Home Helper Finder System

**Software Requirements Specification**

Version 1.0

Danang, February 2025

**TABLE OF CONTENTS**

**TABLE OF CONTENTS**. 2

[**1. DOCUMENT HISTORY 3**](#_6fjx8adf9a3p)

[**2. REFERENCE DOCUMENTS 4**](#_jrnxi0z80tsy)

[**3. INTRODUCTION 4**](#_nd9tt24jub99)

[3.1 Purpose 4](#_g3gah9nlcrrh)

[3.2 In Scope 5](#_a6mf6uqioisp)

[**4. OVERVIEW 5**](#_mnrij95yz507)

[4.1 Actors 5](#_q1mabmypqkth)

[4.2 System Use Case Diagram 6](#_jzpxcwe5s51f)

[A. Overall Use Case Diagram 6](#_odadnsew9x6a)

[B. User Use Case Diagram 8](#_hniq9insc0s7)

[C. Helper Use Case Diagram 8](#_wlnt4gxzphci)

[D. Admin Use Case Diagram 10](#_ujb06lwkn46v)

[4.3 Screen Flow 10](#_7e6kktsngcvt)

[**5. USE CASE DESCRIPTION 12**](#_a6a2rrg56rss)

[5.1. Sign Up 13](#_4xv11oq6hnxp)

[5.2. Sign In 14](#_3m9dnl5u1oma)

[5.3. Sign Out 15](#_3ua3uvgc2r4o)

[5.4. Edit User Profile 16](#_9l2sjc93c36h)

[5.5. Post Help Request 17](#_8xvmsq31078f)

[5.6. Edit Help Request 18](#_ggg9m8qm45sw)

[5.7. Delete Help Request 20](#_blx3q4lhrnwn)

[5.8. Search Helpers 21](#_7e4zpmtkwt49)

[5.9. Book Helpers 22](#_9i4925t6kk5f)

[5.10. Make Payment 23](#_mwkz7q8n6rd7)

[5.11. View notification 25](#_qudshjkxkufx)

[5.12. Add favourite Helper 26](#_vw85mtulnysd)

[5.13. View favourite Helper List 27](#_lnjnmvmyhdrr)

[5.14. Delete favourite Helper 29](#_80yh6x2dpbxq)

[5.15. Track Helper Progress 30](#_5trx5fkw153e)

[5.16. Cancel Request 32](#_scikbk9ju7zw)

[5.17. Rate and Feedback Helper 33](#_lw63hckc3r0a)

[5.18. View Request History 35](#_x1431di1bk9l)

[5.19. Edit Helper Request 36](#_c9qmipx5am5d)

[5.20. View Scheduled session 38](#_jfvswqftxgjl)

[5.21. Register as Helper 39](#_au76i78w4tbi)

[5.22. Set available to request 41](#_5lj5wk1su3z)

[5.23. Accept/Deny Request 42](#_39byapys0rp5)

[5.24. View Income 44](#_wtx4m1he6mh3)

[5.25. Withdraw Income 46](#_wtybvhbgq1fq)

[5.26. View Feedback 47](#_amoai6g5gl76)

[5.27. Report Rate/Feedback 49](#_5knr9sofvnsy)

[5.28. Censoring Helper application 50](#_4tlovtqtr45p)

[5.29. Ban/Unban Profiles 51](#_kmvpgjteukkn)

[5.30. View Request on platform 53](#_71l0a9ndmo8r)

[5.31. Create Report/ Analytics 55](#_6ygg30i7w9au)

[**6. DATA STRUCTURE 56**](#_sdhvcmijd48o)

[6.1 Database 57](#_xy2k9q1748rp)

[**7. NON-FUNCTIONAL 57**](#_d3i69mvhx2a6)

[1. External Interfaces 57](#_wmxvumb6nss0)

[1.1. User Interface 57](#_x9qt55hbxp0f)

[1.2. Software Interface 57](#_atikik8u0gn9)

[2. Quality Attributes 58](#_tksite4503yr)

[2.1. Availability 58](#_wi4qynu8dh4m)

[2. 2 Usability 58](#_rpp0n36ncocj)

[2.3. Reliability 58](#_efl5uld7cuuj)

[2.4. Performance 58](#_k6iworfdnq1e)

[2.5. Security 58](#_3wrr6qqwwpdd)

[**8. APPENDICES 58**](#_2zee42rutoz6)

[Appendix A: Acronyms and Abbreviations 58](#_vmmt5en55y22)

[Appendix B: References 59](#_pdl2x97smcz8)

[Appendix C: Tools and Technologies 59](#_vwksjl9v3poo)

[Appendix D: Glossary 59](#_shlxvbxut6l0)

[Appendix E: Contact Information 59](#_oageuojqqdfd)

# **1. DOCUMENT HISTORY**

| **Date** | **Summary of Changes** | **Version** |
| --- | --- | --- |
| 25/01/2025 | Analyse software requirements | CMS\_SRS\_sample\_v1.0 |
|  |  |  |
|  |  |  |

# 

# **2.** **REFERENCE DOCUMENTS**

| **Document Name** | **Description** |
| --- | --- |
| [1] What is Class Diagram? - visual-paradigm | Reference document |
| [2] Slide bài giảng SWD392 | Reference document |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

# **3.** **INTRODUCTION**

The following subsections of the Software Requirements Specifications (SRS) document provide an overview of the entire SRS.

## **3.1** **Purpose**

* The Software Requirements Specification (SRS) defines the requirements for the Mobile Home Helper Finder Application. It ensures the system automates the process of finding, booking, and managing home cleaning and assistance services, providing users with a fast, reliable, and user-friendly platform. This SRS will guide developers in building the application effectively and ensure it aligns with stakeholder expectations. Feedback from users and service providers will be incorporated to deliver a solution tailored to the needs of households seeking trusted home helpers.

## **3.2** **In Scope**

The Mobile Home Helper Finder Application will serve as a platform connecting users with local home service providers. It will support service listing, booking, and management features. The system will enable location-based searches, handle user and helper registrations, and facilitate secure, role-based access for users, helpers, and administrators. It will also support efficient operations, service tracking, and basic reporting to ensure reliability and ease of use.

# **4.** **OVERVIEW**

## **4.1** **Actors**

| **Actor** | **Description** |
| --- | --- |
| Guest | A Guest is a person who accesses the app without logging in. They can view the public information about platform and general services |
| User | A User is a registered customer. They can post service requests, search and book helpers, make payments, track job status, rate helpers, manage their account, and chat with helpers. |
| Helper | A Helper is a service provider. They can create a profile, view and accept nearby job requests, manage schedules, track earnings, respond to ratings, and receive notifications. |
| Admin | An Admin can monitor, moderate and adjust the platform |

