NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

# SC2006 – Software Engineering

## Lab 1 Deliverables

| | |
|---|---|
| **Lab Group** | SCSF |
| **Team** | 4 (Fantastic Four) |
| **Members** | Dylan Quek Zhi En (U2321239L)<br>Lee Zhuo Yang, Nicholas (U2423112F)<br>Sheth Shaivi Sachin (U2323665B)<br>Rashi Ojha (U2323323D)<br>Richard Chong Wing Liong (U2222047G) |

## Table Of Contents

## Project Mission Statement

The Fantastic Four team will develop a user-friendly web application called ParkIT, which is aimed at making parking in Singapore more efficient. Through ParkIT, drivers can quickly find available car parks, along with their cost and availability, saving them time and money. The app allows users to set personalized filters like price and parking type, so they can easily choose the best option for them. With real-time data, ParkIT aims to make the parking process smoother and support Singapore's goal of becoming a smarter, more connected city. The project will be complete when the website is fully tested and working as expected.

## Target Users

ParkIT is designed for a diverse range of Singaporean drivers including:

1. Everyday Commuters: The main target users for ParkIT app consist of individuals who drive regularly. ParkIT helps them quickly locate available parking locations and offers affordable parking rates around their residential, office and public areas.
2. Electric Vehicle Owners: ParkIT helps electric vehicle drivers who need car parks with EV Charging lots.
3. Disabled Drivers: ParkIT makes it easy for drivers with disabilities to find accessible car parks. It allows drivers to choose a parking type and shows all the available spaces that are suitable to their preference.
4. Fleet Owners and Businesses: ParkIT helps fleet businesses optimize parking for their vehicles by saving both cost and money. The data analytics of their parking history gives insight into their spendings and so help them optimize their fleet's parking usage.

# 1. Documentation of functional and non-functional requirements

## A. Functional Requirements

1. ParkIT shall allow drivers to be authenticated and use the ParkIT application.

   1.1 ParkIT shall allow drivers to create an account

   1.1.1 ParkIT shall support account creation for Singaporean drivers via singpass.

   1.1.2 ParkIT shall support account creation via Google gmail.

   1.1.3 ParkIT shall allow drivers to create an account by entering the following details.

   1.1.3.1 Name

   1.1.3.2 Contact number

   1.1.3.3 Profile photo

   1.2 ParkIT shall allow drivers to login to their account

   1.1.1 ParkIT shall support login for Singaporean drivers via singpass.

   1.1.2 ParkIT shall support login via Google.

   1.1.3 ParkIT shall allow drivers to login via their account

2. ParkIT shall allow drivers to search for parking

   2.1 ParkIT shall allow the driver to input their destination

   2.2 ParkIT shall validate the input destination

   2.3 ParkIT shall use the input destination to determine nearby car parks.

3. ParkIT shall display the available car parks

    3.1 ParkIT shall display all the available car parks within a 3 km radius of the input destination.

    3.2 ParkIT shall allow drivers to switch between map and list view

4. ParkIT shall allow drivers to apply filters to refine the parking search.

    4.1 ParkIT shall allow drivers to use a sidebar to select the distance radius

    4.2 Park IT shall allow drivers to sort by price or distance by the following.

        4.2.1 Ascending

        4.2.2 Descending

    4.3 Park IT shall allow drivers to filter by the following type of parking

        4.3.1 Normal parking

        4.3.2 Disabled parking

        4.3.3 EV charging lots

5. ParkIT shall allow the drivers to select and view car park details

    5.1 ParkIT shall display the car park's pricing data

    5.2 ParkIT shall display the lot availability

    5.3 PartIT shall display EV charging information

    5.4 ParkIT shall display the car park's name and address

    5.5 ParkIT shall allow navigation to the selected car park

6. Once the driver selects their desired car park, ParkIT shall allow the driver to navigate to the selected car park.

6.1 ParkIT shall use OneMap Route API to generate directions and navigate the route.

6.2 ParkIT shall display and update the estimated time and distance to the selected car park as the driver drives.

7. ParIT shall manage parking sessions

7.1 ParkIT shall allow drivers to record their parking sessions.

7.1.1 ParkIT shall pin the parking spot location in the driver's account.

7.1.2 ParkIT shall allow drivers to save the location as favourite

7.1.3 ParkIT shall display the pinned parking spot on the map.

7.1.4 ParkIT shall provide drivers with navigation to the pinned parking spot.

7.2 ParkIT shall display saved parking session

7.2.1 ParkIT shall display the total parking duration.

7.2.2 ParkIT shall save statistics, including cost and time elapsed, for each parking session in the database.

8. Park IT shall provide drivers with a dashboard.

8.1 ParkIT shall display parking history.

8.1.1 ParkIT shall display the time spent for each parking session.

8.1.2 ParkIT shall display the amount spent for each parking session.

8.2 ParkIT shall display statistics.

8.2.1 ParkIT shall calculate the total time and amount spent on parking within a given month.

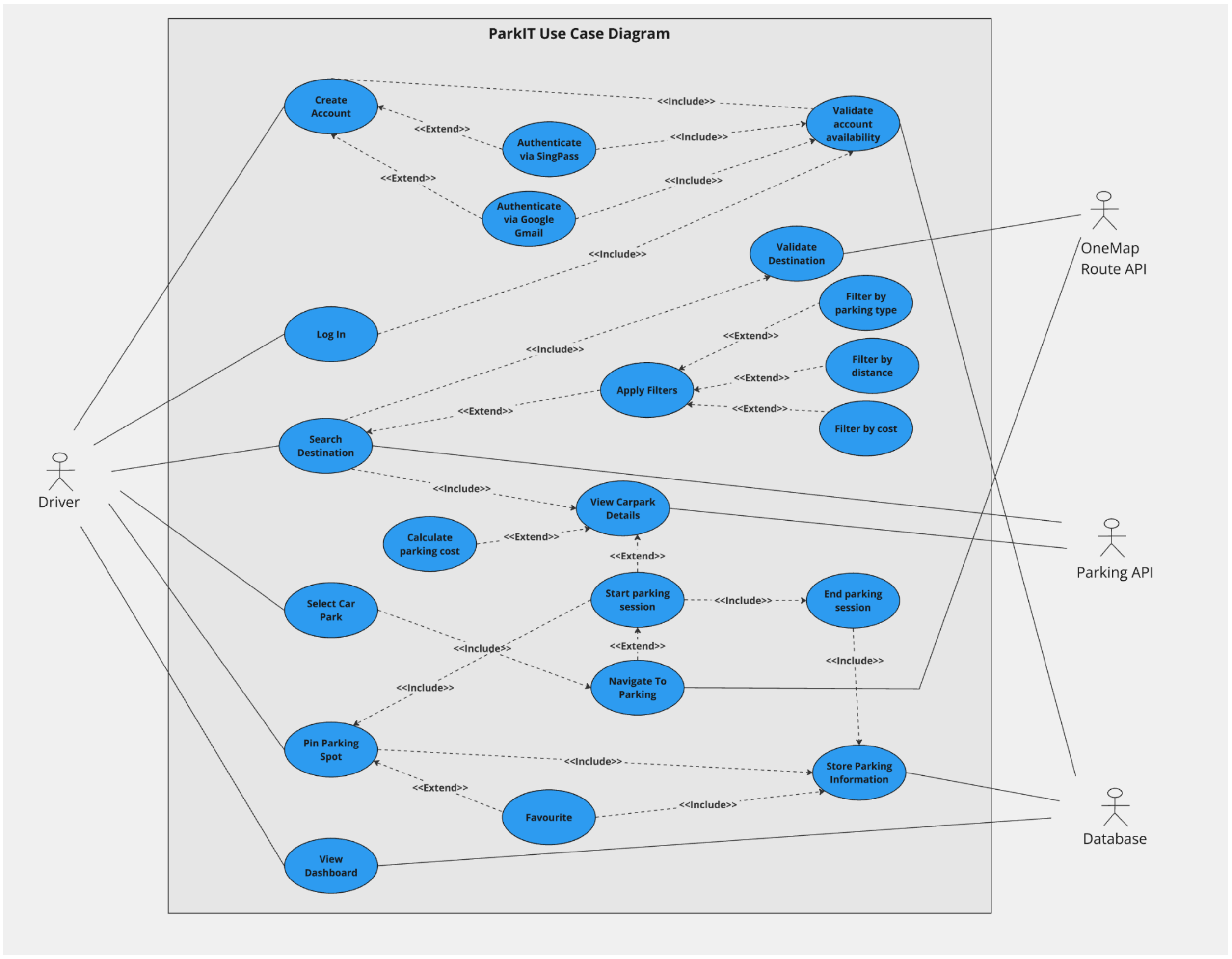
8.2.2 ParkIT shall allow drivers to compare time and cost spent on parking on a weekly or monthly basis using bar charts.

