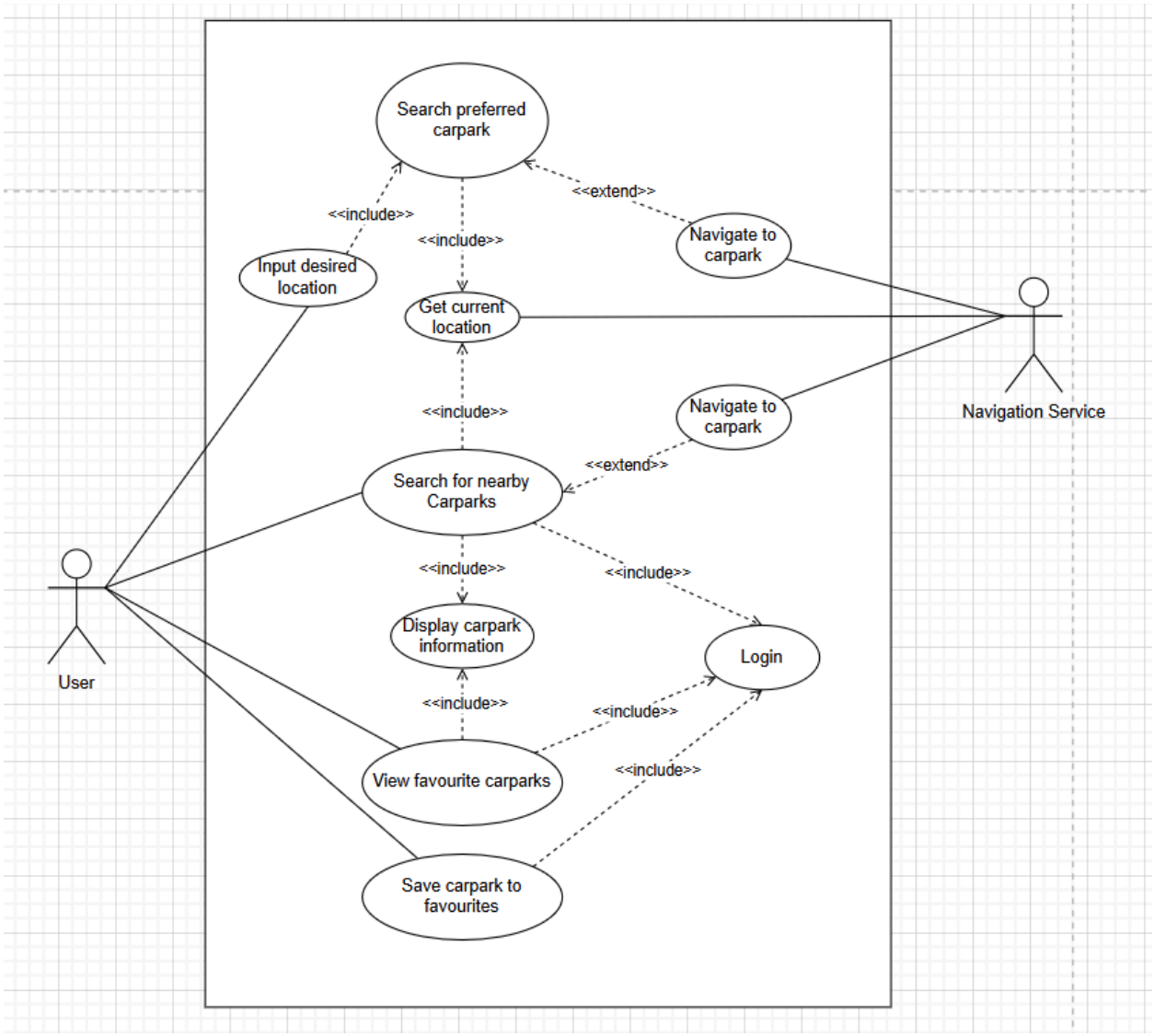
Diagrams

Use Case Diagram



Use Case Templates

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

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:	The user can login to the app.
Priority:	High
Frequency of Use:	Once for new users
Flow of Events:	1. The user selects "Sign Up" on the login page. 2. The system prompts the user to enter: a. Username b. Email address c. Phone number d. Password 3. User enters their information and clicks confirm. 4. 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 certain requirements) 5. The system saves their information into the database. 6. The user can now log in and use the system.
Alternative Flows:	1. AF-S5: Email already registered a. The system displays "Email already in use. Please log in or reset your password." until user clicks "OK" 2. AF-S5: Phone number already registered a. The system displays "Phone number already in use. Please log in or reset your password." until user clicks "OK" 3. AF-S4: Weak password (min. 10 char, special characters, upper/lower case) a. The system displays the message "Password is too weak" until user clicks "OK" b. Return to step 2
Exceptions:	1. EX1: System unavailable a. The app displays that the system is unavailable.
Includes:	
Special Requirements:	
Assumptions:	
Notes and Issues:	

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

Actor:	User Database
Description:	User logs in to the app using 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:	High
Frequency of Use:	Low
Flow of Events:	1. User opens the app. 2. The system displays the login page. 3. User enters their email and password, then clicks the 'login' button. 4. The system authenticates the user's information with the database. 5. User is logged in to their respective account, and can access the app.
Alternative Flows:	1. AF-S4: System cannot authenticate the user's information a. The system displays "Incorrect information. Try again" 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:	
Special Requirements:	
Assumptions:	
Notes and Issues:	

Use Case ID:	3(marvin)		
Use Case Name:	Nearby Carpark Search in Map View		
Created By:	Marvin	Last Updated By:	
Date Created:	20/02/25	Date Last Updated:	

Actor:	User, Google Maps
Description:	User wants to find carparks near their current location in map view
Preconditions:	1. The user's device must have access to the internet. 2. User must be logged in 3. User must have location services on their device, and given permission to the system to access it
Postconditions:	1) The user successfully finds a carpark and chooses to navigate there. 2) The user exits the search without selecting a carpark.
Priority:	High Priority
Frequency of Use:	Very Frequent
Flow of Events:	1) User opens app 2) The system requests the user's current location from the device's GPS. 3) The system retrieves carpark data (i.e. availability of parking lots, parking rates, carpark type, carpark height) within the user's current location. 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. The user clicks a bubble to get more carpark details (i.e. availability of parking lots, parking rates, carpark type, carpark height) 5) User selects a carpark from the map to view detailed information. 6) The system displays carpark details. 7) User may choose to navigate to the carpark by selecting "Get Directions." 8) System displays an in-app map view that users can access directly without needing to switch to an external map service. 9) The map will help the user navigate to his desired location with clear guidance. 10) Use case ends when the user either selects a carpark or exits the search.
Alternative Flows:	1. AF-S2: System cannot retrieve users' current location

