Software Requirements Specification

for

TripSmart

Version 1.0 approved

Prepared by
Lim Boon Hian (U2120791F)
Joanne Christina Salimin (U2120304J)
Koh Hao Sheng (U2122672K)
Lim Dong Wan (U2122455D)
Loke Yong Jian (U2120476G)
Ng Woon Yee (U2120622C)
Tiwana Teg Singh (U2122816B)

Team-7th

13 April 2023

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Loke Yong Jian	13/4/2023	Initial Document	1.0

1. Introduction

1.1 Purpose

This document outlines the software requirements and specifications for TripSmart, a mobile application designed to make travel more accessible and stress-free for passengers and commuters in Singapore. Available on both Android and iOS mobile platforms, TripSmart serves as a comprehensive aggregator platform that allows users to compare and select the most affordable and effective modes of transportation, based on their needs and preferences. This document will detail the purpose and features of the system, its interfaces, and the constraints and external stimuli it must consider. It is intended for both the stakeholders and the developers of the system.

1.2 Document Conventions

Text formats
Font: Times New Roman
Font size: 12 for Body and Layer 2 Sub-Headings, 14 for Layer 1 Sub-Headings and 18 for Headings
Line height: 1.15

1.3 Intended Audience and Reading Suggestions

The document is meant for TripSmart's stakeholders, including customers and developers, such as designers, coders, testers, and maintainers. It is not necessary to read the document in sequence, as users are encouraged to navigate to the relevant sections of their choice. The following is a summary of each part of the document:

Part 1 (Introduction): This section provides an overview of the TripSmart project, outlining its goals, objectives, project scope, and general system details.

Part 2 (Overall Description): This section describes the TripSmart system in detail, class by class, including interface details, class hierarchies, performance/design constraints, and process details.

Part 3 (External Interface Requirements): This section covers the structure of the Graphical User Interface (GUI), including preliminary mockups of the TripSmart mobile application. Readers can view this section to get an idea of what the final product will look like.

Part 4 (System Features): This section describes in depth the functional requirements of the functions of TripSmart. It includes how the user should interact with the application and how the application should respond accordingly for each said interjection.

Part 5 (Other Non-Functional Requirements): This section encompasses the various aspects of non-functional requirements of TripSmart, such as the performance and security requirements, that support the usability of TripSmart for customers and/or developers.

Part 6 (Other Requirements): This section includes a data dictionary, pertinent analysis models, a list of test cases, testing and expected output, and other relevant information.

Part 7 (Appendix): This section includes additional information that may be useful to readers and stakeholders.

1.4 Product Scope

The 7th team will develop an application that makes instant travel more accessible and stress-free for commuters and passengers within Singapore. The app does so by offering a comprehensive, practical and integrated aggregator platform to compare and select the most affordable and efficient mode of transportation, according to the commuters' needs and wants. Our platform will compile the estimated travel time and fare data from multiple sources and transport modes, such as public transportation, ride-hailing and taxi services, as well as private vehicle rental firms to assist commuters in making informed choices and saving money and/or time on their journeys. In order to help Singapore achieve its vision of becoming a smart nation, we seek to streamline the transportation industry and encourage smart mobility, by empowering regular commuters to openly compare the transportation options available to them.

1.5 References

[1]
MongoDB, "MongoDB API Documentation." https://api.mongodb.com/ (accessed Apr. 13, 2023).

[2]
Meta, "Core Components and APIs · React Native," Jan. 12, 2023. https://reactnative.dev/docs/components-and-apis (accessed Apr. 13, 2023).

[3]
JavaTPoint, "Software Engineering | Software Design Principles - javatpoint." https://www.javatpoint.com/software-engineering-software-design-principles (accessed Apr. 13, 2023).

2. Overall Description

2.1 Product Perspective

TripSmart is a comprehensive application designed specifically for use in Singapore. It incorporates multiple transport-related Application Programming Interfaces (APIs) that enable users to check the fares of various transportation services, including BlueSG, Grab, ComfortDelgro, and public transport, all within a single application. This feature allows users to choose the most affordable or efficient service based on their needs, thereby addressing the daily commuting challenges that Singaporeans encounter.

2.2 Product Functions

These are the six main functions of TripSmart:

- **Register:** Our application allows users to create an account that will bind all of their search history, filters and settings to the database, hence allowing a more customized browsing experience.
- **Compare transportation fares between services:** Our application allows users to compare all the transportation fares on a singular platform without having to open up the individual applications (Grab, BlueSG, etc.) to check for fares.
- **View history:** Our application allows users to save their search history after they have deep-linked to other applications, enabling them to track the transportation services they have used in the past.
- **Deep-link:** Upon searching for locations and viewing fares in our application, they can click on the results, and our application will automatically redirect them to the respective applications, saving them extra time and effort.
- **Changing of language:** Our application enables users to change the language setting, making it accessible to non-English readers and allowing them to utilize the application with ease in their own respective languages.
- **Change password:** Our application provides the option for users to change their password if they suspect that their password may have been compromised.

2.3 User Classes and Characteristics

Commuters: These are the primary users of the application and will likely use it frequently. They will use the application to compare and select the most affordable and efficient mode of transportation to get to their destinations within Singapore.

Tourists: These users may have limited knowledge of the local transportation options, making the application particularly useful to provide insight on the various transport options available in

Singapore. They may use the application to find the most affordable and efficient mode of transportation to get them to their destinations within Singapore.

2.4 Operating Environment

To use the application, which can be accessed on both iOS and Android platforms, location services are required. As the application is built on top of Expo, the mobile app compatibility depends on the Expo SDK version. As for now, the application is built using Expo SDK 48, which supports the following version:

a. For Android users, their device must have at least Android 5 (Lollipop) operating system.
b. For iOS users, their device must have at least iOS 13 operating system.

As for the development environment, the app requires Node 19+ and Python 3.9+ to run. To start the app server locally, type “yarn starts” under the front-end directory. More about this is stated in the README.md file.

2.5 Design and Implementation Constraints

In terms of information availability, TripSmart relies heavily on free APIs, which may reduce the feasibility of the application. For example, as we use Google Maps to store our user database, we may face credit limitations since Google Map API only provides access until a certain limit. We anticipate showcasing the full potential of our application once our budget is expanded.

Currently, we do not possess the necessary monetary resources or permissions from Grab to access their APIs. As an interim solution, we will utilize Flask to extract data from Grab locally. However, we intend to transition to the authorized Grab APIs once we are granted the appropriate permissions and have an allocated budget for it.

2.6 User Documentation

1. Well-explained Youtube video to guide the users.
2. User manual (Appendix A)

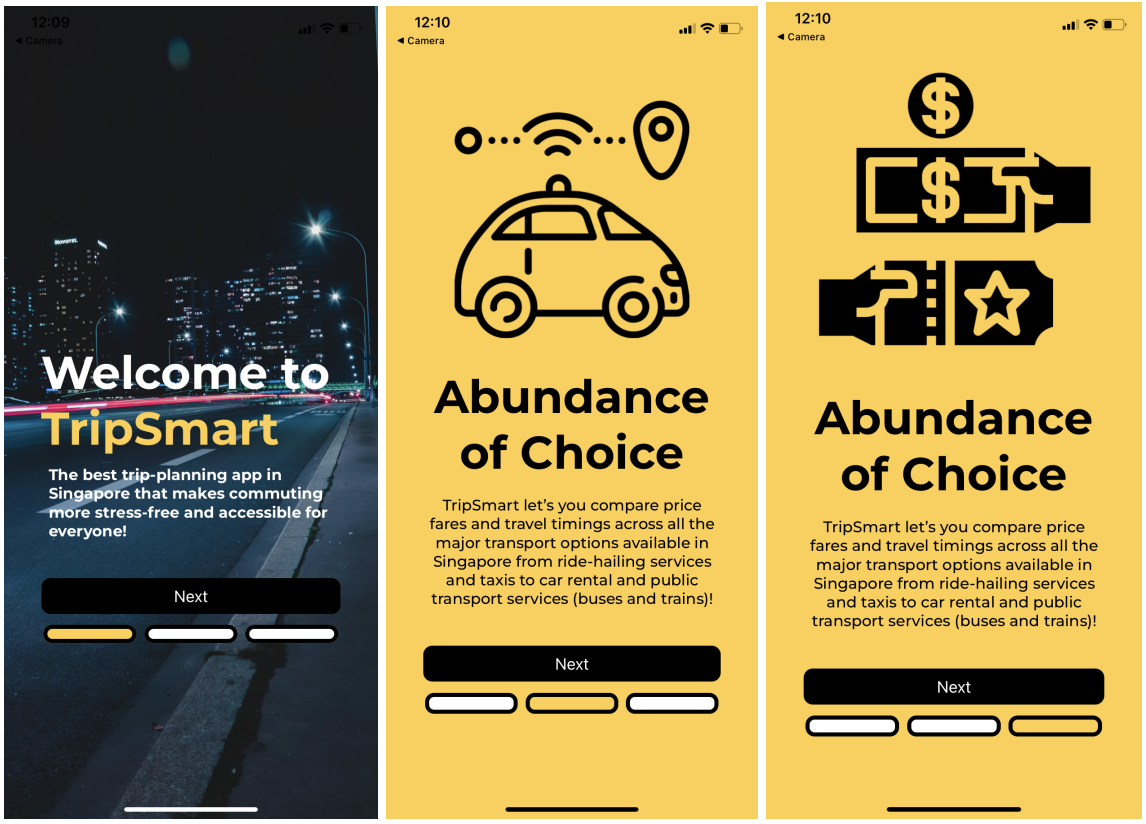
2.7 Assumptions and Dependencies

The primary premise, in this case, is that users must have access to and maintain an internet connection at all times. Without an internet connection, the application may not be able to locate all results that meet the specified requirements.

3. External Interface Requirements

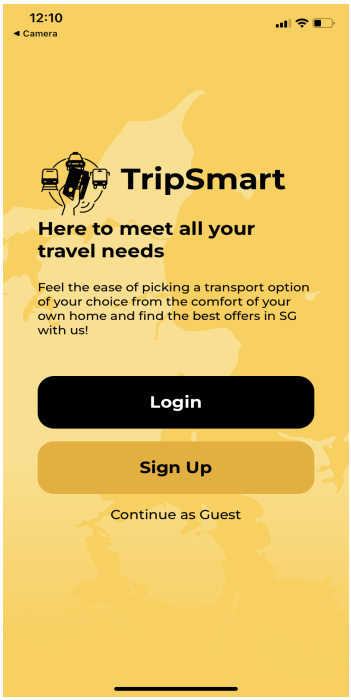
3.1 User Interfaces

3.1.1 Greeting Pages



The greeting pages consist of descriptions of our application, TripSmart. The 'Next' button at the bottom allows users to view the 3 greeting pages that our application has respectively. Once the 'Next' button on the third page has been clicked, users will be directed to the selection page.

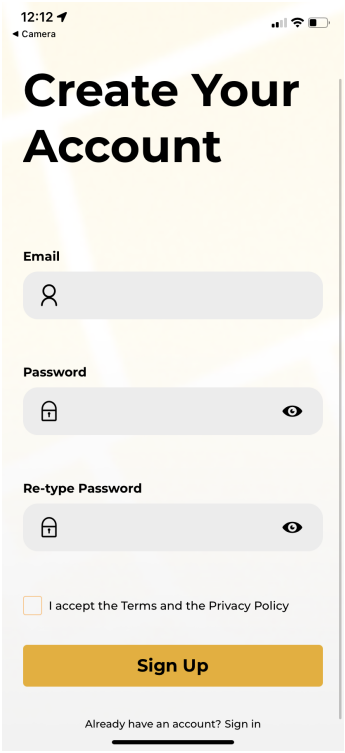
3.1.2 Selection Page



At the selection page, users have 3 options to choose from:

- **Login**: This button will redirect users to the login page where they can log in to their existing TripSmart account.
- **Sign Up**: This button will redirect users to the registration page where they can register an account with TripSmart.
- **Continue as Guest**: This button will bring users directly to the search page. Users will then be treated as guests. However, they will not be able to view their search history, as the history functionality is only applicable to registered users with an existing TripSmart account.

3.1.3 Registration Page



At the registration page, there are 2 fields that require user inputs, namely email address and password, and an additional input field for users to re-type their desired password to ensure the consistency of their password.

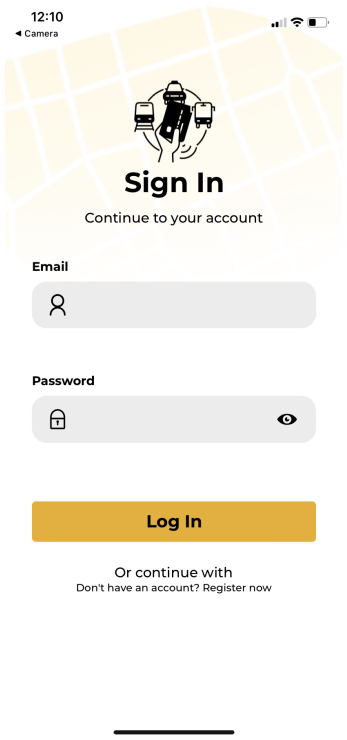
In addition to these fields, users have to tick the checkbox of accepting the Terms and Conditions of the application. There is also validation on the format of the email address and the complexity of the password.

If any of the requirements are not being met, there would be an alert to prompt users about the issue.

In the password field, there is a masked feature to allow the password to be masked when typed for enhanced security. Users can also choose to unmask them when necessary.

Below the “Sign Up” button, users can directly press the “Sign In” button to redirect them to the login page if they wish to sign in with their existing account instead.

3.1.4 Login Page



At the login page, there are 2 fields that require user inputs, namely email address and password.

In the email field, users are required to input the email address they have earlier registered at the registration page.

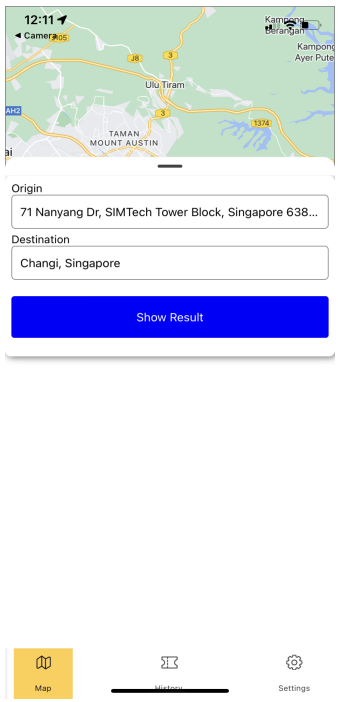
In the password field, users are required to input the password they have earlier registered at the registration page. There is a masked feature to allow the password to be masked when typed. Users can choose to unmask it where necessary.

If the input email has not been registered earlier, or the input password is incorrect, there would be an alert to prompt users about the issue.

If the input email and password are correct, users will be redirected to the search page when the “Log In” button is clicked.

If the user has not registered for an account yet, users can press the ‘Register now’ button, which will redirect users to the registration page to register.

3.1.5 Search Page



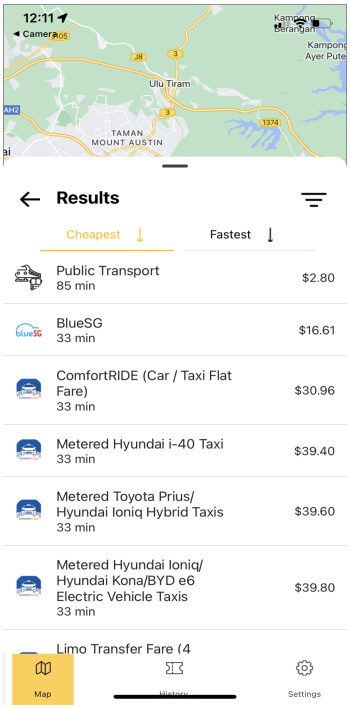
At the search page, there are 2 fields - Origin and Destination.

Upon typing in the inputs, there will be a drop-down box that suggests the locations that best match the user input. Users have to choose one location from the drop-down box. The suggestions are made using an API from Google Maps.

After pressing the “Show Result” button, TripSmart will call the various APIs to deliver the fares and relevant details and display them accordingly.

At the same time, users can lower down the search panel and view the map. There will be markers and lines to indicate the distance between the origin and destination that users have chosen.

3.1.6 Result List Page



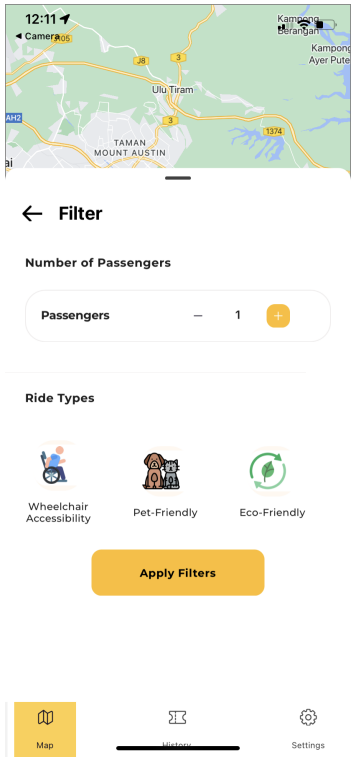
At the result list page, it will display all the results fetched from the various APIs.

By default, the results are sorted by Cheapest. Users also have a choice of sorting them by Fastest by pressing on the "Fastest" button.

At the top right panel of the result component, there is a button that will allow users to apply some filters to the results in order to suit their preferences.

After pressing on any of the results, the application will redirect them to the respective application or websites through deep-link. These actions will be saved to the database and are retrievable at the history page.

3.1.7 Result Filter Page



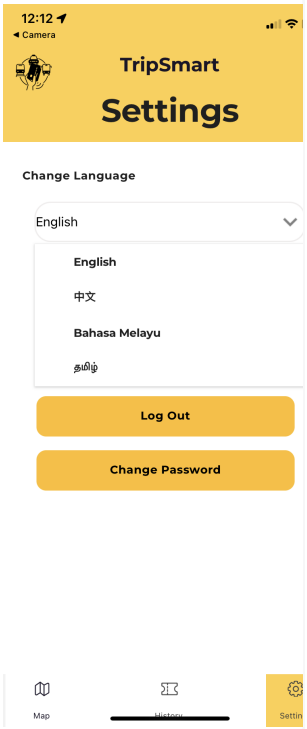
At the result filter page, users can choose to filter the result list in 2 main ways, either by the Number of Passengers or Ride Types.

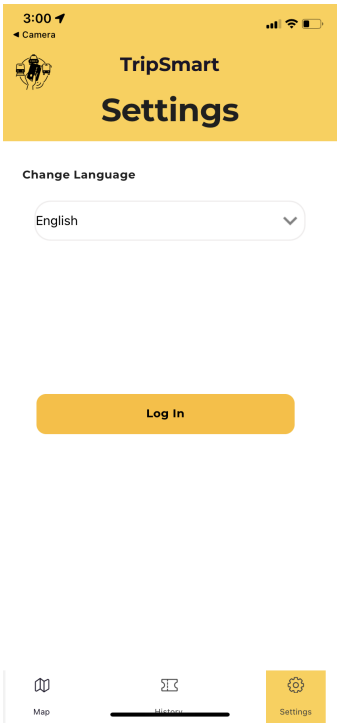
For the Number of Passengers filter, users can increase or decrease the number of passengers who will be commuting to suit their needs.

For the Ride Types filter, users can click on the 3 respective buttons - Wheelchair Accessibility, Pet-Friendly, Eco-Friendly - to change the ride type based on their needs.

When the ‘Apply Filters’ button is clicked, the filters will be applied. After that, when the left arrow button next to the ‘Filter’ heading is clicked, users will be directed to the result page and the updated fare of the respective transport options will be reflected.

3.1.8 Settings Page





There are 2 versions of the settings page - when users continue as a guest and when users log in to their account.

At the settings page (Guest), users have 2 options to choose from: change language and log in.

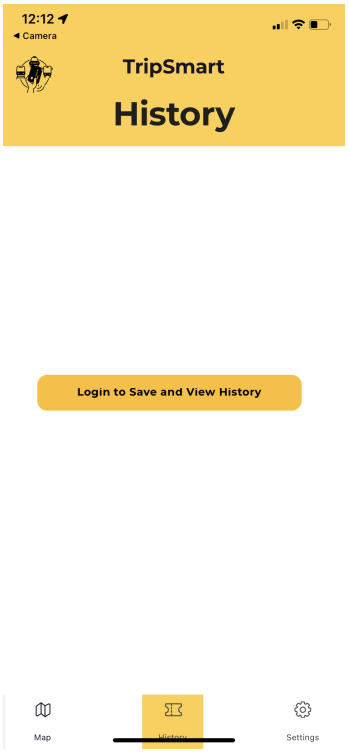
The default language displayed is English. Users can choose to change the language displayed by clicking on the drop-down button. The other languages available are Mandarin, Bahasa Melayu and Tamil.

