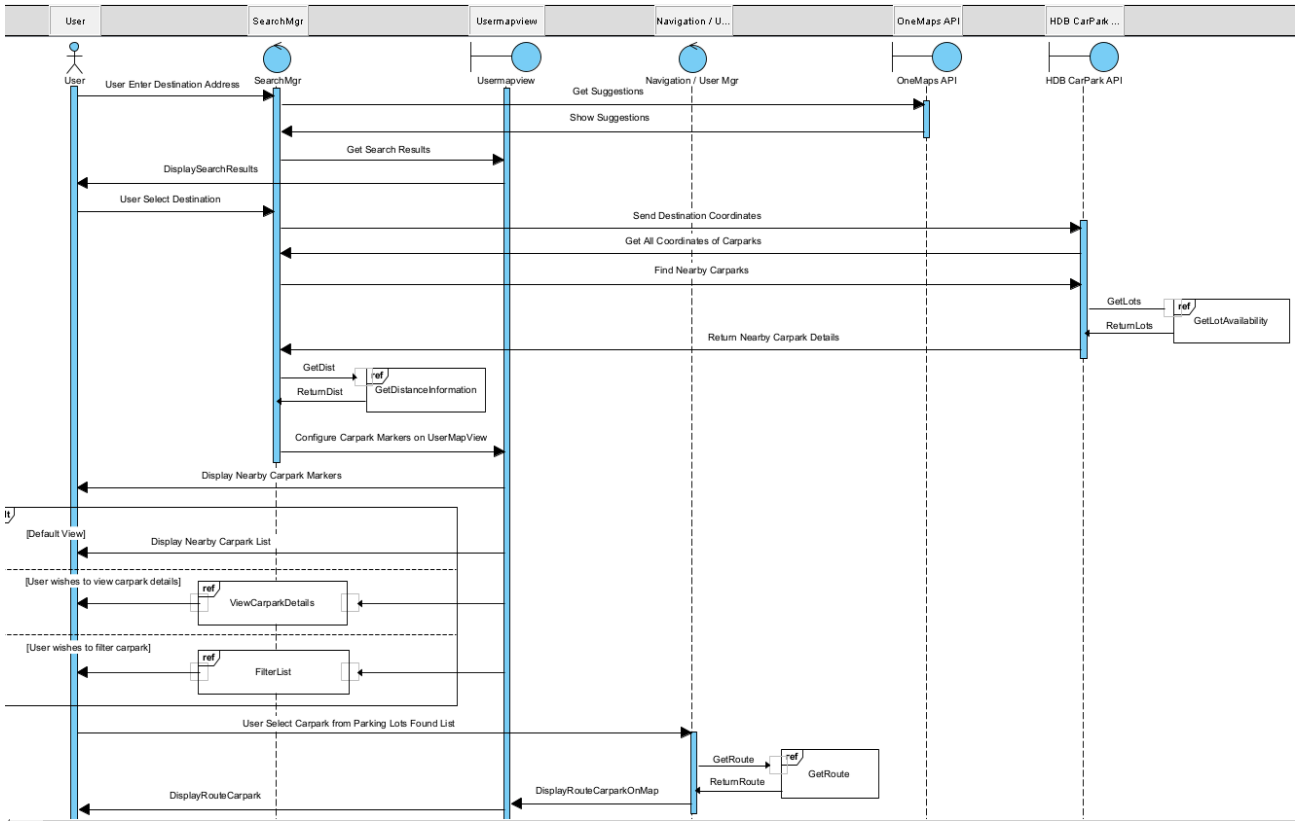
Login/Guest


User
LoginPage
Database
Usermapview
sd [Login/Guest]
User
LoginPage
Database
Usermapview
alt
[User Login]
SubmitLoginForm
ValidateCredentials
alt
Verified By Database
[Valid Credentials]
Load Map Interface
ref
Recent Parking History
[Invalid Credentials]
Invalid Credentials
DisplayErrorMessage
[Guest Account]
User Click Guest
Load Map Interface