	(a) System displays the message “Cannot find your current location, please enable location services” until user clicks “OK” (b) System returns to step 1 2. AF-S8: Device does not have a navigation service installed (a) System displays the message “Cannot direct you to a navigation service” until user clicks “OK” (b) System returns to step 6
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	
Notes and Issues:	

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

Actor:	User, Google Maps
Description:	User wants to find carparks near their current location in list view
Preconditions:	1. The user's device must have access to the internet. 2. User must be logged in 3. User must have location services on their device, and given permission to the system to access it
Postconditions:	3) The user successfully finds a carpark and chooses to navigate there. 4) The user exits the search without selecting a carpark.
Priority:	High Priority
Frequency of Use:	Very Frequent
Flow of Events:	1) User opens app 2) The system requests the user's current location from the device's GPS. 3) The system retrieves carpark data (i.e. availability of parking lots, parking rates, carpark type, carpark height) within the user's current location. 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. The user clicks a bubble to get more carpark details (i.e. availability of parking lots, parking rates, carpark type, carpark height) 5) The user selects the search bar on the top of the screen and enters list mode 6) The system displays a list of nearby carparks (sorted from nearest to farthest) 7) User selects a carpark from the list to view detailed information. 8) The system displays carpark details. 9) User may choose to navigate to the carpark by selecting "Get Directions." 10) System displays an in-app map view that users can access directly without needing to switch to an external map service. 11) The map will help the user navigate to his desired location with clear guidance. 12) Use case ends when the user either selects a carpark or exits the search.
Alternative Flows:	3. AF-S2: System cannot retrieve users' current location

	(c) System displays the message “Cannot find your current location, please enable location services” until user clicks “OK” (d) System returns to step 1 4. AF-S8: Device does not have a navigation service installed (c) System displays the message “Cannot direct you to a navigation service” until user clicks “OK” (d) System returns to step 6
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	
Notes and Issues:	

Use Case ID:	5		
Use Case Name:	Filtering and Selecting Destination in Map View		
Created By:	Ethan	Last Updated By:	
Date Created:	20/02/25	Date Last Updated:	

Actor:	User
Description:	User Searches for a Carpark by Desired Location, and filters by available lots
Preconditions:	1. The user’s device must have access to the internet. 2. The user must be logged in 3. The user must have location services available on their device, and give permission to the system to access them.
Postconditions:	1. The user finds a carpark with available lots and chooses to navigate there. 2. The user exits the search without selecting a carpark.
Priority:	High
Frequency of Use:	Very frequent
Flow of Events:	1. User opens app 2. The system requests the user's current location from the device's GPS. 3. The system retrieves carpark data (i.e. availability of parking lots, carpark type, carpark height) within the user's current location.

	4. The system displays a map of the users' surroundings. The map includes numbers in bubbles. a. The bubbles represent a single carpark b. The number in each bubble represents the number of available lots in that carpark. 5. The user may choose to view the filtered results in the map view (default) or in a list view (sorted nearest to farthest). 6. The user selects a location by either: a. Clicking a point on the map to search that location; or b. Typing a location in the search bar and selecting a result. 7. The system retrieves carpark data within the user's newly selected location. 8. The user selects the filter option, and filters carparks based on their availability (e.g. only show carparks with more than 30 available lots) 9. The system applies the filter and updates the map or list accordingly. 10. The user selects a carpark to view its details. 11. The system displays carpark details (i.e. availability of parking lots, carpark type, carpark height) 12. The user may choose to navigate to the carpark by selecting "Get Directions." 13. The system redirects the user to a navigation service (Google Maps). 14. Use case ends when the user either selects a carpark or exits the search.
Alternative Flows:	1. AF-S7: User's current location cannot be determined a. The system displays the message: "Cannot find your current location, please enable location services" until user clicks "OK" b. The user can: i. Enable location services and retry. ii. Manually enter a location in the search bar. 2. AF-S8: System fails to retrieve real-time parking data a. The system displays the message: "Live parking data unavailable. Showing last updated information." b. The user can: i. Continue with the last available data. ii. Try refreshing the search later.
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	

Notes and Issues:	Instead of filtering by available lots, the user may also filter by carpark type, or carpark height.

Use Case ID:	6		
Use Case Name:	Save a carpark		
Created By:	Yuhe	Last Updated By:	
Date Created:	20/02/25	Date Last Updated:	

Actor:	User
Description:	This use case allows a logged-in user to save a selected carpark to their account, enabling quick access to real-time parking information from the "Saved Carparks" section. The user can also remove a previously saved carpark if they no longer need it.
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 have performed a carpark search.
Postconditions:	1. Success: a. The carpark is saved to the user's account and appears in the "Saved Carparks" section. b. If the user chooses to unsave a carpark, it is removed from their "Saved Carparks" list. 2. Failure: The user cancels the action or exits before saving.
Priority:	High
Frequency of Use:	Frequent
Flow of Events:	Saving a Carpark 1. The user searches for a carpark. 2. The system displays the search results. 3. The user selects a carpark from the list. 4. The user clicks the "Save Carpark" button. 5. The system checks if the carpark is already in the user's saved list. (Alternative Flow: AF-S10) If the carpark is already saved, proceed to AF-S10. 6. The system saves the selected carpark to the user's account. 7. The user can access their saved carparks from the "Saved Carparks" section and view real-time parking availability. Unsaving a Carpark 1. The user navigates to the "Saved Carparks" section. 2. The system displays the list of saved carparks. 3. The user selects a carpark they want to remove. 4. The user clicks the "Unsave" button. 5. The system prompts for confirmation: "Are you sure you want to remove this carpark from your saved list?" 6. The user confirms the action. 7. The system removes the carpark from the user's saved list.
Alternative Flows:	AF-S10: Carpark Already Saved

	a. The system displays a message: "This carpark is already in your favorites." b. The user acknowledges the message by clicking “OK”.
Exceptions:	
Includes:	Search for a Carpark
Special Requirements:	The system must support real-time parking availability updates.
Assumptions:	
Notes and Issues:	