## 

## 

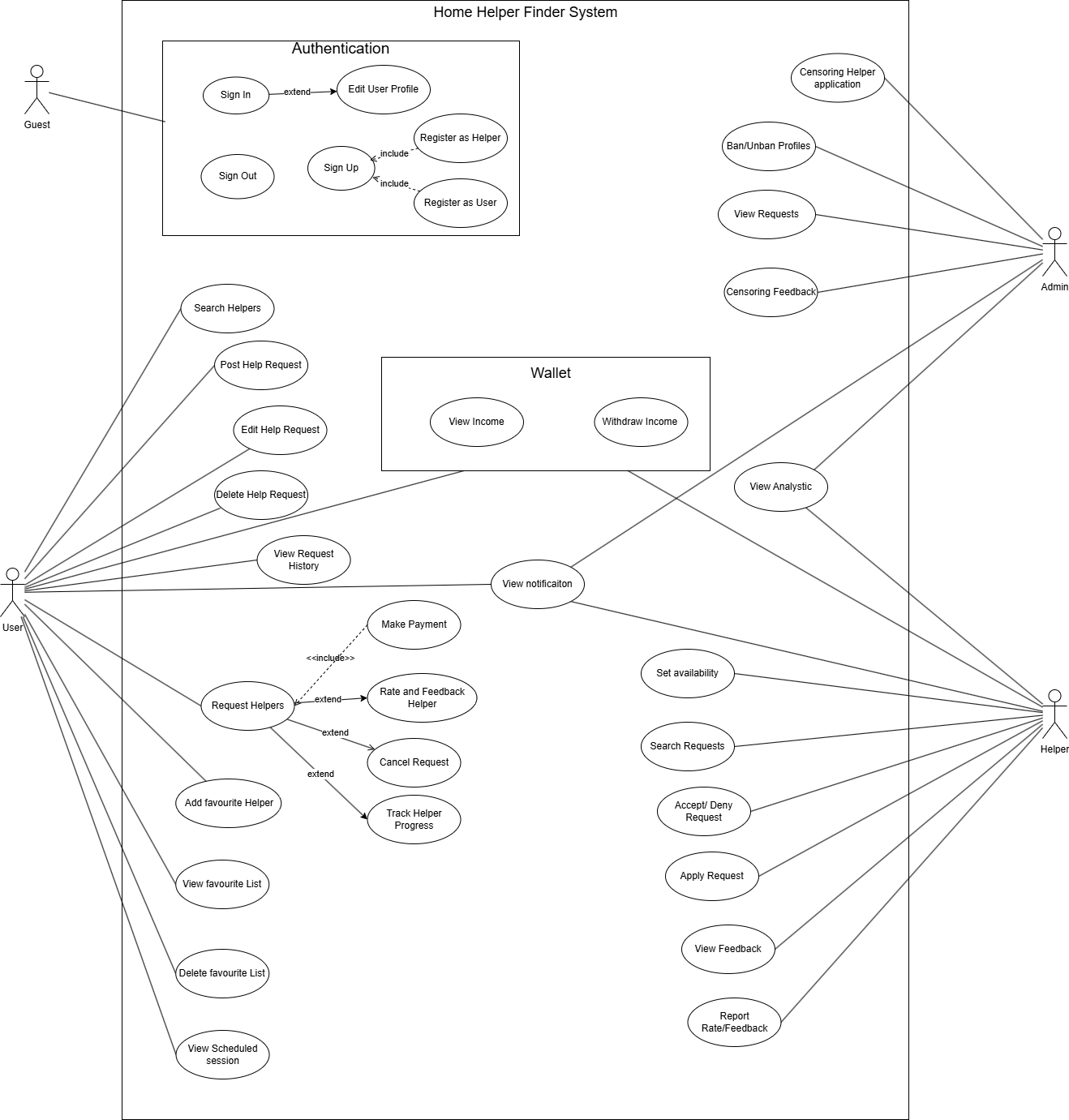
## 

## 

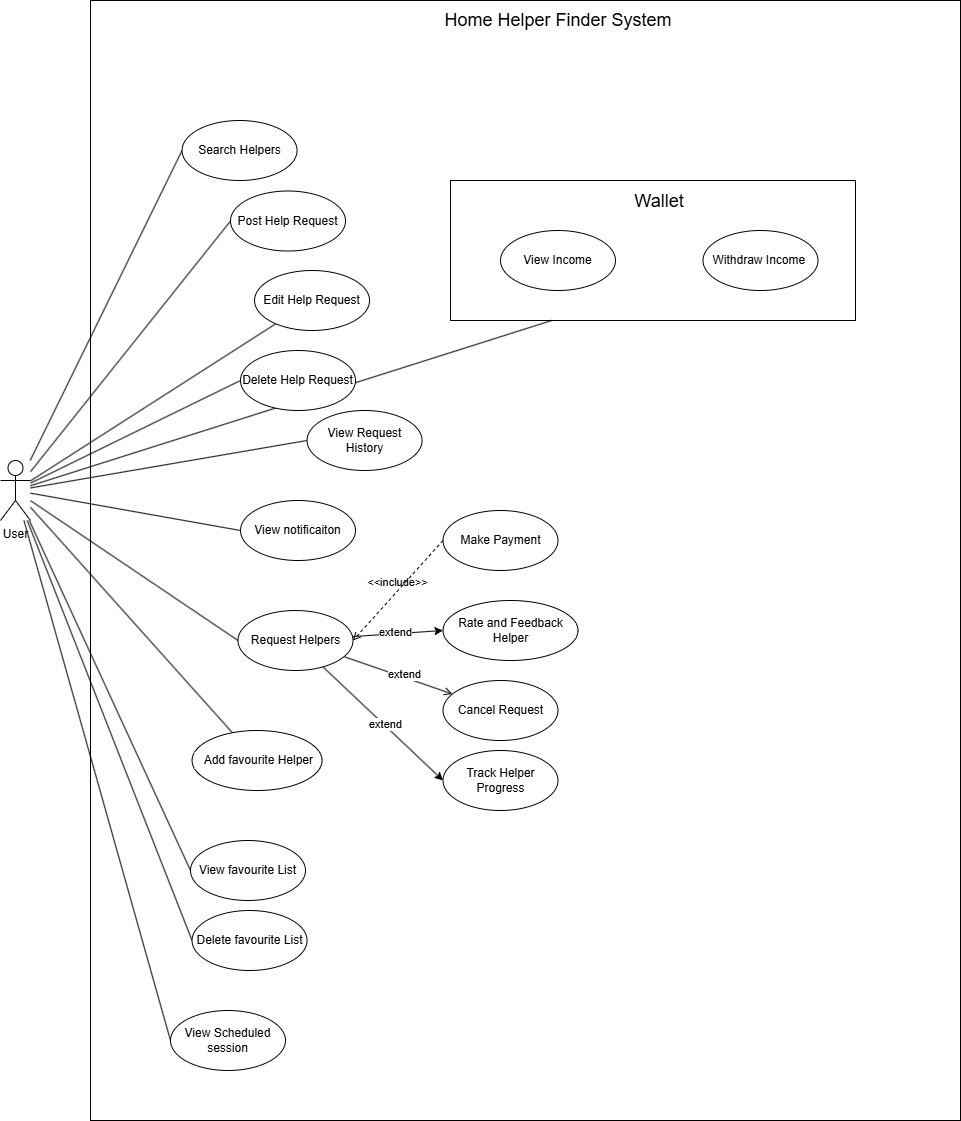
## 

## **4.2** **System Use Case Diagram**

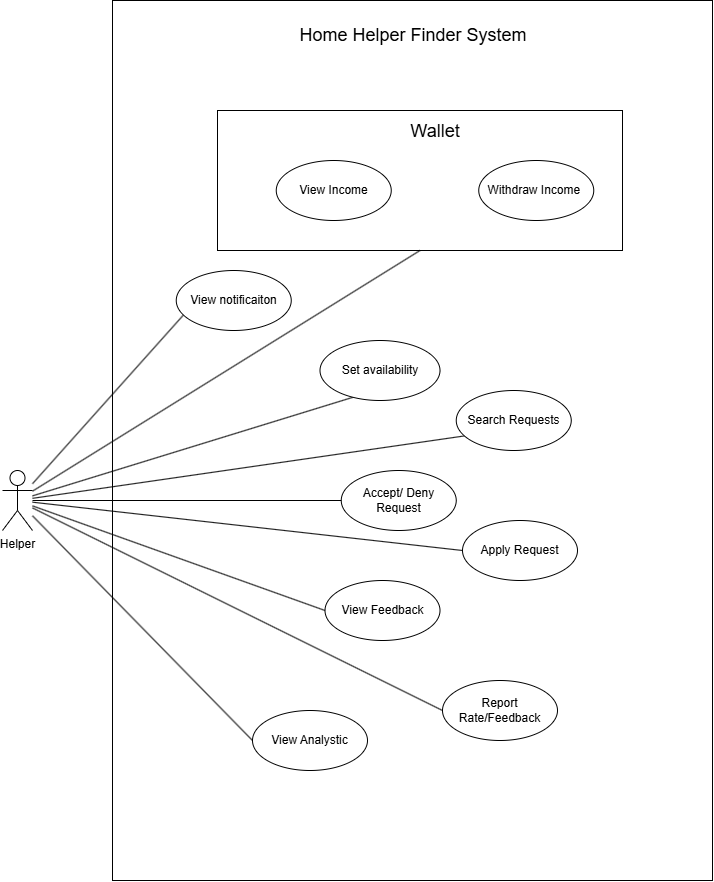
### **Overall Use Case Diagram**



### **User Use Case Diagram**



### **Helper Use Case Diagram**

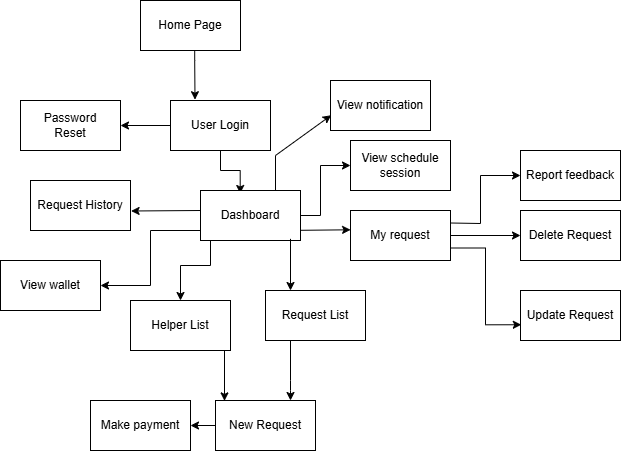


### **Admin Use Case Diagram**

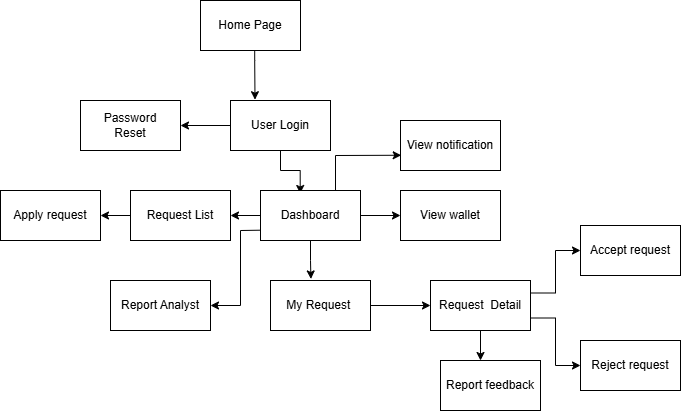
### 

## **4.3** **Screen Flow**

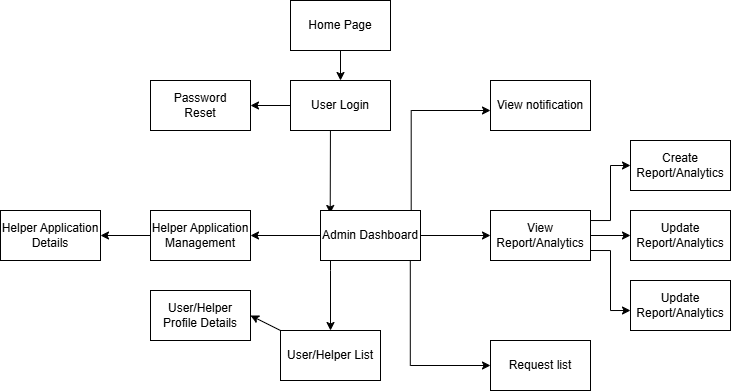
1. User



1. Helper

******

1. Admin



# **5.** **USE CASE DESCRIPTION**

## **5.1. Sign Up**

| Use Case ID: | **UC-01** | USE CASE NAME | **SIGN-UP** |
| --- | --- | --- | --- |
| Create By: | PhuongPH | Date Created: |  |
| Primary Actor: | Guest | Secondary Actor | None |
| Description | This use case describes the process of a new visitor signing up for an account, providing their personal details such as name, email, and password to become a registered user. | | |
| Priority: | High | | |
| Trigger | The guest clicks on the "Sign Up" button on the login page or accesses the registration page via a link. | | |
| Preconditions: | PRE-01**:** The guest has a valid email address.  PRE-02**:** The guest provides a valid password. | | |
| Post-Conditions | POST-01**:** The system works stably without errors.  POST-02**:** A new user account is created, and the guest receives a unique identifier upon successful verification.  POST-03**:** The guest can log in using their newly created account | | |
| Basic Flow: | 1. The guest accesses the home page. 2. Clicks on the avatar icon, then selects the “Sign In” button. 3. The system redirects to the sign-in page. 4. Clicks on “Sign Up.” 5. Fills in all required fields for account creation. 6. Clicks the “Create Account” button. 7. Verifies email with the code sent to their registered email. 8. The system redirects to the sign-in page. | | |
| Alternative Flow: | None | | |
| Exception: | **1a.** The guest enters an invalid username or email.   * **1a1.** The system displays an error message and prompts the guest to enter a valid username, email, or password.   **1b.** The guest enters a username or email that already exists.   * **1b1.** The system displays an error message and prompts the guest to enter a different username or email. | | |

## **5.2. Sign In**

| Use Case ID: | **UC-02** | USE CASE NAME | **SIGN-IN** |
| --- | --- | --- | --- |
| Create By: | PhuongPH | Date Created: |  |
| Primary Actor: | Guest | Secondary Actor | None |
| Description | This use case describes how a user can sign in to their account. The system verifies the credentials and grants access to the user's account. | | |
| Priority: | High | | |
| Trigger | User clicks on the "Sign In" Button in Home Page. | | |
| Preconditions: | PRE-01: User has a created and valid account  PRE-02: User account has been authorised. | | |
| Post-Conditions | POST-01: The system works stably without errors.  POST-02:User successfully logged into the app. | | |
| Basic Flow: | 1. User access to the sign-in page.  2. User to choose login with their account.  3. User type in account info and press sign in button.  4. System verifies the user's account information and allows the user  to access the system.  5. System redirects to Home page. | | |
| Alternative Flow: | 2a. User to login with Gmail method.  2a1. System redirects to the Google sign in page.  3a. Users type in their Google account and click log in.  4a. Google verifies the account’s information and allows the user access to the system. | | |
| Exception: | 4c. The system fails to verify the user's account information and display messages.  4c1. Users stop the login process.  Use case stops.  4c2. Users click on Forgot password | | |

## **5.3. Sign Out**

| Use Case ID: | **UC-03** | USE CASE NAME | **SIGN-OUT** |
| --- | --- | --- | --- |
| Create By: | PhuongPH | Date Created: |  |
| Primary Actor: | User, Helper | Secondary Actor | None |
| Description | This use case describes how a user can sign out of their account, which clears any active cookies and logs them out of the system | | |
| Priority: | High | | |
| Trigger | User clicks on the "Logout" Button in the sidebar. | | |
| Preconditions: | PRE-01: User has logged in into app | | |
| Post-Conditions | POST-01: The system works stably without errors.  POST-2: System navigates the users back to the home page. | | |