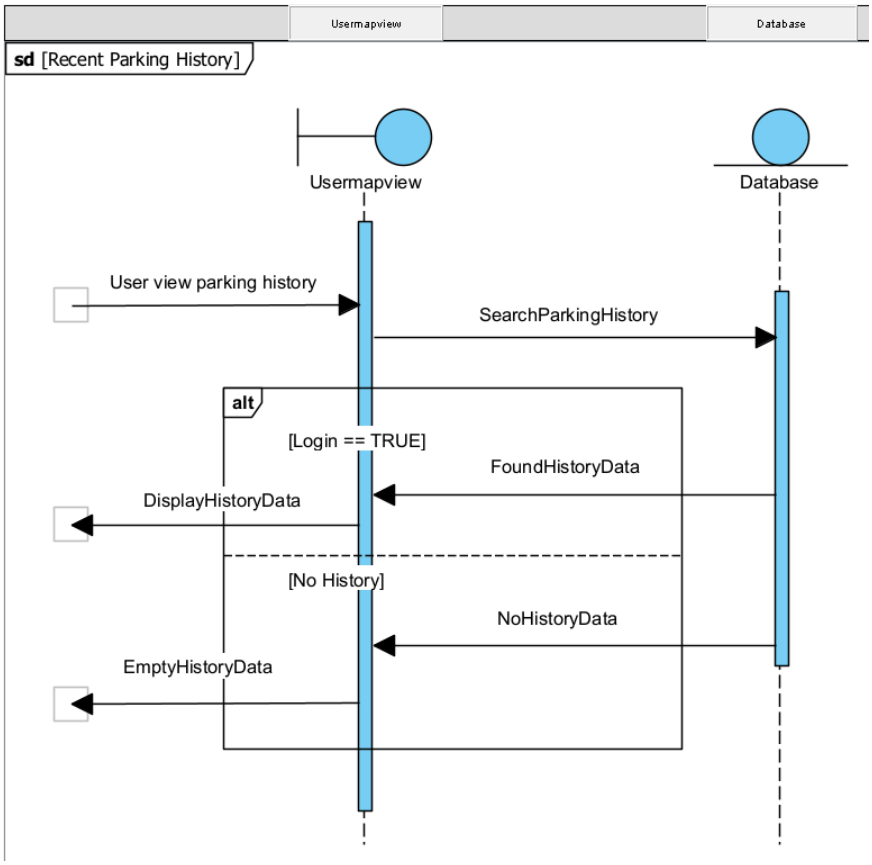
ForgetPassword



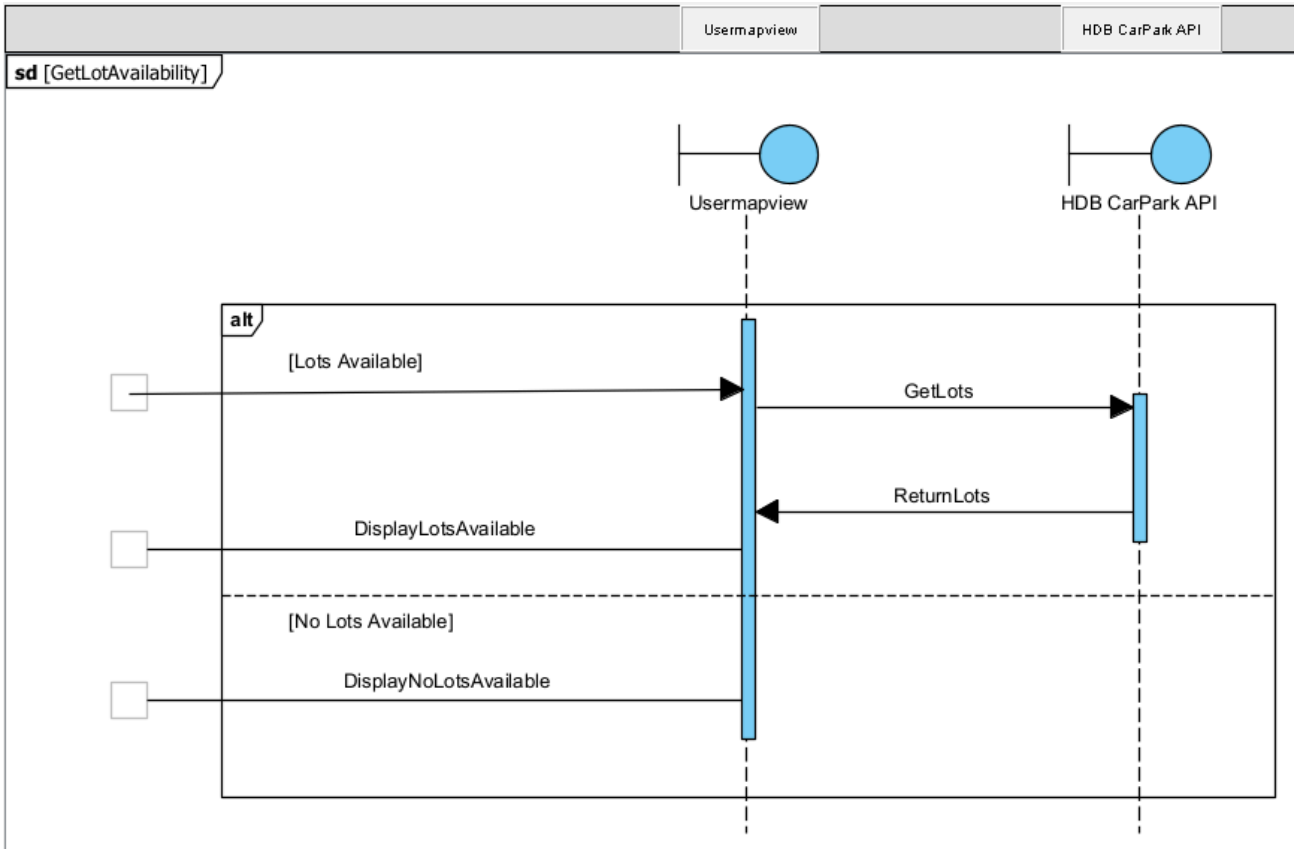
Search and Select Destination



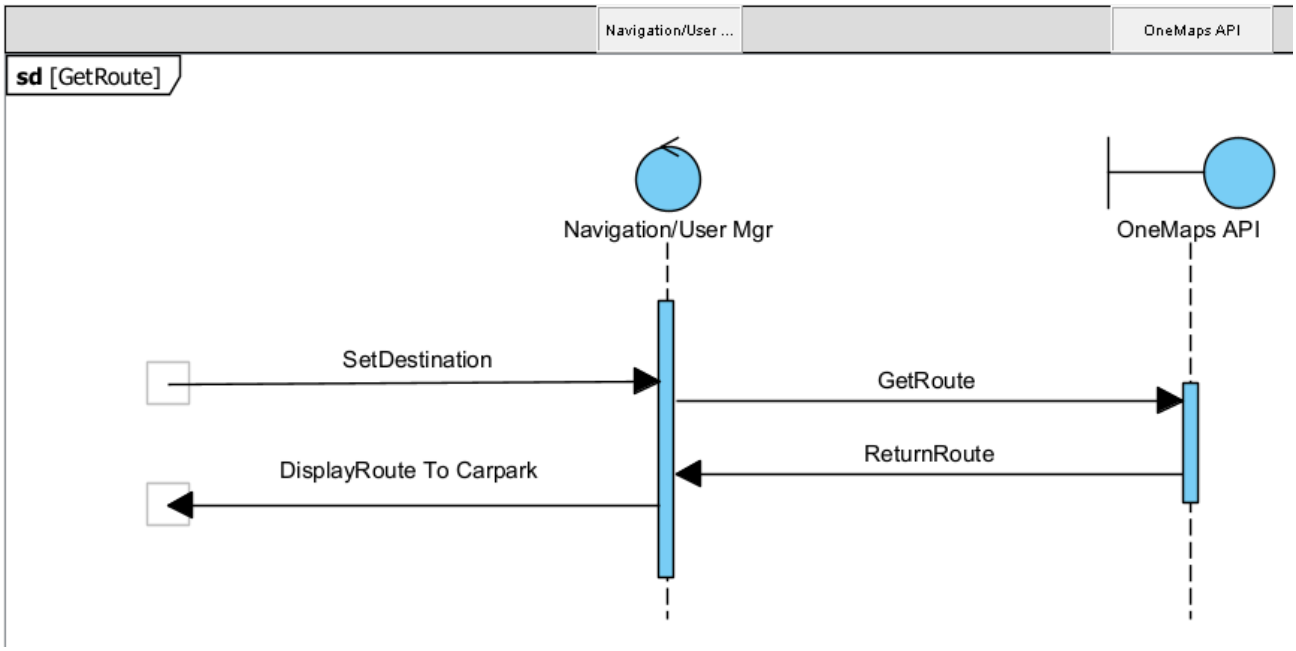