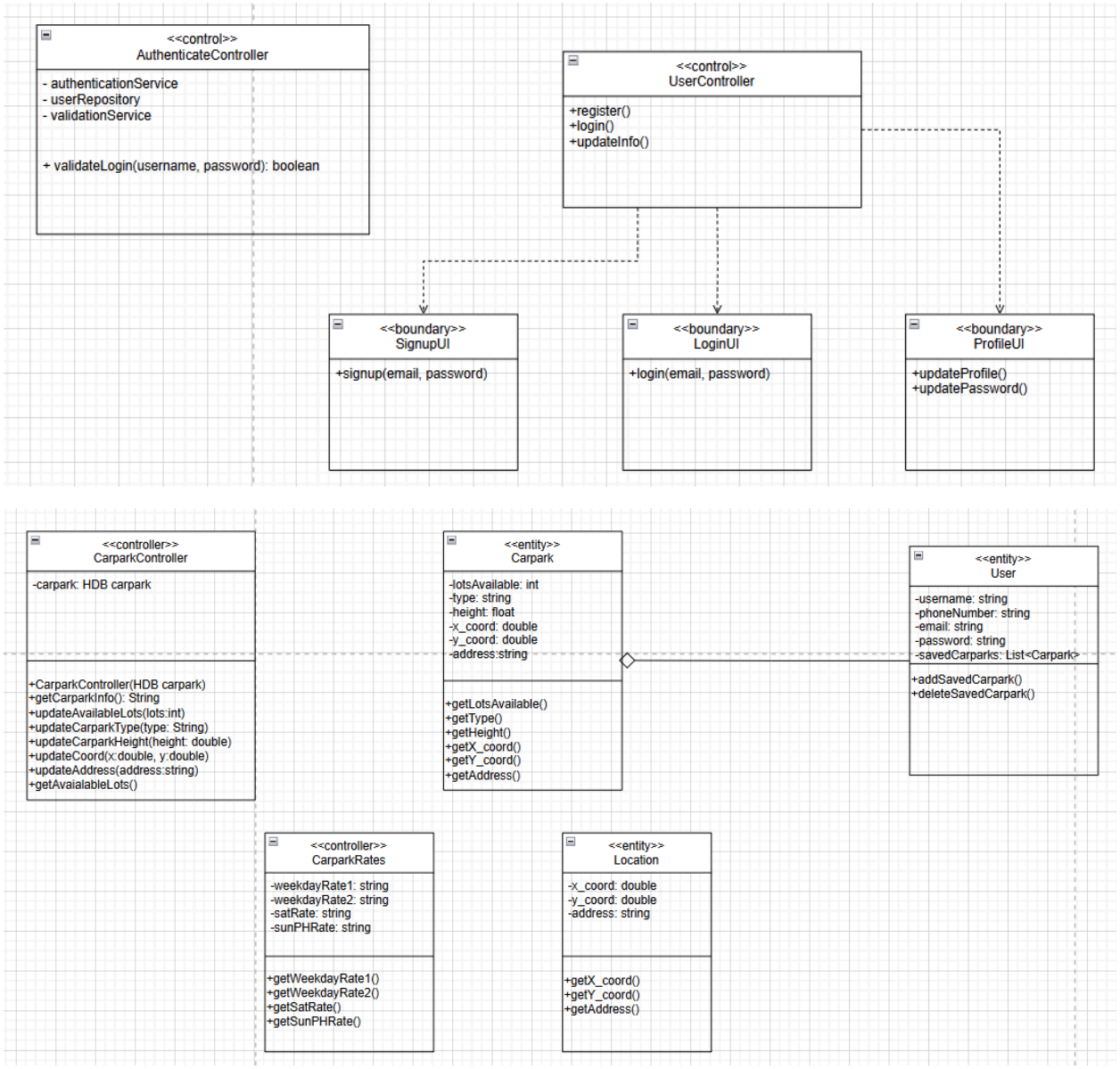
Use Case ID:	7		
Use Case Name:	Manage profile		
Created By:	Marvin	Last Updated By:	
Date Created:	19/03/25	Date Last Updated:	