| Basic Flow: | 1. User click on “mark 3 line” to expand the sidebar .  2. Click on the log out option.  3. The user is now no longer determined as any role but Guest.  4. The guest will be navigated back to the home page. | | |
| Alternative Flow: | None | | |
| Exception: | None | | |

## **5.4. Edit User Profile**

| Use Case ID: | **UC-04** | USE CASE NAME | **EDIT-PROFILE** |
| --- | --- | --- | --- |
| Create By: | PhuongPH | Date Created: |  |
| Primary Actor: | User | Secondary Actor | None |
| Description | This use case describes how a registered user can view and update their personal details. | | |
| Priority: | High | | |
| Trigger | The user clicks on the profile section. | | |
| Preconditions: | PRE-01: The user must be logged into the system. | | |
| Post-Conditions | POST-01: The system updates and saves the user’s profile changes.  POST-02: The user sees the updated profile information. | | |
| Basic Flow: | 1. The user logs into the system. 2. Clicks on the profile section. 3. Edits profile details (e.g., name, email, profile picture). 4. Clicks the "Save" button. 5. The system updates and displays the changes. | | |
| Alternative Flow: | None | | |
| Exception: | **4a.** The user enters invalid information.   * **4a1.** The system displays an error message and prompts the user to correct the information. | | |

## **5.5. Post Help Request**

| Use Case ID: | **UC-05** | USE CASE NAME | **POST HELP REQUEST** |
| --- | --- | --- | --- |
| Create By: | PhuongPH | Date Created: |  |
| Primary Actor: | User | Secondary Actor | None |
| Description | The user creates a new help request by selecting service type, time, location, and notes. | | |
| Priority: | High | | |
| Trigger | User clicks “Post Request” or “Request Help” from home/ dashboard. | | |
| Preconditions: | PRE-01: User is logged in.  PRE-02: User profile has location info. | | |
| Post-Conditions | POST-01: Help request is saved in system.  POST-02: Request is visible to nearby helpers. | | |
| Basic Flow: | 1. User logs in. 2. Click “Post Help Request.” 3. Select service type, time, and location. 4. Enter notes (optional). 5. Click “Submit.” | | |
| Alternative Flow: | None | | |
| Exception: | * 1a. Missing required fields → system shows validation errors. * 1b. Invalid location → system asks to correct it. | | |

## **5.6. Edit Help Request**

| Use Case ID: | **UC-06** | USE CASE NAME | **EDIT HELP REQUEST** |
| --- | --- | --- | --- |
| Create By: | PhuongPH | Date Created: |  |
| Primary Actor: | User | Secondary Actor | None |
| Description | The user modifies the details (service type, time, location, notes) of an existing help request they previously created. | | |
| Priority: | High | | |
| Trigger | User clicks “Edit” or an edit icon associated with one of their existing, active help requests from a list of their requests. | | |
| Preconditions: | PRE-01: User is logged in.  PRE-02: The help request exists in the system and was created by the user.  PRE-03: The help request is in a state that allows editing (e.g., not yet accepted by a helper or in progress). | | |
| Post-Conditions | POST-01: Help request is updated in the system with the new details.  POST-02: The updated request is visible to nearby helpers (potentially with an indication of being updated).  POST-03: (Optional) Helpers who were previously viewing or matched with the request may be notified of the changes. | | |
| Basic Flow: | 1. User logs in. 2. User navigates to their list of posted help requests. 3. User selects the specific help request they wish to edit. 4. User clicks an “Edit” button/link for that request. 5. System displays the current details of the help request (service type, time, location, notes) in an editable form. 6. User modifies one or more of the following: service type, time, location, or notes. 7. User clicks “Save Changes” or “Update.” | | |
| Alternative Flow: | 6a. User decides not to save changes:   1. User clicks "Cancel" or navigates away. 2. System discards changes and returns the user to their list of requests or the request details view (unchanged). | | |
| Exception: | * 1a. Missing required fields → system shows validation errors. * 1b. Invalid location → system asks to correct it. * 1c. User is not authorized to edit the request → the system displays an error. | | |