Recent Parking History



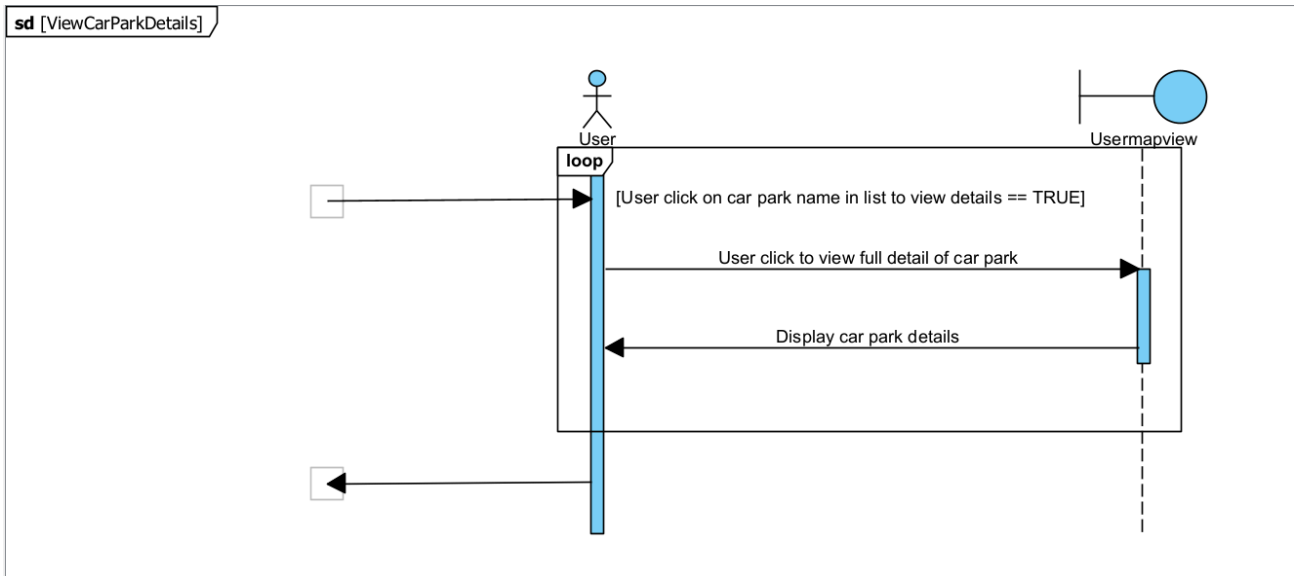
Get Lot Availability



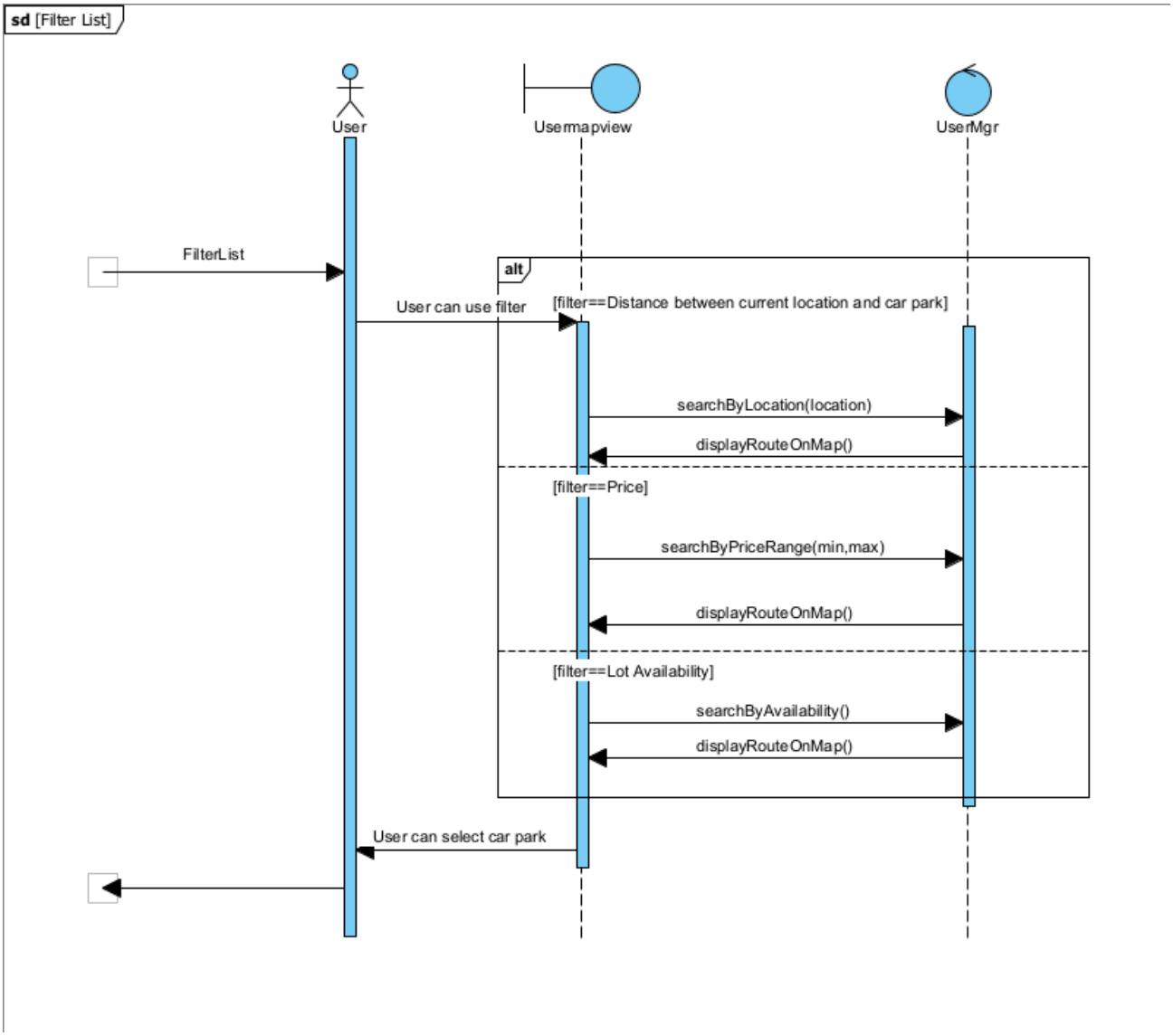
GetRoute