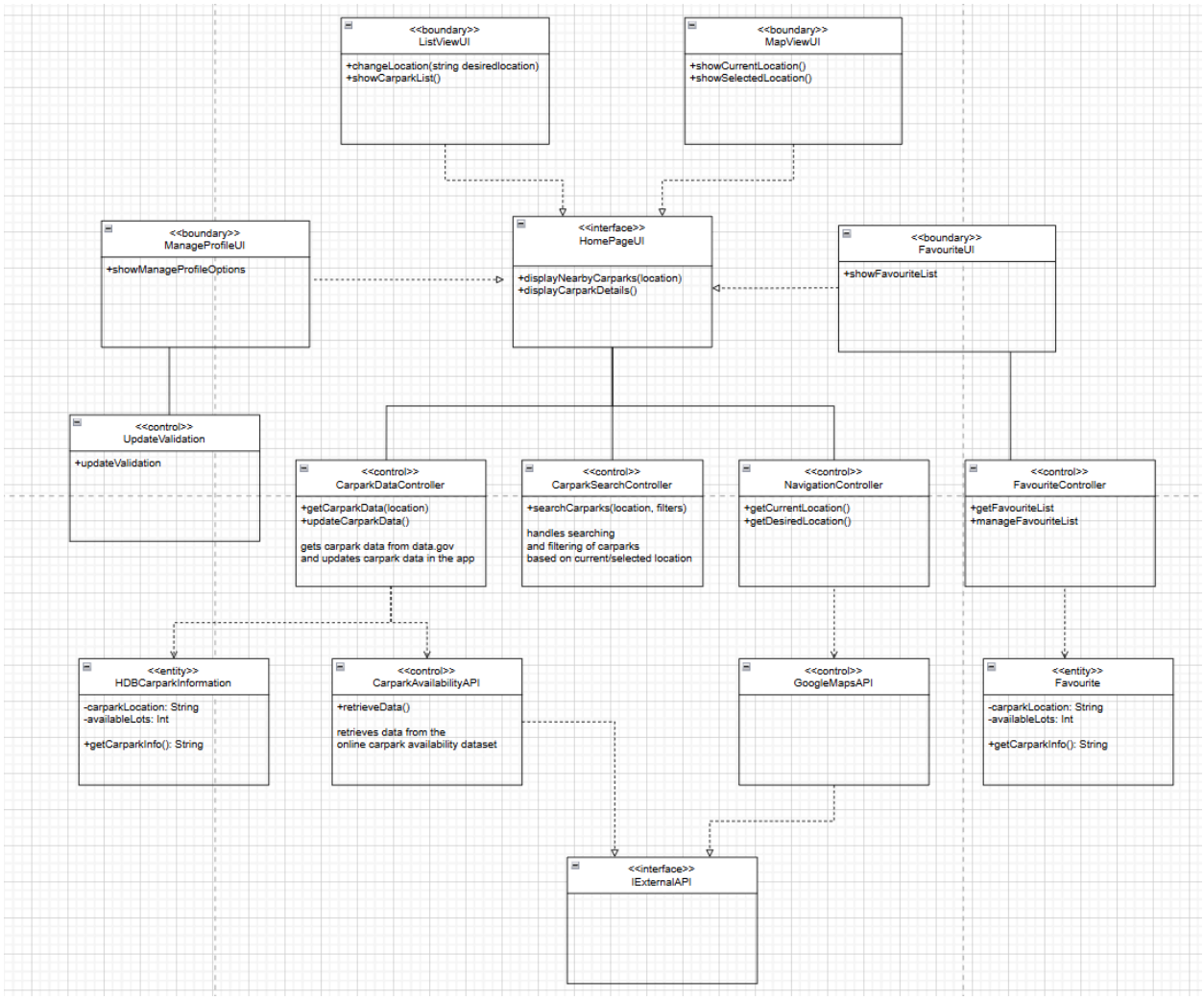
Actor:	User
Description:	Allows user to view their profile, change their email, password, username and phone number.
Preconditions:	1. User must already have an account registered 2. User must be logged in
Postconditions:	1. The user’s profile changes are updated in the database 2. The user is directed to the profile page with the relevant changes updated and can be seen
Priority:	Low
Frequency of Use:	Occasional(Users may update their profile settings periodically)
Flow of Events:	1. The user accesses the “Manage Profile” section 2. The system displays the current user’s information 3. The user selects the information they want to update(Username, Email, Password, Phone Number) 4. The user submits the updates 5. The system validates the inputs 6. The system updates the profile with the new information
Alternative Flows:	AF-5: The user’s input is invalid a. The system raises a prompt to the user to correct their input to meet the requirements AF-3: The user cancels the update process a. The system returns to the previous page without making any profile changes
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	
Notes and Issues:	

Use Case ID:	8		
Use Case Name:	View and Manage Favourites		
Created By:	Yuhe	Last Updated By:	
Date Created:	19/03/25	Date Last Updated:	

Actor:	User
Description:	This use case allows a logged-in user to save a selected carpark to their account, enabling quick access to real-time parking information from the "Saved Carparks" section. The user can also remove a previously saved carpark if they no longer need it.
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 have performed a carpark search.
Postconditions:	1. Success: If the user chooses to unsave a carpark, it is removed from their "Saved Carparks" list. 2. Failure: The user cancels the action or exits before saving.
Priority:	High
Frequency of Use:	Frequent
Flow of Events:	1. The user can access their saved carparks from the "Saved Carparks" section and view real-time parking availability. Unsaving a Carpark 1. The user navigates to the "Saved Carparks" section. 2. The system displays the list of saved carparks. 3. The user selects a carpark they want to remove. 4. The user clicks the "Unsave" button. 5. The system prompts for confirmation: "Are you sure you want to remove this carpark from your saved list?" 6. The user confirms the action. 7. The system removes the carpark from the user's saved list.
Alternative Flows:	
Exceptions:	
Includes:	
Special Requirements:	The system must support real-time parking availability updates.
Assumptions:	
Notes and Issues:	