## **5.7. Delete Help Request**

| Use Case ID: | **UC-07** | USE CASE NAME | **DELETE HELP REQUEST** |
| --- | --- | --- | --- |
| Create By: | PhuongPH | Date Created: |  |
| Primary Actor: | User | Secondary Actor | None |
| Description | The user permanently removes one of their existing help requests from the system. | | |
| Priority: | High | | |
| Trigger | User clicks “Delete” or a delete icon associated with one of their existing, active help requests from a list of their requests. | | |
| Preconditions: | PRE-01: User is logged in.  PRE-02: The help request exists in the system and was created by the user.  PRE-03: The help request is in a state that allows deletion. | | |
| Post-Conditions | POST-01: Help request is marked as deleted or permanently removed from the system.  POST-02: Request is no longer visible to nearby helpers.  POST-03: (Optional) Helpers who were previously viewing or matched with the request may be notified of the cancellation. | | |
| Basic Flow: | 1. User logs in. 2. User navigates to their list of posted help requests. 3. User selects the specific help request they wish to delete. 4. User clicks a “Delete” button/link for that request. 5. System prompts the user for confirmation (e.g., "Are you sure you want to delete this request?"). 6. User confirms the deletion. 7. System removes the help request. | | |
| Alternative Flow: | 6a. User cancels deletion:   1. User clicks "Cancel" or "No" on the confirmation prompt. 2. System cancels the deletion process and the request remains active. | | |
| Exception: | * 1a. Request no longer deletable → system displays an error message and prevents deletion. * 1b. User is not authorized to delete the request → the system displays an error. * 1c. Request not found -> System displays an appropriate message. | | |

## 

## **5.8. Search Helpers**

| Use Case ID: | **UC-08** | USE CASE NAME | **SEARCH HELPERS** |
| --- | --- | --- | --- |
| Create By: | PhuongPH | Date Created: |  |
| Primary Actor: | User | Secondary Actor | None |
| Description | Allows users to find suitable helpers based on location and filters. | | |
| Priority: | High | | |
| Trigger | User opens the “Find Helpers” page. | | |
| Preconditions: | PRE-01: User is logged in.  PRE-02: User has set location. | | |
| Post-Conditions | POST-01: Matching helpers are displayed.  POST-02: User can view helper profiles. | | |
| Basic Flow: | 1. User opens “Find Helpers.” 2. System auto-detects user location. 3. User applies filters (skills, rating, etc.). 4. System shows list of matching helpers. 5. User clicks on a profile to view details. | | |
| Alternative Flow: | None | | |
| Exception: | * 1a. No helpers found → system shows “No matches found.” | | |

## **5.9. Book Helpers**

| Use Case ID: | **UC-09** | USE CASE NAME | **BOOK HELPER** |
| --- | --- | --- | --- |
| Create By: | PhuongPH | Date Created: |  |
| Primary Actor: | User | Secondary Actor | None |
| Description | This use case describes the process of booking a helper by selecting service, time, and confirming the request. | | |
| Priority: | High | | |
| Trigger | User selects a helper and clicks “Book Now.” | | |
| Preconditions: | PRE-01: User is logged in.  PRE-02: Helper is available at selected time. | | |
| Post-Conditions | POST-01: Booking is saved in the system.  POST-02: User is redirected to payment step. | | |
| Basic Flow: | 1. User logs in. 2. Searches and selects a helper. 3. Chooses date and time. 4. Clicks “Book Now.” 5. System confirms availability and saves booking. 6. User is redirected to payment. | | |
| Alternative Flow: | 3a. Helper is unavailable → system prompts to choose another time. | | |
| Exception: | * 1a. Booking fails due to system error → system shows retry message. | | |

## **5.10. Make Payment**

| Use Case ID: | **UC-10** | USE CASE NAME | **MAKE PAYMENT** |
| --- | --- | --- | --- |
| Create By: | PhuongPH | Date Created: |  |
| Primary Actor: | User | Secondary Actor | Payment Gateway |
| Description | This use case describes how a user completes the payment for a confirmed booking using available payment options. | | |
| Priority: | Medium | | |
| Trigger | User is redirected to payment after booking confirmation. | | |
| Preconditions: | PRE-01: A valid booking exists.  PRE-02: User has access to chosen payment method. | | |
| Post-Conditions | POST-01: Payment status is recorded.  POST-02: User receives confirmation message. | | |
| Basic Flow: | 1. User selects a payment method (cash, Momo, ZaloPay, card). 2. Confirms payment. 3. System processes transaction. 4. Payment gateway returns success. 5. System updates booking to “Paid.” 6. User sees confirmation. | | |
| Alternative Flow: | 1a. User cancels payment → system keeps booking in “Pending Payment.” | | |
| Exception: | * 3a. Payment fails → system shows error and retry option. | | |

## 

## **5.11. View notification**

| Use Case ID: | **UC-010** | USE CASE NAME | **VIEW NOTIFICATIONS** |
| --- | --- | --- | --- |
| Create By: | PhuongPH | Date Created: |  |
| Primary Actor: | User, Helper, Admin | Secondary Actor | System |
| Description | This use case describes how a user views in-app notifications regarding bookings, system updates, promotions, and real-time alerts such as job status or schedule changes. | | |
| Priority: | Medium | | |
| Trigger | User opens the notification center or clicks on the notification icon. | | |
| Preconditions: | PRE-01: User is logged in.  PRE-02: There are one or more notifications available for the user. | | |
| Post-Conditions | POST-01: Notifications are marked as read after being viewed.  POST-02: User can take action based on the content of the notification (e.g., view booking, respond to message). | | |
| Basic Flow: | 1. User opens the notification center. 2. System retrieves all unread and recent notifications. 3. User views the list of notifications. 4. User selects a notification to view details. 5. System displays detailed information and possible actions. 6. Notification is marked as read. | | |
| Alternative Flow: | 3a.1 (No Notifications): If no notifications exist, system displays a “No new notifications” message.  5a.1 (Actionable Notification): User may tap an action button within the notification (e.g., “View Booking”, “Open Chat”). | | |
| Exception: | 2e.1 (Fetch Error): If the system fails to retrieve notifications, it displays an error message and retry option.  4e.1 (Invalid Notification Link): If notification links to a missing resource, the system informs the user. | | |

## **5.12. Add favourite Helper**

| Use Case ID: | **UC-12** | USE CASE NAME | **ADD FAVOURITE HELPER** |
| --- | --- | --- | --- |
| Create By: | PhuongPH | Date Created: |  |
| Primary Actor: | User | Secondary Actor | Helper |
| Description | This use case describes how a user adds a helper to their list of favorites, allowing easier future bookings and prioritizing the helper in future service matches. | | |
| Priority: | High | | |
| Trigger | User selects the "Add to Favorites" option from a helper’s profile or after a completed service. | | |
| Preconditions: | PRE-01: User is logged in.  PRE-02: User has previously interacted with or viewed the helper’s profile. | | |
| Post-Conditions | POST-01: Helper is added to the user's favorites list.  POST-02: System updates user preferences for future matches. | | |
| Basic Flow: | 1. User views a helper’s profile or service history. 2. User selects the "Add to Favorites" option. 3. System confirms the selection and adds the helper to the favorites list. 4. User receives a confirmation message. | | |
| Alternative Flow: | 1a.1 (From Booking History): User accesses completed service details and selects “Add to Favorites” from there.  2a.1 (Already Favorited): If helper is already in the list, system informs the user. | | |
| Exception: | 3e.1 (System Error): If there’s an error while saving the favorite, the system displays a message and prompts user to retry. | | |

## **5.13. View favourite Helper List**

| Use Case ID: | **UC-13** | USE CASE NAME | **VIEW FAVOURITE HELPER LIST** |
| --- | --- | --- | --- |
| Create By: | PhuongPH | Date Created: |  |
| Primary Actor: | User | Secondary Actor | Helper |
| Description | This use case describes how a user accesses and views a list of helpers they have previously marked as favorites, allowing quick reference and selection for future bookings. | | |
| Priority: | Medium | | |
| Trigger | User selects the “Favourite Helpers” option from the menu or profile section. | | |
| Preconditions: | PRE-01: User is logged in.  PRE-02: User has at least one helper marked as a favorite. | | |
| Post-Conditions | POST-01: System displays the list of favorite helpers with profile and availability details. | | |
| Basic Flow: | 1. User navigates to the “Favourite Helpers” section. 2. System retrieves the list of helpers marked as favorites. 3. System displays each helper’s name, rating, service types, availability, and quick action options (e.g., “View Profile,” “Book Now”). 4. User can select a helper to view full profile or initiate a new booking. | | |
| Alternative Flow: | 2a.1 (No Favorites): If the list is empty, system informs the user and provides a link to browse or search for helpers. | | |
| Exception: | 2e.1 (Data Load Error): If the system fails to load the list, an error message is displayed with a retry option. | | |