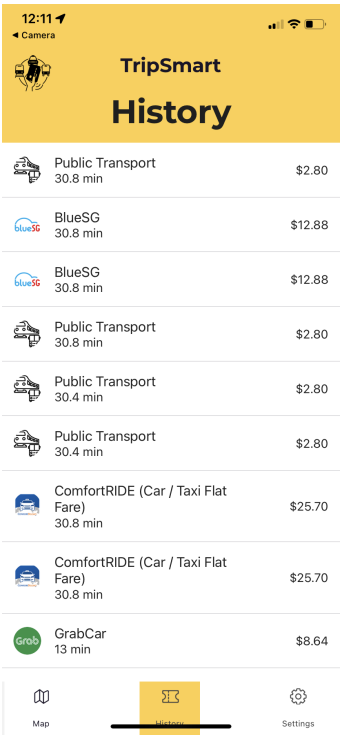
Users have the option to log in by clicking on the 'Log In' button in order to use other functionalities that only registered users have, such as the view search history functionality. When the button is clicked, users will be redirected to the login page.

At the settings page (for registered users), users have an additional option to choose from, which is to change password, on top of the change language and log out options.

When the 'Change Password' button is clicked, the user will be redirected to the change password page.

3.1.9 History Page



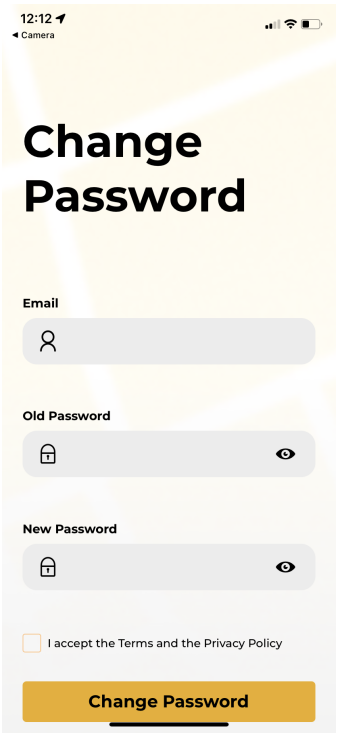


There are 2 versions of the history page - when users continue as a guest and when users log in to their account.

At the history page (Guest), users are not able to view any history and only have the option to log in to save and view history. When the button is clicked, users will be redirected to the login page.

At the history page (for registered users), users can view the history of what was previously selected as the preferred mode of transport option.

3.1.10 Change Password Page



Users will be able to change their account password at this page.

There are some validations to the input fields:

- The email address must have been registered before.
- The old password needs to be the current password associated with the registered email address.
- The new password cannot be the same password as the current password.

After the "Change Password" button is pressed, the change to the password should be immediate.

3.2 Hardware Interfaces

Our application can be accessed only through mobile platforms, on either iOS or Android platforms. The compatibility of the app depends on the Expo SDK version, and currently, the operating system must be at least Android 5 for Android, or at least iOS 13 for iOS. Wi-Fi is required for all functionalities to be accessed and enabled within the application.

3.3 Software Interfaces

3.3.1 APIs Used

Multiple APIs are used in our application.

- Taxi API (ComfortDelgro API): This API is used to check the availability of Taxi (ComfortDelgro) around the user.
- BlueSG API: This API is used to check the availability of BlueSG vehicles around the user.
- Grab API: This API is used to check the availability of Grab around the user, as well as the estimated fare and the estimated duration of the trip using Grab.

- Google Maps API: This API is used to obtain the estimated distance, time of travel between two points. Google Maps API is also utilized to obtain formatted addresses by passing the latitude and longitude values.

3.3.2 Database Used

We made use of MongoDB, a NoSQL database in our application.

In TripSmart, we utilize MongoDB as our database management system to store all the user account information, such as email and password. Our application communicates with the database using MongoDB REST API. When a new user registers for an account, we verify whether the account already exists in the database. If the account does not exist, we allow the user to create a new account. Moreover, when a user selects any travel option, such as JustGrab, the application records this data as part of the user's history, which is then uploaded to the MongoDB database. Later, when the user navigates to the history page, the application retrieves this data from the database and renders it for the user.

3.3.3 Frameworks and Libraries Used

- Node environment (Featured)
  Framework:
  - Expo
  - React Native

  Libraries:
  - react-native-google-places-autocomplete
  - fetch

- Python environment (Featured)
  Framework:
  - Flask

  Libraries:
  - Selenium

Our code is written in multiple languages, but mainly in TypeScript, JavaScript and Python. As we utilize various libraries in our application, we will only highlight the main framework and libraries used in our project.

Expo is a free and open-source framework built on top of Node.js that simplifies the development of mobile applications for iOS and Android platforms. It provides a set of pre-built tools and services that make it easier for developers to build and deploy mobile apps. Expo integrates seamlessly with React Native, enabling developers to build native mobile apps on Mac

or Windows easily. It also allows developers to test the applications easily on their devices on the Expo app without the need for manual deployment.

React Native is a cross-platform mobile app development framework that uses Node.js. It allows developers to build native mobile apps for both iOS and Android platforms using JavaScript and offers many pre-built UI components, enabling faster and easier app development with a consistent user interface.

React-native-google-places-autocomplete is a library for React Native that provides an easy-to-use interface for searching and selecting places using the Google Places API, which is used in the search page to autofill the places inputted by the user.

Fetch is a web API used to request resources from servers. It provides a flexible and powerful way to retrieve data asynchronously in JavaScript. It is used to fetch the data from various APIs.

Flask is a Python web framework that is lightweight and commonly used to create web applications. In our case, we employ Flask to initiate the local server on the localhost:5000, which in turn enables us to extract data from Grab.

Selenium is a suite of open-source tools for automating web browsers. It enables the creation of automated tests for web applications, and the ability to interact with web elements, simulate user actions and behaviors, and perform web scraping and data extraction. We use Selenium to scrape the data from the Grab fare checker website to obtain the estimated fare and estimated travel time for Grab.

3.4 Communications Interfaces

To communicate with MongoDB REST API, the TripSmart application will need to use the fetch function in JavaScript to send HTTP requests and receive responses. The messages will be formatted using the REST protocol, which specifies the use of standard HTTP verbs such as GET, POST, PUT, and DELETE. For example, to retrieve a user's account information, the application might send a GET request to the API with the user's ID as a parameter in the URL.

To communicate with Google Maps APIs, the application will also use the fetch function in JavaScript to send HTTP requests and receive responses. The messages will be formatted using the REST protocol and will include an API key for authentication. The Google Maps APIs support a range of endpoints for geocoding, directions, and distance matrix, and the application will need to send appropriate requests based on the user's input.

To communicate with BlueSG and ComfortDelgro APIs, which are GraphQL APIs, the application will use the fetch function in JavaScript to send GraphQL queries and receive responses. The messages will be formatted using the GraphQL protocol, which defines a query language for APIs and allows the application to specify exactly the data it needs. For example, to

obtain information about nearby BlueSG services, the application sends a GraphQL query that includes the user's location as input.

The application can use Promises for async operations to ensure synchronization between different sources. When the application sends requests to multiple APIs, we send the useState hook to the API controller, such that they return the responses whenever the data is ready. After that, the data will be sorted accordingly afterwards. For example, if the application wants the travel option data from BlueSG, Grab and ComfortDelgro, it can send requests and the useState hook to the corresponding controllers. Thereafter, whenever one of the APIs has fetched the data, it will be passed to the useState hook, which will then trigger the application to re-arrange the data and render them in the application.

4. System Features

4.1 Greeting Page

4.1.1 Description and Priority

The greeting page of the TripSmart application is a critical component that presents the application name and description. As the first point of contact with the user, the greeting page must be well-designed, informative, and user-friendly. Its purpose is to provide an introduction to the application's features and functionalities. Therefore, this use case has a high development priority to ensure that the greeting page effectively communicates the application's purpose and provides a seamless and intuitive experience for the user.

4.1.2 Stimulus/Response Sequences

The greeting page of the website must appear when the application is first launched.

4.1.3 Functional Requirements

.1 Upon entering the application, the first greeting page must be displayed.

.1.1 The first greeting page must showcase the application's name and provide a brief description of its functionality.

.1.1.2 The first greeting page must feature a slider. The slider's first option must be pre-selected as it's the first page.

.1.1.2.1 The slider must have three buttons. The first button must take the user back to the first page, the second button must take them to the second page, and the third button must take them to the third page.

.1.1.3 The first greeting page must feature a next button. The next button must allow the user to navigate to the second greeting page.

.2 The second greeting page must provide a more detailed explanation of what the user can expect from using the application.

.2.1 The second greeting page must feature a slider. The slider's second option must be pre-selected as it's the second page.

.2.1.1 The slider must have three buttons. The first button must take the user back to the first page, the second button must take them to the second page, and the third button must take them to the third page.

.2.2 The second greeting page must feature a next button. The next button must allow the user to navigate to the third greeting page.

.3 The third greeting page must further elaborate on what the user can expect from using the application.

.3.1 The third greeting page must feature a slider. The slider's third option must be pre-selected as it's the third page.

.3.1.1 The slider must have three buttons. The first button must take the user back to the first page, the second button must take them to the second page, and the third button must take them to the third page.

.3.2 The third greeting page must feature a next button. The next button must allow the user to navigate to the selection page.

4.2 Selection Page

4.2.1 Description and Priority

The selection page of the TripSmart application presents users with three exclusive options, "Login," "Sign Up," and "Continue as Guest." This page appears after the greeting page, and each option directs the user to a different page. The selection page is critical to the application's functionality and provides users with a clear path to continue using the application. Therefore, it has a high development priority to ensure that users can easily navigate and select the appropriate option without confusion or errors.

4.2.2 Stimulus/Response Sequences

Click the "Next" button on the third greeting page.

4.2.3 Functional Requirements

.1 The selection page must present users with three distinct and mutually exclusive options: "Login," "Sign Up," and "Continue as Guest."

.1.1 If the user selects "Login," they must be redirected to the login page.

.1.2 If the user chooses "Sign Up," they must be directed to the registration page.

.1.3 If the user decides to "Continue as Guest," they must be redirected to the search page.

.1.3.1 If the user continues as a guest, the user must be treated as a guest, and not a logged-in user.

4.3 Login Page

4.3.1 Description and Priority

The login page allows users to access their accounts and utilize the full functionality of the platform. The page comprises a series of input text boxes where users must enter their email address and password to log in. The system then verifies the accuracy of the login information entered by the user, and if it is correct, it redirects them to the search page. In case the login credentials entered are incorrect, an error message is displayed, prompting the user to re-enter

their login credentials. Given that users can still utilize the app's primary functionalities without logging in, the priority for the login page is considered to be medium.

4.3.2 Stimulus/Response Sequences

1. Clicking the "Login" button on the selection page
2. Clicking the “Bring me to Log in page” button in the alert message upon successful registration on the registration page
3. Clicking the "Sign in" button on the registration page

4.3.3 Functional Requirements

.1 Users who have an existing account must be able to log in to the application using their respective email address and password.

.1.1 The user must click on the email input text box.

.1.2 The user must enter into the email input text box the email address used when registering for an account.

.1.2.1 The email address must be in alphanumeric form.

.1.3 The user must enter into the password input text box the password created when registering for an account.

.1.3.1 The password must be in alphanumeric form.

.1.3.2 There must be a button that toggles the password between being hidden and visible, allowing the user to see the actual text for checking purposes.

.2 The application must verify that the email and password pair entered by the user exists in the database.

.2.1 If the login information cannot be verified, the system must display an alert message to the user stating “Email or password is incorrect”.

.2.1 The alert message must have an “OK” button.

.2.1.1 The user must press the “OK” button to close the alert message.

.2.1.2 The application must redirect the user back to the login page.

.2.2 If the login information can be verified, the system must allow the user to log in to the application.

.2.2.1 Upon successful login, the application must redirect the user to the search page.

.3 The login page must have a "Register Now" button that directs the user to the registration page upon clicking.

4.4 Registration Page

4.4.1 Description and Priority

The registration process for the application involves inputting a valid email address, a 12-18 character-long alphanumeric password, and accepting the terms and conditions. The application must validate all inputs and display appropriate error messages if the inputs are invalid or incomplete. Upon successful validation, the application must store the email and password in the database and display a success message with a redirect button to the Login page. The priority for this feature is medium because while it is essential for users to have accounts to use the application, the application can still function without an account.

4.4.2 Stimulus/Response Sequences

1. Clicking the "Sign Up" button on the selection page
2. Clicking the “Register Now” button on the login page

4.4.3 Functional Requirements

.1 The user must input their email address into an input textbox.
- .1.1 The user must click on the email input text box.
- .1.2 The email address must be in a valid email format.
  - .1.2.1 If the email address is not valid, an alert message "Invalid email format" must be displayed when the user clicks on the "Sign Up" button.
    - .1.2.1.1 The alert message must have an "OK" button.
    - .1.2.1.2 Pressing the "OK" button must close the alert message and return the user to the register page.
  - .1.2.2 The email address must be alphanumeric.
  - .1.2.3 The email address must not exist in the database before.
    - .1.2.3.1 If the email address has been used to register before, an alert message "Email already registered" must be displayed when the user clicks on the "Sign Up" button.
      - .1.2.3.1.1 The alert message must have an "OK" button and a "Log me in" button.
        - .1.2.3.1.1.1 Pressing the "OK" button must close the alert message and return the user to the register page.
        - .1.2.3.1.1.2 Pressing the "Log me in" button must redirect the user to the login page.

.2 The user must input their password into the password input text box.
- .2.1 The user must focus on the password input text box.

.2.1.1 There must be a button that toggles the password between being hidden and visible, allowing the user to see the actual text.

.2.2 The password must be at least 12-18 characters long and alphanumeric.

.2.2.1 If the password is not valid, an alert "Password must be at least 12 digits and contain an alphabetical letter" must be displayed when the user clicks on the "Sign Up" button.

.2.2.1.1 The alert message must have an "OK" button.

.2.2.1.2 Pressing the "OK" button must close the alert message and return the user to the register page.

.3 The user must retype their password into the retype password input text box.

.3.1 The user must focus on the retype password input text box.

.3.1.1 There must be a button that toggles the retype password between being hidden and visible, allowing the user to see the actual text.

.3.2 The valid retype password value must have the same value as the one inside the password input text box.

.3.2.1 If the retype password does not match, an alert message "Passwords do not match" must be displayed when the user clicks on the "Sign Up" button.

.3.2.1.1 The alert message must have an "OK" button.

.3.2.1.2 Pressing the "OK" button must close the alert message and return the user to the register page.

.4 The user must check the checkbox beside "I accept the Terms and the Privacy Policy."

.4.1 If the user hasn't checked the checkbox, an alert message "Please agree to the terms and conditions" must be displayed.

.4.1.1 The alert message must have an "OK" button.

.4.1.2 Pressing the "OK" button must close the alert message and return the user to the register page.

.5 Once all required inputs are valid, and the user clicks the "Sign Up" button, the application must store the entered email and password in the database.

.5.1 If the storage is successful, the alert message "Success! You have been registered with us" along with a "Bring me to the Login Page" button must be displayed.

.5.1.1 Clicking the "Bring me to the Login Page" button must redirect the user to the Login Page.

.6 The registration page must have a "Sign in" button that directs the user to the login page upon clicking.

4.5 Search Page

4.5.1 Description and Priority

The Search Input Functionality provides an efficient way for users to input their starting location, which is crucial in navigating through the application. It includes GPS integration, where the Starting Location Input is automatically filled with the current location's address if the user grants GPS permission. Additionally, the application suggests up to five addresses in a dropdown format, which is limited to locations within Singapore, when the user types in the Starting Location Input. If the user selects an address from the dropdown, it will be automatically filled in the input box. The priority for this feature is high because it enhances the user experience by providing an intuitive and seamless way to input the starting location.

4.5.2 Stimulus/Response Sequences

1. Clicking the "Continue as Guest" button on the selection page
2. Successfully logging in to an account from the login page
3. Clicking the “Search” button in the navigation bar

4.5.3 Functional Requirements

.1 Upon accessing the search page, the user must be prompted to grant GPS permission.

.2 The search page must implement the feature: Starting Location Input Feature, Destination Location Input.

.3 The search page must display a Google Map in the background with Singapore focused on it.

.3.1 The user must be able to move the map around by dragging it.

.4 If a valid address is provided in the Starting Location Input Feature, a marker must be placed on the map.

.4.1 The map must zoom in on the position of the marker.

.4.2 An invalid address will not generate a marker.

.5 If a valid address is provided in the Destination Location Input Feature, a second marker must be placed on the map.

.5.1 The map must zoom in on the position of the marker.

.5.2 An invalid address will not generate a marker.

.6 When both Starting Location Input Feature and Destination Location Input Feature have valid addresses:

.6.1 A route will be displayed on the map connecting the two markers.

.6.1.1 The route must show the most optimal path to travel from the starting location to the destination location.

.6.2 Clicking on the "Show Result" button will take the user to the result page.

.6.3 If there is an invalid address in either the Starting Location Input or Destination Location Input

.6.3.1 "Show Result" button will not be clickable.

.6.3.2 Route between markers will not be displayed.

.7 The Navigation Bar Feature must be rendered at the bottom.

4.6 Starting Location Input

4.6.1 Description and Priority

The Search Input Functionality is a high-priority feature that streamlines the input of the user's starting location. It includes GPS integration and suggests up to five Singapore-based addresses in a dropdown format when the user types in the Starting Location Input. The user's current location's address is automatically filled in if GPS permission is granted, and the selected address is automatically filled in the input box. This feature improves the user experience by providing an intuitive and seamless way to input the starting location.

4.6.2 Stimulus/Response Sequences

Implemented inside the search page

4.6.3 Functional Requirements

.1 The starting location input is a text input that holds the starting location address.

.2 The starting location input must get information from the search page on whether or not the User has granted GPS permission or not.

.2.1 If permission is granted, the text input must be automatically filled with the current location's address.

.2.1.1 The address must be in the form of an alphanumeric string.

.3 When the User types a character or number into the starting location input, it must display a dropdown list of suggested addresses.

.3.1 The dropdown must provide a maximum of five suggested addresses.

.3.2 The suggested addresses in the dropdown must be located in Singapore.

.3.3 If the User selects an address from the dropdown, the starting location input must be automatically filled with the chosen address.

.3.4 Only the addresses selected from the dropdown are considered valid.

4.7 Destination Location Input

4.7.1 Description and Priority

The User can input the Destination Location as an alphanumeric string in a text input box. When the User types into the Destination Location Input, a dropdown with a maximum of five suggested addresses located in Singapore will appear. If the User selects an option from the dropdown, the application will autofill the Destination Location Input with the chosen address. The priority for this functionality is high as it's crucial for users to easily input their desired destination.

4.7.2 Stimulus/Response Sequences

Implemented inside the search page

4.7.3 Functional Requirements

.1 The destination location input is a text input that holds the destination location address.

.2 When the User types a character or number into the destination location input, it must display a dropdown list of suggested addresses.

.2.1 The dropdown must have a maximum of five suggested addresses.

.2.2 The suggested addresses in the dropdown must be located in Singapore.

.2.3 If the User selects an address from the dropdown, the destination location input must be automatically filled with the chosen address.

.2.4 Only the addresses selected from the dropdown are considered valid.

4.8 Travel Time and Fare Comparison

4.8.1 Description and Priority

The system must return a report containing multiple result entries, each displaying a transportation mode and the corresponding travel time and price. The report should be sorted in ascending order of either pricing fare or travel time, depending on the user's selection of the "Cheapest" or "Fastest" button. The pricing and travel time must be calculated according to specific rules for each transportation mode and displayed in a standardized format. The recommended travel routes and distances must be obtained from the Google Maps API, and for ride-hailing companies, pricing and travel time must be obtained from their respective APIs. The priority for this functionality is high.

4.8.2 Stimulus/Response Sequences

1. Clicking the “Show Result” in the search page.

4.8.3 Functional Requirements

.1 Once the user input both a valid starting address and destination address, the application must display and sort the query results based on travel time or travel fare.

.1.1 The User must be able to toggle between the 2 buttons - “Cheapest” and “Fastest” - to display the query result based on travel fare and travel time respectively.

.1.2 “Cheapest” button must be the default option.

.2 When the “Cheapest” button is selected, the query result will be displayed in ascending order of pricing.

.2.1 The query result must contain multiple result entries.

.2.2 Every result entry must include one and only one transportation mode, which allows the User to travel from starting location to the destination.

.2.3 The result list must be sorted in the order of increasing pricing fare, with the first travel option at the top of the list having the lowest cost of travel involved.

.2.4 Each result entry must include the price of the corresponding transportation.

.2.4.1 The price must begin with a dollar sign and be formatted into two decimal places.

.2.4.2 For public transport, the price returned must be calculated based on the travel distance between the starting point and the destination.

.2.4.2.1 The system must decide the travel distance based on the recommended public transportation travel route by the Google Maps API.

.2.4.3 For car-rental, the price returned must be calculated by multiplying estimated travel time with rental fee per unit time.

.2.4.3.1 The system must decide the travel distance based on the recommended travel route by the Google Maps API.

.2.4.4 For ride-hailing, the price returned must be obtained from the ride-hailing companies’ API.

.2.4.5 For taxi, the price returned must be calculated by multiplying travel distance with taxi fare per distance.

.2.4.5.1 The system must decide the travel distance based on the recommended travel route by the Google Maps API.

.2.5 Each result entry must include the travel time of the corresponding transportation method.

.2.5.1 The travel time in the results must be in minutes.

.2.5.2 For public transport, the time returned must be calculated based on the travel distance between the starting point and the destination.

.2.5.2.1 The system must decide the travel distance based on the recommended public transportation travel route by the Google Maps API.

.2.5.2.2 The result must consider the travel time of traveling to the bus stations and/or MRT stations.

