Use Cases

for

Parked Up

Version 1.2 approved

Prepared by Dora

NTU SC2006 FDAF Group 2

20/02/25

Revision History

Name	Date	Reason For Changes	Version
Dora	20/02/25	Initial Version	1.0
Yuhe, Marvin	26/03/25	Added Use Case 7 & 8	1.1
Dora, Ethan	09/04/25	Final Version	1.2

Guidance for Use Case Template

Document each use case using the template shown in the Appendix. This section provides a description of each section in the use case template.

1. Use Case Identification

1.1. Use Case ID

Give each use case a unique numeric identifier, in hierarchical form: X.Y. Related use cases can be grouped in the hierarchy. Functional requirements can be traced back to a labeled use case.

1.2. Use Case Name

State a concise, results-oriented name for the use case. These reflect the tasks the user needs to be able to accomplish using the system. Include an action verb and a noun. Some examples:

- View part number information.
- Manually mark hypertext source and establish link to target.
- Place an order for a CD with the updated software version.

1.3. Use Case History

1.3.1 Created By

Supply the name of the person who initially documented this use case.

1.3.2 Date Created

Enter the date on which the use case was initially documented.

1.3.3 Last Updated By

Supply the name of the person who performed the most recent update to the use case description.

1.3.4 Date Last Updated

Enter the date on which the use case was most recently updated.

2. Use Case Definition

2.1. Actor

An actor is a person or other entity external to the software system being specified who interacts with the system and performs use cases to accomplish tasks. Different actors often correspond to different user classes, or roles, identified from the customer community that will use the product. Name the actor(s) that will be performing this use case.

2.2. Description

Provide a brief description of the reason for and outcome of this use case, or a high-level description of the sequence of actions and the outcome of executing the use case.

2.3. Preconditions

List any activities that must take place, or any conditions that must be true, before the use case can be started. Number each precondition. Examples:

1. User's identity has been authenticated.
2. User's computer has sufficient free memory available to launch task.

2.4. Postconditions

Describe the state of the system at the conclusion of the use case execution. Number each postcondition. Examples:AA

1. Document contains only valid SGML tags.
2. Price of item in database has been updated with new value.

2.5. Priority

Indicate the relative priority of implementing the functionality required to allow this use case to be executed. The priority scheme used must be the same as that used in the software requirements specification.

2.6. Frequency of Use

Estimate the number of times this use case will be performed by the actors per some appropriate unit of time.

2.7. Flow of Events

Provide a detailed description of the user actions and system responses that will take place during execution of the use case under normal, expected conditions. This dialog sequence will ultimately lead to accomplishing the goal stated in the use case name and description. This description may be written as an answer to the hypothetical question, "How do I <accomplish the task stated in the use case name>?" This is best done as a numbered list of actions performed by the actor, alternating with responses provided by the system.

2.8. Alternative Flows

Document other, legitimate usage scenarios that can take place within this use case separately in this section. State the alternative course, and describe any differences in the sequence of steps that take place. Number each alternative course using the Use Case ID as a prefix, followed by "AC" to indicate "Alternative Course". Example: X.Y.AC.1.

2.9. Exceptions

Describe any anticipated error conditions that could occur during execution of the use case, and define how the system is to respond to those conditions. Also, describe how the system is to respond if the use

case execution fails for some unanticipated reason. Number each exception using the Use Case ID as a prefix, followed by “EX” to indicate “Exception”. Example: X.Y.EX.1.

2.10. Includes

List any other use cases that are included (“called”) by this use case. Common functionality that appears in multiple use cases can be split out into a separate use case that is included by the ones that need that common functionality.

2.11. Special Requirements

Identify any additional requirements, such as nonfunctional requirements, for the use case that may need to be addressed during design or implementation. These may include performance requirements or other quality attributes.

2.12. Assumptions

List any assumptions that were made in the analysis that led to accepting this use case into the product description and writing the use case description.

2.13. Notes and Issues

List any additional comments about this use case or any remaining open issues or TBDs (To Be Determined) that must be resolved. Identify who will resolve each issue, the due date, and what the resolution ultimately is.