## **5.14. Delete favourite Helper**

| Use Case ID: | **UC-12** | USE CASE NAME | **DELETE FAVOURITE HELPER** |
| --- | --- | --- | --- |
| Create By: | PhuongPH | Date Created: |  |
| Primary Actor: | User | Secondary Actor | Helper |
| Description | This use case describes how a user removes a helper from their list of favorites. | | |
| Priority: | Medium | | |
| Trigger | User selects the "Remove from Favorites" in their "Favourite Helper List" or from the helper's profile if they are currently favorited. | | |
| Preconditions: | PRE-01: User is logged in.  PRE-02: The helper is currently in the user's favorites list. | | |
| Post-Conditions | POST-01: Helper is removed from the user's favorites list.  POST-02: System updates user preferences; the helper is no longer prioritized as a favorite for future matches for this user. | | |
| Basic Flow: | 1. User navigates to their “Favourite Helpers” list.. 2. User identifies the helper they wish to remove from their favorites. 3. User selects the "Remove from Favorites" 4. System prompts the user for confirmation 5. User confirms the deletion. 6. System removes the helper from the user's favorites list. 7. System provides a confirmation message to the user.. 8. If viewing the favorites list, the list is refreshed to reflect the removal. | | |
| Alternative Flow: | 5a.1 (User Cancels Deletion):   1. User selects "Cancel" on the confirmation prompt. 2. System cancels the removal process, and the helper remains in the user's favorites list. | | |
| Exception: | 6e.1 (System Error): If there’s an error while removing the favorite (e.g., database error), the system displays an error message and advises the user to retry.  6e.2 (Helper Not Found in Favorites): If, due to a data inconsistency or an action in another session, the helper is no longer in the favorites list when the removal is attempted, the system informs the user.. | | |

## **5.15. Track Helper Progress**

| Use Case ID: | **UC-14** | USE CASE NAME | **TRACK HELPER PROGRESS** |
| --- | --- | --- | --- |
| Create By: | PhuongPH | Date Created: |  |
| Primary Actor: | User | Secondary Actor | Helper |
| Description | This use case describes how a user tracks the real-time progress of a helper for a scheduled or ongoing service, including location, check-in status, and task updates. | | |
| Priority: | High | | |
| Trigger | User selects a scheduled or active service from the service history or notifications. | | |
| Preconditions: | PRE-01: User is logged in.  PRE-02: A helper is assigned to an active or scheduled service.  PRE-03: The helper is using the system to update progress and/or location. | | |
| Post-Conditions | POST-01: User can monitor helper’s location and service status.  POST-02: System logs progress updates for reference. | | |
| Basic Flow: | 1. User navigates to “My Services” or receives a notification about an active session. 2. User selects the service to track. 3. System displays helper’s current status (e.g., "On the way," "Checked in," "In Progress," "Completed"). 4. If enabled, system shows real-time location of the helper on a map. 5. User can view time logs, helper notes, and live updates on service progress. 6. User may contact helper or support if needed. | | |
| Alternative Flow: | 4a.1 (No Location Sharing): If real-time tracking is disabled, system shows estimated time and last known status.  5a.1 (Helper Adds Notes): Helper can update progress notes, visible to user. | | |
| Exception: | 3e.1 (No Active Service): If there is no service to track, system notifies the user.  4e.1 (Location Error): If location cannot be retrieved, system shows fallback status info.  5e.1 (System Error): On data retrieval failure, system advises user to retry. | | |

## **5.16. Cancel Request**

| Use Case ID: | **UC-15** | USE CASE NAME | **CANCEL REQUEST** |
| --- | --- | --- | --- |
| Create By: | PhuongPH | Date Created: |  |
| Primary Actor: | User | Secondary Actor | Helper |
| Description | This use case describes how a user cancels a previously submitted service request before the service starts, including optional reasons for cancellation. | | |
| Priority: | High | | |
| Trigger | User selects “Cancel Request” from the request details screen. | | |
| Preconditions: | PRE-01: User is logged in.  PRE-02: A valid, pending or scheduled service request exists that has not yet started. | | |
| Post-Conditions | POST-01: The request is canceled and removed from the active queue.  POST-02: Helper (if assigned) is notified of the cancellation.  POST-03: Cancellation is recorded in the system for tracking and reporting. | | |
| Basic Flow: | 1. User navigates to “My Requests” or service history. 2. User selects a pending or scheduled service request. 3. User clicks “Cancel Request.” 4. System prompts for optional cancellation reason. 5. User confirms cancellation. 6. System cancels the request, notifies helper (if assigned), and confirms to user. | | |
| Alternative Flow: | 4a.1 (Quick Cancel): User may choose to cancel immediately without entering a reason.  4a.2 (Predefined Reasons): System offers predefined reasons to choose from. | | |
| Exception: | 2e.1 (Already Started): If the service is already in progress or completed, cancellation is not allowed and system notifies the user.  6e.1 (System Error): If cancellation fails due to a system error, user is prompted to retry or contact support. | | |

## **5.17. Rate and Feedback Helper**

| Use Case ID: | **UC-16** | USE CASE NAME | **RATE HELPER** |
| --- | --- | --- | --- |
| Create By: | PhuongPH | Date Created: |  |
| Primary Actor: | User | Secondary Actor | Helper |
| Description | This use case describes how a user provides a rating and optional feedback for a helper after a completed service session, contributing to the helper's overall performance record. | | |
| Priority: | High | | |
| Trigger | User receives a prompt or navigates to a completed service to submit a rating. | | |
| Preconditions: | PRE-01: User is logged in.  PRE-02: The service with the helper has been completed. | | |
| Post-Conditions | POST-01: Rating and feedback are saved and associated with the helper’s profile.  POST-02: Overall helper rating is updated. | | |
| Basic Flow: | 1. User navigates to “Completed Services” or receives a post-service prompt. 2. User selects the service session to rate. 3. System displays rating interface (e.g., 1 to 5 stars, optional text box for feedback). 4. User selects a rating and optionally enters comments. 5. User submits the rating. 6. System confirms submission and updates helper profile. | | |
| Alternative Flow: | 2a.1 (Rate Later): User may skip and return later to rate from history.  4a.1 (Anonymous Option): User may choose to leave feedback anonymously. | | |
| Exception: | 2e.1 (Service Not Completed): If the service is not marked as completed, rating is disabled.  5e.1 (System Error): If rating fails to submit, system notifies user and allows retry. | | |

## **5.18. View Request History**

| Use Case ID: | **UC-17** | USE CASE NAME | **VIEW REQUEST HISTORY** |
| --- | --- | --- | --- |
| Create By: | PhuongPH | Date Created: |  |
| Primary Actor: | User | Secondary Actor | Helper |
| Description | This use case describes how a user views the history of all past and current service requests, including their status, details, and outcomes. | | |
| Priority: | High | | |
| Trigger | User selects “Request History” from the app’s menu or dashboard. | | |
| Preconditions: | PRE-01: User is logged in.  PRE-02: User has made at least one service request. | | |
| Post-Conditions | POST-01: System displays a list of past and current service requests with status and details.  POST-02: User may view individual request details. | | |
| Basic Flow: | 1. User navigates to the “Request History” section. 2. System retrieves all requests associated with the user. 3. System displays a list with summary info (e.g., service type, date, status). 4. User selects a request to view detailed information (e.g., helper, notes, price, status). 5. User may filter or sort requests (by date, type, or status). | | |
| Alternative Flow: | 3a.1 (Filter/Search): User applies filters or uses search to find specific requests.  4a.1 (Repeat Request): From a completed request, user may select “Book Again” to submit a similar request. | | |
| Exception: | 2e.1 (No History): If no requests exist, system informs the user.  3e.1 (Load Failure): If system fails to load history, user is shown an error with retry option. | | |

## **5.19. Edit Helper Request**

| Use Case ID: | **UC-18** | USE CASE NAME | **EDIT HELPER REQUEST** |
| --- | --- | --- | --- |
| Create By: | PhuongPH | Date Created: |  |
| Primary Actor: | User | Secondary Actor | Helper |
| Description | This use case describes how a user can edit an existing helper service request before it is matched or started. The user may adjust service type, time, location, or additional notes. | | |
| Priority: | Medium | | |
| Trigger | User selects an active or pending request and chooses the “Edit” option. | | |
| Preconditions: | PRE-01: User is logged in.  PRE-02: Request status is still pending or not yet started. | | |
| Post-Conditions | POST-01: The service request is updated with new details.  POST-02: Helper matching may be restarted based on changes (e.g., new time or location). | | |
| Basic Flow: | 1. User navigates to the “My Requests” or “Request History” section. 2. User selects a request with editable status. 3. User clicks “Edit” and is presented with the request form pre-filled with existing details. 4. User updates fields such as service type(s), date/time, location, or special notes. 5. User reviews the updated request and confirms changes. 6. System validates the new data and saves the updated request. 7. System notifies the user of successful update and adjusts any affected processes (e.g., rematching helper). | | |
| Alternative Flow: | 4a.1 (Saved Address Change): User changes to a previously saved address.  6a.1 (No Significant Change): If no key matching criteria change, system may skip rematching. | | |
| Exception: | 2e.1 (Non-Editable Status): If request has already started or is completed, system disables editing.  6e.1 (Validation Error): System shows error if new data is invalid (e.g., past time, unsupported address). | | |