.2.5.3 For car-rental, the time returned must be calculated based on the travel distance.

.2.5.3.1 The system must decide the travel distance based on the recommended travel route by the Google Maps API.

.2.5.3.2 The result must include the time taken to travel to the rental car place.

.2.4.4 For ride-hailing, the time returned must be obtained from the ride-hailing companies' API.

.2.5.5 For taxi, the time returned must be calculated based on the travel distance.

.2.5.5.1 The system must decide the travel distance based on the recommended travel route by the Google Maps API.

.2.6 Each result entry must include the service type name, which is the corresponding transportation method.

.2.7 Each result entry must include the corresponding service type icon.

.3 When "Fastest" button is selected, the query result must be displayed in ascending order of duration.

.3.1 The query result must contain multiple result entries.

.3.2 Every result entry must include one and only one transportation mode, which allows the User to travel from starting location to destination.

.3.3 The result list must be sorted in the order of increasing travel time, with the first travel option at the top of the list having the shortest travel time involved.

.3.4 Each result entry must include the price of the corresponding transportation.

.3.4.1 The price must begin with a dollar sign and formatted into two decimal places.

.3.4.2 For public transport, the price returned must be calculated based on the travel distance between the starting point and the destination.

.3.4.2.1 The system must decide the travel distance based on the recommended public transportation travel route by the Google Maps API.

.3.4.3 For car-rental, the price returned must be calculated by multiplying estimated travel time with rental fee per unit time.

.3.4.3.1 The system must decide the travel distance based on the recommended travel route by the Google Maps API.

.3.4.4 For ride-hailing, the price returned must be obtained from the ride-hailing companies' API.

.3.4.5 For taxi, the price returned must be calculated by multiplying travel distance with taxi fare per distance.

.3.4.5.1 The system must decide the travel distance based on the recommended travel route by the Google Maps API.

.3.5 Each result entry must include the travel time of the corresponding transportation method.

.3.5.1 The travel time in the result entry must be in minutes.

.3.5.2 For public transport, the time returned must be calculated based on the travel distance between the starting point and the destination.

.3.5.2.1 The system must decide the travel distance based on the recommended public transportation travel route by the Google Maps API.

.3.5.2.2 The result must consider the travel time of traveling to the bus stations and/or MRT stations.

.3.5.3 For car-rental, the time returned must be calculated based on the travel distance.

.3.5.3.1 The system must decide the travel distance based on the recommended travel route by the Google Maps API.

.3.5.3.2 The result must include the time taken to travel to the rental car place.

.3.5.4 For ride-hailing, the time returned must be obtained from the ride-hailing companies' API.

.3.5.5 For taxi, the time returned must be calculated based on the travel distance.

.3.5.5.1 The system must decide the travel distance based on the recommended travel route by the Google Maps API.

.3.6 Each result entry must include the service type name, which is the corresponding transportation method.

3.6.1 The service type name must be an alphanumeric string.

.3.7 Each result entry must include the corresponding service type icon.

4.9 Filter Function

4.9.1 Description and Priority

The filter function allows the user to customize their search results based on their preferences such as the number of travelers, pet-friendly, eco-friendly, and wheelchair accessibility. It is of medium priority as it enhances the user experience by providing relevant results based on their preferences, but it is not essential for basic functionality.

4.9.2 Stimulus/Response Sequences

1. Clicking the "Filter" button on the result list page.

4.9.3 Functional Requirements

.1 The application must provide a filter button on the result list page.

.1.1 After the button is clicked, the application must display an extra query menu.

.1.2 The user must be able to indicate the number of travelers in the query menu.

.1.2.1 After the user indicates the number of travelers, the application must update the result list, such that all the results in the result list are updated based on the number of travelers.

.1.2.2 If the User does not indicate how many passengers will be traveling, the application updates the query result list, such that all the results are based on a single passenger traveling, by default.

.1.3 The user must be able to indicate their preference for pet-friendly transportation modes.

.1.3.1 If the user indicates they prefer pet-friendly transportation modes, the application must update the result list, such that all the results in the query results are pet-friendly.

.1.3.2 If the user indicates they do not prefer pet-friendly transportation modes, the application must update the result list, such that all the results in the query results are either pet-friendly or not pet-friendly.

.1.4 The user must be able to indicate their preference for eco-friendly transportation modes.

.1.4.1 If the user indicates they prefer eco-friendly transportation modes, the application must update the result list, such that all the results in the query results are eco-friendly.

.1.4.2 If the user indicates they do not prefer eco-friendly transportation modes, the application must update the result list, such that all the results in the query results are either eco-friendly or not eco-friendly.

.1.5 The user must be able to indicate their preference for wheelchair-accessible transportation modes.

.1.5.1 If the user indicates they prefer wheelchair-accessible transportation modes, the application must update the result list, such that all the results in the query results are wheelchair-accessible.

.1.5.2 If the user indicates they do not prefer wheelchair-accessible transportation modes, the application must update the result list, such that all the results in the query results are either wheelchair-accessible or not wheelchair-accessible.

4.10 Deep Linking

4.10.1 Description and Priority

Deep linking is a functionality that allows users to be redirected to a specific page within an application or website directly from another application or website. In this case, the priority of the deep linking feature is medium as it plays a crucial role in enabling the user to select their preferred mode of transportation efficiently. The deep linking feature ensures that users are redirected to the appropriate mode of transportation they have chosen quickly. Additionally, it

helps to enhance the user experience by reducing the steps required to access the transportation services they need. Overall, the deep linking feature is essential for the efficient functioning of the transportation selection process and, therefore, has a medium priority.

4.10.2 Stimulus/Response Sequences

1. Clicking any one of the results presented in the result list.

4.10.3 Functional Requirements

.1 At the result page, the User needs to choose their preferred mode of transportation based on the report they received.

.2 If the User selects public transportation, they must be redirected to the Google Maps application with the starting location and destination address already provided.

.2.1 In case Google Maps is not installed, the User must be redirected to the Google Maps website on their browser with the starting location and destination address already inputted.

.3 If the User selects a BlueSG car rental, they must be directed to the blueSG website at 'https://www.bluesg.com.sg/car-sharing#map'.

.4 If the User chooses a Grab ride-hailing service, they must be redirected to the appropriate store page based on their device.

.4.1 If the User is using an Android device, they must be redirected to the Grab PlayStore page

.4.2 If the User is using an IOS device, they must be redirected to the Grab App Store page.

.5 When the User chooses a ComfortDelGro taxi, they must be redirected to the appropriate store page based on their device.

.5.1 If the User is using an Android device, they must be redirected to the ComfortDelGro PlayStore page

.5.2 If the User is using an IOS device, they must be redirected to the ComfortDelGro App Store page.

4.11 History Page

4.11.1 Description and Priority

The History feature is designed to save the User's selected mode of transportation and display a list of previous search results. Whenever the User selects their preferred mode of transportation,

it must be saved in a database. If the User is not logged in, the application must display a "Login to Save and View History" button that redirects them to the Login Page when clicked. If the User is logged in, a list of history should be displayed, with each result showing the transportation icon, transportation type, duration, and price. The priority of the History feature is low because it is not a critical function of the application, but it can enhance the user experience by providing them with easy access to their past searches.

4.11.2 Stimulus/Response Sequences

1. Clicking the “History” button in the navigation bar.

4.11.3 Functional Requirements

.1 If the User is not logged in:

.1.1 The history page must display a "Login to Save and View History" button.

.1.1.1 Pressing the "Login to Save and View History" button must redirect the User to the login page.

.2 If the User is logged in:

.2.1 Clicking any result query in the result list must save it into the database.

.2.2 Upon launching the history page, the application must retrieve the user's stored history from the database.

.2.3 Each history result must display the transportation icon, type, duration, and price.

.2.3.1 The results must be sorted in ascending order of time, with the most recent query being placed at the bottom.

.2.3.2 Each result must include the service type name, which is the corresponding transportation method.

.2.3.2.1 The service type name must be an alphanumeric string.

.2.3.3 Each result entry must include the corresponding service type icon.

.2.3.4 Each result must include the price of the corresponding transportation.

.2.3.4.1 The price must begin with a dollar sign and formatted into two decimal places.

.2.3.5 Each result entry must include the travel time of the corresponding transportation method.

.2.3.5.1 The travel time in the result entry must be in minutes.

4.12 Settings Page

4.12.1 Description and Priority

The Setting feature in the application allows the user to customize their language preference and provides access to the Change Password and Log Out options. The description specifies that the

application must have a navigation button to lead to the Setting page and provide language options for the four main native languages in Singapore, with English as the default language. The priority for this feature is low since it doesn't directly impact the core functionality of the application, which is to provide travel route options to the user. However, it is still important to provide language options for users who may not be comfortable with English and to allow them to manage their account information through the Change Password and Log Out options.

4.12.2 Stimulus/Response Sequences

1. Clicking the “Setting” button in the navigation bar

4.12.3 Functional Requirements

.1 The application must have a navigation button on the Search Page, Result Page, and History Page to access the Settings page.

.2 The Settings page must allow users to select one of Singapore's four main native languages: English, Chinese, Malay, or Tamil, to be used throughout the application.

.2.1 The default language must be set to English.

.2.2 Users must be able to choose their preferred language through a dropdown menu.

.2.3 Once a language is selected, all text within the application must be displayed in the chosen language.

.3 If a user is not logged in, the application must display a "Log In" button.

.3.1 If a user selects "Log In", they must be redirected to the Login Page.

.4 If a user is logged in, the application must present two mutually exclusive options: "Change Password" or "Logout".

.4.1 If a user selects "Change Password", they must be redirected to the Change Password Page.

.4.1.1 On the Change Password Page, users must be able to input their email, old password, and new password in separate input boxes.

.4.1.2 The application must verify the accuracy of the email and password entered by the user.

.4.1.2.1 If the email address entered does not exist in the database, the application must display an error message stating "Email does not exist."

.4.1.2.2 If the email address exists in the database, but the old password entered is not associated with it, the application must display an error message stating "Password is incorrect."

4.1.2.3 If the new password the user entered is the same as the old password, the application must reject it, and display an error message stating "New password cannot be the same as old password".

.4.1.3 The application must verify that the new password entered is between 12-18 characters long and alphanumeric.

.4.1.4 The application must update the password associated with the email in the database to match the new password entered by the user.

.4.2 If a user selects "Logout," they must be redirected to the Selection Page and signed out of their account.

4.13 Navigation Bar

4.13.1 Description and Priority

The navigation bar that leads to the settings page from the Search Page, Result Page, and History Page. In the Settings Page, users must be able to choose between four main native languages in Singapore: English, Chinese, Malay, and Tamil. The default language should be English, and the choice of language must be displayed using a dropdown. Once the user selects a language, all text in the application must be displayed in the chosen language. This feature is considered low priority and should only be implemented after more critical features have been completed.

4.13.2 Stimulus/Response Sequences

This navigation bar is rendered whenever the user is in Search Page, Result List Page, Result Filter Page, Settings Page and History Page.

4.13.3 Functional Requirements

.1 The navigation bar must be rendered whenever the user is in Search Page, Result List Page, Result Filter Page, Settings Page and History Page.

.2 The navigation bar must present users with three distinct and mutually exclusive options: "Map", "History" and "Settings"

.2.1 If the user selects "Map" they must be redirected to the search page.

.2.2 If the user chooses "History" they must be redirected to the history page

.2.3 If the user decides to "Setting" they must be redirected to the setting page.

5. Other Non-Functional Requirements

5.1 Performance Requirements

5.1.1 When a User requests a ride, the system must process the request and show the result within 30 seconds.
5.1.2 The application should be able to redirect the User to the respective external application in 10 seconds.
5.1.3 Location should be accurate with a maximum error of 50m.

5.2 Safety Requirements

5.2.1 The application must ensure the security and privacy of user data at all times, to prevent unauthorized access or breaches.
5.2.2 The application must be thoroughly tested for potential security vulnerabilities or safety risks before being released to the public.
5.2.3 The application must be designed to prevent any potential harm to the user or others when in use.

5.3 Security Requirements

5.3.1 Password field should be masked when the user is typing it in.
5.3.2 Deter others from looking at the user's password when typing it in.

5.4 Software Quality Attributes

5.4.1 **Scalability:** More companies and other modes of travel can be integrated into the application, for future updates. In the case where the company has closed down, the application should update itself accordingly.
5.4.2 **Supportability:** The system must be able to support 500 Users simultaneously on the application at the same time
5.4.3 **Reliability:** The app must be available for use 99.9% of the time, with no more than 1 hour of planned downtime per year.

5.5 Business Rules

5.5.1 **User**

- Initiators are free to manage his/her particulars on the platform
- Initiators can choose to accept or reject the terms and conditions
- Initiators can clear his/her history that is associated with their accounts

5.5.2 **Developer**

- Developer can remove any users that do not abide by the rules and regulations defined by TripSmart
- Developer is authorized to remove any user particulars and history of the users

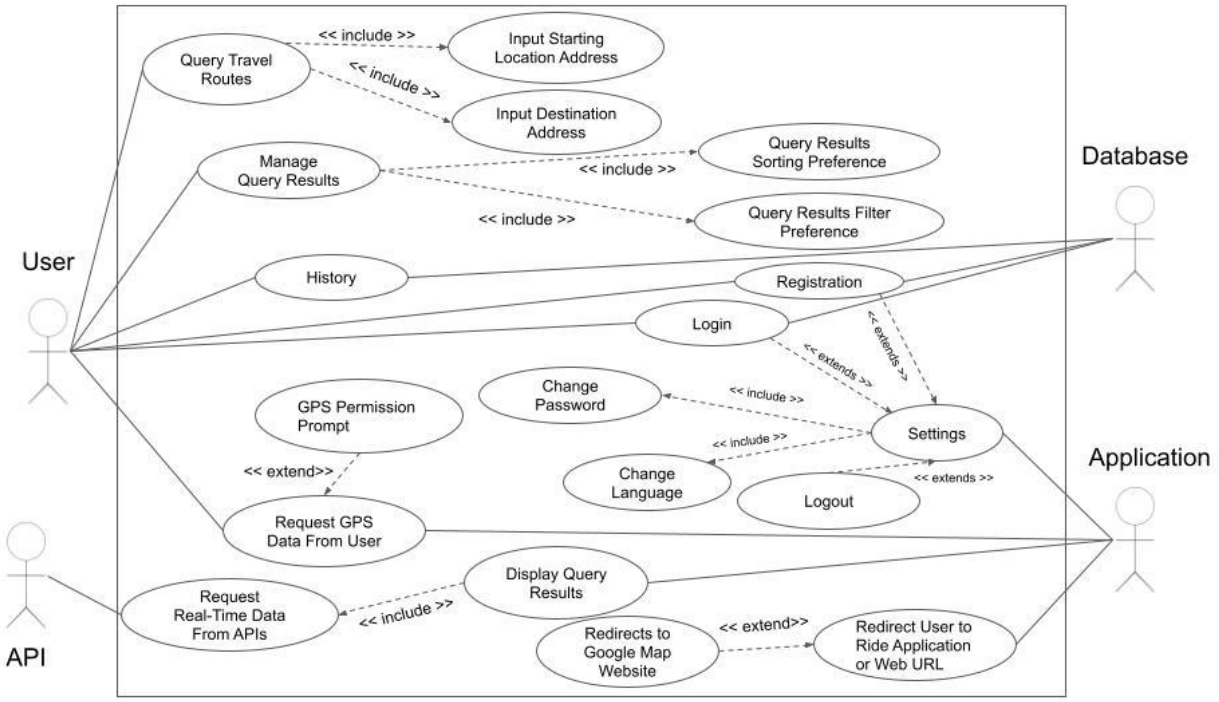
6. Other Requirements

6.1 Data Dictionary

Term	Description
User	An individual who uses the app to compare fares of different modes of transportation to reach their intended destination.
Starting Location	The point where the User wishes to begin his/her journey. It can either be his/her current physical location, as indicated by his/her phone GPS data, a location chosen by the User by pinpointing the location on the map or another location if he/she wants to plan in advance.
Destination	The intended location the User wishes to go. The User enters it when requesting a ride or seeking public transport options, and it serves as the final stop. This data is used by the app to estimate the travel time and travel fares.
Travel Route	The path taken to get from the User's starting point to the intended destination. Upon gathering the User's input of the starting location and destination, the app will display a list of travel routes available. This varies between different travel routes and modes of transportation.
Travel Distance	The distance of a travel route from the User's starting location to the User's destination. This varies between different travel routes and modes of transportation.
Travel Time	The amount of time taken for the User to get from his / her starting point to his / her intended destination. This varies between different travel routes and modes of transportation. This data helps the User make an informed decision in choosing the best route.
Travel Fare	The cost incurred in the travel route between the User's starting point and intended destination. This varies between different travel routes and modes of transportation. This data helps the User to make an informed decision in choosing the best route.

Result List	Contains all the possible travel routes between the starting location and the destination.
Result Entry	One of the possible travel routes between the starting location and the destination found in the result list.
SortResult (Control Class)	This is one of the Key Control Classes, which will sort the result list either by price or time.
FilterResult (Control Class)	This is one of the Key Control Classes, which will filter the result list according to the User's additional input. The User can filter the result list by total passenger number, or filter by whether the transportation mode is pet-friendly.

6.2 Use Case Diagram



6.3 Use Case Descriptions

Use Case ID:	1		
Use Case Name:	Query Travel Routes		
Created By:	Koh Hao Sheng	Last Updated By:	Lim Boon Hian
Date Created:	8/2/2023	Date Last Updated:	13/4/2023

Actor:	User
Description:	User queries for possible travel routes.
Preconditions:	User must open the application.
Post-conditions:	Application requests for Global Positioning System (GPS) permission from User.
Flow of Events:	1. Application renders the result page, which consists of maps and a bottom scroll view. 2. User can key in starting location and destination in the bottom scroll view.
Alternative Flows:	-NIL-
Exceptions:	-NIL-
Assumptions:	-NIL-
Priority:	High

Use Case ID:	2		
Use Case Name:	Input Starting Location Address		
Created By:	Lim Dong Wan	Last Updated By:	Lim Boon Hian
Date Created:	8/2/2023	Date Last Updated:	14/4/2023

Actor:	User
Description:	User enters a starting location address.
Preconditions:	Application must request GPS permission from User, and an input box must be available.
Post-conditions:	User enters a destination address.
Flow of Events:	1. User clicks on the empty space in the input box. 2. User inputs a starting location address. 3. User must click on his or her intended starting location address in a drop box.
Alternative Flows:	2. AC.1 If the user provides GPS permission, the input box is automatically filled with the User's location.
Exceptions:	-NIL-
Assumptions:	-NIL-
Priority:	High

Use Case ID:	3		
Use Case Name:	Input Destination Address		
Created By:	Lim Dong Wan	Last Updated By:	Lim Boon Hian
Date Created:	8/2/2023	Date Last Updated:	14/4/2023

Actor:	User
Description:	User enters a destination address.
Preconditions:	User must enter a valid starting location address, and an input box must be available.
Post-conditions:	User decides how the query results should be sorted and filtered.
Flow of Events:	1. User clicks on the empty space in the input box. 2. User inputs a destination address. 3. User must click on his or her intended destination address in a drop box.
Alternative Flows:	-NIL-
Exceptions:	-NIL-
Assumptions:	-NIL-
Priority:	High

Use Case ID:	4		
Use Case Name:	Manage Query Results		
Created By:	Lim Boon Hian	Last Updated By:	Ng Woon Yee
Date Created:	11/2/2023	Date Last Updated:	13/4/2023

Actor:	User
Description:	User manages query results displayed in the application interface.
Preconditions:	User must enter a valid starting location address, destination address and click the “Show Result” button, and the application must redirect User to the Query Results page.
Post-conditions:	User decides how the query results should be sorted and filtered.
Flow of Events:	1. Upon entering a valid starting location address, destination address and clicking the “Show Result” button, the application redirects User to the result list page. 2. Application requests data from various APIs, and displays a list of query results, sorted based on the cheapest travel fare by default. 3. User decides whether to sort the results based on travel time or travel fare. 4. User decides whether to filter the results based on the number of passengers traveling, or if the mode of transport is wheelchair-accessible, pet-friendly or eco-friendly.
Alternative Flows:	-NIL-
Exceptions:	-NIL-
Assumptions:	-NIL-
Priority:	High

Use Case ID:	5		
Use Case Name:	Query Results Sorting Preference		
Created By:	Lim Boon Hian	Last Updated By:	Ng Woon Yee
Date Created:	8/2/2023	Date Last Updated:	13/4/2023

Actor:	User
Description:	User decides how the query results should be sorted.
Preconditions:	User must enter a valid starting location address and destination address, and the application must redirect User to the Query Results page.
Post-conditions:	Application requests data from various APIs to provide the cheapest or fastest route as the first option in the list of query results for the requested trip, based on the User's sorting preference.
Flow of Events:	1. Application requests data from various APIs, and displays a list of query results, sorted based on the cheapest travel fare by default. 2. User can toggle between the 2 buttons - "Cheapest" and "Fastest" - to display the query result based on travel fare and travel time respectively. a. If the button displays "Fastest" when the User clicks on it, the application sorts the query results based on the fastest travel time. b. If the button displays "Cheapest" when the User clicks on it, the application sorts the query results based on the cheapest travel fare.
Alternative Flows:	5.AC.1 User clicks on the "Filter" button, which enables the User to input extra filter requirements for the query results, so as to re-render or trim the list of query results.
Exceptions:	-NIL-
Assumptions:	The application can always find an available transportation mode, as long as both the starting location and destination addresses are valid Singapore addresses.
Priority:	High