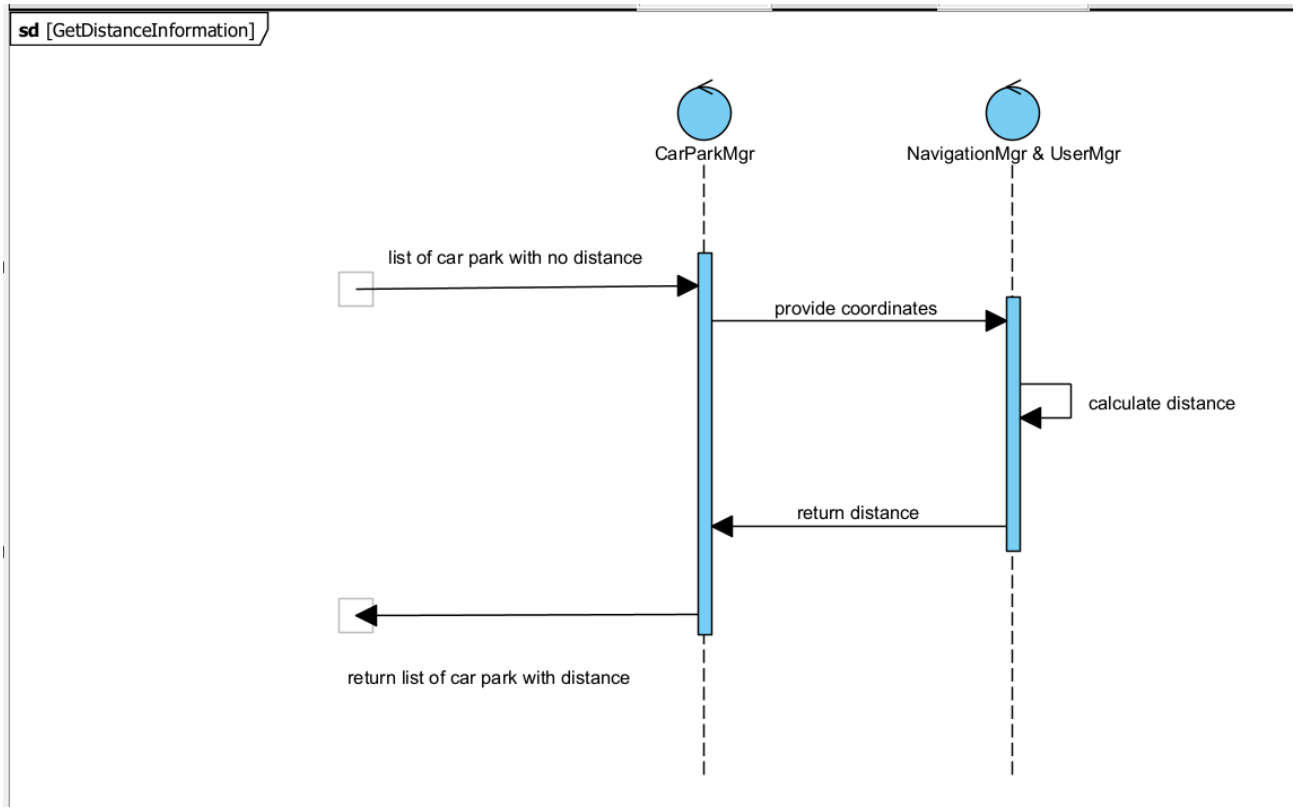
ViewCarParkDetails



FilterList


sd [Filter List]
User
Usermapview
UserMgr
FilterList
alt
User can use filter
[filter==Distance between current location and car park]
searchByLocation(location)
displayRouteOnMap()
[filter==Price]
searchByPriceRange(min,max)
displayRouteOnMap()
[filter==Lot Availability]
searchByAvailability()
displayRouteOnMap()
User can select car park

GetDistanceInformation



Source:

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