Use Case Template

Use Case ID:	1		
Use Case Name:	Sign Up		
Created By:	Dora	Last Updated By:	Dora
Date Created:	20/02/25	Date Last Updated:	09/04/25

Actor:	User, Database
Description:	User signs up for an account.
Preconditions:	1. The user's device must have access to the internet. 2. The system must be connected to the database. 3. The user must not already have an account.
Postconditions:	User is redirected to login page.
Priority:	Medium
Frequency of Use:	Once for new users
Flow of Events:	1. The user clicks on "Signup" from the navigation bar. 2. The system displays the Signup page. 3. The user is required to enter: a. Username b. Email address c. Phone number d. Password 4. User enters their information and clicks the "Signup" button to sign up. 5. The system validates the information: a. Ensures all fields are filled b. Checks for valid and unused email c. Checks for valid and unused phone number d. Checks password strength based on the following conditions: i. Minimum 8 characters. ii. At least 1 alphabet should be of Upper Case [A-Z] iii. At least 1 number or digit between [0-9]. iv. At least 1 symbol. 6. The system saves their information into the database. 7. The system displays "Signup successful! Please login.". 8. The system displays the login page when user clicks "OK".
Alternative Flows:	1. AF-S5: Email already registered a. The system displays "Email already exists!" until the user clicks "OK" b. Return to step 2 2. AF-S5: Phone number already registered a. The system displays "Failed to create account!" until the user clicks "OK" b. Return to step 2

	3. AF-S4: Weak password a. The system displays the message "Failed to create account!" until the user clicks "OK" b. Return to step 2
Exceptions:	1. EX1: System unavailable a. The app displays that the system is unavailable.
Includes:	N/A
Special Requirements:	N/A
Assumptions:	N/A
Notes and Issues:	N/A

Use Case ID:	2		
Use Case Name:	Login		
Created By:	Dora	Last Updated By:	Dora
Date Created:	20/02/25	Date Last Updated:	09/04/25

Actor:	User, Database
Description:	User logs in to the app using their email and password.
Preconditions:	1. The user's device must have access to the internet. 2. The system must be connected to the database. 3. The user must already have an account.
Postconditions:	The app displays the homepage in map view.
Priority:	Medium
Frequency of Use:	Low
Flow of Events:	1. User clicks on "Login" from the navigation bar. 2. The system displays the login page. 3. The user is required to enter: a. Email b. Password 4. User clicks the "Login" button to login. 5. The system authenticates the user's credentials with the database. 6. User is logged in to their account.
Alternative Flows:	1. AF-S4: Email not in database a. The system displays "Invalid email or password" until the user clicks "OK" b. Return to step 3 2. AF-S4: Wrong password a. The system displays "Invalid email or password" until the user clicks "OK" b. Return to step 3
Exceptions:	1. EX1: System unavailable a. The app displays that the system is unavailable.
Includes:	N/A
Special Requirements:	N/A
Assumptions:	N/A
Notes and Issues:	N/A

Use Case ID:	3		
Use Case Name:	Carpark Search in Map View		
Created By:	Marvin	Last Updated By:	Dora
Date Created:	20/02/25	Date Last Updated:	09/04/25

Actor:	User, Google Maps
Description:	User searches for carparks in map view and navigates to a carpark
Preconditions:	1. The user's device must have access to the internet. 2. User must have location services on their device, and given permission to the system to access it.
Postconditions:	1. User is redirected to Google Maps if they choose to navigate to a carpark. 2. Default map view is displayed if the user exits the search without selecting a carpark.
Priority:	High
Frequency of Use:	High
Flow of Events:	1. User opens the app or clicks "Map View" from the navigation bar. 2. The system requests the user's current location from the device's GPS. 3. The system retrieves carpark data from the API: a. Carpark name b. Address c. Available Lots d. Gantry Height e. Carpark Type 4. The system displays a map of the users' surroundings. The map includes numbers in bubbles. The bubbles represent a single carpark, and the numbers represent the number of available lots in that carpark. 5. User selects a carpark bubble from the map. 6. The system displays carpark details. 7. Users clicks on the "Directions" button to navigate to carpark. 8. User is redirected to Google Maps website for navigation to the selected carpark.
Alternative Flows:	1. AF-S2: System cannot retrieve users' current location a. System displays the message "Error fetching geolocation". b. Return to step 2
Exceptions:	N/A
Includes:	N/A
Special Requirements:	N/A
Assumptions:	N/A
Notes and Issues:	N/A