Use Case ID:	6		
Use Case Name:	Query Results Filter Preference		
Created By:	Lim Boon Hian	Last Updated By:	Lim Dong Wan
Date Created:	11/2/2023	Date Last Updated:	11/4/2023

Actor:	User
Description:	User decides how the query results should be filtered.
Preconditions:	User must click on the “Filter” button.
Post-conditions:	Application requests data from various APIs, based on the extra filter requirements requested by the User, and updates the query results accordingly.
Flow of Events:	1. User clicks on the “Filter” button. 2. Application displays an extra query menu as a pop-up. 3. User decides whether to indicate the number of passengers who will be traveling, or their preference for an eco-friendly, pet-friendly or wheelchair-accessible transportation mode, in the query menu. a. If the User indicates the number of passengers, the application updates the query result list with travel options that are based on the indicated number of travelers. b. If the User indicates their preference for an eco-friendly transportation mode, the application updates the result list with eco-friendly transportation modes. c. If the User indicates their preference for a pet-friendly transportation mode, the application updates the result list with pet-friendly transportation modes. d. If the User indicates their preference for a wheelchair-accessible transportation mode, the application updates the result list with wheelchair-accessible transportation modes.
Alternative Flows:	6.AC.1 If the User does not indicate how many passengers will be traveling, the application updates the query result list, such that all the results are based on a single passenger traveling, by default. 6.AC.2 If the User indicates no preference for eco-friendly transportation mode, the application updates the query result list, such that all the results are either eco-friendly or non-eco friendly transportation modes. 6.AC.3 If the User indicates no preference for pet-friendly transportation mode, the application updates the query result list, such

	that all the results are either pet-friendly or non-pet friendly transportation modes. 6.AC.4 If the User indicates no preference for wheelchair-accessible transportation modes, the application updates the query result list, such that all the results are either wheelchair-accessible or non-wheelchair accessible transportation modes.
Exceptions:	-NIL-
Assumptions:	-NIL-
Priority:	High

Use Case ID:	7		
Use Case Name:	Request GPS Data From User		
Created By:	Ng Woon Yee	Last Updated By:	Ng Woon Yee
Date Created:	8/2/2023	Date Last Updated:	13/4/2023

Actor:	User, Application
Description:	Application requests GPS access from the User in order to obtain the User's current location for the purpose of inputting a starting location address for a trip.
Preconditions:	User must open the application.
Post-conditions:	User enters a starting location address to start querying for a trip.
Flow of Events:	1. User is currently in the Search page of the application. 2. Application checks if the User has granted GPS access. 3. If the User has granted GPS access, the application retrieves the User's current location.
Alternative Flows:	7.AC.1 If the User has not granted GPS access before, the application prompts the User to give GPS permission to the application.
Exceptions:	-NIL-
Assumptions:	User’s phone has a GPS installed.
Priority:	High

Use Case ID:	8		
Use Case Name:	GPS Permission Prompt		
Created By:	Ng Woon Yee	Last Updated By:	Ng Woon Yee
Date Created:	8/2/2023	Date Last Updated:	13/4/2023

Actor:	User, Application
Description:	Application prompts the User to grant GPS permission, in the event that permission has not been granted.
Preconditions:	User has not granted GPS Permission to the application.
Post-conditions:	User enters a starting location address to start querying for a trip.
Flow of Events:	1. Application displays a pop-up window to prompt the User to grant GPS access. 2. User decides whether to grant GPS access. 3. If the User grants GPS access, the application retrieves the User's current location. 4. Application redirects the User to Input Starting Location Address Page, as granting GPS permission is optional for the User.
Alternative Flows:	8.AC.1 If the User denies GPS access, the application must allow the User to input address information through the input box.
Exceptions:	-NIL-
Assumptions:	User’s phone has a GPS installed.
Priority:	High

Use Case ID:	9		
Use Case Name:	Display Query Results		
Created By:	Loke Yong Jian	Last Updated By:	Ng Woon Yee
Date Created:	8/2/2023	Date Last Updated:	13/4/2023

Actor:	Application, API
Description:	Application displays a list of query results returned by the various APIs, based on the User's inputs of starting location and destination addresses.
Preconditions:	User must enter a valid starting location address and destination address.
Post-conditions:	Application displays a list of query results, sorted based on the cheapest travel fare by default.
Flow of Events:	1. User enters valid starting location and destination addresses in the input box. 2. Application extracts the address information and query the result through the various APIs. 3. Application redirects the User to the Result List page, which displays a list of query results based on the User's requested trip.
Alternative Flows:	-NIL-
Exceptions:	-NIL-
Assumptions:	-NIL-
Priority:	High

Use Case ID:	10		
Use Case Name:	Request Real-Time Data From APIs		
Created By:	Ng Woon Yee	Last Updated By:	Ng Woon Yee
Date Created:	8/2/2023	Date Last Updated:	13/4/2023

Actor:	Application, API
Description:	Application requests data from API to display a list of query results.
Preconditions:	User must input valid starting location and destination addresses and click the “Show Result” button.
Post-conditions:	Application displays a list of query results, sorted based on the cheapest travel fare by default.
Flow of Events:	1. Application has the addresses of the starting location and destination inputted by the User. 2. Application sends a request to the API to obtain relevant information, including travel time and estimated price. 3. API responds to the application's request and sends back the relevant information. 4. Application redirects the User to the Result List page, which displays a list of query results based on the User’s requested trip.
Alternative Flows:	-NIL-
Exceptions:	-NIL-
Assumptions:	The transportation APIs respond correctly and successfully to the application's requests.
Priority:	High

Use Case ID:	11		
Use Case Name:	Redirect User to Ride Application or Web URL		
Created By:	Lim Boon Hian	Last Updated By:	Ng Woon Yee
Date Created:	8/2/2023	Date Last Updated:	13/4/2023

Actor:	Application
Description:	Application redirects the User to the corresponding ride application, based on the User's preferred transportation mode.
Preconditions:	User must select a transportation mode (result entry) from the result list.
Post-conditions:	Application redirects the User to the said ride application or web url.
Flow of Events:	1. User selects the preferred transportation mode from the result list. 2. Application invokes the corresponding ride application based on the transportation mode that the User has selected. a. If BlueSG is selected, the application redirects the User to the BlueSG's website. b. If Grab is selected, the application redirects the User to the Grab's app store page. c. If public transportation is selected, the application redirects the User to Google Maps. d. If ComfortDelgro taxi is selected, the application redirects the User to the Comfortdelgro's app store page. 3. The corresponding ride application will be opened for the User if the said application is installed in the User's device.
Alternative Flows:	-NIL-
Exceptions:	11.EX.1 If Google Maps is not installed in the User's device, the User is redirected to the Google Maps website on their browser with the starting location and destination addresses already inputted.
Assumptions:	User has the corresponding ride application and web browsers installed on the User's device.
Priority:	Medium

Use Case ID:	12		
Use Case Name:	Redirect User to Google Map Website		
Created By:	Lim Boon Hian	Last Updated By:	Ng Woon Yee
Date Created:	8/2/2023	Date Last Updated:	13/4/2023

Actor:	Application
Description:	Application redirects the User to the Google Map page.
Preconditions:	If google map is not installed in the User's device, the User is redirected to the Google Maps website on their browser with the starting location and destination address already inputted.
Post-conditions:	Application redirects the User to Google Maps.
Flow of Events:	1. User selects public transportation as the transportation mode. 2. Application detects that Google Maps is not installed on the User's device. 3. Application redirects the User to Google Map website.
Alternative Flows:	-NIL-
Exceptions:	-NIL-
Assumptions:	User does not have Google Maps installed on the User's device.
Priority:	Medium

Use Case ID:	13		
Use Case Name:	Search History		
Created By:	Lim Dong Wan	Last Updated By:	Lim Dong Wan
Date Created:	11/2/2023	Date Last Updated:	19/2/2023

Actor:	User, Database
Description:	User views his / her past trip searches using the application.
Preconditions:	Application must have a “History” button available in the interface.
Post-conditions:	-NIL-
Flow of Events:	1. User clicks on the “History” button. 2. Application displays the User’s search history, sorted from oldest to latest. 3. Application displays information like starting location address, destination address, date and time that the User made the search query, in each entry of the search history.
Alternative Flows:	13.AC.1 Application displays an empty search history if the User has not made any search queries in the application yet.
Exceptions:	-NIL-
Assumptions:	-NIL-
Priority:	Medium

Use Case ID:	14		
Use Case Name:	Registration		
Created By:	Ng Woon Yee	Last Updated By:	Ng Woon Yee
Date Created:	10/2/2023	Date Last Updated:	13/4/2023

Actor:	User, Database
Description:	User registers for an account on the application to store their personal information.
Preconditions:	User must have a valid email address, and the "Register" button must be available in the Selection page.
Post-conditions:	User has a registered account on the application and proceeds to enter a starting location address to start querying for a trip.
Flow of Events:	1. User clicks on the "Register" button in the selection page and is redirected to the Registration page. 2. User provides a valid email address and creates a password with 12-18 characters, including alphanumeric characters. 3. User re-type the same password into the retype password input text box. 4. User submits the registration information. 5. Application verifies the email address and password format inputted by the User. 6. If the information is valid, the application creates a User account. 7. Application notifies the User of successful registration. 8. Application redirects User to the Login page.
Alternative Flows:	-NIL-
Exceptions:	14.EX.1 If the email address is already registered, the application displays an error message and prompts the User to try again with a different email address. 14.EX.2 If the email address is not valid, the application displays an error message and prompts the User to try again with a different email address. 14.EX.3 If the password does not meet the required format, the application displays an error message and prompts the User to try again with a password that meets the requirements. 14.EX.4 If retype password does not match, the application displays an error message and prompts the User to try again with a password that

	meets the requirements.
Assumptions:	User has access to a valid email address.
Priority:	Medium

Use Case ID:	15		
Use Case Name:	Login		
Created By:	Ng Woon Yee	Last Updated By:	Ng Woon Yee
Date Created:	10/2/2023	Date Last Updated:	13/4/2023

Actor:	User, Database
Description:	User logs in to their account on the application.
Preconditions:	User must have a pre-existing account on the application, and the "Login" button must be available in the selection page.
Post-conditions:	User has access to their personal information stored on the application, and proceeds to enter a starting location address to start querying for a trip.
Flow of Events:	1. User clicks on the "Login" button on the selection page. 2. User is redirected to the login page. 3. User provides his / her registered email address and password. 4. User submits the login information. 5. Application verifies if the email address and password match a registered account in the database. 6. If the information is valid, the application grants the User access to their account. 7. User is redirected to the Search page.
Alternative Flows:	-NIL-
Exceptions:	15.EX.1 If the email address and password do not match a registered account, the application displays an error message “Email or password is incorrect” and prompts the User to try again.
Assumptions:	-NIL-
Priority:	Medium

Use Case ID:	16		
Use Case Name:	Settings		
Created By:	Lim Boon Hian	Last Updated By:	Ng Woon Yee
Date Created:	19/2/2023	Date Last Updated:	13/4/2023

Actor:	User
Description:	Application must have a “Settings” icon in the interface.
Preconditions:	-NIL-
Post-conditions:	-NIL-
Flow of Events:	1. User clicks on the Settings icon. 2. Application directs the User to Settings page. 3. The application must display a dropdown to change language settings. 4. If the User is not logged in, the application must display a “Log in” option: a. If “Log in” is chosen, User must be redirected to the Login page. 5. If the User is logged in, the application must display two mutually exclusive options: “Change Password” & “Log Out”: a. If “Change Password” is chosen, User must be redirected to the Change Password page. b. If “Log Out” is chosen, User must be redirected back to the Login page.
Alternative Flows:	16.AC.1 After the user clicks the return button, the user must be able to return to the previous page they are browsing in the application.
Exceptions:	-NIL-
Assumptions:	Application must have a “Settings” icon in the interface, which redirects the User to the settings page.
Priority:	Medium

Use Case ID:	17		
Use Case Name:	Change Language		
Created By:	Lim Dong Wan	Last Updated By:	Lim Dong Wan
Date Created:	10/2/2023	Date Last Updated:	11/4/2023

Actor:	User
Description:	User selects the preferred language for the application.
Preconditions:	Application must have a “Settings” icon in the interface and a “Select Language” button in the Settings page.
Post-conditions:	-NIL-
Flow of Events:	1. User clicks on the Settings icon. 2. Application directs the User to Settings page. 3. User clicks on the “Change Language” dropdown. 4. Application displays 4 different language buttons - “English”, “Chinese”, “Malay” and “Tamil”. 5. User decides and clicks on the desired language button. 6. Application displays all text in the language chosen by User.
Alternative Flows:	17.AC.1 If the User never changes the language before, the application displays all text in English, by default.
Exceptions:	-NIL-
Assumptions:	User is able to read and understand at least 1 of the 4 languages provided by the application.
Priority:	Medium

Use Case ID:	18		
Use Case Name:	Change Password		
Created By:	Lim Dong Wan	Last Updated By:	Lim Dong Wan
Date Created:	20/2/2023	Date Last Updated:	11/4/2023

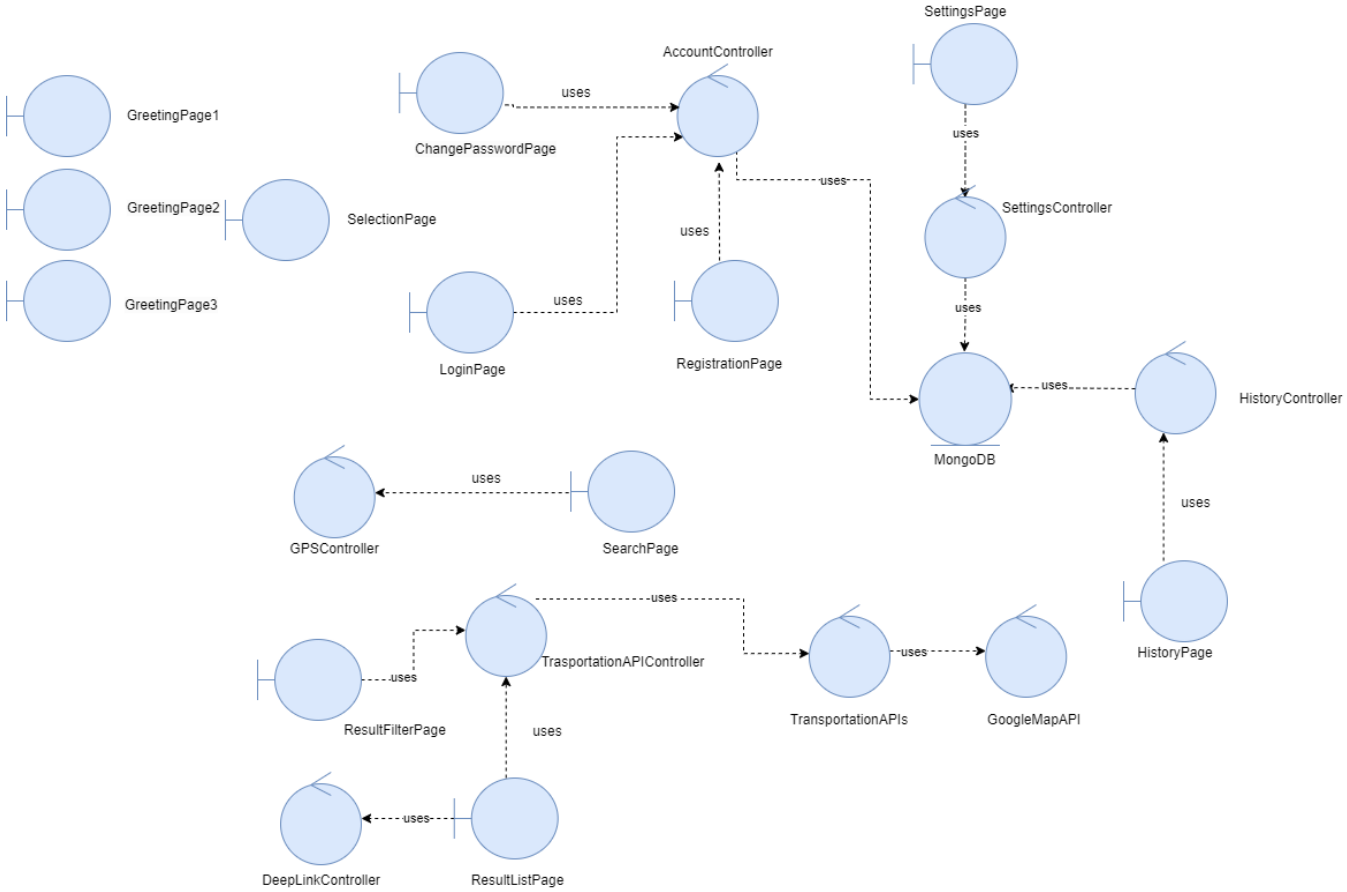
Actor:	User
Description:	User changes the password used in registering for an account to another password.
Preconditions:	Application must have a “Settings” icon in the interface and a “Change Password” button in the Settings page.
Post-conditions:	-NIL-
Flow of Events:	1. User clicks on the Settings icon. 2. Application directs the User to Settings page. 3. User clicks on the “Change Password” button. 4. Application directs the User to the Change Password page. 5. User enters the email address used for the account. 6. User enters the current old password used for the account. 7. User enters a new password, which must be of 12-18 alphanumeric characters, for the account. 8. Application validates if the User has entered a valid email address, correct old password and valid new password. 9. If the User has entered the old password correctly, the application prompts the User that the password has been changed successfully.
Alternative Flows:	18.AC.1 User clicks on the “Back” button. 18.AC.2 The application directs the User back to the Settings page.
Exceptions:	18.EX.1 If the email address entered does not exist in the database, the application displays an error message and prompts the user to enter another email address. 18.EX.2 If the User has entered the old password incorrectly, the application prompts the User that the User has entered the old password incorrectly. 18.EX.3 If the User has provided an invalid new password, the application highlights the input text box in red and displays an “Invalid Password” message. 18.EX.4 If the new password does not meet the required format, the application displays an error message and prompts the User to try again

	with a password that meets the requirements. 18.EX.5 If the new password the user entered is the same as the old password, the application must reject it, and display an error message stating "New password cannot be the same as old password".
Assumptions:	User has a pre-existing account with the application.
Priority:	Medium

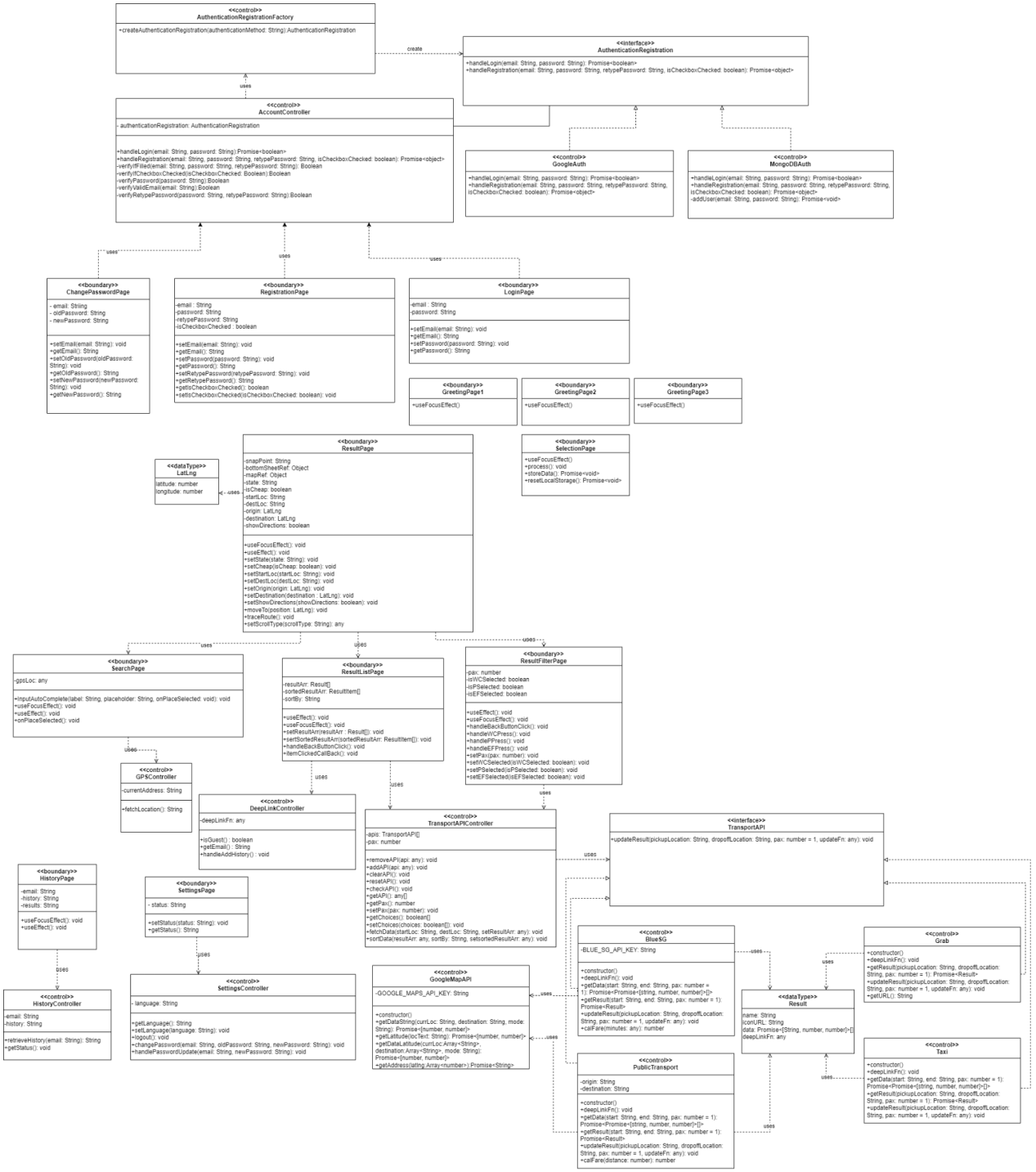
Use Case ID:	19		
Use Case Name:	Logout		
Created By:	Ng Woon Yee	Last Updated By:	Ng Woon Yee
Date Created:	22/2/2023	Date Last Updated:	13/4/2023