## 1. Documentation of functional and non-functional requirements

### B. Non- Functional Requirements

| | |
|---|---|
| **Usability** | Different Languages<br>● Content must be displayed in the local language ( English, Chinese, Malay, Tamil) according to the user's locale.<br><br>Responsive Design<br>● Users must be able to see 100% of all the content regardless of their device.<br>● No content should be chipped off  or lost due to device size.<br><br>Responsive User Interface<br>● Users must experience minimal delays (less than 5 seconds) when performing common tasks.<br><br>User Accessibility<br>● The design must be clear and users can effortlessly find key features such as parking availability and pricing. |
| **Reliability** | ● After a system reboot, the fill system functionality must be restored within 5 seconds.<br>● ParkIT must follow data visualisation standards when displaying driver's parking statistics. |
| **Performance** | Data Retrieval Efficiency<br>● The app should efficiently retrieve and display data to users (eg: Parking lot availability and their prices must be available to the user)<br><br>Real-Time Data Updates:<br><br>● Parking availability and pricing data must be updated in real-time to ensure that users have the most current information when making decisions. |
| **Supportability** | The database must be replaceable with any commercial product supporting standard SQL / NoSQL queries. |

## 2. UML Use Case Diagram & Description



## 1. Account Management

### 1.1. Create an Account

| | |
|---|---|
| **Use case ID** | 1 |
| **Use Case** | Create Account |
| **Primary Actor** | Driver |
| **Description** | This use case describes the process by which a driver registers for a new account on the ParkIT platform. The driver can create an account using Singpass (for local drivers) or Google Gmail or sign up using their own email (for guest drivers). |
| **Pre Conditions** | 1. The driver has navigated to the ParkIT website.<br>2. The driver has selected the option to create an account. |
| **Post Conditions** | A new driver account is created in the system with the role "Driver."<br>The driver's provided details (name, contact number, and optional profile picture) are stored in the database. |
| **Priority** | High |
| **Frequency of Use** | 1-3 times per lifetime per user |
| **Flow of Events** | 1. The driver selects "Create Account" from the homepage.<br>2. The system presents the account creation form.<br>3. The driver enters the required details (name, contact number, optional profile picture).<br>4. The system offers a choice between Singpass (for Singaporean drivers) and Google Gmail (for guest/tourist drivers) for authentication.<br>5. The driver selects an authentication method and completes the process.<br>6. The system validates the input and creates the account with the "Driver" role.<br>7. A confirmation message is displayed, and the driver is redirected to the main dashboard. |
| **Alternate Flow** | **Invalid Input:** If the driver enters invalid or incomplete data, the system displays error messages and requests corrections before account creation can proceed. |

| | |
|---|---|
| **Exceptions** | None |
| **Includes** | Validate Account Availability |
| **Special Requirements** | None |
| **Assumptions** | 1. The system integrates with external authentication services (Singpass and Google).<br>2. Database storage is functioning as expected. |
| **Notes and Issues** | None |

## 1.2. Log in

| | |
|---|---|
| **Use Case ID** | 2 |
| **Use case** | Log In |
| **Primary Actor** | Driver |
| **Description** | This use case describes the process by which a driver logs into the ParkIT platform. The driver can authenticate using Singpass (for local users) or Google Gmail (for guest users). |
| **Pre Conditions** | The driver already has an account created in the system. |
| **Post Conditions** | The driver is authenticated and granted access to the ParkIT system. |
| **Priority** | High |
| **Frequency of Use** | Daily or as needed |
| **Flow of Events** | 1. The driver selects “Log In” on the login page.<br>2. The system displays the login form with options for Singpass (for local drivers) and Google Gmail (for guest drivers).<br>3. The driver enters credentials and selects the appropriate authentication method.<br>4. The system verifies the credentials against stored account information.<br>5. On successful verification, the driver is logged in and redirected to the main dashboard. |

|  |  |
|---|---|
| **Alternate Flow** | Failed authentication attempt:<br>1. The system detects the unsuccessful login attempt<br>2. An error message is displayed indicating that the login attempt was unsuccessful.<br>3. The driver is given the option to retry logging in or reset the password. |
| **Exceptions** | None |
| **Includes** | Validate Account Credentials |
| **Special Requirements** | None |
| **Assumptions** | 1. Authentication services (Singpass, Google) are fully operational.<br>2. The system securely stores and retrieves account information. |
| **Notes and Issues** | None |

### 1.3. Validate Account Availability

|  |  |
|---|---|
| **Use case ID** | 3 |
| **Use case** | Validate Account Availability |
| **Primary Actor** | System |
| **Description** | This use case describes how the system validates the availability of a driver's account during the registration or login process to ensure that the account is unique (during registration) or exists (during login). |
| **Pre Conditions** | 1. The system has received account details (e.g., email address or Singpass/Google ID).<br>2. The system has access to the account database. |
| **Post Conditions** | 1. The system verifies whether the account exists or is available for registration.<br>2. The system either proceeds with the registration or login process or displays an error message. |
| **Priority** | High |

| | |
|---|---|
| **Frequency of Use** | During every login or account registration attempt |
| **Flow of Events** | 1. The system receives account details (e.g., email address, Singpass/Google ID) from the user.<br>2. The system queries the account database to check if the account exists.<br>3. If the account exists (during login), the system continues to authenticate the user.<br>4. If the account does not exist (during registration), the system allows the user to proceed with creating a new account. |
| **Alternate Flow** | AF-1: Account Already Exists (During Registration):<br>1. The system detects that the provided account details already exist in the database.<br>2. The system displays an error message indicating that the account is already registered.<br>3. The user is prompted to try a different email or authentication method.<br><br>AF-2: Account Not Found (During Login):<br>1. The system detects that the provided account details do not exist in the database.<br>2. The system displays an error message indicating that the account is not found.<br>3. The user is prompted to check their input or register for a new account. |
| **Exceptions** | Database Connection Error:<br>1. The system is unable to query the account database.<br>2. The system displays an error message indicating that the service is temporarily unavailable.<br>3. The user is advised to try again later. |
| **Includes** | Validate Account Input |
| **Special Requirements** | Database access must be reliable and secure. |
| **Assumptions** | The account database contains up-to-date records of all registered users. |
| **Notes and Issues** | None |