Use Case ID:	4		
Use Case Name:	Carpark Search in List View		
Created By:	Ethan	Last Updated By:	Dora
Date Created:	20/02/25	Date Last Updated:	09/04/25

Actor:	User, Google Maps
Description:	User searches for carparks in list view and navigates to a carpark
Preconditions:	1. The user's device must have access to the internet. 2. User must have location services on their device, and given permission to the system to access it
Postconditions:	1. User is redirected to Google Maps if they choose to navigate to a carpark. 2. Default list view is displayed if the user exits the search without selecting a carpark.
Priority:	High
Frequency of Use:	High
Flow of Events:	1. User opens the app. Default map view is displayed. 2. The system requests the user's current location from the device's GPS. 3. The system retrieves carpark data from the API: a. Carpark name b. Address c. Available Lots d. Gantry Height e. Carpark Type 4. User clicks on "List View" from the navigation bar. 5. The system displays a list of nearby carparks sorted from nearest to furthest, and their available lots. 6. User selects a carpark from the list to view detailed information. 7. The system displays carpark details. 8. Users clicks on the "Directions" button to navigate to carpark. 9. User is redirected to Google Maps website for navigation to the selected carpark.
Alternative Flows:	1. AF-S2: System cannot retrieve users' current location a. System displays the message "Error fetching geolocation". b. Return to step 2
Exceptions:	N/A
Includes:	N/A
Special Requirements:	N/A
Assumptions:	N/A
Notes and Issues:	N/A

Use Case ID:	5		
Use Case Name:	Filtering in Map View		
Created By:	Ethan	Last Updated By:	Dora
Date Created:	20/02/25	Date Last Updated:	08/04/25

Actor:	User, Google Maps
Description:	User filters carparks by available lots and navigates to a carpark
Preconditions:	1. The user's device must have access to the internet. 2. The user must have location services available on their device, and give permission to the system to access them.
Postconditions:	1. User is redirected to Google Maps if they choose to navigate to a carpark. 2. Default map view is displayed if the user exits the search without selecting a carpark.
Priority:	High
Frequency of Use:	High
Flow of Events:	1. User opens the app or clicks "Map View" from the navigation bar. 2. The system requests the user's current location from the device's GPS. 3. The system retrieves carpark data from the API: a. Carpark name b. Address c. Available Lots d. Gantry Height e. Carpark Type 4. The system displays a map of the users' surroundings. The map includes numbers in bubbles. The bubbles represent a single carpark, and the numbers represent the number of available lots in that carpark. 5. User clicks the filter icon on the page. 6. The system displays the carpark filters panel, allowing the user to filter by available lots, gantry height, or carpark type. 7. The user filters carparks based on available lots (e.g. at least 30 available lots) 8. The system applies the filter and updates the map accordingly, only displaying carparks that fulfill the requirements. 9. User selects a carpark bubble from the map. 10. The system displays carpark details. 11. Users clicks on the "Directions" button to navigate to carpark. 12. User is redirected to Google Maps website for navigation to the selected carpark.
Alternative Flows:	1. AF-S2: System cannot retrieve users' current location a. System displays the message "Error fetching geolocation".

	b. Return to step 2
Exceptions:	N/A
Includes:	N/A
Special Requirements:	N/A
Assumptions:	N/A
Notes and Issues:	Instead of filtering by available lots, the user may also filter by carpark type, or gantry height.