Actor:	User
Description:	User ends the session by logging out of their account.
Preconditions:	Users must have logged into their own account within the application.
Post-conditions:	-NIL-
Flow of Events:	1. User clicks on the Settings icon. 2. Application directs the User to Settings page. 3. User clicks on the “Log Out” button. 4. User logouts and will be redirected to Selection Page.
Alternative Flows:	-NIL-
Exceptions:	-NIL-
Assumptions:	User has a pre-existing account with the application, and the user is logged in.
Priority:	Medium

6.4 Key Boundary Classes & Control Classes

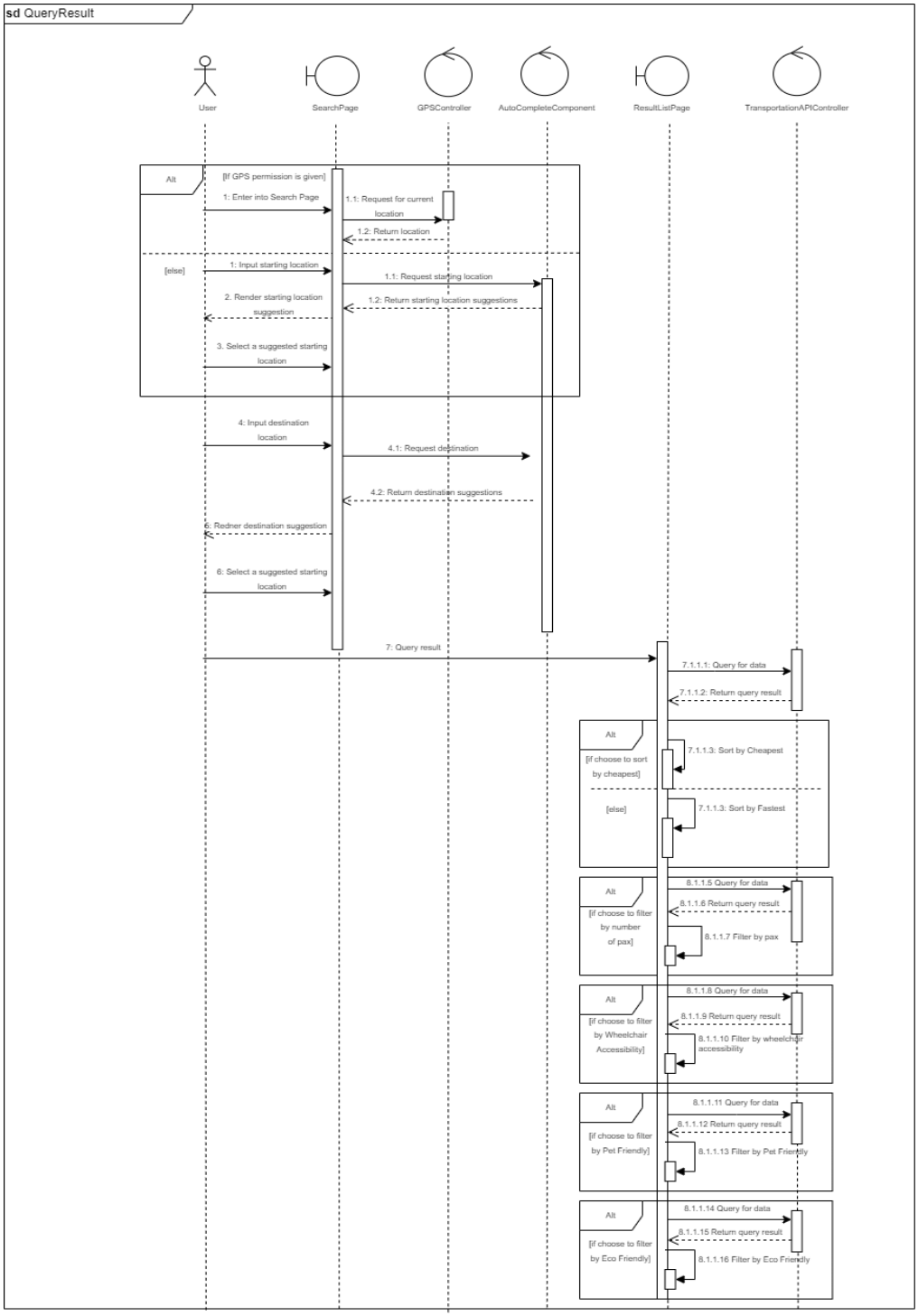


6.5 Class Diagram



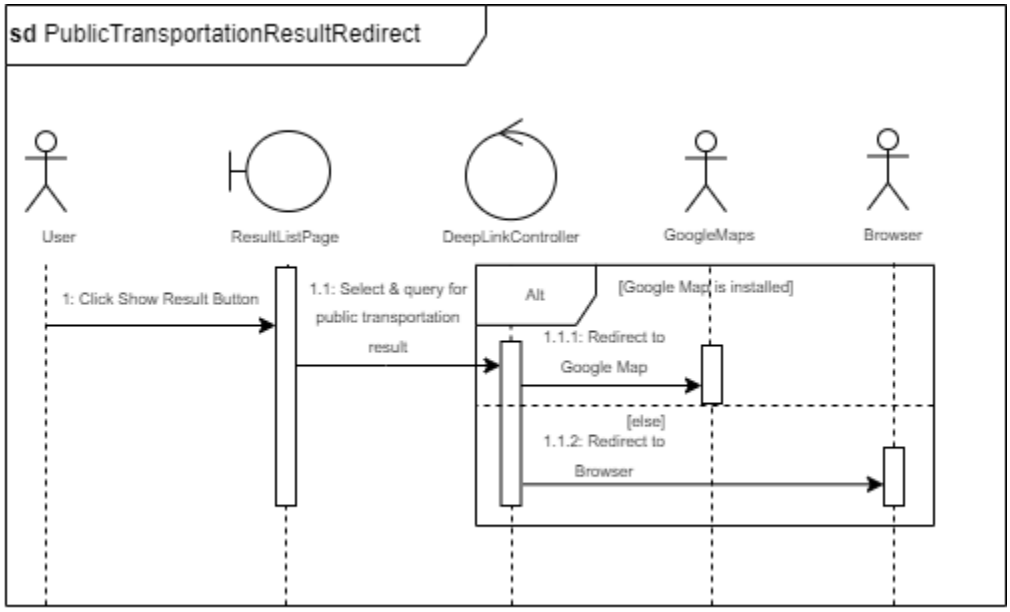
6.6 Sequence Diagrams

6.6.1 Query Result

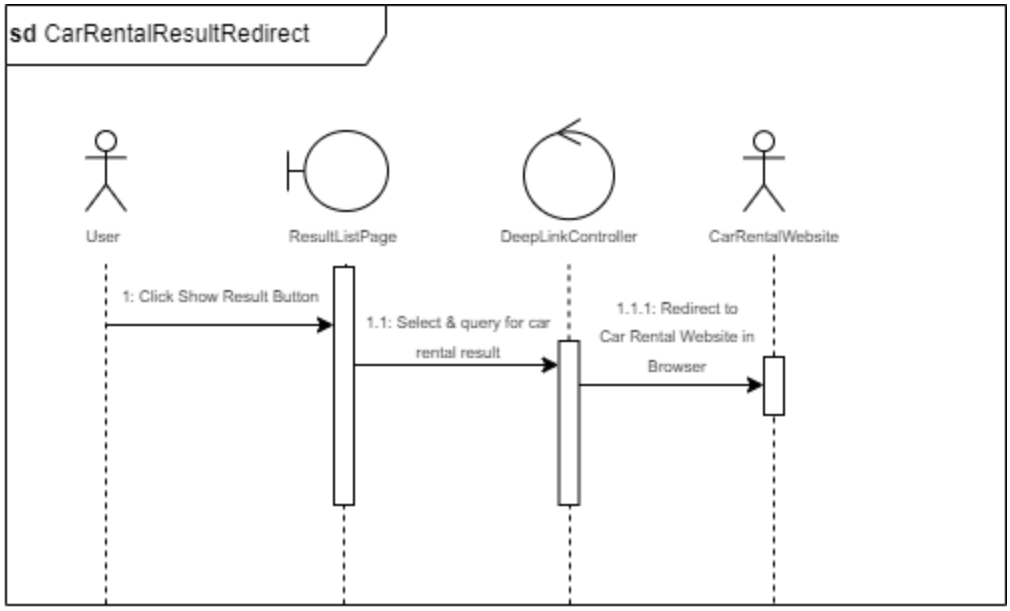


6.6.2 Redirecting to Relevant Transportation Applications

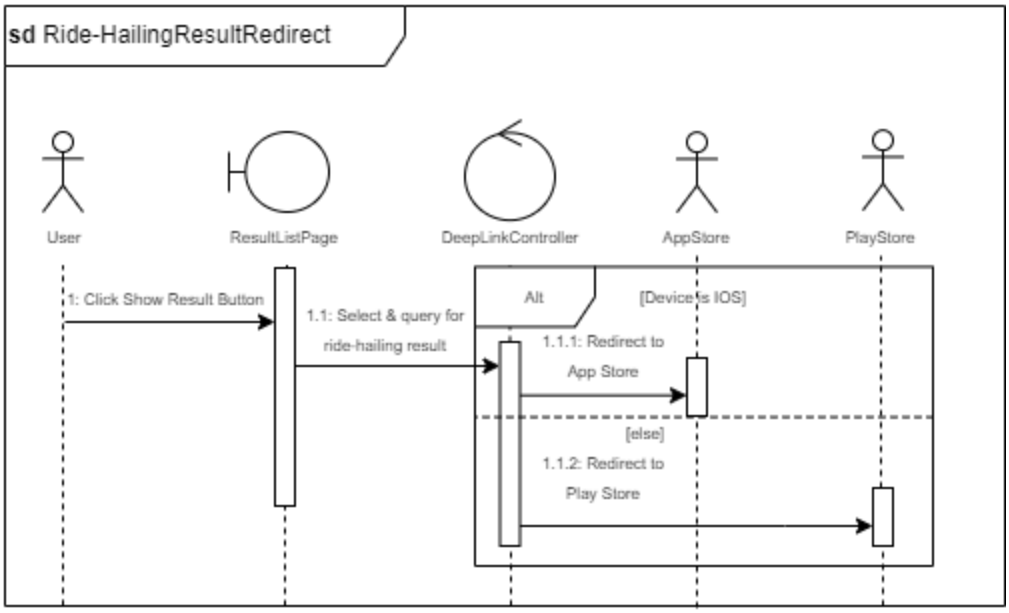
6.6.2.1 Public Transportation Result Redirect