## 1.4. Authenticate via SingPass

| | |
|---|---|
| **Use case ID** | 4 |
| **Use case** | Authenticate via SingPass |
| **Primary Actor** | Driver |
| **Description** | This use case describes how a driver logs in or registers on the ParkIT platform using Singpass for authentication. |
| **Pre Conditions** | 1. The driver has selected the Singpass authentication option during login or account creation.<br>2. The Singpass service is operational and reachable. |
| **Post Conditions** | 1. The system successfully authenticates the driver using Singpass.<br>2. The driver gains access to the ParkIT system. |
| **Priority** | High |
| **Frequency of Use** | Regular use for login or registration by local drivers |
| **Flow of Events** | 1. The driver selects "Singpass" as the authentication method.<br>2. The system redirects the driver to the Singpass authentication portal.<br>3. The driver logs in using their Singpass credentials.<br>4. Singpass verifies the credentials and returns an authentication token to the system.<br>5. The system validates the token and grants access to the driver. |
| **Alternate Flow** | Authentication Failure:<br>1. Singpass detects invalid credentials and denies access.<br>2. The driver is redirected back to the ParkIT system with an error message.<br>3. The driver can retry the authentication process. |
| **Exceptions** | Singpass Service Unavailable:<br>1. The system displays an error message indicating that Singpass is temporarily unavailable.<br>2. The driver is advised to try again later. |
| **Includes** | Validate Token |
| **Special** | Integration with Singpass API must comply with security and |

| | |
|---|---|
| **Requirements** | privacy standards. |
| **Assumptions** | The Singpass authentication service is available. |
| **Notes and Issues** | None |

### 1.5. Authenticate via Google Gmail

| | |
|---|---|
| **Use case ID** | 5 |
| **Use case** | Authenticate via Google Gmail |
| **Primary Actor** | Driver |
| **Description** | This use case describes how a driver logs in or registers on the ParkIT platform using Google Gmail for authentication. |
| **Pre Conditions** | 1. The driver has selected the Google Gmail authentication option during login or account creation.<br>2. The Google authentication service is operational and reachable. |
| **Post Conditions** | 1. The system successfully authenticates the driver using their Google account.<br>2. The driver gains access to the ParkIT system. |
| **Priority** | High |
| **Frequency of Use** | Regular use for login or registration |
| **Flow of Events** | 1. The driver selects "Google Gmail" as the authentication method.<br>2. The system redirects the driver to the Google OAuth authentication portal.<br>3. The driver logs in using their Google account credentials.<br>4. Google verifies the credentials and returns an OAuth token to the system.<br>5. The system validates the token and grants access to the driver. |
| **Alternate Flow** | Authentication Failure:<br>1. Google detects invalid credentials and denies access.<br>2. The driver is redirected back to the ParkIT system with an error message. |

| | |
|---|---|
| | 3. The driver can retry the authentication process. |
| **Exceptions** | Google Service Unavailable:<br>1. The system displays an error message indicating that Google authentication is temporarily unavailable.<br>2. The driver is advised to try again later. |
| **Includes** | Validate Token |
| **Special Requirements** | Integration with Google OAuth API must comply with security and privacy standards. |
| **Assumptions** | The Google OAuth authentication service is reliable and available. |
| **Notes and Issues** | None |

## 2. Parking Search and Navigation

### 2.1. Search Car Park

| | |
|---|---|
| **Use Case ID** | 6 |
| **Use case** | Search Car Park |
| **Primary Actor** | Driver |
| **Description** | This use case allows a driver to search for available car parks near a specified destination. The driver enters a destination, and the system retrieves and displays a list and map of nearby car parks along with pertinent details such as availability, pricing, and parking type. This search functionality optionally supports the application of filters to refine results. |
| **Pre Conditions** | 1. The driver has entered or selected a destination. |
| **Post Conditions** | The system displays a list and map of available car parks near the specified destination. |
| **Priority** | High |
| **Frequency of Use** | Frequently – every time a driver needs to find parking. |
| **Flow of Events** | 1. The driver inputs a destination into the search field. |

| | |
|---|---|
| | 2. The system validates the destination.<br>3. The system initiates the search by querying the database for car parks within a 3 km radius. *(Uses Database)*<br>4. The use case **"Display Parking"** (included as a mandatory behavior) is triggered, and the system displays the car park details such as availability, pricing, and parking type. |
| **Alternate Flow** | 1. The driver selects various filters (e.g., distance radius, sort by price, parking type such as normal, disabled, or EV charging).<br>2. The **"Apply Filters"** extension modifies the search query results.<br>3. The filtered car parks are then re-displayed as part of the “Display Parking” process. |
| **Exceptions** | 1. If the destination cannot be validated, an error message is displayed prompting the driver to correct the input.<br>2. In case of a database connection failure, the system informs the driver that the search is temporarily unavailable. |
| **Includes** | "Display Parking" (mandatory behavior that shows the search results). |
| **Special Requirements** | 1. Real-time data must be used so that the parking availability and pricing are current.<br>2. The search response should be delivered promptly (e.g., within 3 seconds). |
| **Assumptions** | 1. The system maintains an up-to-date database of available car parks.<br>2. The driver’s device has a stable internet connection. |
| **Notes and Issues** | None |

## 2.2. Select Car Park

| | |
|---|---|
| **Use Case ID** | 7 |
| **Use case** | Select Car Park |
| **Primary Actor** | Driver |
| **Description** | In this use case, a driver selects a car park from a list of search results. After selection, the system displays detailed information about the chosen car park and provides navigational directions to guide the driver to the parking location. |
| **Pre Conditions** | A list of car parks has been displayed following a search. The driver has reviewed the options. |
| **Post Conditions** | The system displays detailed information about the selected car park and allows the user to start a parking session, and provide navigational directions to it. |
| **Priority** | High |
| **Frequency of Use** | Frequently – each time a driver decides on a specific car park. |
| **Flow of Events** | 1. The driver clicks on a specific car park from the search results.<br>2. The system triggers the **"View Car park Details"** use case (included), which displays information such as real-time pricing, availability, distance from the destination, and estimated driving time.<br>3. After reviewing the details, the driver confirms the selection.<br>4. The system then initiates the **"Navigate to Parking"** use case (also included) to generate directions using the OneMap API.<br>5. Navigation details (route, estimated time, and distance) are displayed to the driver. |
| **Alternate Flow** | **If navigation fails:** An error message is displayed with options to retry or select another car park. |
| **Exceptions** | If the "View Car park Details" use case fails to retrieve necessary information, the system will display a relevant error message and suggest that the driver try again later. |

| | |
|---|---|
| **Includes** | 1. "View Car park Details" (to display detailed parking information).<br>2. "Navigate to Parking" (to provide directions). |
| **Special Requirements** | 1. The system must integrate with the OneMap API reliably.<br>2. Directions should be updated in real-time if conditions change. |
| **Assumptions** | 1. The car park details (pricing, availability, etc.) are current and accurate.<br>2. The OneMap API is available and responsive. |
| **Notes and Issues** | None |