## **5.20. View Scheduled session**

| Use Case ID: | **UC-20** | USE CASE NAME | **VIEW SCHEDULED SESSION OF HELPER** |
| --- | --- | --- | --- |
| Create By: | PhuongPH | Date Created: |  |
| Primary Actor: | User | Secondary Actor | Helper |
| Description | This use case describes how a user views all their upcoming scheduled service sessions that they have booked, including date, time, helper details, location, and service types. | | |
| Priority: | High | | |
| Trigger | User selects “My Bookings,” “My Schedule,” or equivalent option from the app menu or dashboard. | | |
| Preconditions: | PRE-01: User is logged in.  PRE-02: At least one service session has been booked by the user. | | |
| Post-Conditions | POST-01: User can view a list or calendar of their upcoming booked sessions.  POST-02: User can access details of each scheduled session, including information about the assigned/chosen helper. | | |
| Basic Flow: | 1. User navigates to the “My Schedule” section. 2. System retrieves all future scheduled sessions booked by the user. 3. System displays sessions in a list or calendar format with summary.. 4. User selects a specific session to view full details, agreed time, location, service type, any notes provided by the user or helper). 5. User can review the session details, potentially with options to contact the helper, view directions, or navigate to cancel/reschedule options. | | |
| Alternative Flow: | 3a.1 (Filter): User may filter sessions by date or service type.  4a.1 (Map View): User can view session locations on a map. | | |
| Exception: | 2e.1 (No Scheduled Sessions): System shows message like “No upcoming sessions found.”  3e.1 (Load Error): If sessions fail to load, system displays an error and retry option. | | |

## **5.21. Register as Helper**

| Use Case ID: | **UC-21** | USE CASE NAME | **REGISTER AS HELPER** |
| --- | --- | --- | --- |
| Create By: | PhuongPH | Date Created: |  |
| Primary Actor: | Helper | Secondary Actor | none |
| Description | This use case describes how a new user or guest registers as a helper by providing required personal, professional, and service-related information for review and approval. | | |
| Priority: | Medium | | |
| Trigger | User selects the “Register as Helper” option from the app. | | |
| Preconditions: | PRE-01: User is not already registered as a helper.  PRE-02: Required documents and information are ready for submission. | | |
| Post-Conditions | POST-01: Helper registration request is submitted for review.  POST-02: User is notified about registration status (approved/pending/rejected). | | |
| Basic Flow: | 1. User selects “Register as Helper.” 2. System displays the registration form with required fields:   – Personal details (name, ID, contact info)   – Skills/services offered (e.g., cleaning, childcare)   – Availability   – Experience and certifications (optional)   – Upload ID and relevant documents 3. User completes and submits the form. 4. System validates the information. 5. System stores the application and notifies the admin. 6. Admin reviews and approves or rejects the application. 7. User is notified of the result via system message or email. | | |
| Alternative Flow: | 2a.1 (Logged-in User): If the user is already logged in, some personal information is pre-filled.  4a.1 (Missing Info): User is prompted to complete any missing required fields. | | |
| Exception: | 3e.1 (Submission Failure): If submission fails, system displays an error and prompts retry.  6e.1 (Delayed Review): If approval takes too long, user may contact support or receive auto-status updates. | | |

## **5.22. Set available to request**

| Use Case ID: | **UC-22** | USE CASE NAME | **REGISTER AS HELPER** |
| --- | --- | --- | --- |
| Create By: | PhuongPH | Date Created: |  |
| Primary Actor: | Helper | Secondary Actor | User |
| Description | This use case describes how a helper sets their availability status to allow users to send them direct service requests. | | |
| Priority: | High | | |
| Trigger | Helper selects “Set Available” or toggles availability status from their dashboard. | | |
| Preconditions: | PRE-01: Helper is registered and approved.  PRE-02: Helper is logged in. | | |
| Post-Conditions | POST-01: System updates the helper’s status to available.  POST-02: Users can now send direct requests to the helper. | | |
| Basic Flow: | 1. Helper logs into the system. 2. Helper navigates to their profile or availability settings. 3. Helper toggles availability to "Available for Requests." 4. System updates the helper’s status in real-time. 5. Users browsing for helpers can now see and request this helper. | | |
| Alternative Flow: | 3a.1 (Set Time Range): Helper may optionally define specific available dates/times.  5a.1 (Auto-Timeout): If availability expires after a certain time, system auto-sets to "Unavailable." | | |
| Exception: | 2e.1 (Not Approved): If helper is not yet verified/approved, system blocks the action with a message.  4e.1 (Update Failure): If system fails to update status, error is shown and retry option is provided. | | |

## **5.23. Accept/Deny Request**

| Use Case ID: | **UC-23** | USE CASE NAME | **ACCEPT / DENY REQUEST** |
| --- | --- | --- | --- |
| Create By: | PhuongPH | Date Created: |  |
| Primary Actor: | Helper | Secondary Actor | none |
| Description | This use case describes how a helper can accept or deny a service request assigned or suggested to them. Acceptance confirms their availability, while denial allows the system to reassign the request to another helper. | | |
| Priority: | High | | |
| Trigger | Helper receives a service request notification or opens the request list. | | |
| Preconditions: | PRE-01: Helper is logged in and marked as available.  PRE-02: A service request has been matched to the helper and is awaiting a response. | | |
| Post-Conditions | POST-01 (Accept): Request is confirmed and scheduled with the helper.  POST-02 (Deny): System removes the request from the helper and continues searching for another match. | | |
| Basic Flow: | 1. Helper accesses “Incoming Requests.” 2. Helper selects a new request to view details. 3. Helper clicks “Accept.” 4. System updates the request status and notifies the user. 5. Request is now scheduled with this helper. —--------------------------------------------------------- 6. Helper accesses “Incoming Requests.” 7. Helper selects a request. 8. Helper clicks “Deny” and optionally adds a reason. 9. System removes the helper from the request and continues the matching process. 10. User is notified that the request is still being matched. | | |
| Alternative Flow: | 3a.1 (Auto Accept): If auto-accept is enabled, system accepts the request on behalf of the helper.  3a.2 (Set Response Time): If helper does not respond within a set time, system auto-denies the request. | | |
| Exception: | 2e.1 (Expired Request): If request has expired or been accepted by another helper, system shows error.  4e.1 (System Error): If acceptance/denial fails, helper is shown a retry or contact support message. | | |

## **5.24. View Income**

| Use Case ID: | **UC-24** | USE CASE NAME | **MANAGE INCOME** |
| --- | --- | --- | --- |
| Create By: | PhuongPH | Date Created: |  |
| Primary Actor: | User, Helper | Secondary Actor | None |
| Description | This use case describes how a helper views and manages their income from completed service sessions, including transaction history, total earnings. | | |
| Priority: | High | | |
| Trigger | Helper selects “Income” or “Earnings” from the app dashboard. | | |
| Preconditions: | PRE-01: Helper is logged in.  PRE-02: Helper has completed at least one paid session. | | |
| Post-Conditions | POST-01: System displays total income, breakdowns, and history.  POST-02: Helper may request payout or export earnings report. | | |
| Basic Flow: | 1. Helper navigates to the “Manage Income” section. 2. System displays total earnings, pending balance, and available balance. 3. Helper views transaction history (dates, service type, user, amount). 4. Helper optionally filters by date or status (paid, pending). 5. Helper may request withdrawal to a registered bank or e-wallet. 6. System confirms the withdrawal request and updates the balance accordingly. | | |
| Alternative Flow: | 3a.1 (Download Report): Helper can export income report as PDF/CSV.  5a.1 (Set Auto-Payout): Helper sets auto-withdrawal preferences (e.g., weekly, monthly). | | |
| Exception: | 2e.1 (No Income Data): If no transactions exist, system displays “No income yet.”  5e.1 (Invalid Bank Info): If withdrawal account is invalid, system notifies helper to update it.  6e.1 (Withdrawal Failed): If payout request fails, system displays error and logs issue. | | |