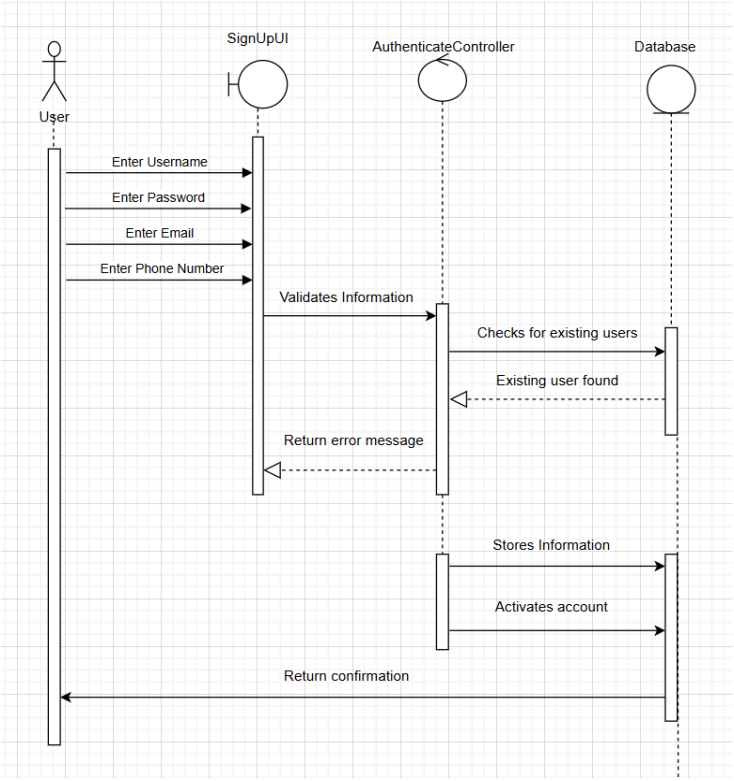
Class Diagram



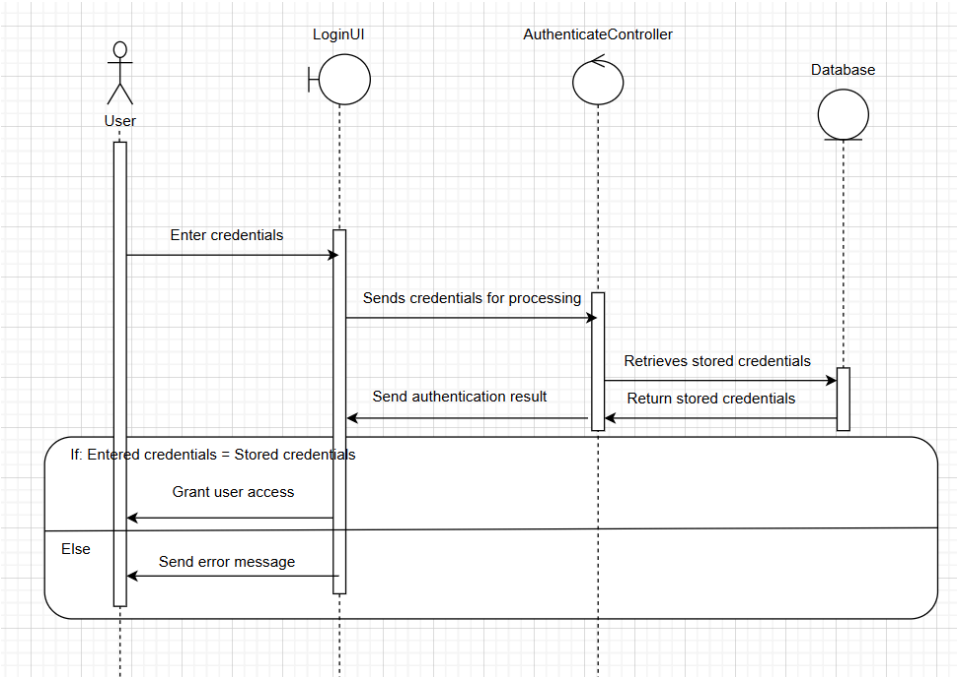

<<boundary>>
ListViewUI
+changeLocation(string desiredlocation)
+showCarparkList()
<<boundary>>
MapViewUI
+showCurrentLocation()
+showSelectedLocation()
<<boundary>>
ManageProfileUI
+showManageProfileOptions
<<interface>>
HomePageUI
+displayNearbyCarparks(location)
+displayCarparkDetails()
<<boundary>>
FavouriteUI
+showFavouriteList
<<control>>
UpdateValidation
+updateValidation
<<control>>
CarparkDataController
+getCarparkData(location)
+updateCarparkData()
gets carpark data from data.gov
and updates carpark data in the app
<<control>>
CarparkSearchController
+searchCarparks(location, filters)
handles searching
and filtering of carparks
based on current/selected location
<<control>>
NavigationController
+getCurrentLocation()
+getDesiredLocation()
<<control>>
FavouriteController
+getFavouriteList
+manageFavouriteList
<<entity>>
HDBCarparkInformation
-carparkLocation: String
-availableLots: Int
+getCarparkInfo(): String
<<control>>
CarparkAvailabilityAPI
+retrieveData()
retrieves data from the
online carpark availability dataset
<<control>>
GoogleMapsAPI
<<entity>>
Favourite
-carparkLocation: String
-availableLots: Int
+getCarparkInfo(): String
<<interface>>
IExternalAPI