## 2.3. Pin Parking Spot

| | |
|---|---|
| **Use case ID** | 8 |
| **Use case** | Pin Parking Spot |
| **Primary Actor** | Driver |
| **Description** | This use case allows a driver to save the location of a parked vehicle by “pinning” the parking spot on the map. This feature helps the driver quickly retrieve the parking location later and obtain navigational directions back to the car. |
| **Pre Conditions** | 1. User has logged in<br>2. The driver has parked their vehicle in a selected parking spot. |
| **Post Conditions** | The parking spot location is saved (pinned) to the driver’s account, and a visual marker appears on the map. |
| **Priority** | Medium |
| **Frequency of Use** | Occasionally – typically used when a driver parks and wants to remember the spot. |
| **Flow of Events** | 1. Once the driver has parked, they select the “Pin Parking Spot” option.<br>2. The system captures the current location of the vehicle. |

| | |
|---|---|
| | 3. The system saves the location data to the driver’s account (usually in the background, storing it in the database).<br>4. The pinned parking spot is then displayed on the map, and the driver can later request directions back to it. |
| **Alternate Flow** | **If saving fails:** An error message is displayed and the driver is prompted to try pinning the spot again. |
| **Exceptions** | If the device’s GPS is unavailable or inaccurate, the system may be unable to capture the correct location, in which case an error is shown. |
| **Includes** | None |
| **Special Requirements** | The user interface must clearly indicate when a parking spot is successfully pinned. |
| **Assumptions** | The device has an active GPS signal. |
| **Notes and Issues** | Evaluate privacy considerations related to storing location data. |

### 2.4. Validate Destination

| | |
|---|---|
| **Use case ID** | 9 |
| **Use case** | Validate Destination |
| **Primary Actor** | Driver |
| **Description** | This use case ensures that the destination entered by the driver is valid and recognized by the system. It confirms that the provided location is in the correct format and can be resolved to a geographical area, enabling accurate parking search results. |
| **Pre Conditions** | The driver has entered a destination into the search field. |
| **Post Conditions** | If the destination is valid, the system confirms it and allows the parking search process to proceed.<br>If the destination is invalid, the system displays an error message and prompts the driver to enter a correct destination. |

| | |
|---|---|
| **Priority** | High |
| **Frequency of Use** | Every time a driver initiates a parking search by entering a destination. |
| **Flow of Events** | The driver inputs a destination into the search field.<br>The system receives the destination input and checks the format (e.g., proper address or landmark notation).<br>The system verifies that the destination exists and can be mapped to a geographical location using its internal logic or external mapping services.<br>The system confirms the destination as valid.<br>The validated destination is then used to proceed with the parking search process. |
| **Alternate Flow** | If the destination is not in a recognized format or cannot be resolved, the system identifies the issue.<br>An error message is displayed to the driver (e.g., "Destination not found. Please enter a valid address or landmark.").<br>The driver is prompted to re-enter the correct destination. |
| **Exceptions** | None |
| **Includes** | This use case is typically only invoked as part of the “Search Parking” process, and ensures that only validated destinations are used for querying available car parks. |
| **Special Requirements** | The validation process should be performed quickly to avoid any noticeable delay in the search process (e.g., within 2–3 seconds).<br>The error messages must be clear and user-friendly, guiding the driver on how to correct the input. |
| **Assumptions** | The driver enters destination data in a language or format that the system is configured to recognize. |
| **Notes and Issues** | None |

## 2.5. View Car park Details

| | |
|---|---|
| **Use case ID** | 10 |
| **Use case** | View Car park Details |
| **Primary Actor** | Driver |
| **Description** | This use case allows the driver to review detailed information about a selected car park. The system displays data such as real-time pricing, current availability, parking type (e.g., normal, disabled, EV charging), distance from the destination, and estimated driving time. This detailed view supports the driver's decision-making before selecting a car park. |
| **Pre Conditions** | A list of car parks has been generated from a prior search. The driver has selected a specific car park from the search results. |
| **Post Conditions** | The system presents detailed information about the chosen car park to the driver. |
| **Priority** | High |
| **Frequency of Use** | Frequently – each time the driver reviews a parking option. |
| **Flow of Events** | The driver clicks on a car park from the list or map.<br>The system retrieves detailed information for the selected car park from its internal data store or a real-time data source.<br>The system displays information including pricing per hour (or per kWh for EV lots), availability status, parking type, distance from the destination, and estimated driving time.<br>The detailed view remains available for the driver to review further before proceeding with navigation or selection. |
| **Alternate Flow** | If certain details (e.g., real-time availability) cannot be retrieved, the system displays a placeholder or “information not available” message while providing any other available details. |
| **Exceptions** | None |
| **Includes** | This use case is typically included as part of the "Select Car Park" flow, which leads into navigation or further actions. |
| **Special** | The layout should be clear and accessible on various |

| | |
|---|---|
| **Requirements** | devices. |
| **Assumptions** | The system has access to accurate, real-time data for each car park. |
| **Notes and Issues** | None |

### 2.6 Store Parking Info

| | |
|---|---|
| **Use case ID** | 11 |
| **Use case** | Store parking info |
| **Primary Actor** | Database |
| **Description** | This use case handles the storage and management of parking information, such as cost, location, and duration. |
| **Pre Conditions** | 1. The driver is logged in to the system.<br>2. The driver has ended a parking session. |
| **Post Conditions** | Parking session information, including total duration, cost and location, is stored in a database. This information will be accessible for future reference from the driver's dashboard. |
| **Priority** | High |
| **Frequency of Use** | Very frequent - used whenever a parking session is completed. |
| **Flow of Events** | The driver ends a parking session. The system captures parking session details. The following information is stored in a database: Start and end time, total duration, total cost. |
| **Alternate Flow** | None |
| **Exceptions** | None |
| **Includes** | None |
| **Special Requirements** | Database must be secure. When data is fetched from the database, it should take a maximum of 3 seconds. |
| **Assumptions** | None |
| **Notes and Issues** | None |

## 2.7 Navigate to Parking

| | |
|---|---|
| **Use case ID** | 12 |
| **Use case** | Navigate to parking |
| **Primary Actor** | Driver |
| **Description** | This use case enables the system to provide real-time navigation instructions to guide the driver to the selected carpark using the OneMap Route API. It also provides the estimated time to arrival on a real-time basis. |
| **Pre Conditions** | The driver is logged into the system. The driver has selected a carpark. The driver has a working GPS connection. |
| **Post Conditions** | The route to the selected carpark is shown on the dashboard. Estimated time to arrival is also shown. Both the route and estimated arrival time are updated in real-time until the driver reaches the destination |
| **Priority** | High |
| **Frequency of Use** | Very frequent - every time the driver needs directions to their selected carpark |
| **Flow of Events** | The driver selects a carpark. The system obtains the driver's location via GPS. The system calls the OneMap Route API and uses it to generate the optimal route. |
| **Alternate Flow** | GPS unavailable<br>1. An error message is shown while the system reattempts locating the driver.<br>2. The driver will be given an option to manually input location.<br>3. In this case, the route will be generated once via the OneMap Route API but will not be updated in real-time. |
| **Exceptions** | API cannot be fetched:<br>1. An error message will be displayed.<br><br>Internet Connection Lost:<br>1. The system cannot communicate with the API due to a network issue.<br>2. The system attempts to reconnect and retrieve the data. |

| | |
|---|---|
| **Includes** | 1. OneMap Route API Integration<br>2. Update Route in Real Time |
| **Special Requirements** | None |
| **Assumptions** | 1. The driver’s device has GPS and internet connectivity for real-time navigation.<br>2. The OneMap Route API service is operational and reliable. |
| **Notes and Issues** | None |