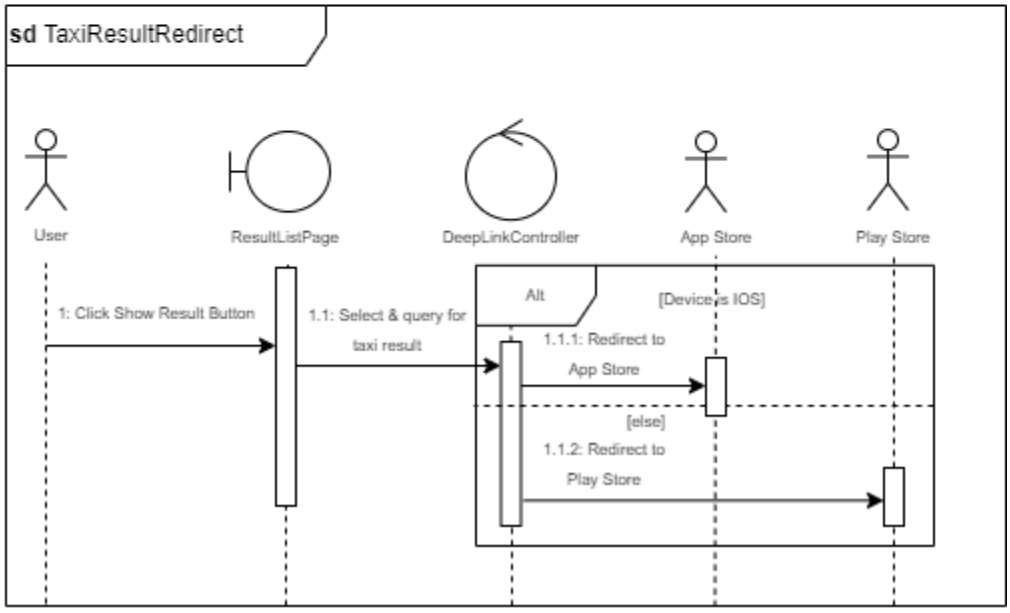
6.6.2.2 Car Rental Result Redirect



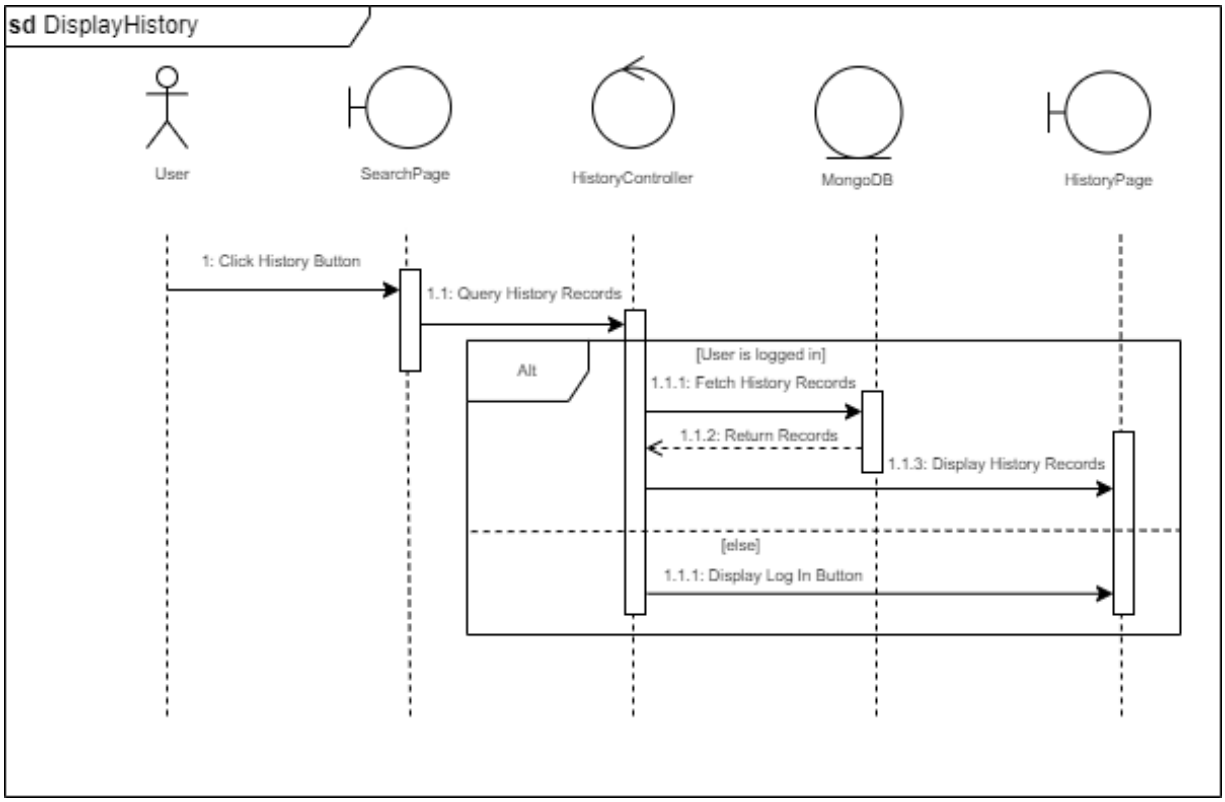
6.6.2.3 Ride-Hailing Result Redirect



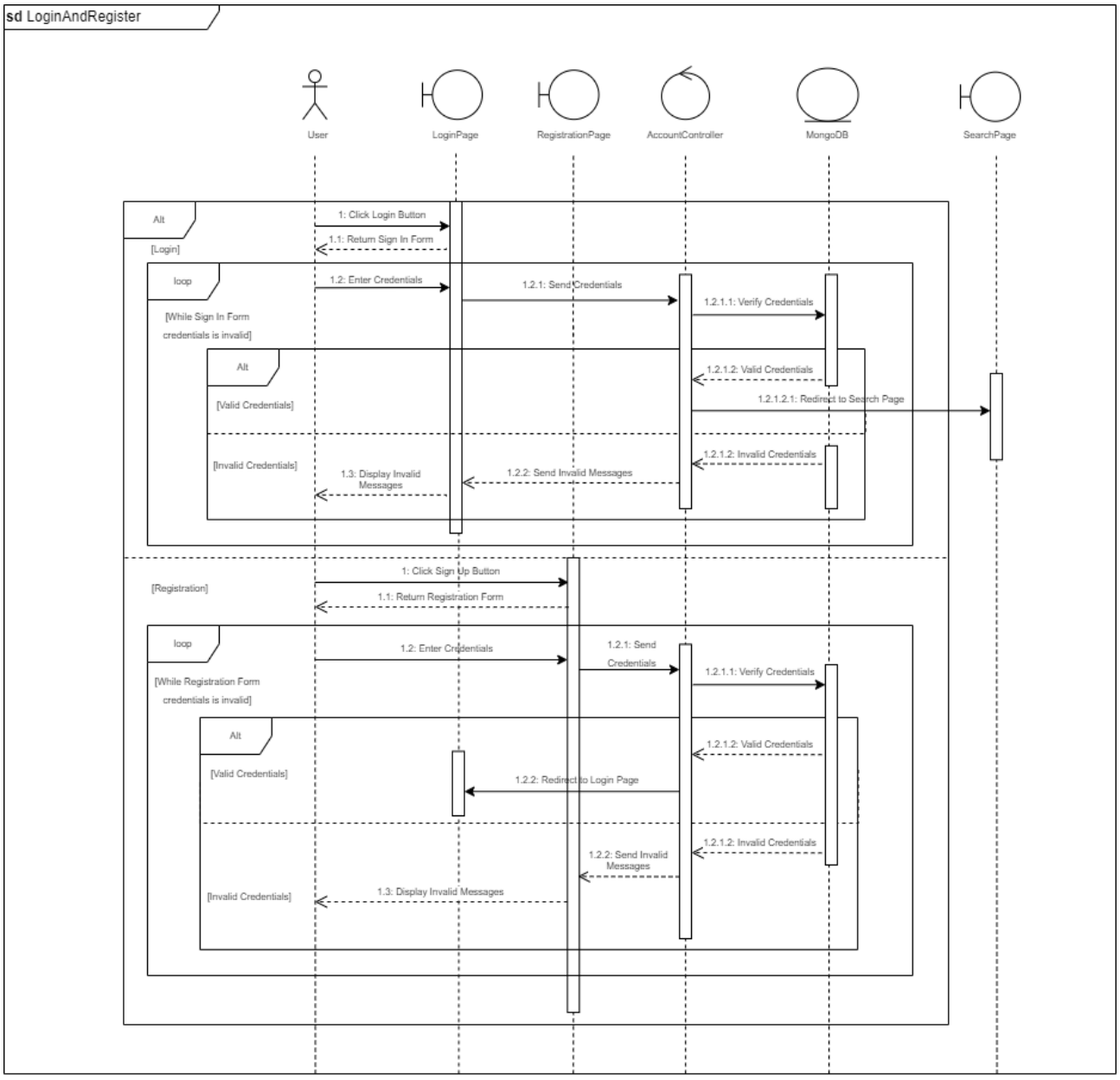
6.6.2.4 Taxi Result Redirect



6.6.3 Display History

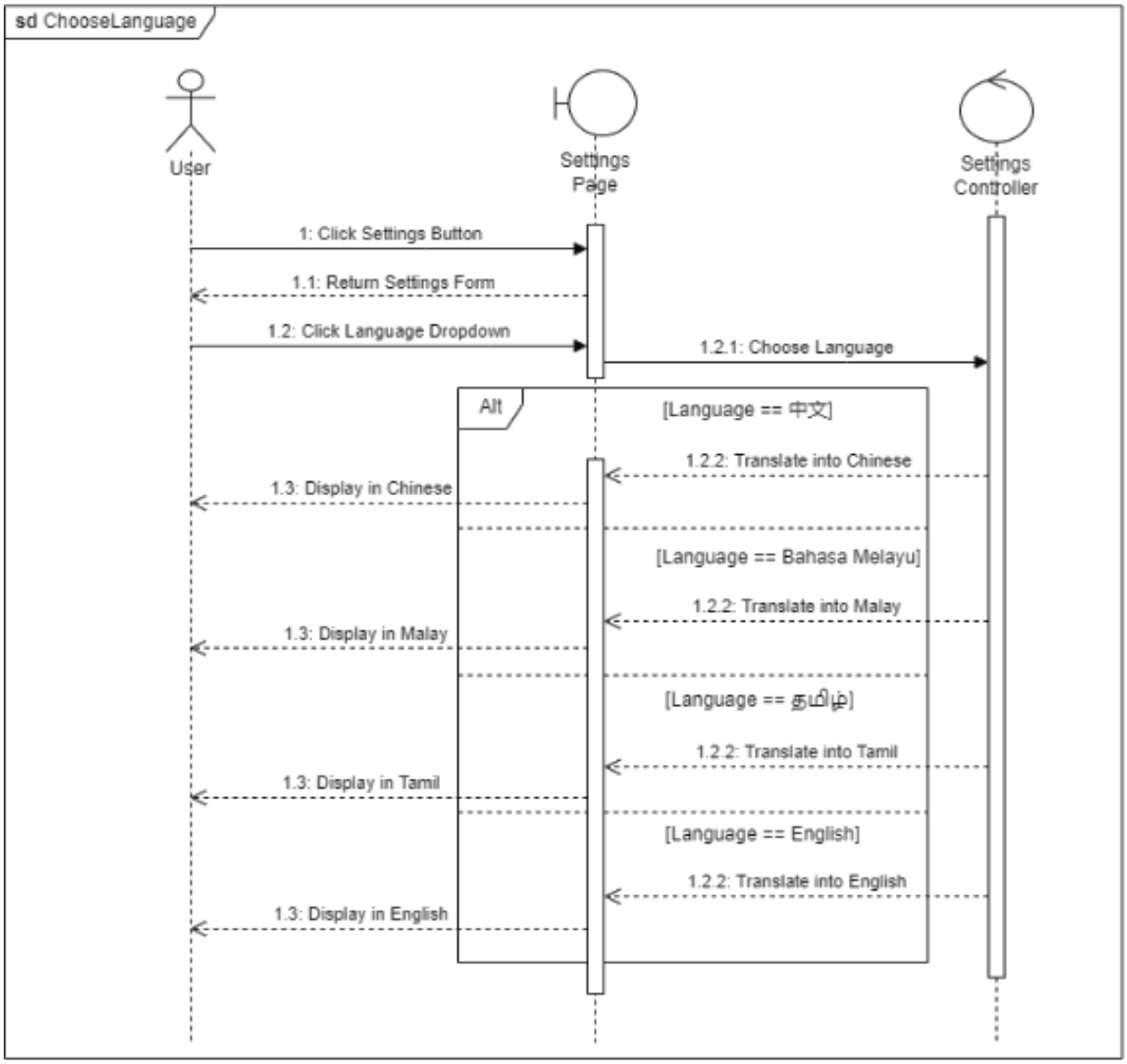


6.6.4 Login and Register

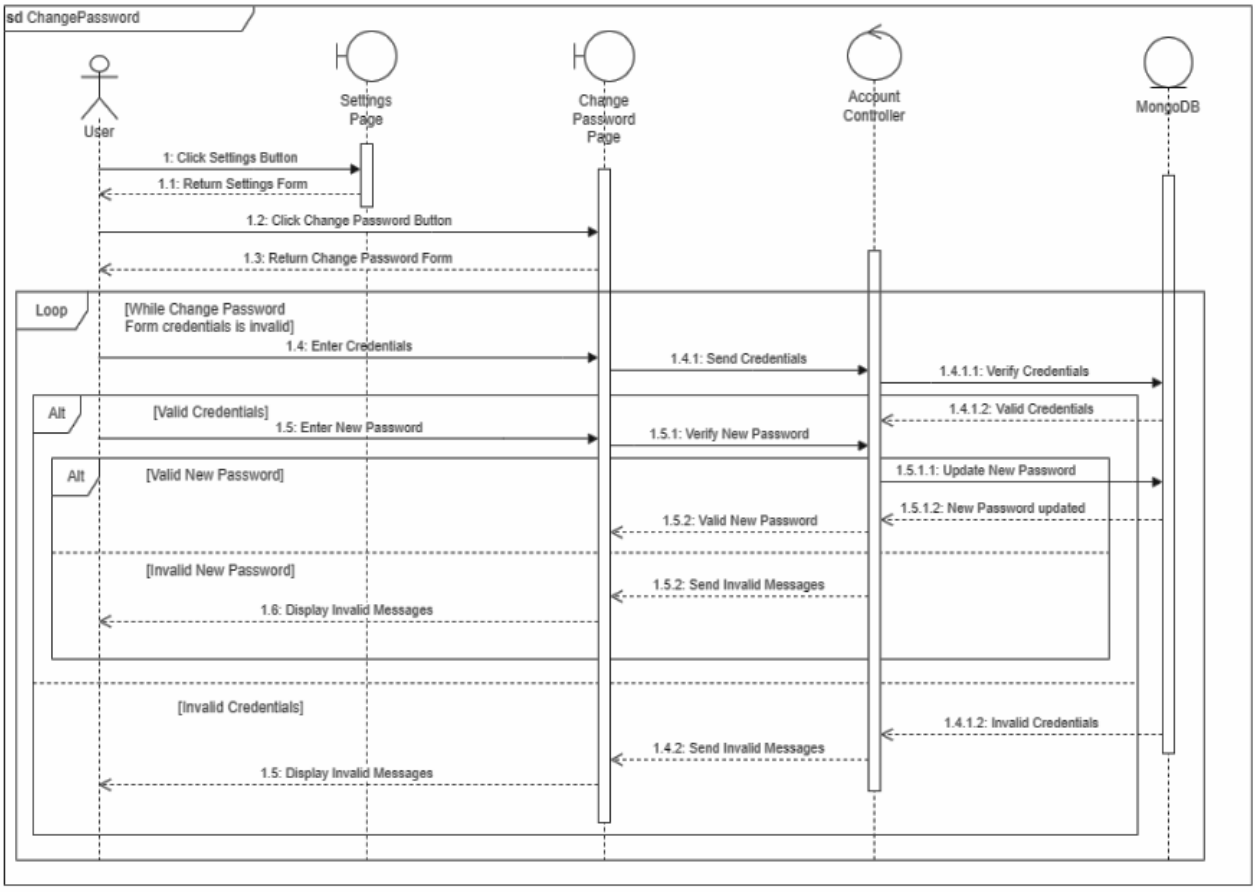


6.6.5 Settings

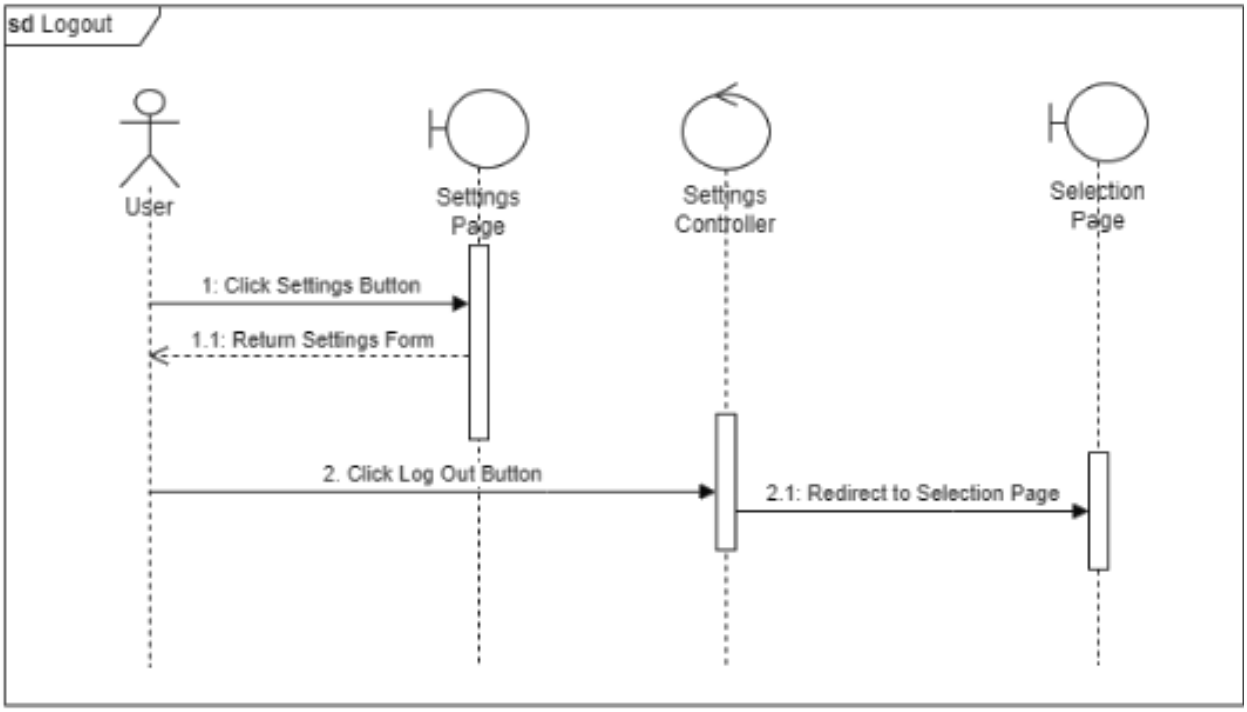
6.6.5.1 Choose Language



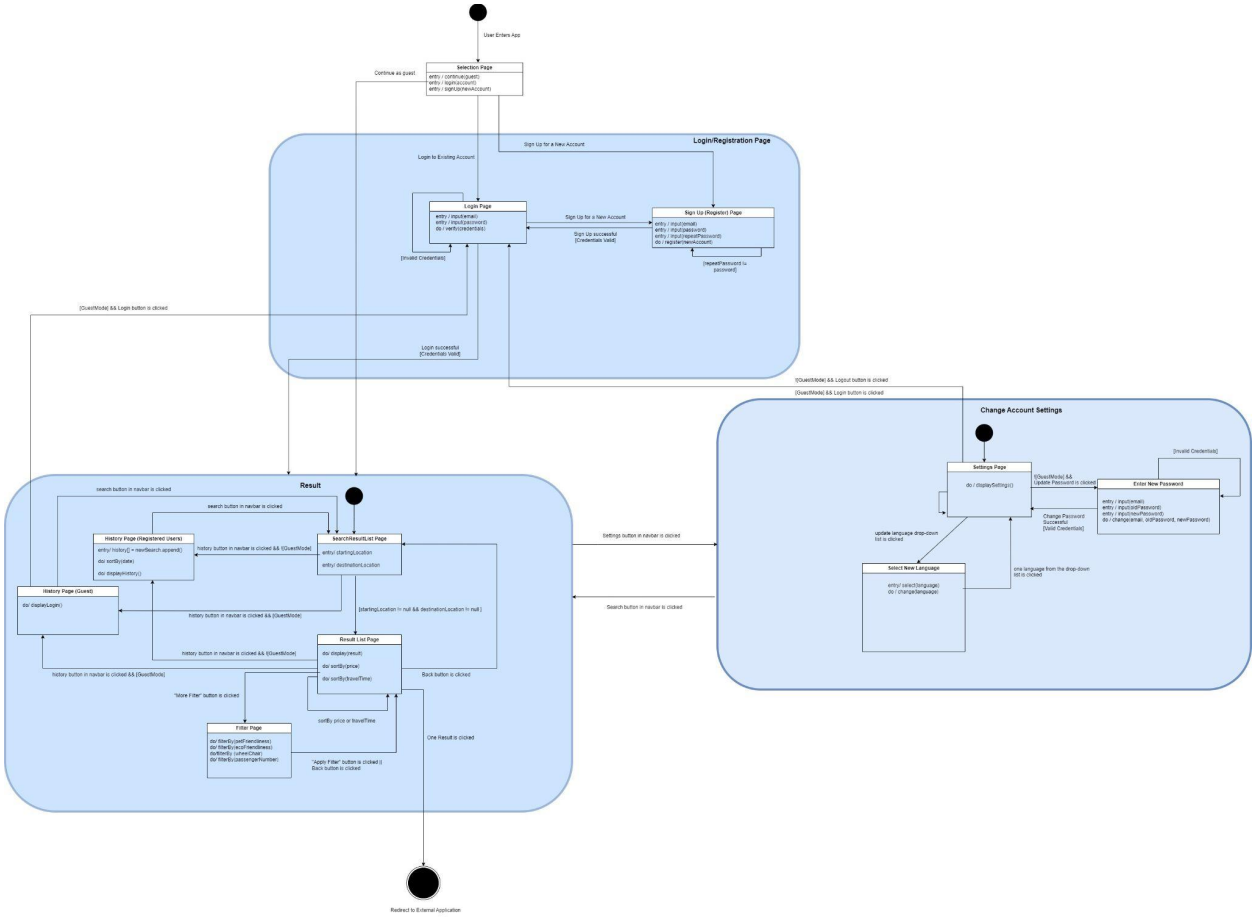
6.6.5.2 Change Password



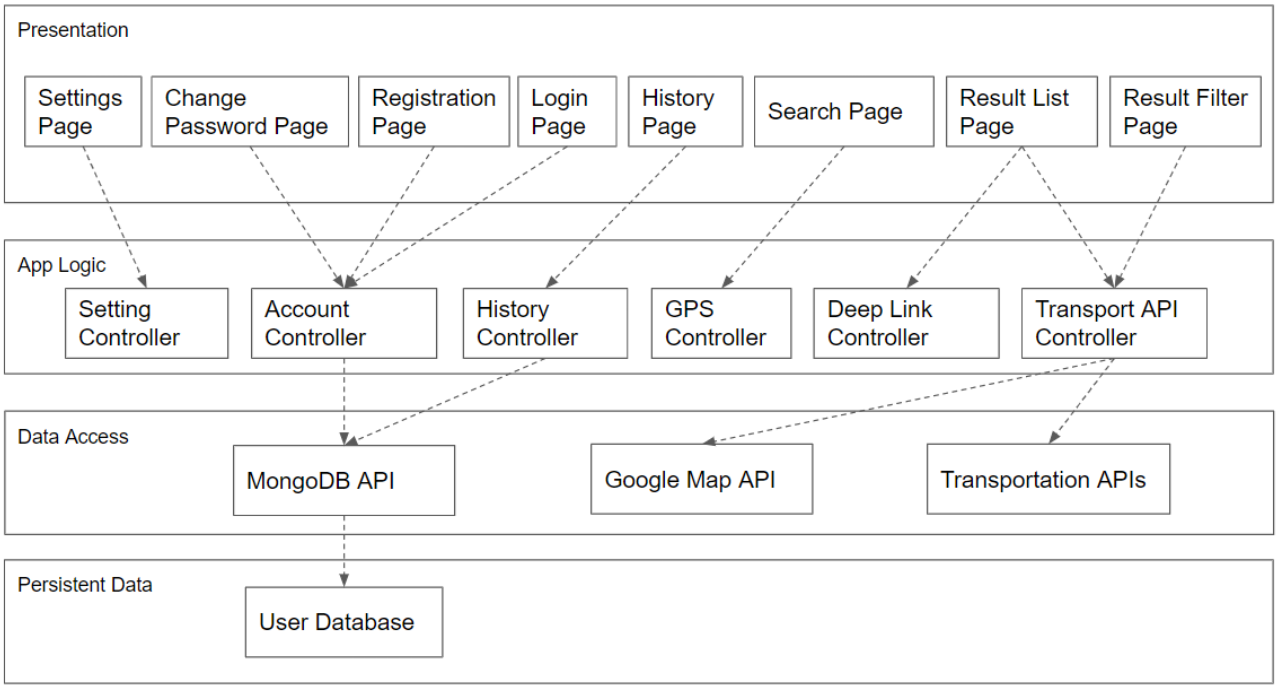
6.6.5.3 Logout


sd Logout
User
Settings Page
Settings Controller
Selection Page
1: Click Settings Button
1.1: Return Settings Form
2. Click Log Out Button
2.1: Redirect to Selection Page

6.7 Dialog Map



6.8 System Architecture



6.9 Testing

6.9.1 Black Box Testing

Test Case 1: Login Function

Test scenario	Login to the system	**Test case ID**	login
Test case description	Invalid inputs result in unsuccessful login.	**Test priority**	High
Prerequisite	Email should have been registered before.		

#	Description	Action	Inputs	Expected Output	Actual Output	Test Result
1	Users manage to log in successfully	1. In Selection Page, users press on "Login". 2. In Login Page, users type in his/her email and password. 3. Users press the "Log In" button.	email: jerry@gmail.com password: correct_password123	Users will be redirected to the search page.	Users are redirected to the search page.	Pass
2	Users typed in the wrong password	1. In Selection Page, users press on "Login". 2. In Login Page, users type in his/her email and password. 3. Users press the "Log In" button.	email: jerry@gmail.com password: wrong_password123	Users will be displayed a message box that says "Email or password is incorrect".	Users are being displayed a message box that says "Email or password is incorrect".	Pass
3	Users typed in the wrong email address	1. In Selection Page, users press on "Login". 2. In Login Page, users type in his/her email and password. 3. Users press the "Log In" button.	email: wrongEmail@gmail.com password: correct_password123	Users will be displayed a message box that says "Email or password is incorrect".	Users are being displayed a message box that says "Email or password is incorrect".	Pass

4	There is missing field for the inputs	1. In Selection Page, users press on "Login". 2. Users press the "Log In" button.	-	Users will be displayed a message box that says "Email or password is incorrect".	Users is being displayed a message box that says "Email or password is incorrect".	Pass

Test Case 2: Register Function

Test scenario	Register for account	**Test case ID**	register
Test case description	Invalid inputs result in unsuccessful registration.	**Test priority**	High
Prerequisite	Email should not have been registered before.		

#	Description	Action	Inputs	Expected Output	Actual Output	Test Result
1	Users manage to register successfully	1. In Selection Page, users press on "Register". 2. In Register page, users fill up the inputs accordingly - email address, password and retype password. 3. Users check the checkbox "Agree to Terms and Conditions". 4. Users press the "Sign up" button.	email: jerry@gmail.com password: correct_password123 retype password: correct_password123	User will be presented with a message box that says "You have successfully registered with us".	User is presented with a message box that says "You have successfully registered with us".	Pass
2	User's password does not match	1. In Selection Page, users press on "Register". 2. In Register page, users fill up the inputs accordingly - email address, password and retype password. 3. Users check the checkbox "Agree to Terms and Conditions". 4. Users press the "Sign up" button.	email: jerry@gmail.com password: correct_password123 retype password: different_password123	User will be presented with a message box that says "Passwords do not match".	User is presented with a message box that says "Passwords do not match".	Pass
3	Users type in invalid email	1. In Selection Page, users press on "Register". 2. In Register page, users fill up the inputs accordingly - email address, password and retype password.	email: jerry password: correct_password123 retype password: correct_password1	User will be presented with a message box that says "Invalid email format".	User is presented with a message box that says "Invalid email format".	Pass

		3. Users check the checkbox "Agree to Terms and Conditions". 4. Users press the "Sign up" button.	23			
4	Users did not agree to the terms and conditions	1. In Selection Page, users press on "Register". 2. In Register page, users fill up the inputs accordingly - email address, password and retype password. 3. Users press the "Sign up" button.	email: jerry@gmail.com password: correct_password123 retype password: correct_password123	User will be presented with a message box that says "Please agree to the terms and conditions".	User is presented with a message box that says "Please agree to the terms and conditions".	Pass
5	Email has been registered before	1. In Selection Page, users press on "Register". 2. In Register page, users fill up the inputs accordingly - email address, password and retype password. 3. Users check the checkbox "Agree to Terms and Conditions". 4. Users press the "Sign up" button.	email: jerry@gmail.com password: correct_password123 retype password: correct_password123	User will be presented with a message box that says "You have already registered with us before".	User is presented with a message box that says "You have already registered with us before".	Pass
6	Users have missing inputs (e.g. did not type in email or password)	1. In Selection Page, users press on "Register". 2. Users check the checkbox "Agree to Terms and Conditions". 3. Users press the "Sign up" button.	-	User will be presented with a message box that says "Please fill in all the fields".	User is presented with a message box that says "Please fill in all the fields".	Pass

Test Case 3: Language Changing

Test scenario	Change language	**Test case ID**	lang
Test case description	If no preferred language is chosen, the default language of the application interface is English.	**Test priority**	Medium
Prerequisite	User has not selected a preferred language for the contents of the application.		