Sequence Diagrams

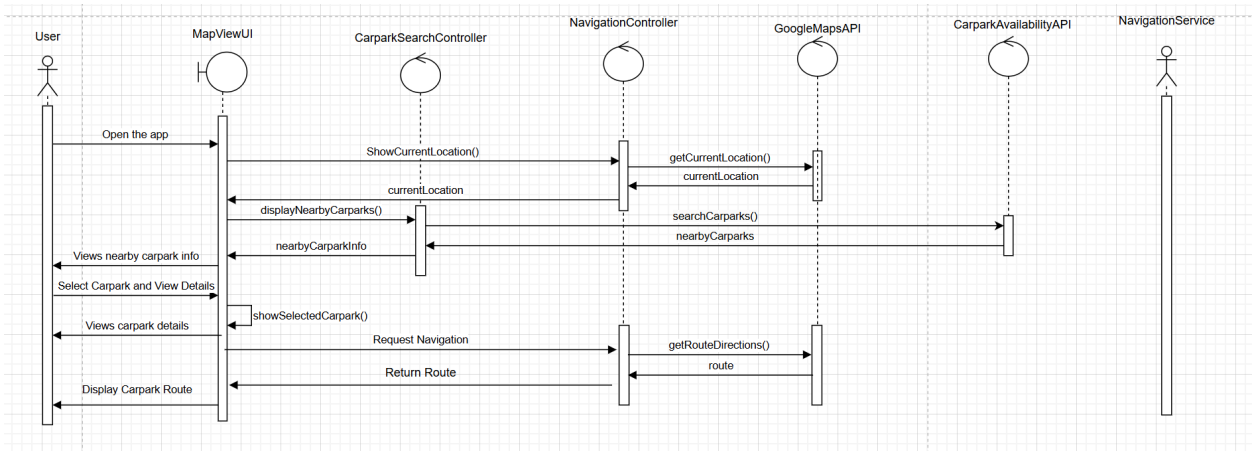
Use case 1: sign up



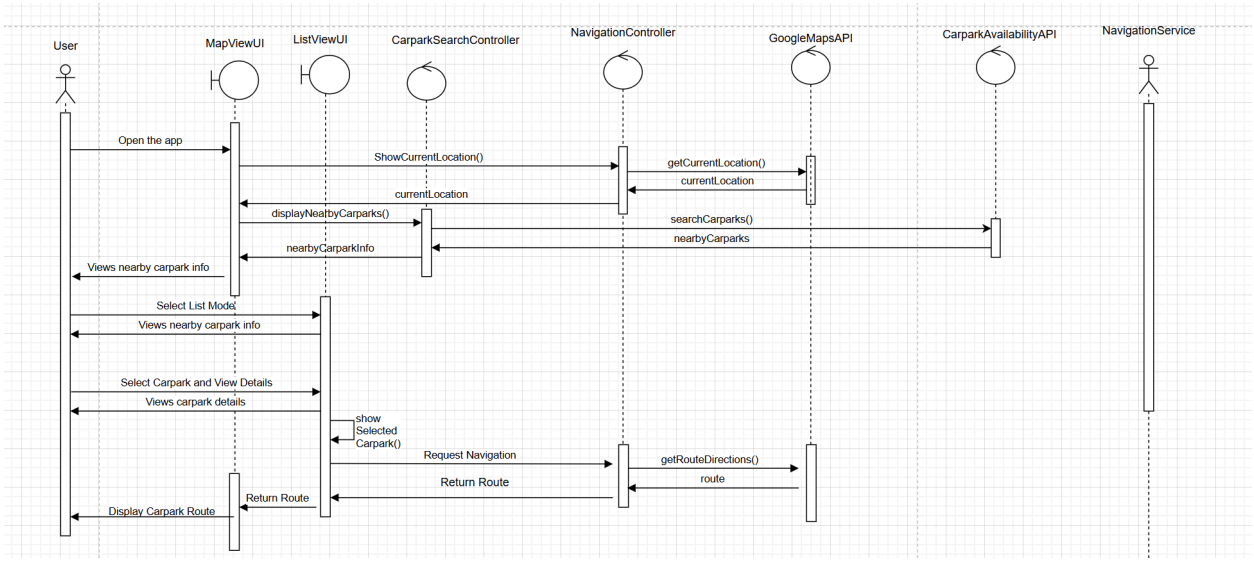
Use case 2: login



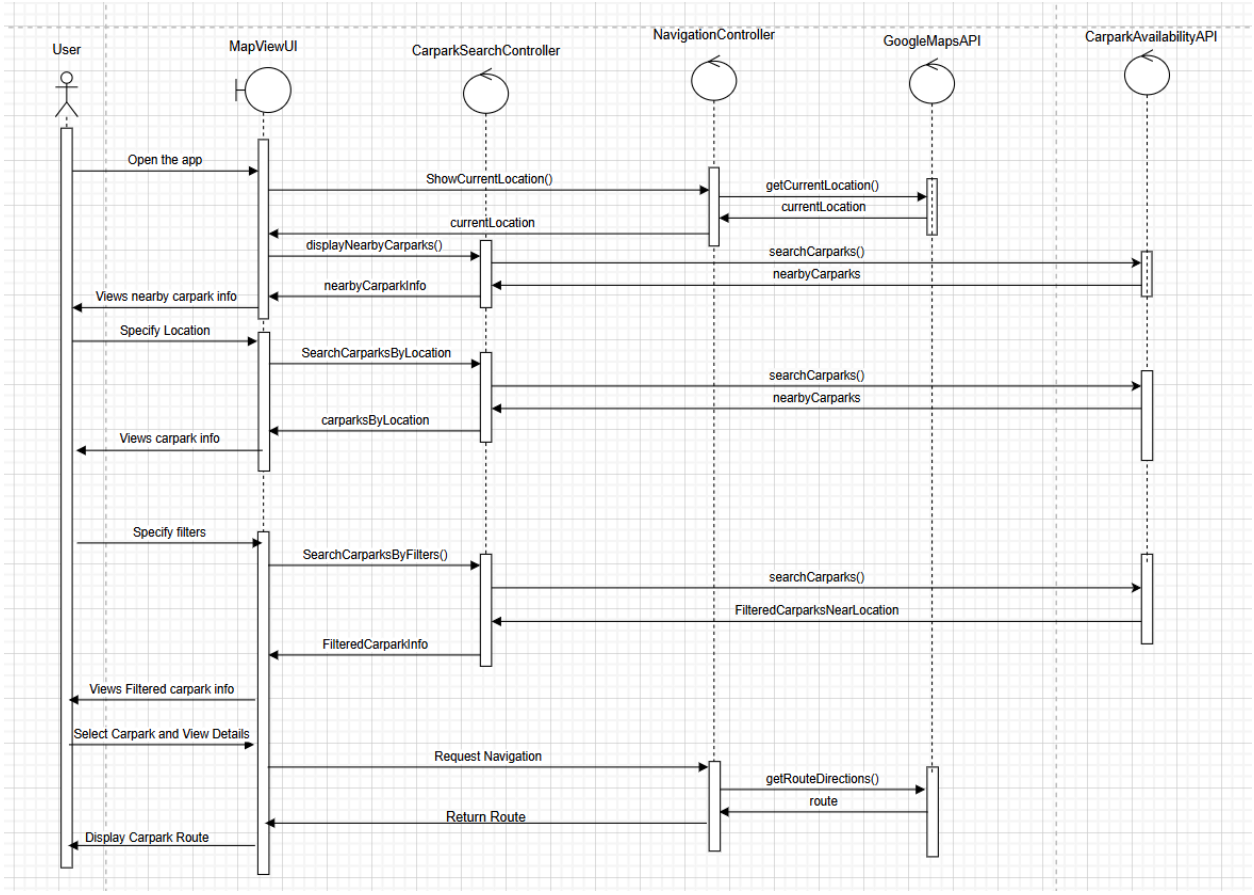
Use case 3: Nearby Carpark Search in Map View



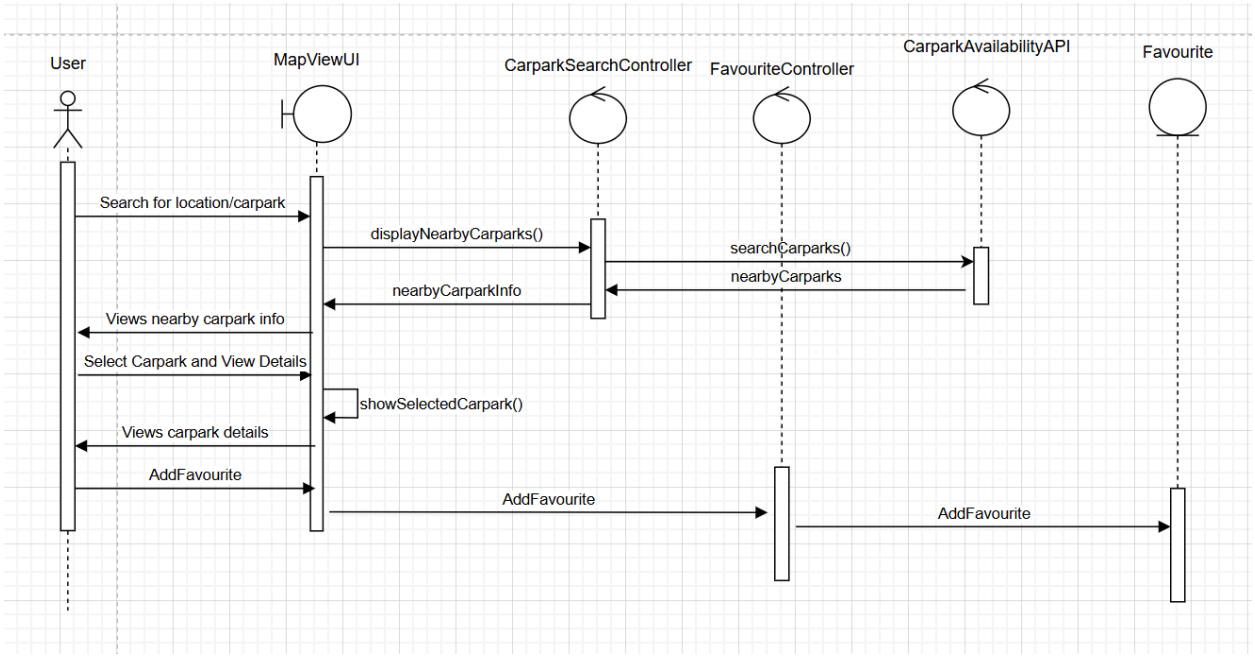
Use case 4: Nearby Carpark Search in List View