## 3. Filter Management

### 3.1. Apply Filter

| | |
|---|---|
| **Use case ID** | 13 |
| **Use case** | Apply Filter |
| **Primary Actor** | Driver |
| **Description** | This use case is an optional feature. It allows the driver to apply filters such as filter by parking type, filter by distance radius and filter by cost to optimise based on their preferences. |
| **Pre Conditions** | 1. The driver has searched for the destination.<br>2. The destination has been validated |
| **Post Conditions** | 1. The driver can choose which filters to apply and not<br>2. After filter has been chosen, the driver can select the desired car park |
| **Priority** | High |
| **Frequency of Use** | Very Frequently as most drivers would want to optimise the results |
| **Flow of Events** | 1. The driver opens the filter menu.<br>2. The system displays available filter options, including:<br>3. Price<br>4. Parking type<br>5. Distance radius |

| | |
|---|---|
| | 6. The driver selects and applies one or more filters.<br>7. The system validates the filter input and updates the search results accordingly.<br>8. The system re-displays the filtered search results in both list and map views. |
| **Alternate Flow** | No Matching Results:<br>1. The system displays a message indicating that no car parks match the selected filter criteria.<br>2. The driver is prompted to adjust or reset the filters. |
| **Exceptions** | Filter Input Error:<br>1. The system detects invalid input (e.g., negative price values or out-of-range distance).<br>2. The system displays an error message and prompts the driver to correct the input. |
| **Includes** | 1. Filter by Price<br>2. Filter by Parking Type<br>3. Filter by Distance Radius |
| **Special Requirements** | None |
| **Assumptions** | The system provides up-to-date data for all filter criteria (price, parking type, distance). |
| **Notes and Issues** | None |

### 3.2. Filter by Price

| | |
|---|---|
| **Use case ID** | 14 |
| **Use case** | Filter by price |
| **Primary Actor** | Driver |
| **Description** | This use case allows drivers to refine their search by filtering and sorting parking locations based on price. They can set a price range, as well as sort in ascending or descending order. |
| **Pre Conditions** | 1. The driver has performed a parking search.<br>2. There are multiple parking locations available. |

| | |
|---|---|
| **Post Conditions** | Search results that fall within the price range will be shown in a list. Search results will be sorted in either ascending or descending order, depending on which was chosen. In map view, only parking spaces that fall within the price range will be shown. |
| **Priority** | Medium |
| **Frequency of Use** | Regular, based on user needs |
| **Flow of Events** | 1. Driver performs a parking search.<br>2. Driver sets a price range by selecting a minimum and maximum price.<br>3. Driver sorts the options by either ascending or descending order. |
| **Alternate Flow** | 1. Driver uses one of the other filters e.g. filter by type<br>2. Driver then filters by price. |
| **Exceptions** | Price information unavailable: Driver will be notified of the parking spaces where up to date price information is not available. By default, these parking spaces will be filtered out. There will be an option to include these results. |
| **Includes** | Validate Filter Input |
| **Special Requirements** | Filtered results should be displayed within 3 seconds. Can be combined with the other filters. |
| **Assumptions** | Accurate and up-to-date pricing data is available. |
| **Notes and Issues** | None |

## 3.3. Filter by Parking Type

| | |
|---|---|
| **Use case ID** | 15 |
| **Use case** | Filter by parking type |
| **Primary Actor** | Driver |
| **Description** | This use case allows drivers to refine their search by filtering based on the type of parking lots available. This allows drivers to quickly find a parking space that suits their needs. |
| **Pre Conditions** | 1. The driver has performed a parking search.<br>2. There are multiple parking spaces available. |
| **Post Conditions** | 1. The system updates and displays search results filtered by the selected parking type. |
| **Priority** | Medium |
| **Frequency of Use** | Regular, based on user needs |
| **Flow of Events** | 1. The driver opens the filter options and selects the parking type filter.<br>2. The system presents options for parking types (e.g., normal, disabled, EV charging).<br>3. The driver selects one or more parking types.<br>4. The system updates and re-displays the filtered parking results. |
| **Alternate Flow** | No Results for Selected Parking Type:<br>1. The system displays a message indicating that no parking spots match the selected parking type.<br>2. The driver is prompted to adjust the filter settings |
| **Exceptions** | None |
| **Includes** | Validate Filter Input |
| **Special Requirements** | None |
| **Assumptions** | Accurate and up-to-date data for parking types is available. |
| **Notes and Issues** | None |

## 3.4. Filter by Distance Radius

| | |
|---|---|
| **Use case ID** | 16 |
| **Use case** | Filter by distance radius |
| **Primary Actor** | Driver |
| **Description** | This use case describes how a driver refines their parking search by adjusting the distance radius from the destination. |
| **Pre Conditions** | 1. The system has displayed initial parking search results.<br>2. Distance data for each car park is available. |
| **Post Conditions** | 1. The system updates and displays search results based on the selected distance radius. |
| **Priority** | Medium |
| **Frequency of Use** | Regular, based on user needs |
| **Flow of Events** | 1. The driver opens the filter options and selects the distance radius filter.<br>2. The system provides a slider or input field to adjust the radius (e.g., 1 to 10 km).<br>3. The driver sets the desired distance radius.<br>4. The system updates and re-displays the filtered parking results. |
| **Alternate Flow** | No Results Within Distance Radius:<br>1. The system displays a message indicating that no car parks match the selected distance radius.<br>2. The driver is prompted to adjust the filter settings |
| **Exceptions** | None |
| **Includes** | Validate Filter Input |
| **Special Requirements** | None |
| **Assumptions** | Accurate and up-to-date distance data is available. |
| **Notes and Issues** | None |

## 4. Save Favourites

| | |
|---|---|
| **Use case ID** | 17 |
| **Use case** | Favorites |
| **Primary Actor** | Driver |
| **Description** | This use case allows drivers to save their favorite parking locations for easy access in the future. Drivers can save frequently used carparks as favorites, providing convenience for regular destinations. |
| **Pre Conditions** | Driver must be logged in to their account. |
| **Post Conditions** | Selected parking location is saved to the driver's favorite list. Saved locations will appear as favorites in future searches. Drivers may jump to these locations during a search. |
| **Priority** | Medium |
| **Frequency of Use** | Moderate - Saving a location as favorite will likely be infrequent, but using a favorited location will be more frequent. |
| **Flow of Events** | 1. Driver selects a parking location.<br>2. Driver selects the "Add to favorites" option.<br>3. System prompts the driver to add an optional label to name the location<br>4. System saves the location as a favorite. |
| **Alternate Flow** | Using a favorited location<br>1. During the search, the driver opens up the list of favorites<br>2. By selecting the favorited location, the screen will be taken directly to the chosen location.<br><br>Deleting a favorited location<br>1. Driver selects a favorited location<br>2. Driver chooses to remove from favorites<br>3. System prompts the driver for confirmation<br>4. If confirmed, system removes location from favorites |
| **Exceptions** | Maximum number of favorites reached: Error message is displayed and driver is prompted to remove favorites before adding new ones. |

| | |
|---|---|
| **Includes** | None |
| **Special Requirements** | None |
| **Assumptions** | None |
| **Notes and Issues** | None |

## 5. View Dashboard

| | |
|---|---|
| **Use case ID** | 18 |
| **Use case** | View Dashboard |
| **Primary Actor** | Driver |
| **Description** | This use case enables a driver to review detailed statistics of their parking history. The dashboard displays data such as session durations, total costs, and monthly summaries, often using visual aids like bar charts for clarity. |
| **Pre Conditions** | The driver has completed one or more parking sessions, and the system has been tracking parking history. |
| **Post Conditions** | The driver is presented with detailed statistics on their parking usage, including monthly summaries and visual representations (bar charts). |
| **Priority** | Medium |
| **Frequency of Use** | Occasionally – typically used by drivers interested in reviewing their parking expenses and usage patterns over time. |
| **Flow of Events** | 1. The driver selects the “View Parking Statistics” option from the menu.<br>2. The system retrieves the driver’s parking history, including session times and amounts spent, from its internal data store. *(Note: The database interaction here is considered an internal process and is not shown as a secondary actor.)*<br>3. The system processes the retrieved data to calculate monthly statistics (total time and cost).<br>4. The system then displays the statistics using visual |

| | |
|---|---|
| | aids such as bar charts for easy comparison on a weekly or monthly basis. |
| **Alternate Flow** | **No History Available:** If no parking sessions have been recorded, the system displays a message such as “No parking history available.”<br>**Error in Data Retrieval:** If the system encounters an error retrieving the data, an appropriate error message is shown, and the driver is advised to try again later. |
| **Exceptions** | None |
| **Includes** | None |
| **Special Requirements** | The dashboard must load within a reasonable time (e.g., under 5 seconds).<br>Data visualizations should be clear, user-friendly, and accessible across devices. |
| **Assumptions** | The parking history is accurately recorded and stored by the system |
| **Notes and Issues** | None |