#	Description	Action	Inputs	Expected Output	Actual Output	Test Result
1	Users did not select a preferred language.	1. In Settings Page, users press on the dropdown menu. 2. From the dropdown menu, users do not press on any of the dropdown buttons.	-	Contents of application will be displayed in English language.	Contents of application are displayed in English language.	Pass
2	Users set English as their preferred language.	1. In Settings Page, users press on the dropdown menu. 2. From the dropdown menu, users press the "English" dropdown button.	Language: English	Contents of application will be displayed in English language.	Contents of application are displayed in English language.	Pass
3	Users set Chinese as their preferred language.	1. In Settings Page, users press on the dropdown menu. 2. From the dropdown menu, users press the "中文" dropdown button.	Language: 中文	Contents of application will be displayed in English language.	Contents of application are displayed in English language.	Pass

4	Users set Malay as their preferred language.	1. In Settings Page, users press on the dropdown menu. 2. From the dropdown menu, users press the "Bahasa Melayu" dropdown button.	Language: Bahasa Melayu	Contents of application will be displayed in English language.	Contents of application are displayed in English language.	Pass
5	Users set Tamil as their preferred language.	1. In Settings Page, users press on the dropdown menu. 2. From the dropdown menu, users press the "தமிழ்" dropdown button.	Language: தமிழ்	Contents of application will be displayed in English language.	Contents of application are displayed in English language.	Pass

Test Case 4: Location Searching

Test scenario	Searching for start and end locations	**Test case ID**	Search a location function
Test case description	Invalid inputs result in unsuccessful search.	**Test priority**	High
Prerequisite	Location should be valid and within Singapore.		

#	Description	Action	Inputs	Expected Output	Actual Output	Test Result
1	Users manage to search and Display results successfully	1. In Search Page, users press on "Allow location usage" to allow automatic input of origin location. 2. In Search page, users fill up the address inputs accordingly & accurately – destination address. 3. Users check and click on their respective address from the dropdown Google suggested List. 4. Users press "Show Result" button to display the Results List in the default Cheapest Fares using the sortByCheapestPrice() function.	GPS permission: Allow Destination input: Marina Bay Sands clickOn: Marina Bay Sands, 10 Bayfront Ave, Singapore 018956 clickOn: "Show Result" Button	User will be presented with a result list that says "Cheapest"/"Fastest" and displays all results by the cheapest fares, with the option to check the fastest fare times as well.	User is presented with a result list that says "Cheapest"/"Fastest" and displays all results by the cheapest fares, with the option to check the fastest fare times as well.	Pass
2	User does not allow usage of GPS permissions	1. After being redirected to search page, users press on “Do Not Allow location usage”	GPS permission: DO NOT Allow	The origin will not be pre-filled with the user’s	The origin is not pre-filled with the user’s current	Pass

				current location.	location.	
3	User does not input an alphanumeric address in the input Text Box	1. In Search Page, users press on "Allow location usage" to allow automatic input of origin location. 2. In Search page, users fill up the address inputs with a non-alphanumeric address.	GPS permission: Allow Destination input: MBS🌈🎉🐾💻🚀😎Sg	User will be NOT Allowed to continue to the result list page and not allowed to click on the "Show Result" button.	User is NOT Allowed to continue to the result list page and not allowed to click on the "Show Result" button.	Pass
4	User did not select a valid address from the dropdown google maps suggested addresses list or the address inputted is not in the suggested dropdown list	1. In Search Page, users press on "Allow location usage" to allow automatic input of origin location. 2. In Search page, users fill up the address inputs accordingly & accurately – destination address. 3. Users do not check and do not click on their respective address from the dropdown Google suggested List or the search query inputted is not suggested by the Google maps dropdown address suggestion list. 4. Users try pressing the "Show Result" button to display Results List but cannot progress to the Results List Page.	GPS permission: Allow Destination input: Maria Boy Sonds clickOn: <NIL> clickOn: "Show Result" Button	User will be NOT Allowed to click on the "Show Result" button and not allowed to continue to the result list page.	User is NOT Allowed to click on the "Show Result" button and not allowed to continue to the result list page.	Pass

5	User's inputted origin or end addresses are not set properly or not able to get validated	1. In Search Page, users press on "Allow location usage" to allow automatic input of origin location. 2. In Search page, users fill up the address inputs accordingly – destination address. 3. Users check and click on their respective address from the dropdown Google suggested List. 4. Users try pressing on the "Show Result" button.	GPS permission: Allow Destination input: Marina BKJKay Sands, Singapore 999999 clickOn: <NIL> clickOn: "Show Result" Button	User will be NOT Allowed to click on the "Show Result" button and not allowed to continue to the result list page.	User is NOT Allowed to click on the "Show Result" button and not allowed to continue to the result list page.	Pass

Test Case 5: Filter Function

Test scenario	Filter the results	**Test case ID**	filter
Test case description	Applying a filter that can customized the result	**Test priority**	High
Prerequisite	Search result must be fetched from APIs successfully		

#	Description	Action	Inputs	Expected Output	Actual Output	Test Result
1	Users increase the number of passengers to 2	1. After user has pressed "Show Result" and the results are being shown on the result list page, user presses on the filter button. 2. User increases the number of passengers to 2. 3. User presses the “Apply Filter” button.	Number of Passengers: 2	The fares at the result page should be adjusted accordingly.	The fares at the result page are being adjusted accordingly.	Pass
2	Users select the "Wheelchair Accessibility" filter	1. After user has pressed "Show Result" and the results are being shown on the result list page, user presses on the filter button. 2. User presses the "Wheelchair Accessibility" option under Ride Types. 3. User presses the “Apply Filter” button.	Ride type: Wheelchair accessibility	The results should show only services that support wheelchair accessibility now.	The results show only services that support wheelchair accessibility.	Pass
3	Users select "Pet-friendly" filter	1. After user has pressed "Show Result" and the results are being shown on the result list page, user presses	Ride type: Pet-friendly	The results should show only show services that are pet friendly now.	The results only show services that are pet friendly now.	Pass

		on the filter button. 2. User press on the "Pet-friendly" option under Ride Types. 3. User presses “Apply Filter” button.				
4	Users select "Eco-Friendly" filter	1. After user has pressed "Show Result" and the results are being shown on the result list page, user presses on the filter button. 2. User presses on the "Eco-Friendly" option under Ride Types. 3. User presses the “Apply Filter” button.	Ride type: Eco-friendly	The results should only show services that are eco-friendly now.	The results only show services that are pet friendly now.	Pass
5	Users set the filter but forgot to press on “Apply Filter”	1. After user has pressed "Show Result" and the results are being shown on the result list page, user presses on the filter button. 2. User presses on the "Eco-Friendly" option under Ride Types.	Ride type: Eco-friendly	The results should be the same as the previous result that has no filter applied on it.	The result is the same as the previous result that has no filter applied on it.	Pass

Test Case 6: Deep-Link

Test scenario	Deep Link to other apps or websites	**Test case ID**	deep link
Test case description	To check if is able to redirect to other apps or websites	**Test priority**	Medium
Prerequisite	Results have to be returned from APIs successfully		

#	Description	Action	Inputs	Expected Output	Actual Output	Test Result
1	Deep link on grab	1. User presses on a service from "Grab".	-	User will be redirected to the Grab page in either itunes store or google play store depending on the operating system.	User is redirected to the Grab page in either itunes store or google play store depending on the operating system.	Pass
2	Deep link on Comfort Delgro	1. User presses on a service from "Comfort Delgro".	-	User will be redirected to the Comfort Delgro page in either itunes store or google play store depending on the operating system.	User is redirected to the Comfort Delgro page in either itunes store or google play store depending on the operating system.	Pass
3	Deep link on public transport	1. User presses on "Public Transport".	-	User will be redirected to the Google Map with the starting locations and destination pre-filled.	User is redirected to the Google Map with the starting locations and destination pre-filled.	Pass

4	Deep link on BlueSG	1. User presses on "BlueSG".	-	User will be redirected to the BlueSG website that shows the availability of cars in Singapore.	User is redirected to the BlueSG website that shows the availability of cars in Singapore.	Pass
5	Deep link on public transport (However user does not have Google Map installed)	1. User press on "Public Transport".	-	User will be redirected to the Google Map website on a browser with the starting locations and destination pre-filled.	User is redirected to the Google Map website on a browser with the starting locations and destination pre-filled.	Pass

Test Case 7: Change Password Function

Test scenario	Change password of an account	**Test case ID**	change password
Test case description	Invalid inputs result in unsuccessful changing of password.	**Test priority**	Medium
Prerequisite	Email should have been registered before.		

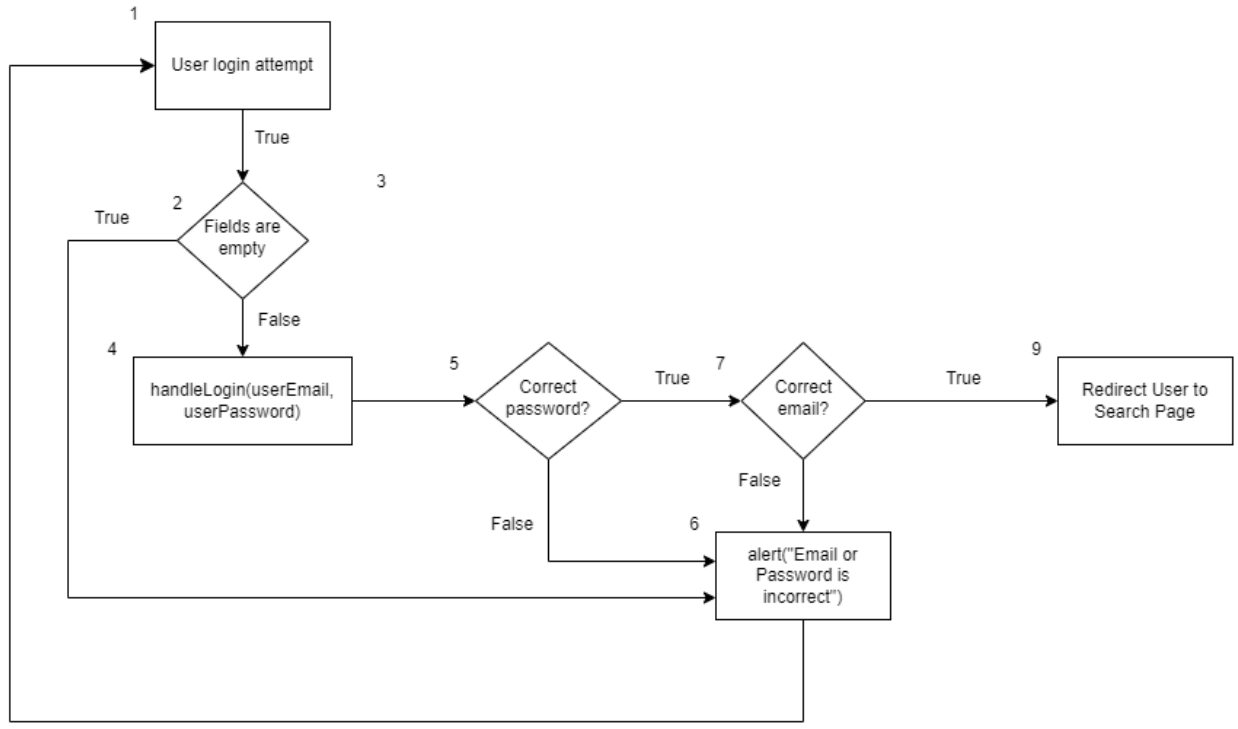
#	Description	Action	Inputs	Expected Output	Actual Output	Test Result
1	Users manage to update password successfully	1. In Settings Page, users press on "Change Password". 2. In Change Password page, users fill up the inputs accordingly - email address, old password and new password. 3. Users check the checkbox "Agree to Terms and Conditions". 4. Users press the "Change Password" button.	email: jerry@gmail.com Old password: correct_password123 New password: new_password123	User will be presented with a message box that says "Password Successfully Updated".	User is presented with a message box that says "Password Successfully Updated".	Pass
2	User's old password input does not match password from database	1. In Settings Page, users press on "Change Password". 2. In Change Password page, users fill up the inputs accordingly - email address, old password and new password. 3. Users check the	email: jerry@gmail.com Old password: wrong_password123 New password: new_password123	User will be presented with a message box that says "Password is incorrect".	User is presented with a message box that says "Password is incorrect".	Pass

		checkbox "Agree to Terms and Conditions". 4. Users press the "Change Password" button.				
3	User's new password input is the same as old password	1. In Settings Page, users press on "Change Password". 2. In Change Password page, users fill up the inputs accordingly - email address, old password and new password. 3. Users check the checkbox "Agree to Terms and Conditions". 4. Users press the "Change Password" button.	email: jerry@gmail.com Old password: correct_password123 New password: correct_password123	User will be presented with a message box that says "New password cannot be the same as old password".	User is presented with a message box that says "New password cannot be the same as old password".	Pass
4	Users did not agree to the terms and conditions	1. In Settings Page, users press on "Change Password". 2. In Change Password page, users fill up the inputs accordingly - email address, old password and new password. 3. Users press the "Change Password" button.	email: jerry@gmail.com Old password: correct_password123 New password: new_password123	User will be presented with a message box that says "Please agree to the terms and conditions".	User is presented with a message box that says "Please agree to the terms and conditions".	Pass
5	Email has not been registered before	1. In Settings Page, users press on "Change Password". 2. In Change Password page, users fill up the	email: jerry123@gmail.com Old password: correct_password1	User will be presented with a message box that says "Email does not exist".	User is presented with a message box that says "Email does not exist".	Pass

		inputs accordingly - email address, old password and new password. 3. Users check the checkbox "Agree to Terms and Conditions". 4. Users press the "Change Password" button.	23 New password: new_password123			
6	Users have missing inputs (e.g. did not type in email or password)	1. In Settings Page, users press on "Change Password". 2. In Change Password page, users fill up the inputs accordingly - email address, old password and new password. 3. Users check the checkbox "Agree to Terms and Conditions". 4. Users press the "Change Password" button.	-	User will be presented with a message box that says "Please fill in all the fields".	User is presented with a message box that says "Please fill in all the fields".	Pass

6.9.2 White Box Testing

Test Case 1: Login Function



Basis Set of Path
Path 1: 1,2,4,5,7,9
Path 2: 1,2,6,1
Path 3: 1,2,4,5,6,1
Path 4: 1,2,4,5,7,6,1

Test Cases
Path 1: User is able to login into his/her account successfully.

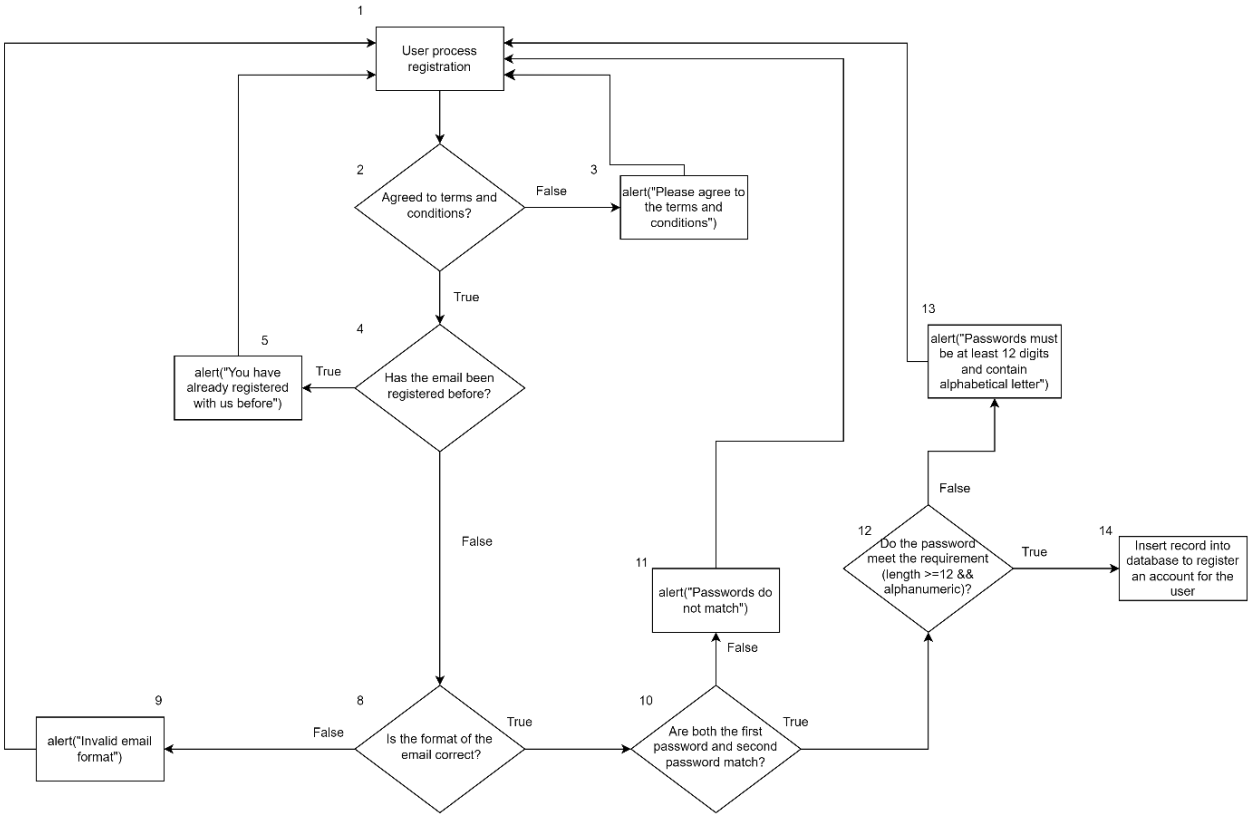
Path 2: User did not type in his/her account credentials.

Path 3: The account password entered does not match with that found in the application database.

Path 4: The account registered email entered does not match with that found in the application database.

Cyclomatic Complexity
|Decision Points| + 1 = 3 + 1 = 4

Test case 2: Register Function



Basis Set of Path

Path 1: 1,2,4,8,10,12,14
Path 2: 1,2,3,1
Path 3: 1,2,4,5,1
Path 4: 1,2,4,8,9,1
Path 5: 1,2,4,8,10,11,1
Path 6: 1,2,4,8,10,12,13,1

Test Cases

Path 1: User is able to register his/her records into the database successfully.

Path 2: User did not agree to the terms and conditions.

Path 3: The email has been registered before.

Path 4: Some of the inputs are currently left blank.

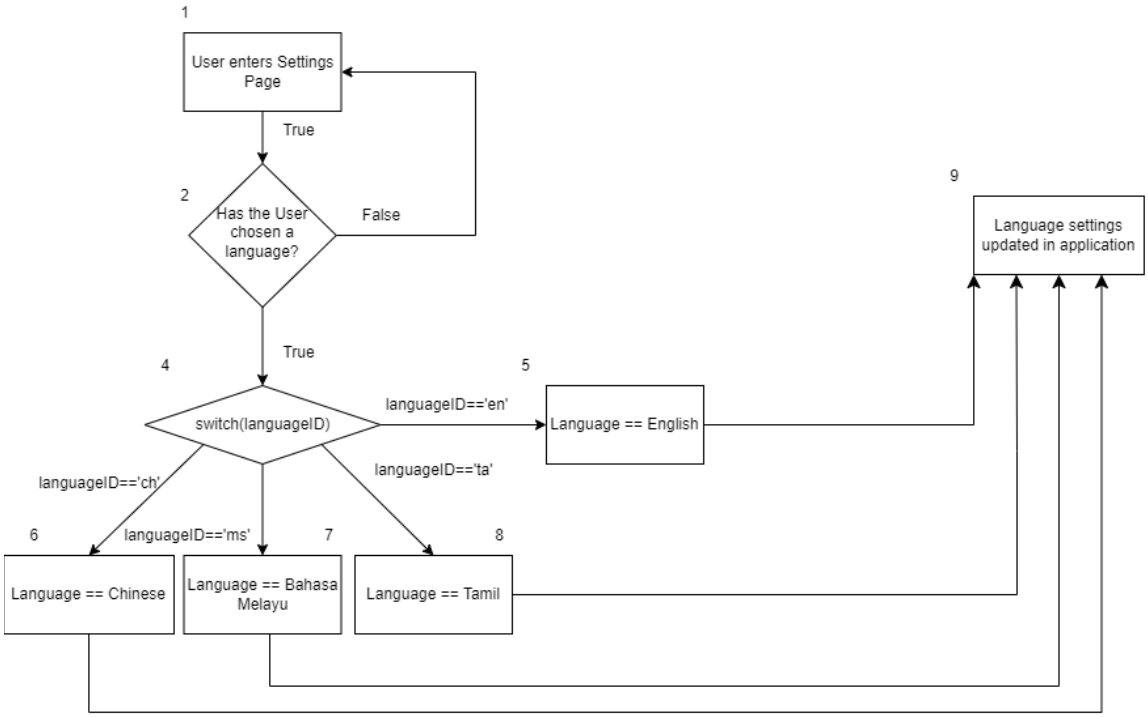
Path 5: The format of the email is incorrect.

Path 6: The first password is not the same as the second password.

Path 7: The password did not meet the requirement (length >= 12 && must be alphanumeric).

Cyclomatic Complexity
|Decision Points| + 1 = 5 + 1 = 6

Test Case 3: Language Changing



Basis Set of Path
Path 1: 1,2
Path 2: 1,2,4,5,9
Path 3: 1,2,4,6,9
Path 4: 1,2,4,7,9
Path 5: 1,2,4,8,9

Test Cases
Path 1: User did not set a preferred language.

Path 2: User sets English as his/her preferred language.