Use Case ID:	6		
Use Case Name:	Add Favourite Carpark in Map View		
Created By:	Yuhe	Last Updated By:	Dora
Date Created:	20/02/25	Date Last Updated:	09/04/25

Actor:	User, Google Maps, Database
Description:	User adds a favourite carpark to their account from the map.
Preconditions:	1. The user's device must have access to the internet. 2. The user must be logged into their account.
Postconditions:	1. The carpark is saved to the user's account and appears in the "Favourites" tab if the user chooses to add it. 2. The carpark is not saved and does not appear in the "Favourites" tab if the user exits before adding it.
Priority:	High
Frequency of Use:	High
Flow of Events:	1. User opens the app or clicks "Map View" from the navigation bar. 2. The system requests the user's current location from the device's GPS. 3. The system retrieves carpark data from the API: a. Carpark name b. Address c. Available Lots d. Gantry Height e. Carpark Type 4. The system displays a map of the users' surroundings. The map includes numbers in bubbles. The bubbles represent a single carpark, and the numbers represent the number of available lots in that carpark. 5. User selects a carpark bubble from the map. 6. The system displays carpark details. 7. User clicks on the heart icon. Carpark is saved to their favourites. 8. User clicks on "Favourites" from the navigation bar. 9. The system displays the user's list of favourite carparks.
Alternative Flows:	N/A
Exceptions:	N/A

Includes:	N/A
Special Requirements:	N/A
Assumptions:	N/A
Notes and Issues:	N/A

Use Case ID:	7		
Use Case Name:	Manage profile		
Created By:	Marvin	Last Updated By:	Dora
Date Created:	19/03/25	Date Last Updated:	09/04/25

Actor:	User, Database
Description:	Users views their profile and may update their information (i.e. email, password, username and/or phone number)
Preconditions:	1. The user's device must have access to the internet. 2. User must be logged into their account.
Postconditions:	1. The system displays the updated profile page if the user successfully updates their information. 2. The system displays the old information if the user exits before successfully updating their information.
Priority:	Medium
Frequency of Use:	Low
Flow of Events:	1. User clicks on the profile icon on the navigation bar. 2. The system displays the current user's information. 3. User inputs any updated fields. 4. User inputs their current password for verification. 5. User clicks on "Update Profile" to confirm the update. 6. The system validates the information: a. Checks for valid and unused email b. Checks for valid and unused phone number c. Checks password strength based on the following conditions: i. Minimum 8 characters. ii. At least 1 alphabet should be of Upper Case [A-Z] iii. At least 1 number or digit between [0-9]. iv. At least 1 symbol. d. Checks that the current password is correct. 7. The system updates their information into the database.
Alternative Flows:	1. AF-S3: The user cancels the update process a. The system returns to the previous page without making any profile changes 2. AF-S4: The user enters an incorrect password a. The system displays "Current password is incorrect!" until the user clicks "OK" b. Return to step 4 3. AF-S6: The user's input is invalid a. The system displays "Failed to update profile!" until the user clicks "OK".

	b. Return to step 3
Exceptions:	N/A
Includes:	N/A
Special Requirements:	N/A
Assumptions:	N/A
Notes and Issues:	N/A

Use Case ID:	8		
Use Case Name:	Remove a Favourite Carpark		
Created By:	Yuhe	Last Updated By:	Dora
Date Created:	19/03/25	Date Last Updated:	09/04/25

Actor:	User
Description:	User removes a favourite carpark from the “Favourites” page.
Preconditions:	1. The user’s device must have access to the internet. 2. The user must be logged into their account. 3. The user must already have a carpark in their “Favourites” list
Postconditions:	1. The system displays the updated “Favourites” page if the user successfully removes the carpark from their favourites. 2. The “Favourites” page remains the same if the user exits before removing the carpark from their favourites.
Priority:	High
Frequency of Use:	High
Flow of Events:	1. The user clicks on "Favourites" from the navigation bar. 2. The system displays the list of the user’s favourite carparks. 3. The user selects the carpark they want to remove. 4. The system displays the carpark details. 5. The user clicks the "Remove" button. 6. The system removes the carpark from the user’s favourite carparks in the database.
Alternative Flows:	N/A
Exceptions:	N/A
Includes:	N/A
Special Requirements:	N/A
Assumptions:	N/A
Notes and Issues:	N/A