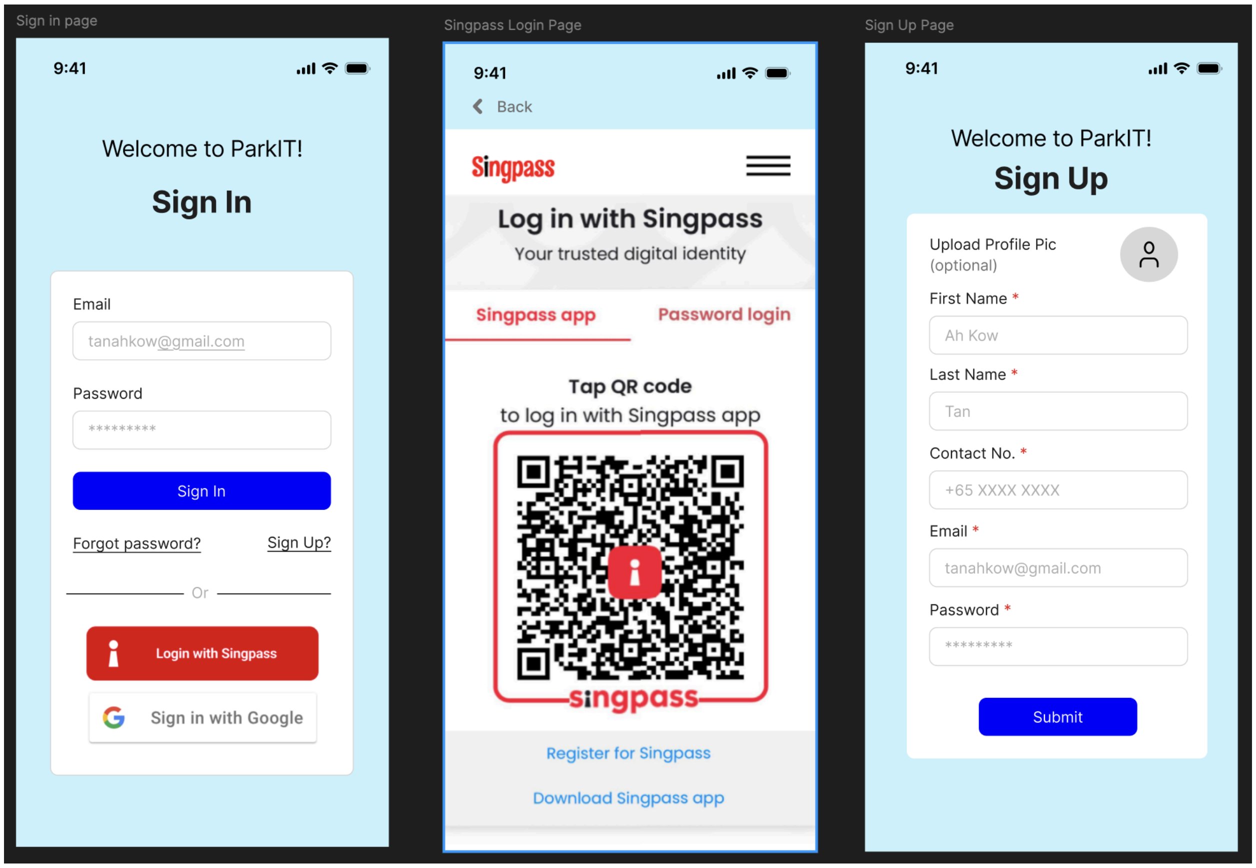
## 3. Data Dictionary

| Term | Definition |
|---|---|
| Account | A registered user's profile in ParkIT that contains their personal information, contact details, vehicle information, and parking history. Accounts can be created using either Singpass (for Singaporean drivers) or email (for tourists). |
| Application | A programme which is installed on a user’s device which is the means by which ParkIT’s features are accessed. Internet access is required. |
| Driver | A registered user who uses ParkIT to find and navigate to car parks. |
| Car park | A facility or location designed to accommodate multiple vehicles for parking. |
| Parking Lot | An individual parking space within a car park where a vehicle can be parked. |
| Parking Spot | Location where driver has parked his car |
| EV Charging Lot | A parking space which includes an electric vehicle charger, including specific charging rates in SGD per kWh. |
| Parking Session | A period during which a vehicle is parked, tracked for duration and cost calculations. |
| Parking History | A record of a driver's past parking sessions, including additional information such location, duration, and amount spent. |
| Parking Rate | The cost of parking per hour in SGD. The rate may vary with time and day. |
| Destination | A target location chosen by the driver. This location will be the reference point for finding nearby car parks. |
| Distance Radius | The maximum distance (default 3 km) from the destination within which ParkIT will display available car parks. |
| Route | The navigation path generated by OneMap API to direct drivers to their destination. |
| Filter | A set of criteria such as distance, price, parking type that drivers can use to refine their search. |

| | |
|---|---|
| Pin | A location marked by a driver to indicate where their vehicle was parked. |
| Dashboard | A summary page in the application that displays the driver's parking history, current sessions, and saved locations. |
| Authentication | The process by which a driver logs into the ParkIT application, either through Singpass or Google Gmail authentication services. |
| Database | A storage system where ParkIT keeps user accounts, parking history, preferences, and real-time parking data. |
| Error Message | A system-generated message displayed to the driver when an error occurs (e.g., invalid login, failed search, or API service errors). |
| Session Timeout | A system security feature that automatically logs out users after a period of inactivity to prevent unauthorized access. |
| Account Verification | A process in which users confirm their identity via a verification code sent to their registered email or phone number. |
| Search Query | A set of input parameters (e.g., destination, filters) that the system uses to retrieve parking results. |
| Parking API | An external service that provides real-time parking data such as availability, rates, and car park locations. |
| OneMap Route API | An external service that generates navigation routes for the driver, including travel time, distance, and directions. |
| Search Result View | The layout options (map view or list view) that display the available car parks after a search. |