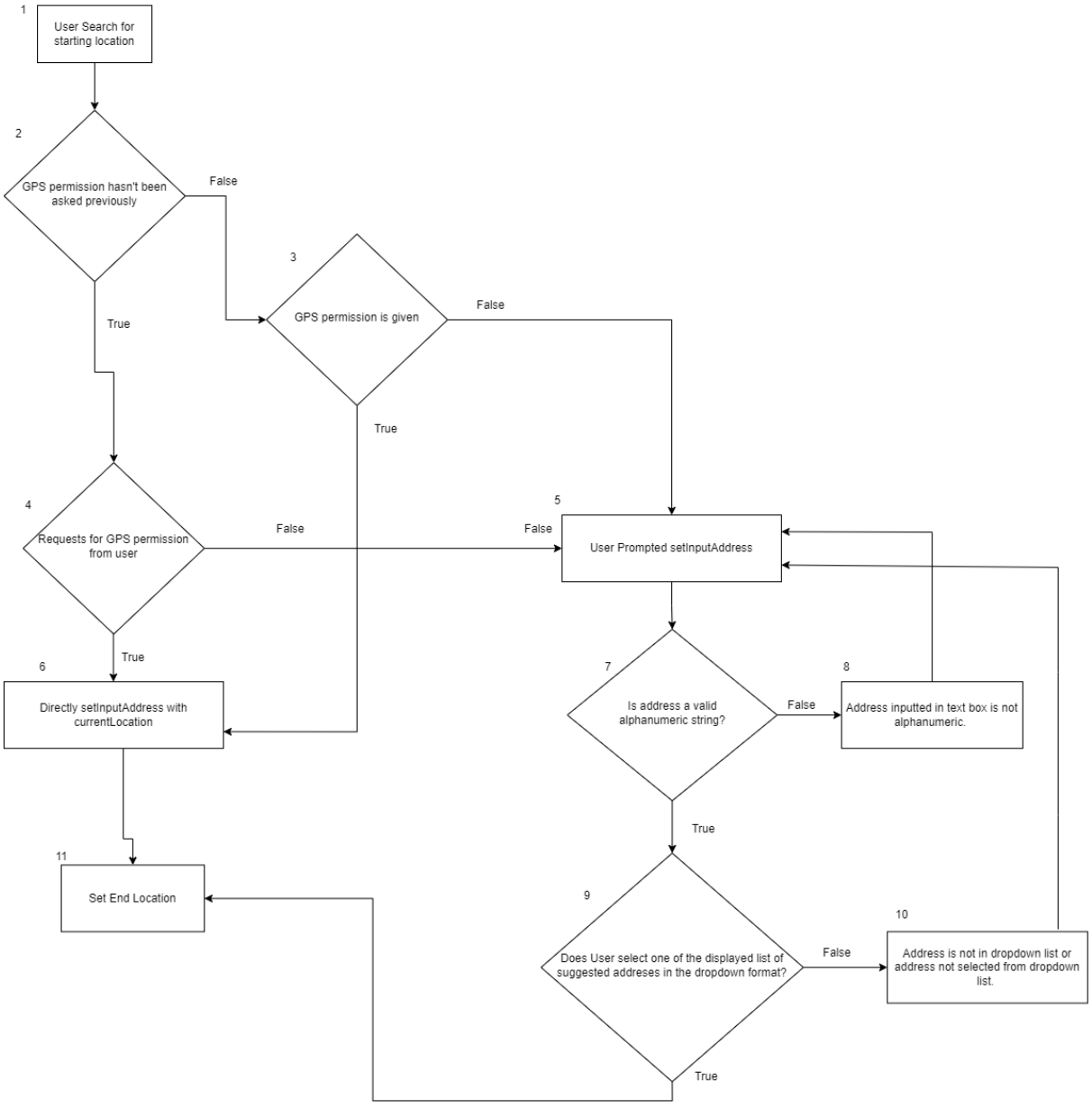
Path 3: User sets Chinese as his/her preferred language.

Path 4: User sets Bahasa Melayu as his/her preferred language.

Path 5: User sets Tamil as his/her preferred language.

Cyclomatic Complexity
|edges| = 11
|nodes| = 8
|edges| - |nodes| +2 = 11 - 8 + 2 = 5

Test Case 4: Starting Location Searching



Basis Set of Path

Path 1: 1,2,3,5,7,9,11
Path 2: 1,2,4,5,7,9,11
Path 3: 1,2,3,6,11
Path 4: 1,2,4,6,11
Path 5: 1,2,3,5,7,8,5
Path 6: 1,2,3,5,7,9,10,5

Test Cases

Path 1: User has been asked for GPS permission, then User doesn't give GPS permission. User type in address, address is alphanumeric, and User selects one of the displayed list of suggested addresses in the dropdown format.

Path 2: User hasn't been asked for GPS permission, then User doesn't give GPS permission upon request from the application. User type in address, address is alphanumeric, and User selects one of the displayed list of suggested addresses in the dropdown format.

Path 3: User has been asked for GPS permission, then User gives GPS permission. The application directly sets the starting address with the current location.

Path 4: User hasn’t been asked for GPS permission, then User gives GPS permission upon request from the application. The application directly sets the starting address with the current location.

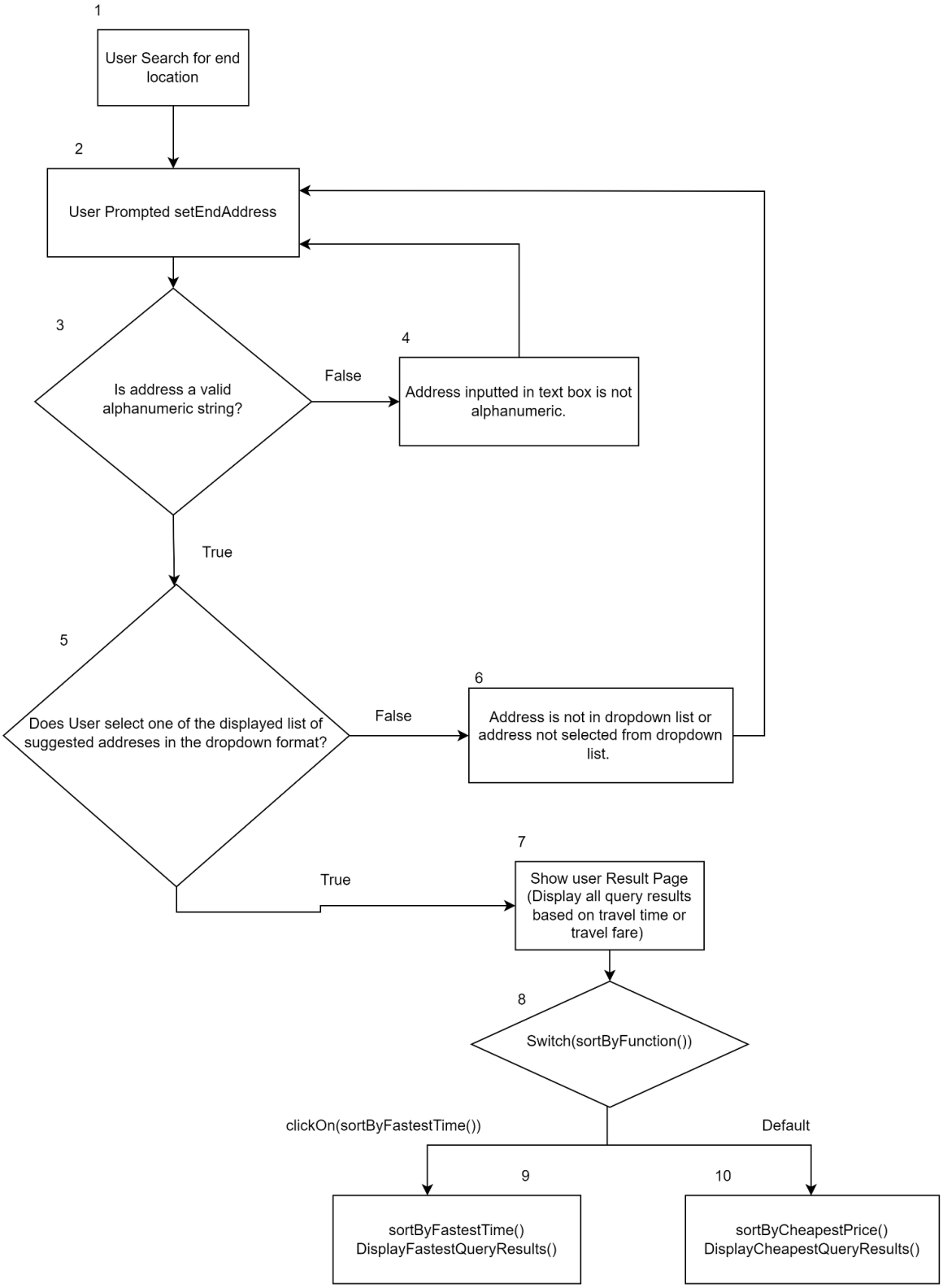
Path 5: User has been asked for GPS permission, then User doesn't give GPS permission. User type in address, address is not alphanumeric.

Path 6: User has been asked for GPS permission, then User doesn't give GPS permission. User type in address, address is alphanumeric, and User doesn't select one of the displayed list of suggested addresses in the dropdown format.

Cyclomatic Complexity

|Decision Points| + 1 = 5 + 1 = 6

Test Case 5: Destination Location Searching



Basis Set of Path
Path 1: 1,2,3,5,7,9
Path 2: 1,2,3,4,2
Path 3: 1,2,3,4,5,6,2
Path 4: 1,2,3,5,7,8,10

Test Cases
Path 1: User prompted to input end address. End address is alphanumeric, address is selected from the dropdown list. The application will show the Result Page, by default it will order it in Fastest Time.

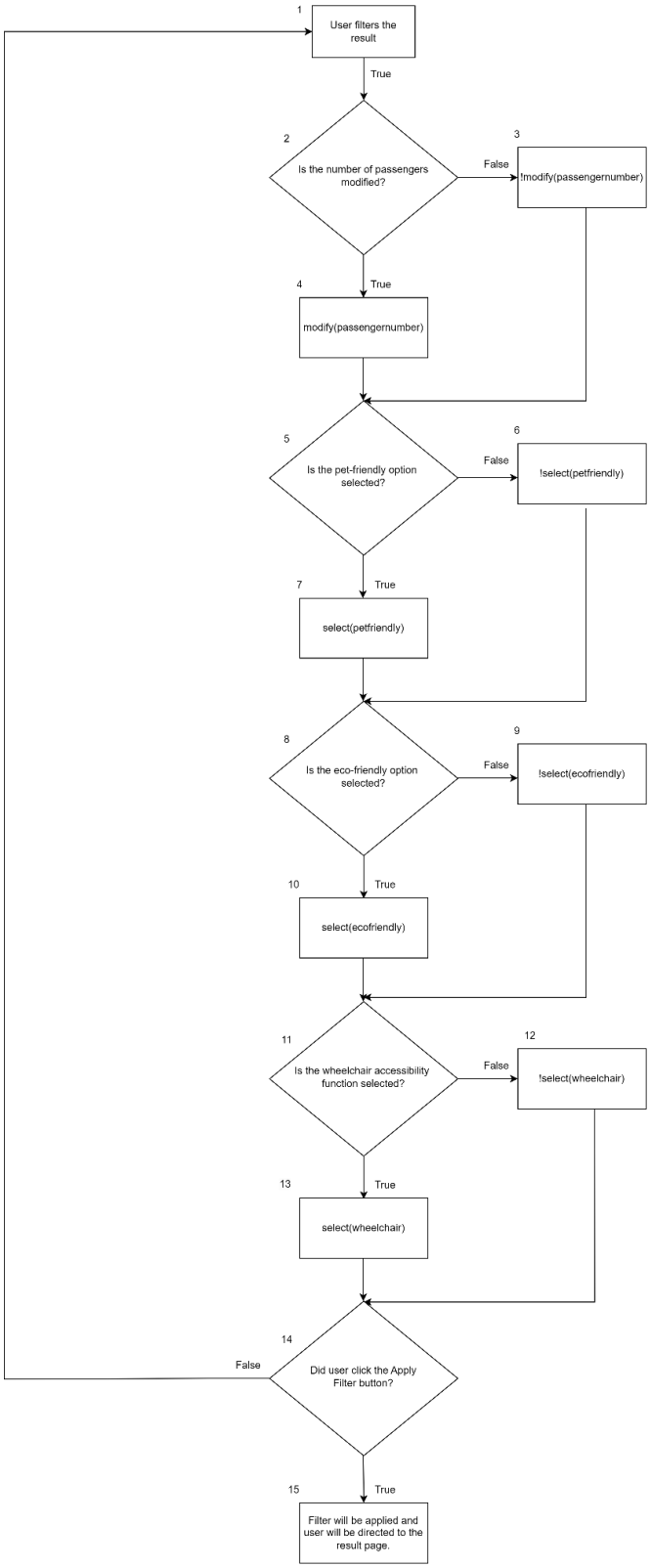
Path 2: User prompted to input end address. End address is not alphanumeric, User is prompted to input the end address again.

Path 3: User prompted to input end address. End address is alphanumeric, address is not selected from the dropdown list, User is prompted to input the end address again.

Path 4: User prompted to input end address. End address is alphanumeric, address is selected from the dropdown list. The application will show the Result Page, User select sort by fastest time, and the result will be filtered based on travel time.

Cyclomatic Complexity
|edges| = 11
|nodes| = 10
|edges| - |nodes| +2 = 11 - 10 + 2 = 3

Test Case 6: Filter Function



Basis Set of Path
Path 1: 1,2,4,5,7,8,10,11,13,14,15
Path 2: 1,2,3,5,7,8,10,11,13,14,15
Path 3: 1,2,4,5,6,8,10,11,13,14,15
Path 4: 1,2,4,5,7,8,9,11,13,14,15
Path 5: 1,2,4,5,7,8,10,11,12,14,15
Path 6: 1,2,4,5,7,8,10,11,13,14,1

Test Cases
Path 1: User selected all possible filter options and clicked the apply filter button.

Path 2: User only selected modify passenger filter option and clicked the apply filter button.

Path 3: User only selected pet-friendly filter option and clicked the apply filter button.

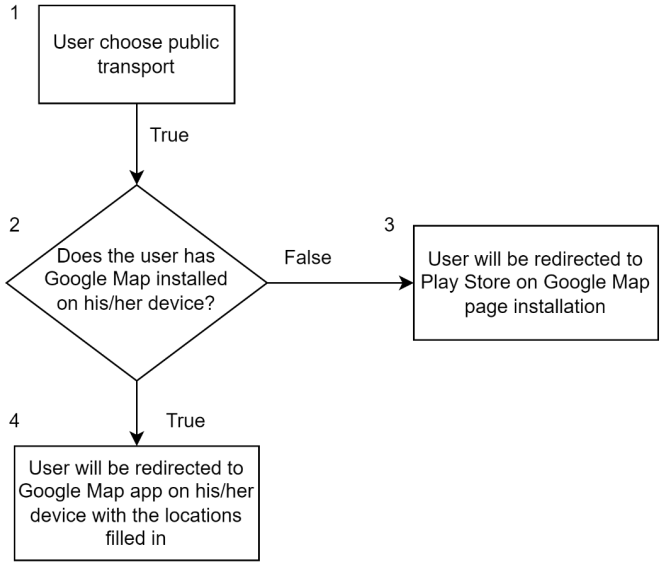
Path 4: User only selected eco-friendly filter option and clicked the apply filter button.

Path 5: User only selected wheelchair accessibility filter option and clicked the apply filter button.

Path 6: User selected all of the filter options and clicked the apply filter button.

Cyclomatic Complexity
|Decision Points| + 1 = 5 + 1 = 6

Test Case 7: Deep-Link



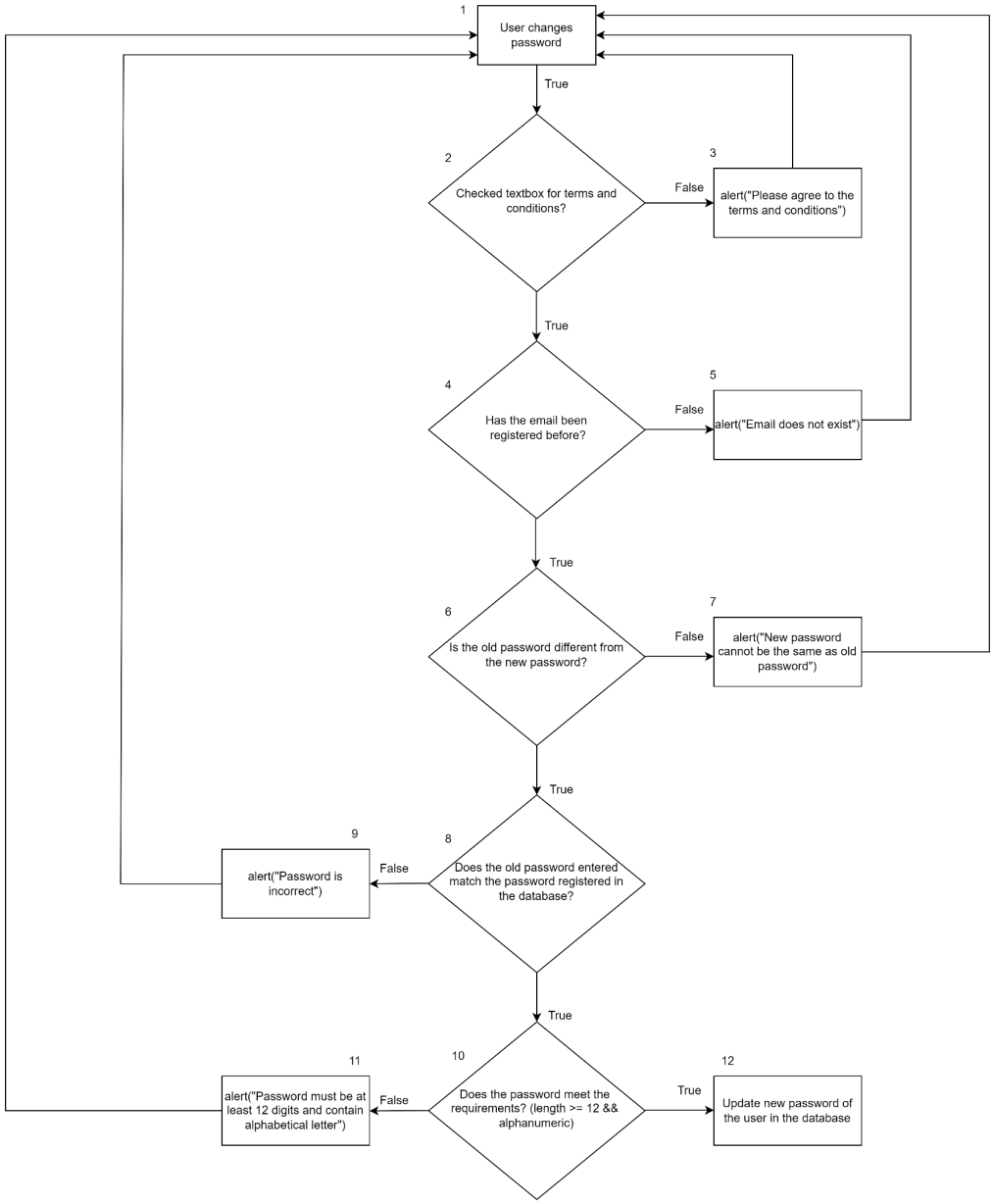
Basis Set of Path
Path 1: 1,2,4
Path 2: 1,2,3

Test Cases
Path 1: User has Google Map application installed on his/her device. He/she is being redirected to the application with the locations that he/she searched for pre-filled.

Path 2: User will be redirected to Play Store on Google Map page installation if he/she doesn't have Google Map installed on his/her device.

Cyclomatic Complexity
|Decision Points| + 1 = 1 + 1 = 2

Test Case 8: Change Password Function



<u>Basis Set of Path</u>

Path 1: 1,2,4,6,8,10,12
Path 2: 1,2,3,1
Path 3: 1,2,4,5,1
Path 4: 1,2,4,6,7,1
Path 5: 1,2,4,6,8,9,1
Path 6: 1,2,4,6,8,10,11,1

Test Cases

Path 1: User has fulfilled all the criteria and is able to change password successfully.

Path 2: User did not agree to the terms and conditions.

Path 3: User did not input a valid email which has been registered and stored in the database before.

Path 4: User did not input a different new password.

Path 5: User inputs an old password which does not match with the password stored in the database.

Path 6: User input a password which did not meet the requirements (length >= 12 && must be alphanumeric).

Cyclomatic Complexity

|Decision Points| + 1 = 5 + 1 = 6

Appendix

Appendix A: User Manual

Welcome to TripSmart - your go-to app for making your travels within Singapore easy, affordable and stress-free. Here's a step-by-step guide on how to use the app:

Step 1: Open the Application
To access TripSmart, open the app on your Android or iOS device.

Step 2: Login or Register
You have the option to continue as a guest or log in using your existing account. If you don't have an account, you can quickly register a new one to start using the app.

Step 3: Input Start and Destination
Once you're logged in, you can input your start location and destination in the relevant fields. You can either manually type in the address or use the GPS feature to detect your current location.

Step 4: View All Trip Options
After inputting your start location and destination, you will be presented with a list of trip options. The list will be arranged in order of either cheapest or fastest, depending on your preferences. The app pulls data from multiple sources and transport modes, such as public transportation, ride-hailing and taxi services, as well as private vehicle rental firms to help you make informed choices and save time and money on your travels.

That's it! With these simple steps, you can easily plan your travels within Singapore with TripSmart. If you encounter any issues or have any questions, feel free to contact our support team for assistance. We hope that you will have an enjoyable experience using TripSmart!