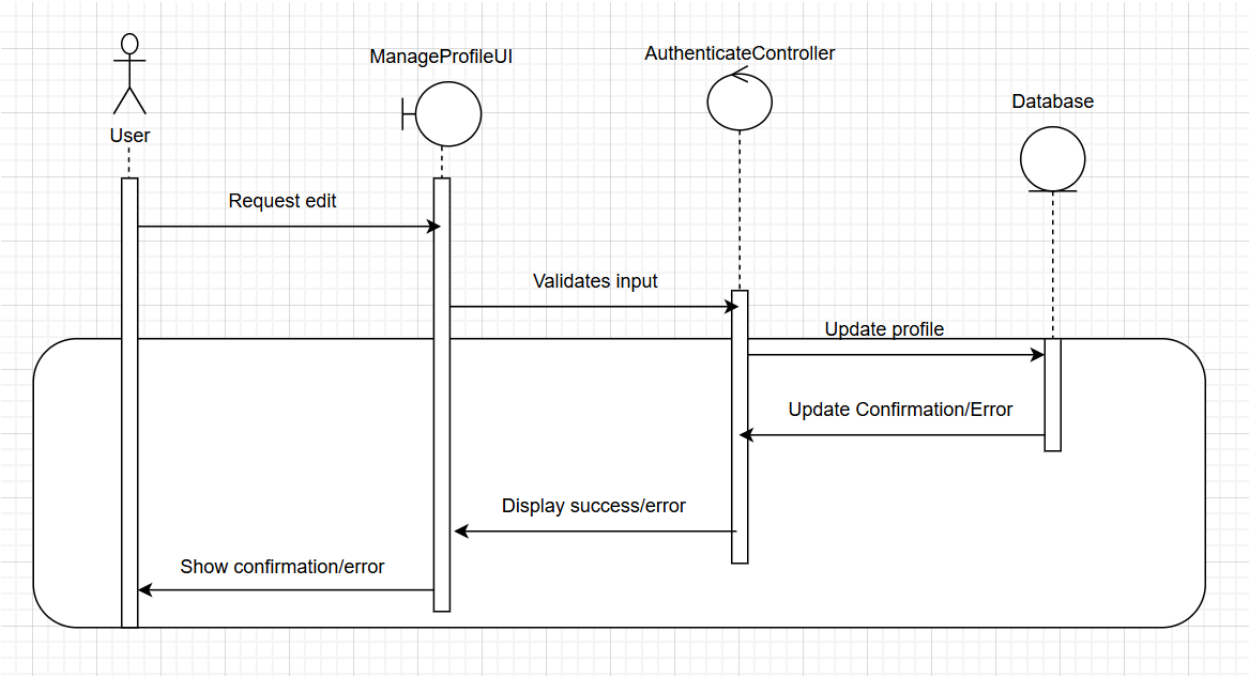
Use case 5: Filtering and Selecting Destination in Map View