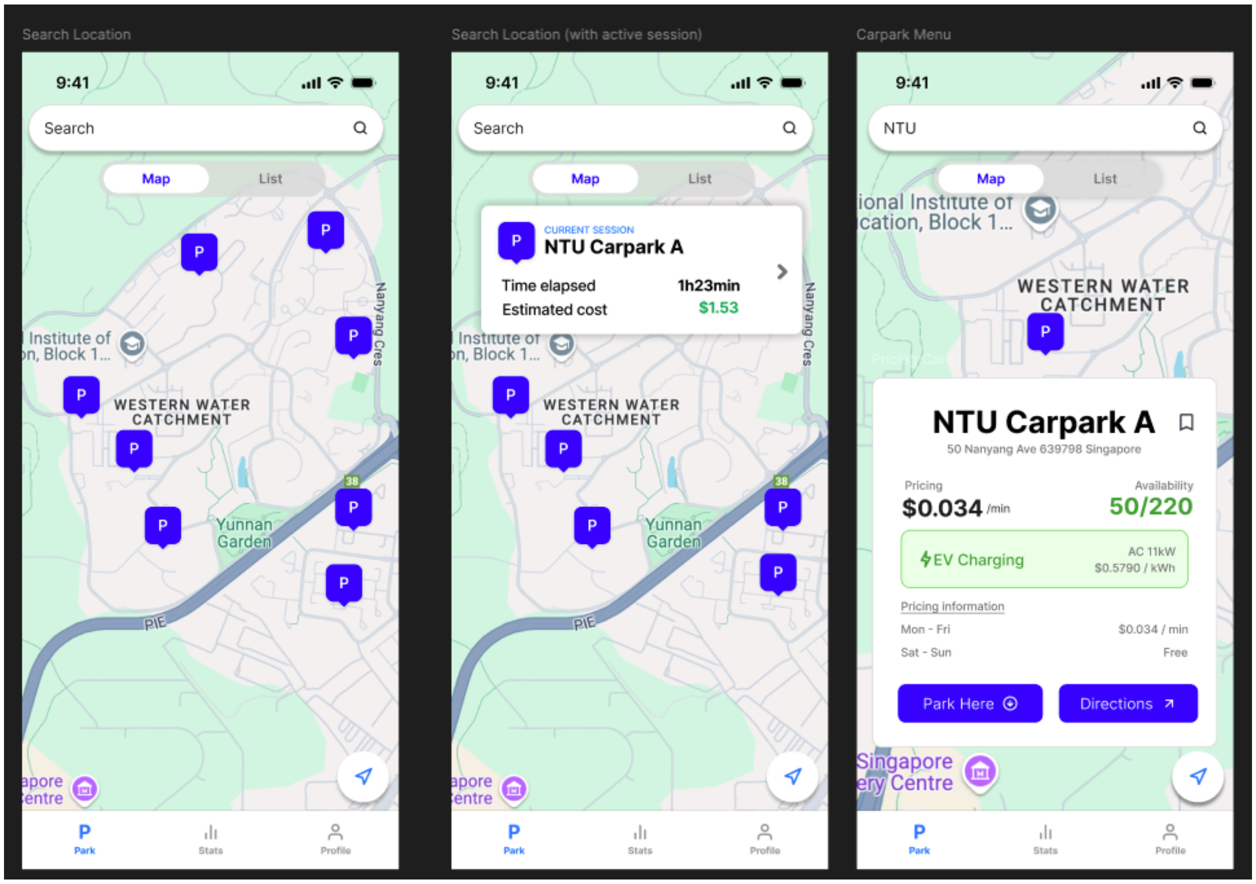
## 4. UI Mockups

### 4.1 Account Login and Sign Up pages

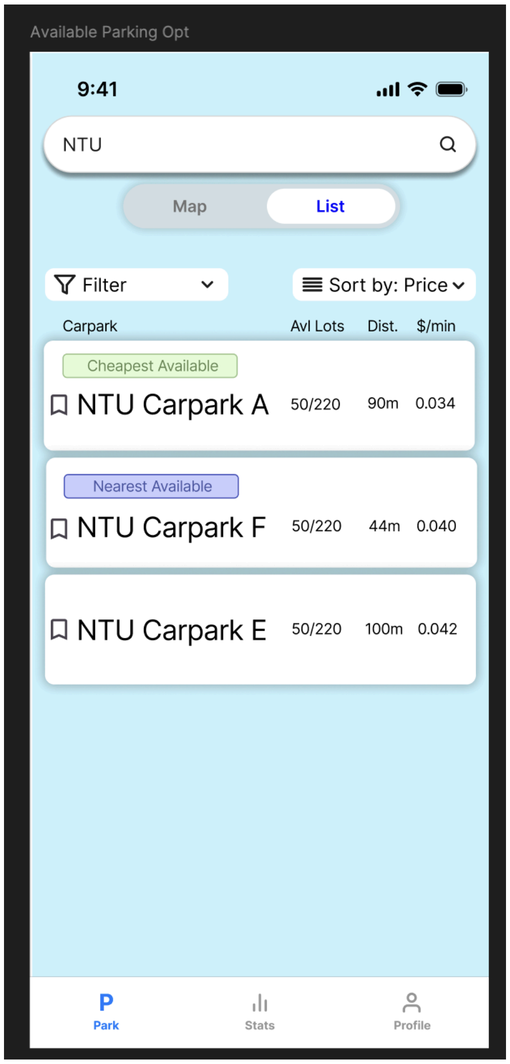


## 4.2 Park Menu

### 4.2.1 Map view - With and without active parking session + car park information

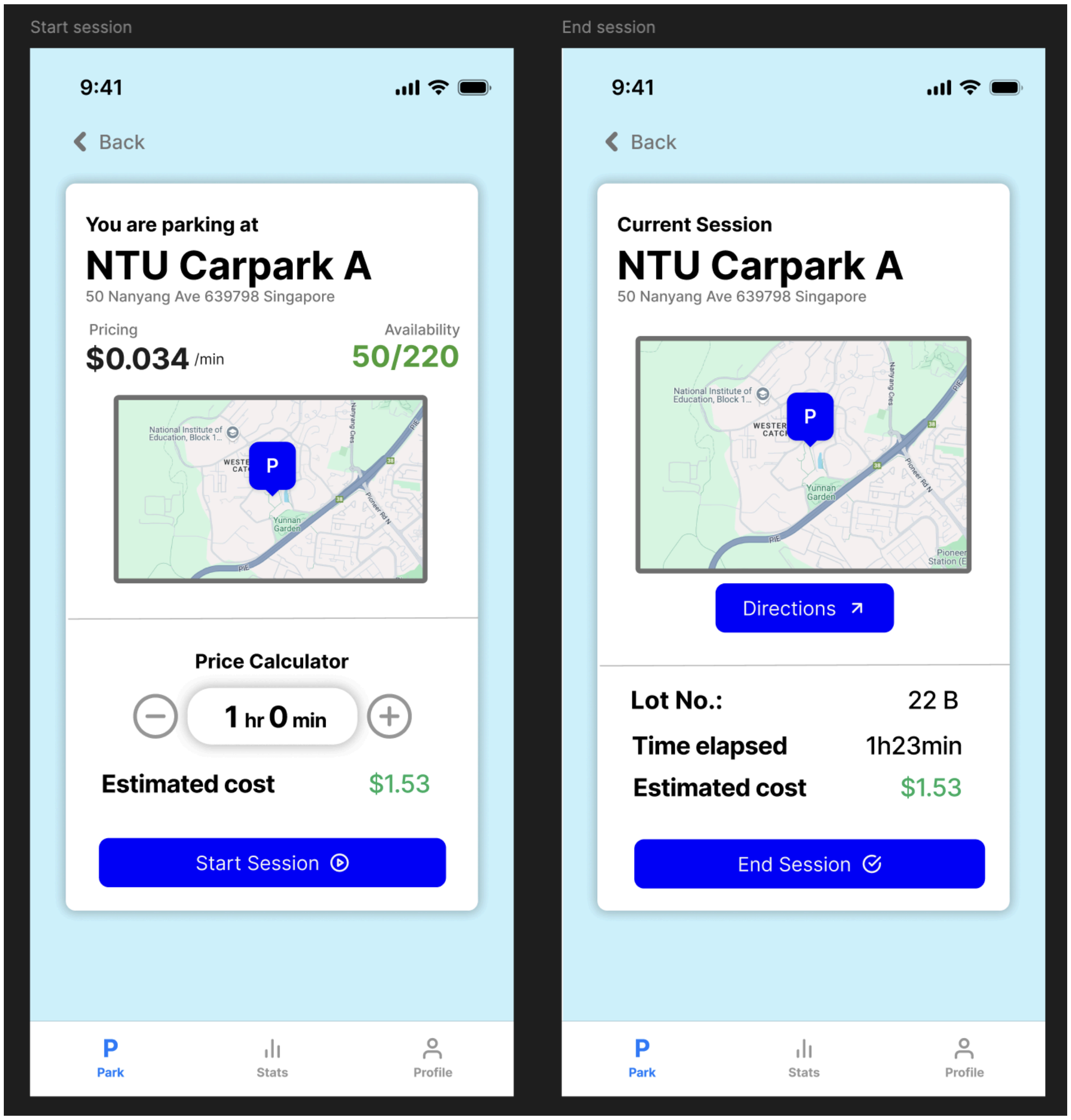


### 4.2.2 List view - View nearby car parks, with filter and sort functionality



| Carpark | Avl Lots | Dist. | $/min |
|---|---|---|---|
| Cheapest Available NTU Carpark A | 50/220 | 90m | 0.034 |
| Nearest Available NTU Carpark F | 50/220 | 44m | 0.040 |
| NTU Carpark E | 50/220 | 100m | 0.042 |

### 4.2.3 Start parking session, End parking session



You are parking at

NTU Carpark A

50 Nanyang Ave 639798 Singapore

Pricing $0.034 /min

Availability 50/220