## **5.25. Withdraw Income**

| Use Case ID: | **UC-25** | USE CASE NAME | **WITHDRAW INCOME** |
| --- | --- | --- | --- |
| Create By: | PhuongPH | Date Created: |  |
| Primary Actor: | User, Helper | Secondary Actor | None |
| Description | This use case details how a helper requests to withdraw their available income balance to a linked bank account or e-wallet. | | |
| Priority: | High | | |
| Trigger | Helper selects the “Withdraw” option from the income management section. | | |
| Preconditions: | PRE-01: Helper is logged in.  PRE-02: Helper has a sufficient available balance.  PRE-03: A valid withdrawal method (bank or e-wallet) is set up. | | |
| Post-Conditions | POST-01: Withdrawal request is submitted and recorded.  POST-02: System updates balance and notifies the helper.. | | |
| Basic Flow: | 1. Helper navigates to the “Manage Income” or “Withdraw” section. 2. System displays available balance and saved withdrawal methods. 3. Helper selects or adds a withdrawal method (bank, e-wallet). 4. Helper enters the withdrawal amount. 5. System validates the amount and method. 6. Helper confirms the withdrawal. 7. System processes the request and shows confirmation. 8. Helper receives a notification when the transfer is completed. | | |
| Alternative Flow: | 3a.1 (Add New Method): Helper adds a new payout method before proceeding.  4a.1 (Withdraw Full Amount): Helper selects “Withdraw All.” | | |
| Exception: | 2e.1 (Insufficient Balance): System shows error if amount exceeds available balance.  3e.1 (No Withdrawal Method): Helper is prompted to add one before proceeding.  7e.1 (Withdrawal Failure): If transaction fails, system shows error and logs the issue. | | |

## **5.26. View Feedback**

| Use Case ID: | **UC-26** | USE CASE NAME | **VIEW FEEDBACK** |
| --- | --- | --- | --- |
| Create By: | PhuongPH | Date Created: |  |
| Primary Actor: | Helper | Secondary Actor | User |
| Description | This use case describes how a helper can view feedback and ratings provided by users after completed service sessions to understand their performance and improve service quality. | | |
| Priority: | Medium | | |
| Trigger | Helper selects “Feedback” or “Ratings & Reviews” from their dashboard. | | |
| Preconditions: | PRE-01: Helper is logged in.  PRE-02: At least one completed service has received feedback from a user. | | |
| Post-Conditions | POST-01: System displays a list of feedback entries with corresponding ratings.  POST-02: Helper can view details for each feedback item. | | |
| Basic Flow: | 1. Helper navigates to the “Feedback” or “Ratings” section. 2. System retrieves all feedback entries associated with the helper. 3. Feedback list is displayed, showing user name, rating, service type, and review comments. 4. Helper selects a specific feedback to view more details. 5. Helper may use filters (e.g., by rating, date, or service type) | | |
| Alternative Flow: | 2a.1 (Sorted View): Helper sorts feedback by most recent, highest rating, or lowest rating.  4a.1 (Reply Option): Helper may reply to feedback (if feature is enabled). | | |
| Exception: | 2e.1 (No Feedback Yet): System displays “You haven’t received any feedback yet.”  3e.1 (Load Failure): If feedback data fails to load, system shows error with retry option. | | |

## **5.27. Report Rate/Feedback**

| Use Case ID: | **UC-27** | USE CASE NAME | **REPORT RATE/FEEDBACK** |
| --- | --- | --- | --- |
| Create By: | PhuongPH | Date Created: |  |
| Primary Actor: | Helper | Secondary Actor | Admin |
| Description | This use case describes how a helper reports inappropriate, false, or abusive ratings/feedback left by users. The system will then notify the admin for review and possible action. | | |
| Priority: | Medium | | |
| Trigger | Helper selects “Report” on a specific rating or feedback entry. | | |
| Preconditions: | PRE-01: Helper is logged in.  PRE-02: At least one feedback entry is available to report. | | |
| Post-Conditions | POST-01: A report is submitted and stored for admin review.  POST-02: Admin is notified to review and respond to the report. | | |
| Basic Flow: | 1. Helper navigates to the “Feedback” section. 2. Helper selects a feedback/rating they believe is inappropriate or incorrect. 3. Helper clicks on the “Report” button. 4. System prompts for a reason or description. 5. Helper provides the report reason and submits. 6. System stores the report and notifies admin. 7. Admin reviews and takes appropriate action (e.g., remove, warn, reject). | | |
| Alternative Flow: | 4a.1 (Predefined Reasons): System offers predefined report reasons (e.g., offensive language, false accusation).  6a.1 (Status Tracking): Helper can view report status (e.g., pending, resolved). | | |
| Exception: | 3e.1 (Multiple Reports): If the same feedback is already under review, system informs the helper.  6e.1 (Submission Failure): System shows error if the report submission fails. | | |

## **5.28. Censoring Helper application**

| Use Case ID: | **UC-28** | USE CASE NAME | **REPORT RATE/FEEDBACK** |
| --- | --- | --- | --- |
| Create By: | PhuongPH | Date Created: |  |
| Primary Actor: | Helper | Secondary Actor | Admin |
| Description | This use case describes how an admin reviews and takes action on helper registration applications, including approving, rejecting, or requesting revisions before a helper account becomes active. | | |
| Priority: | High | | |
| Trigger | Admin accesses the helper application management section. | | |
| Preconditions: | PRE-01: One or more helper applications have been submitted.  PRE-02: Admin is logged in with appropriate access rights. | | |
| Post-Conditions | POST-01: Helper receives application status (approved, rejected, or revision requested).  POST-02: System updates the helper’s application status in the database. | | |
| Basic Flow: | 1. Admin navigates to the helper application review section. 2. System displays a list of pending helper applications. 3. Admin selects a specific application to review. 4. Admin views submitted data, including documents and service preferences. 5. Admin chooses to approve, reject, or request additional information. 6. System updates the application status and notifies the applicant. | | |
| Alternative Flow: | 5a.1 (Request Revision): Admin sends back the application with comments or missing information notice.  5a.2 (Auto-Approve): If certain criteria are met, system may suggest auto-approval for trusted applicants. | | |
| Exception: | 3e.1 (Invalid Documents): If required documents are unreadable or missing, admin is prompted to flag the issue.  6e.1 (Notification Failure): If system fails to notify the helper, admin is alerted to take manual action. | | |

## **5.29. Ban/Unban Profiles**

| Use Case ID: | **UC-29** | USE CASE NAME | **BAN/UNBAN PROFILES** |
| --- | --- | --- | --- |
| Create By: | PhuongPH | Date Created: |  |
| Primary Actor: | Admin | Secondary Actor | User, Helper |
| Description | This use case describes how an admin bans or unbans a user or helper profile due to violations of terms, inappropriate behavior, or upon resolution of previous issues. | | |
| Priority: | High | | |
| Trigger | Admin selects a profile from the user/helper management dashboard and chooses to ban or unban. | | |
| Preconditions: | PRE-01: Admin is logged in with the appropriate role.  PRE-02: The target profile exists and is active or previously banned. | | |
| Post-Conditions | POST-01: The profile is updated with new status (banned/unbanned).  POST-02: The affected party is notified of the status change.  POST-03: Banned users/helpers are restricted from logging in or accessing services. | | |
| Basic Flow: | 1. Admin navigates to the profile management section. 2. Admin selects a user or helper profile. 3. Admin clicks “Ban” and provides a reason (e.g., violation, abuse, fraud). 4. System updates the profile status to “Banned.” 5. System restricts access and sends a notification to the user/helper.   —--------------------------------------------------------------------   1. Admin searches for and selects a banned profile. 2. Admin clicks “Unban” and optionally adds a note (e.g., issue resolved). 3. System updates the status to “Active.” 4. Profile regains full access and is notified of the unban. | | |
| Alternative Flow: | 3a.1 (Temporary Ban): Admin may set a ban duration (e.g., 7 days).  4a.1 (Appeal Review): Admin may unban based on appeal and internal review. | | |
| Exception: | 2e.1 (Non-existent Profile): System shows error if the profile cannot be found.  5e.1 (Notification Failure): If system fails to notify the banned/unbanned user, admin is prompted to act manually. | | |

## **5.30. View Request on platform**

| Use Case ID: | **UC-30** | USE CASE NAME | **VIEW REQUEST ON PLATFORM** |
| --- | --- | --- | --- |
| Create By: | PhuongPH | Date Created: |  |
| Primary Actor: | Admin | Secondary Actor | User, Helper |
| Description | This use case describes how an admin views all service requests submitted on the platform, including their statuses, associated users/helpers, and session details for monitoring and management purposes. | | |
| Priority: | High | | |
| Trigger | Admin selects “View Requests” or accesses the request management dashboard. | | |
| Preconditions: | PRE-01: Admin is logged in with the appropriate permissions.  PRE-02: At least one service request has been submitted by a user. | | |
| Post-Conditions | POST-01: A list of service requests is displayed with relevant details.  POST-02: Admin may take further actions such as filtering, exporting, or flagging a request. | | |
| Basic Flow: | 1. Admin navigates to the “Requests” section of the admin panel. 2. System displays a list of all user-submitted service requests. 3. Admin can view each request's details:  * Request ID * User and Helper (if matched) * Services selected * Scheduled time and location * Request status (pending, matched, completed, cancelled)  1. Admin may filter by date, status, user, or location. 2. Admin clicks a request to view full details. 3. Admin may take actions such as flagging, marking as urgent, or reviewing history. | | |
| Alternative Flow: | 4a.1 (Export Option): Admin exports request data for reporting (e.g., CSV, Excel).  5a.1 (Quick View): Admin previews details in a pop-up without navigating away. | | |
| Exception: | 2e.1 (No Requests Found): If no data is available, system displays “No requests found.”  5e.1 (Load Failure): If request details fail to load, system shows an error and retry option. | | |