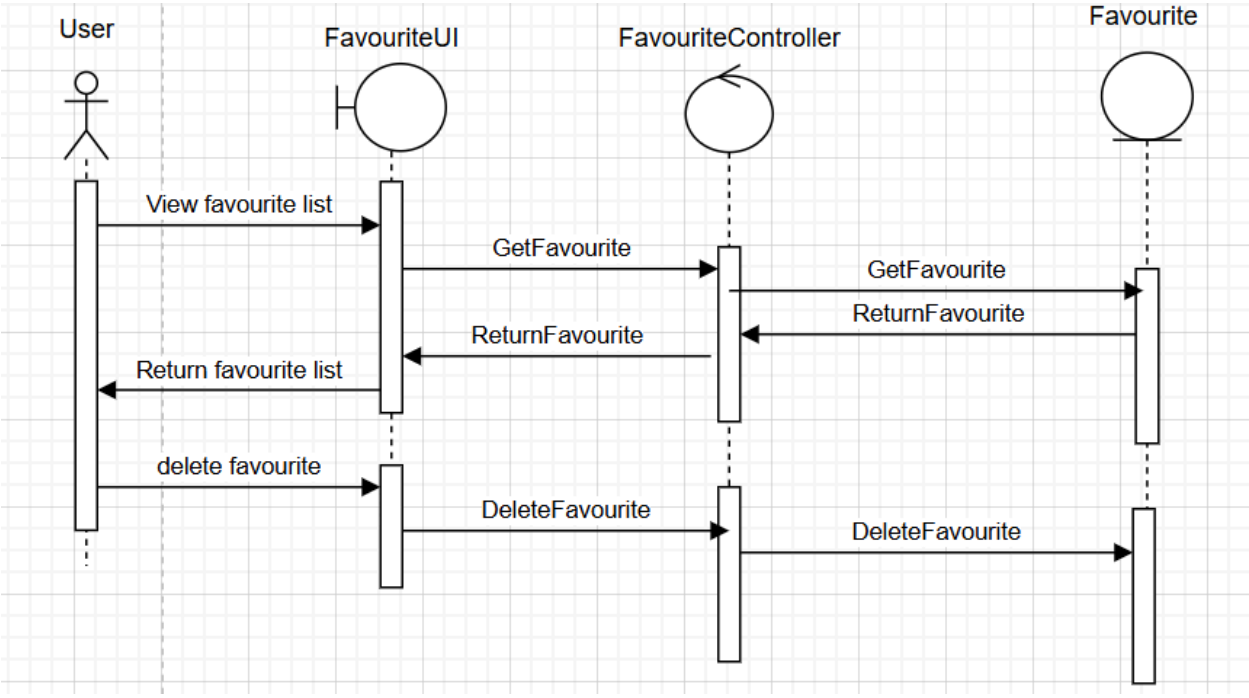
Use case 6: Save a carpark



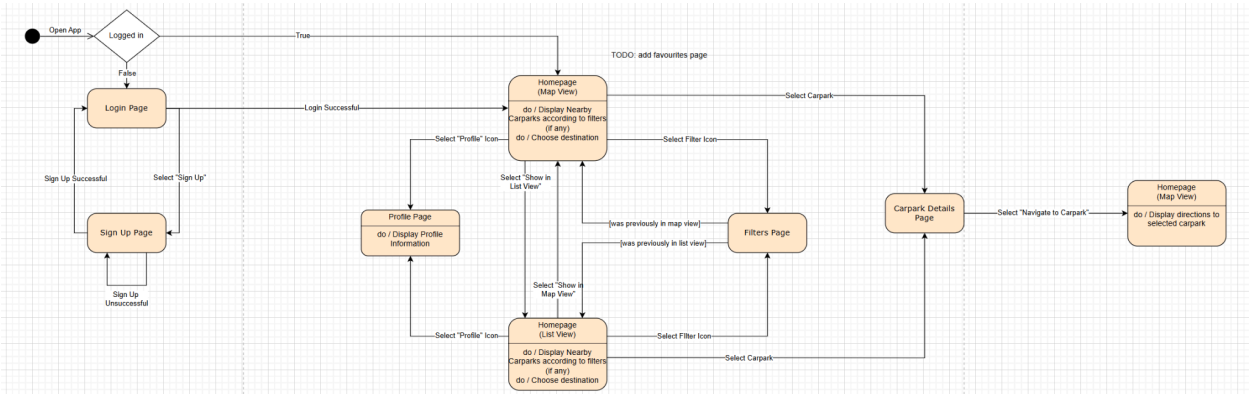
Use case 7: Manage profile



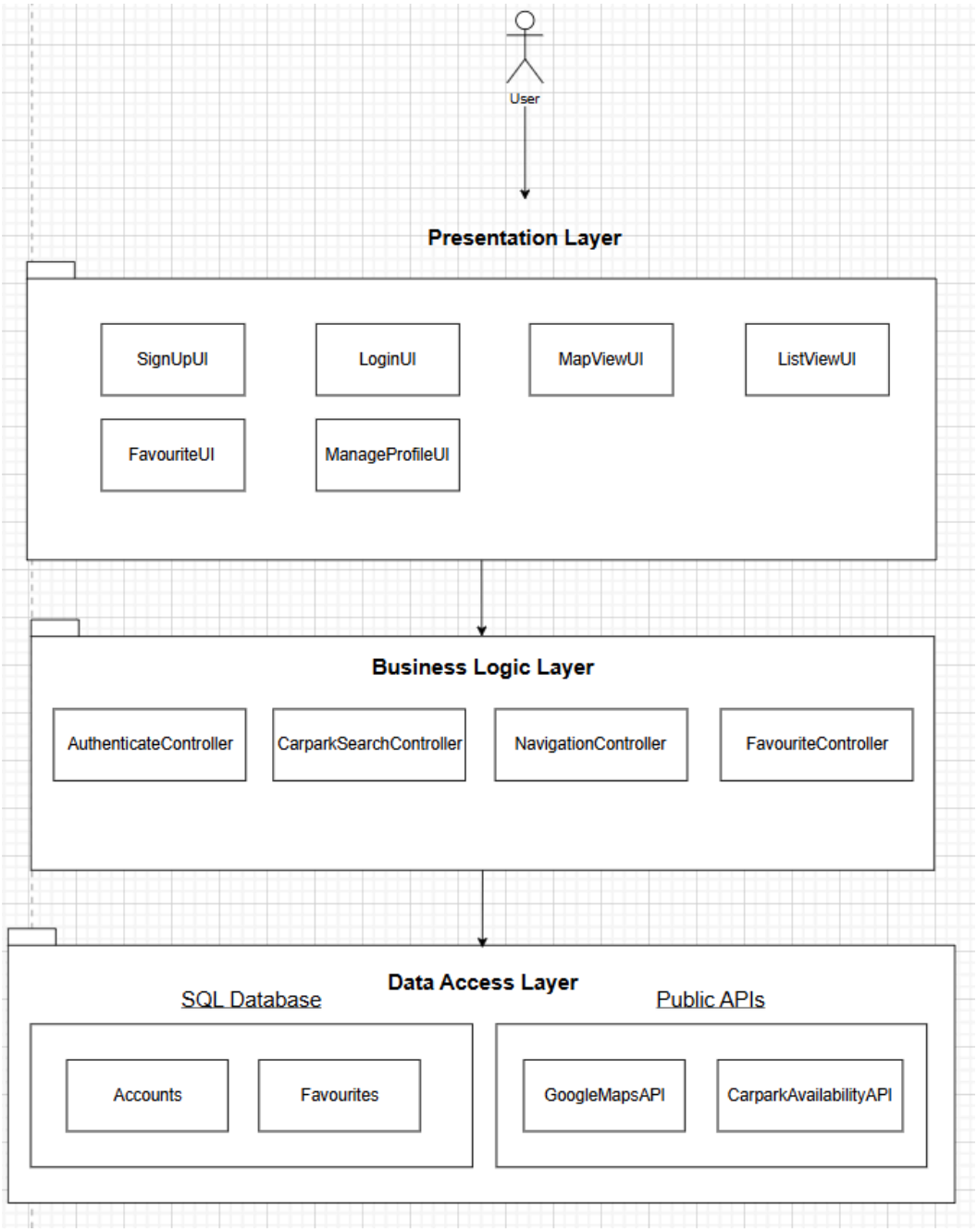
Use case 8: View and Manage Favourites



Initial Dialog Map



System Architecture



Other Deliverables

Identifying and storing persistent data

- SQL database
Tables: Accounts, Favourites

Providing access control

There is no admin role -- no access control needed