Price Calculator

1 hr 0 min

Estimated cost $1.53

Start Session

Current Session

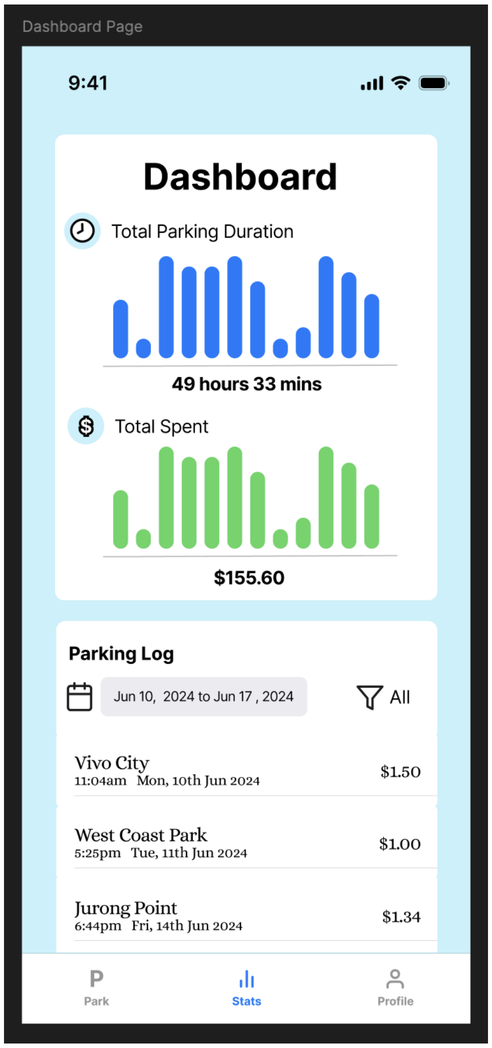
NTU Carpark A

50 Nanyang Ave 639798 Singapore

| | |
|---|---|
| Lot No.: | 22 B |
| Time elapsed | 1h23min |
| Estimated cost | $1.53 |

End Session

## 4.3 Dashboard Menu - Parking statistics + parking log

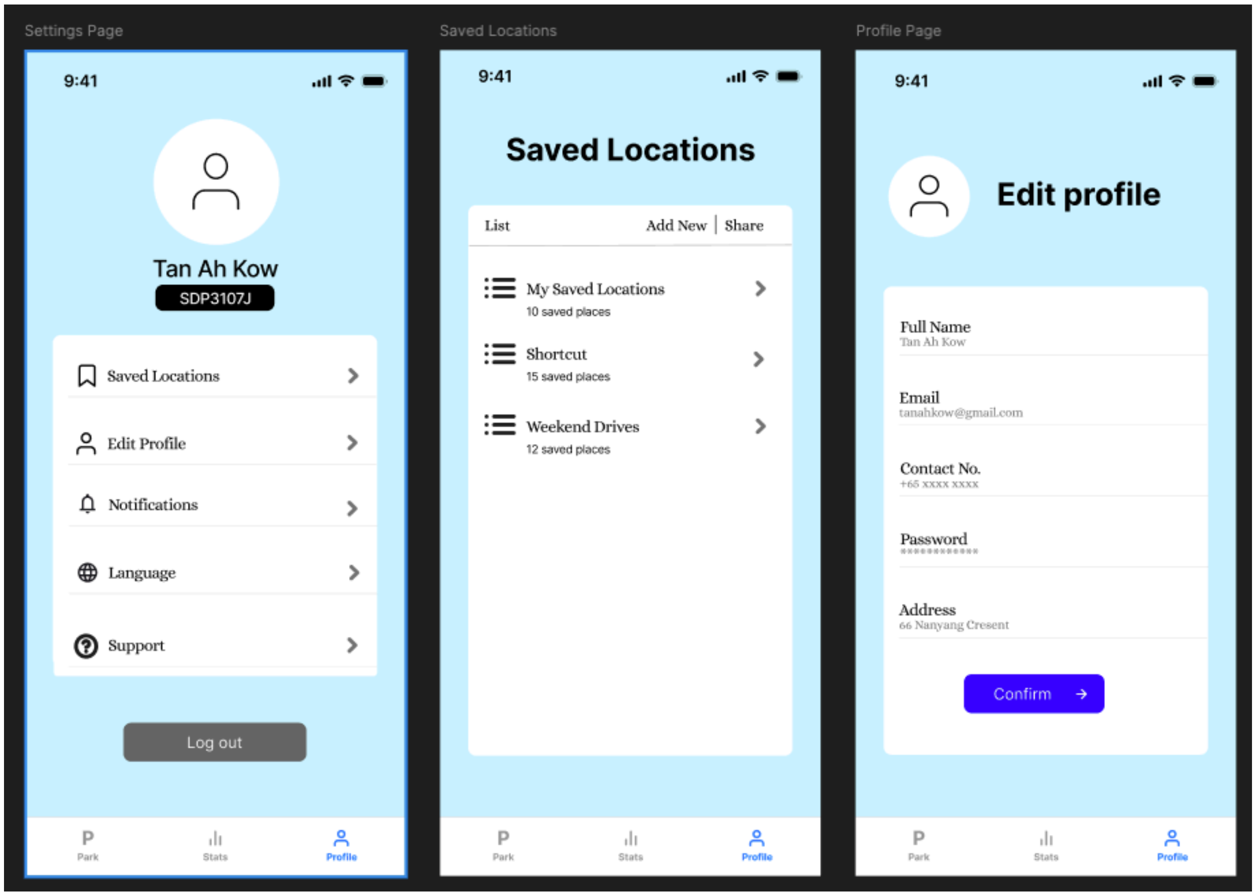


Parking Log

Jun 10, 2024 to Jun 17 , 2024 All

| | |
|---|---|
| Vivo City<br>11:04am Mon, 10th Jun 2024 | $1.50 |
| West Coast Park<br>5:25pm Tue, 11th Jun 2024 | $1.00 |
| Jurong Point<br>6:44pm Fri, 14th Jun 2024 | $1.34 |

## 4.4 Settings and Profile Page



Full Name
Tan Ah Kow

Email
tanahkow@gmail.com

Contact No.
+65 xxxx xxxx

Password
************

Address
66 Nanyang Cresent