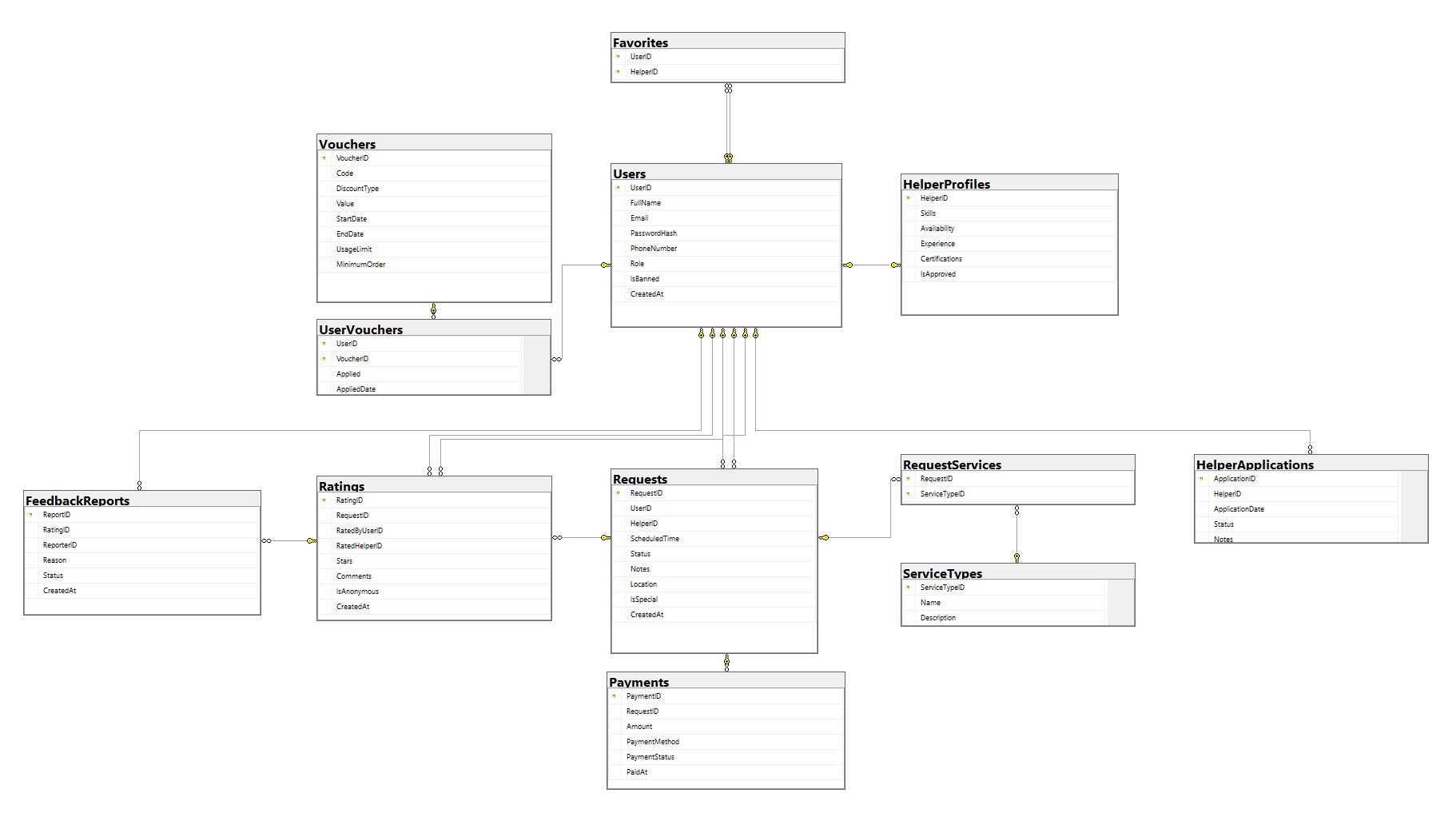
## **5.31. Create Report/ Analytics**

| Use Case ID: | **UC-31** | USE CASE NAME | **CREATE REPORT / ANALYTICS** |
| --- | --- | --- | --- |
| Create By: | PhuongPH | Date Created: |  |
| Primary Actor: | Admin | Secondary Actor | none |
| Description | This use case describes how an admin generates reports or views analytics regarding service usage, user activity, revenue, helper performance, and system trends to support decision-making and platform optimization. | | |
| Priority: | Medium | | |
| Trigger | Admin accesses the “Reports” or “Analytics” section from the admin dashboard. | | |
| Preconditions: | PRE-01: Admin is logged in with reporting privileges.  PRE-02: The system has collected relevant data (e.g., transactions, sessions, ratings). | | |
| Post-Conditions | POST-01: Report or analytics dashboard is generated or displayed.  POST-02: Admin may export, share, or act upon the insights. | | |
| Basic Flow: | 1. Admin navigates to the “Reports” or “Analytics” section. 2. Admin selects the type of report (e.g., service usage, revenue summary, user activity, top helpers). 3. Admin sets filters (e.g., date range, location, service type). 4. System retrieves and compiles the relevant data. 5. System generates a visual report (charts, tables, graphs) or downloadable file (e.g., CSV, PDF). 6. Admin views the report and optionally exports or shares it. | | |
| Alternative Flow: | 3a.1 (Scheduled Report): Admin schedules automated report generation (e.g., daily, weekly, monthly).  4a.1 (Custom Dashboard): Admin customizes the dashboard layout and data sources. | | |
| Exception: | 4e.1 (Data Load Failure): If data cannot be retrieved, system displays an error and suggests retry.  5e.1 (No Data): If no data matches the filters, system displays “No data available for selected criteria.” | | |

## 

# **6.** **DATA STRUCTURE**

## **6.1** **Database**



# **7.** **NON-FUNCTIONAL**

## **1. External Interfaces**

### **1.1. User Interface**

* UI-01: The system must display Success Message or Error Message when completing an action.
* UI-02: When the user is doing the Delete option, there must be a Confirm (Y/N) dialog.
* UI-03: The UI must minimise horizontal scrolling.
* UI-04: All pages in the system must have a Header menu and a Navigation menu.

### **1.2. Software Interface**

* SI-01: Users can use Google services as Login, which will help them to authenticate their account.
* UI-02: When the user is doing the Delete option, there must be a Confirm (Y/N) dialog.
* UI-03: The UI must minimise horizontal scrolling.
* UI-04: All pages in the system must have a Header menu and a Navigation menu.

## **2. Quality Attributes**

### **2.1. Availability**

* The Club management System should be accessible for at least 95% of the time
* The Club management System is accessible 24/7 from Monday to Sunday, except for the maintenance hours of 23:00-24:00 on Sundays.

### **2. 2 Usability**

* An untrained user can use the payment function within a maximum of 15 minutes.
* Links and buttons are easily clickable.

### **2.3. Reliability**

* Under the worst conditions, the mean time between failures should be no more than 5%.
* The platform's ability to recover from any issues should take no longer than 5 hours.

### **2.4. Performance**

* The platform’s loading time should not exceed 2 seconds.
* The platform’s content should be both rendered and readable within a maximum of 3 seconds
* Processing times of the most complexes request must complete within 3s (not include timeout ones)
* Uploads of large images (>20mb) should be complete processing after 3s.
* The platform has the ability to handle up to 200 requests per second.

### **2.5. Security**

* The user’s password should be encrypted using the Bcrypt algorithm before being stored in the database.
* Data backup should be performed every month.

# **8.** **APPENDICES**

## Appendix A: Acronyms and Abbreviations

A list of commonly used acronyms and abbreviations in this document:

* **SDS**: Software Design Specification
* **UI**: User Interface
* **API**: Application Programming Interface
* **DB**: Database
* **RBAC**: Role-Based Access Control

## Appendix B: References

A list of documents, tools, and resources referenced in this specification:

* System architecture diagrams
* Database schema design documents
* Relevant API documentation
* Security and compliance guidelines

## Appendix C: Tools and Technologies

A summary of the tools and technologies used in the system:

* **Backend**: NET 9.0, Kotlin
* **Frontend**: HTML, CSS, JavaScript, Android
* **Database**: SQL Server
* **Version Control**: Git, GitHub
* **Testing Framework**: JUnit

## Appendix D: Glossary

Definitions of key terms used in this document:

* **Scalability**: The ability of the system to handle growth efficiently.
* **Microservices**: An architectural style where applications are composed of loosely coupled services.
* **Authentication**: The process of verifying user identity.
* **Encryption**: The process of encoding information to protect it from unauthorized access.

## Appendix E: Contact Information

Relevant contact details for project stakeholders and support: