Kontroller

Use-Case Specification

Version 3.0

Revision History

| **Date** | **Version** | **Description** | **Author** |
| --- | --- | --- | --- |
| 15/06/2025 | 1.0 | Detail the first draft of the use case’s description | Nguyen Gia Nghi |
| 07/07/2025 | 2.0 | Add more crucial use case (from 12 to 17) to the report, modify some old use cases (6 and 7), as well as the use-case model | Nguyen Gia Nghi |
| 25/07/2025 | 3.0 | Revise All User-case Specification | All members |
|  |  |  |  |

Table of Contents

[**1. Use-case Model 4**](#_heading=h.ghvk8ablbjrc)

[**2. Use-case Specifications 5**](#_heading=h.6zfqgtam836d)

[2.1 Use-case 1- User: Log in to the app 5](#_heading=h.52oz60k8v2jo)

[2.2 Use Case 2 – User: Sign up for a new account 6](#_heading=h.a3a6maexxp9z)

[2.3 Use Case 3 – User: Rate a game 7](#_heading=h.xarag9r92wk5)

[2.4 Use Case 4 – User: Comment on a game 8](#_heading=h.3pdioyenm6d7)

[2.5 Use Case 5 – User: Make a review 9](#_heading=h.3pdioyenm6d7)

[2.6 Use Case 6 – User: Manage personal profile (avatar/ name/ bio/ PlayLater list) 10](#_heading=h.3pdioyenm6d7)

[2.7 Use Case 7 – User: Search for games (or lists and members) 10](#_heading=h.wkd1tgkc8brk)

[2.8 Use Case 8 – User: View content 11](#_heading=h.x2uq3mwc314s)

[2.9 Use Case 9 – User: Follow another user 11](#_heading=h.uswhxdsf9agm)

[2.10 Use Case 10 – User: Block another user 12](#_heading=h.4usvjkxg2sdd)

[2.11 Use Case 11 – User: Like a comment or review 12](#_heading=h.vwnxdeiuhb99)

[2.12 Use Case 12 – User: Create a game list 13](#_heading=h.vshsznjbn9zt)

[2.13 Use Case 13 – User: Delete a diary log 14](#_heading=h.c86oqaz3ta59)

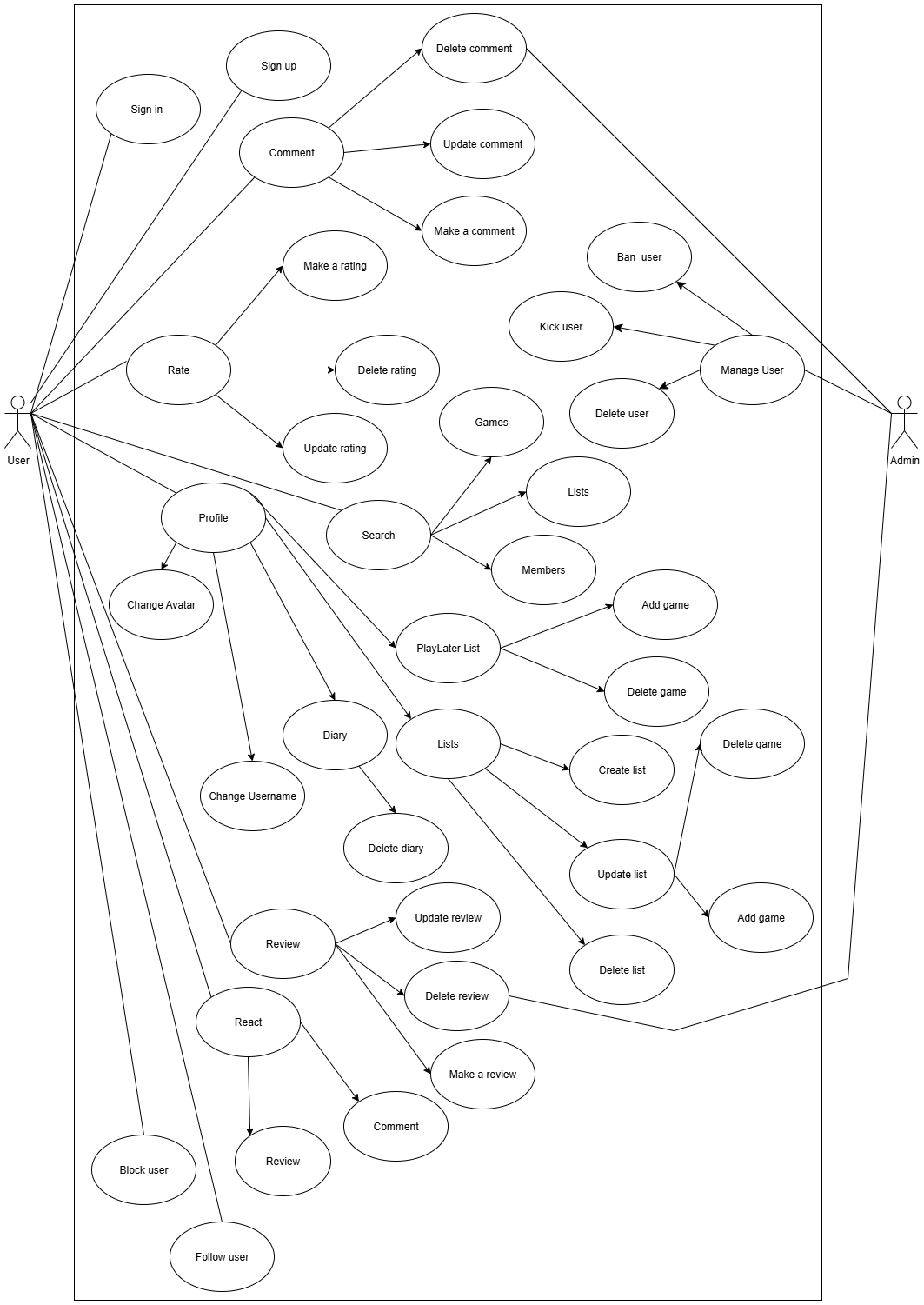
[2.14 Use Case 14 – User: Add/Remove games from a list 15](#_heading=h.43afy4yt9rqk)

[2.15 Use Case 15 – Admin: Ban a user 15](#_heading=h.3pdioyenm6d7)

[2.16 Use Case 16 – Admin: Manage site content 16](#_heading=h.5fsnnjsjqdze)

[2.17 Use Case 17 – Admin: Manage user passwords 16](#_heading=h.3pdioyenm6d7)

# Use-case Model



# Use-case Specifications

## Use-case 1- User: Log in to the app

| **Use case Name** | **Log in to the platform** |
| --- | --- |
| **Brief description** | This use case describes how a user logs into the platform using a registered account or third-party |
| **Actors** | User |
| **Basic Flow** | 1. At the homepage, the user selects a login method (Account, Facebook, or Google)  2. User enters credentials (or continues via selected third-party service)  3. System validates credentials  4. System redirects user to the homepage |
| **Alternative Flows** | **Alternative flow 1: Invalid credentials**  1. From #2 of the basic flow, user enters wrong credentials  2. System notifies login failure and allows retry  **Alternative flow 2: Third-party login fails**  1. From #1 of the basic flow, login with Facebook/Google fails  2. System notifies error and returns user to login options |
| **Pre-conditions** | User is not currently logged in |
| **Post-conditions** | User is successfully logged into their account and redirected to the homepage |

## Use Case 2 – User: Sign up for a new account

| **Use case Name** | **Sign up for a new account** |
| --- | --- |
| **Brief description** | This use case describes how a user registers for a new account on the platform using email or a third-party service. |
| **Actors** | User |
| **Basic Flow** | 1. User selects "Sign up" on the homepage 2. User chooses method: Email, Facebook, or Google 3. User enters required information or authorizes third-party service 4. System validates input and creates new user 5. User is logged in and redirected to homepage |
| **Alternative Flows** | **Alt 1: Email already in use or email doesn’t exist**  - From #3, system rejects the email  - User is asked to choose another email  **Alt 2**: **Third-party registration fails**  - From #3, Facebook/Google authorization fails  - User is returned to signup screen with error message |
| **Pre-conditions** | User is not signed in |
| **Post-conditions** | User account is created, and user is signed in |

## Use Case 3 – User: Rate a game

| **Use case Name** | **Rate a game** |
| --- | --- |
| **Brief description** | This use case describes how a user can submit, update, or delete a rating for a game. |
| **Actors** | User |
| **Basic Flow** | 1. User navigates to a game’s page  2. User clicks on the rating widget  3. User selects a rating value (e.g. 4/5)  4. System saves and displays the rating |
| **Alternative Flows** | **Alternative flow 1: Update rating**  1. From #4, user clicks to change the rating  2. System updates the rating  **Alternative flow 2: Delete rating**  1. From #4, user clicks “Remove rating”  2. System deletes the rating |
| **Pre-conditions** | User must be logged in and on a game’s detail page |
| **Post-conditions** | A new rating is submitted, updated, or deleted successfully |

## 

## Use Case 4 – User: Comment on a game

| **Use case Name** | **Comment on a game** |
| --- | --- |
| **Brief description** | This use case describes how a user can make, update, or delete a comment on a game. |
| **Actors** | User |
| **Basic Flow** | 1. User navigates to a game’s page  2. User clicks on one review of another user and writes a comment  3. User submits the comment  4. System saves and displays the comment under the game |
| **Alternative Flows** | **Alternative flow 1: Update comment**  1. From #4 of the basic flow, user clicks “Edit” on their comment  2. User updates the content and submits  3. System updates and displays the new comment  **Alternative flow 2: Delete comment**  1. From #4 of the basic flow, user clicks “Delete”  2. System prompts for confirmation  3. User confirms, and system removes the comment |
| **Pre-conditions** | User must be logged in and on a game’s detail page |
| **Post-conditions** | The comment is created, updated, or deleted successfully |

## 

## Use Case 5 – User: Make a review

| **Use case Name** | **Make a review** |
| --- | --- |
| **Brief description** | This use case describes how a user can write and manage a detailed review of a game. |
| **Actors** | User |
| **Basic Flow** | 1. User navigates to the game’s page  2. User clicks “Write a Review”  3. User writes and submits the review  4. System saves and displays it |
| **Alternative Flows** | **Alternative flow 1: Edit review**  1. From #4, user clicks “Edit” on an existing review  2. System displays editable form  3. User updates content and submits  **Alternative flow 2: Delete review**  1. From #4, user clicks “Delete”  2. System prompts for confirmation and deletes it |
| **Pre-conditions** | User is logged in and on a game’s detail page |
| **Post-conditions** | Review is added, updated, or deleted successfully |

## 

## Use Case 6 – User: Manage personal profile (avatar/ name/ bio/ PlayLater list)

| **Use case Name** | **Manage personal profile** |
| --- | --- |
| **Brief description** | This use case describes how a user can manage their PlayLater list, diary entries, stats, and avatar. |
| **Actors** | User |
| **Basic Flow** | 1. User navigates to their profile page  2. User selects the feature (e.g. PlayLater list, stats, etc.)  3. System displays editable content  4. User adds or updates content (e.g. adds game to PlayLater list)  5. System saves and reflects the changes |
| **Alternative Flows** | **Alternative flow 1: Avatar change fails (for avatar modification)**  1. User uploads an unsupported file format  2. System notifies user and blocks upload  **Alternative flow 2: Name or bio change fails (for name/ bio modification)**  1. User submits an unsupported name/ writing (too long, weird symbols, etc.)  2. System notifies user of a failed name change attempt |
| **Pre-conditions** | User is logged in |
| **Post-conditions** | Profile updates are saved and reflected in the system |

## 

## Use Case 7 – User: Search for games (or lists and members)

| **Use case Name** | **Search for games** |
| --- | --- |
| **Brief description** | This use case allows a user to search for games by title, genre, platform, or release year, lists or members by name |
| **Actors** | User |
| **Basic Flow** | 1. User enters a keyword or selects filters  2. User clicks on "Search"  3. System queries the database or IGDB API  4. Results are displayed to the user |
| **Alternative Flows** | **Alt 1: No results found**  - From #3, system shows a “No results” message  - User is encouraged to refine their search |
| **Pre-conditions** | User is on the search interface |
| **Post-conditions** | Search results are returned or refined suggestions are shown |

## Use Case 8 – User: View content

| **Use case Name** | **View content (Profiles, Reviews, Lists)** |
| --- | --- |
| **Brief description** | This use case describes how users can view other users' profiles, reviews, and public lists. |
| **Actors** | User |
| **Basic Flow** | 1. User selects a profile, list, or review from activity feed or search  2. System retrieves and displays the selected content  3. User reads or interacts (like, comment if allowed) |
| **Alternative Flows** | None |
| **Pre-conditions** | Content exists and is accessible by current user |
| **Post-conditions** | User successfully views the content |

## 

## Use Case 9 – User: Follow another user

| **Use case Name** | **Follow another user** |
| --- | --- |
| **Brief description** | This use case describes how a user follows another user to see their activity. |
| **Actors** | User |
| **Basic Flow** | 1. User visits another user's profile  2. User clicks the “Follow” button  3. System sends follow request and updates database  4. Followed user is notified  5. System displays updated follower count on both users’ accounts |
| **Alternative Flows** | **Alternative flow 1: User already follows the other user**  – System displays “Unfollow” instead of “Follow” |
| **Pre-conditions** | User is logged in and viewing another profile |
| **Post-conditions** | Follow is successfully recorded; updates shown on feed and profile |

## 

## Use Case 10 – User: Block another user

| **Use case Name** | **Block another user** |
| --- | --- |
| **Brief description** | This use case describes how a user blocks another user to prevent interaction. |
| **Actors** | User |
| **Basic Flow** | 1. User opens another user's profile  2. User clicks “Block”  3. System updates the user’s block list  4. All further interactions from blocked user are restricted |
| **Alternative Flows** | **Alternative flow 1: User attempts to block themselves**  - System prevents action and shows error |
| **Pre-conditions** | User must be logged in and viewing a valid user profile |
| **Post-conditions** | The blocked user can no longer view or interact with the user’s profile |

## 

## Use Case 11 – User: Like a comment or review

| **Use case Name** | **Like a comment or review** |
| --- | --- |
| **Brief description** | This use case describes how a user likes someone’s review or comment. |
| **Actors** | User |
| **Basic Flow** | 1. User reads a review or comment  2. User clicks “Like” icon  3. System registers the like and updates like counter |
| **Alternative Flows** | **Alternative flow 1: User already likes the other user**  – System displays “Unlike” button instead of “Like” |
| **Pre-conditions** | User is logged in and interacting with a visible comment or review |
| **Post-conditions** | The system displays updated like count and stores user interaction |

## 

## Use Case 12 – User: Create a game list

| **Use case Name** | **Create a game list** |
| --- | --- |
| **Brief description** | This use case describes how a user creates and publishes a custom list of games. |
| **Actors** | User |
| **Basic Flow** | 1. User navigates to "Lists" section  2. Clicks "Create New List"  3. User inputs title, description, and selects games  4. User submits the list  5. System stores and displays the new list on the profile |
| **Alternative Flows** | **Alternative flow 1: User submits list without any games**  - System alerts: “Add at least one game”  **Alternative flow 2: User submits list without a title**  - System alerts: “Input the title”  **Alternative flow 3: User chooses to cancel (delete the list before submitting it)**  - System removes the list from database |
| **Pre-conditions** | User is logged in and has access to list creation UI |
| **Post-conditions** | List is created and displayed on profile and public list feed |

## 

## Use Case 13 – User: Delete a diary log

| **Use case Name** | **Delete a diary log** |
| --- | --- |
| **Brief description** | This use case describes how a user removes a previously logged diary entry from their profile. |
| **Actors** | User |
| **Basic Flow** | 1. User goes to their profile  2. User opens the Diary section  3. User selects a specific diary entry  4. Clicks “Delete”  5. System asks for confirmation  6. User confirms  7. System removes the diary log from database |
| **Alternative Flows** | **Alternative flow 1: User cancels deletion at confirmation step**  – System does nothing and returns to diary view |
| **Pre-conditions** | User must be logged in and have existing diary logs |
| **Post-conditions** | Selected diary entry is deleted and no longer visible in diary history |

## 

## Use Case 14 – User: Add/Remove games from a list

| **Use case Name** | **Add/Remove game from a list (Public List or PlayLaterList)** |
| --- | --- |
| **Brief description** | This use case describes how a user can add or remove a game from either a custom public list or their PlayLaterList. |
| **Actors** | User |
| **Basic Flow** | 1. User views a game profile  2. User clicks on “Add to List” or “Add to PlayLater” button  3. System displays available lists or creates a new one  4. User selects the list  5. Game is added to the selected list  6. 6. A confirmation appears on screen |
| **Alternative Flows** | **Alternative flow 1: Game already exists in the selected list**  – System displays "Already in list" message and disables action  **Alternative flow 2: User removes the game**  – User clicks "Remove", system asks for confirmation and then deletes game from list |
| **Pre-conditions** | User must be logged in and the game must exist in the system |
| **Post-conditions** | The selected game is either added to or removed from the specified list successfully |

## 

## Use Case 15 – Admin: Ban a user

| **Use case Name** | **Ban a user** |
| --- | --- |
| **Brief description** | This use case describes how an admin can ban a user from the platform. |
| **Actors** | Admin |
| **Basic Flow** | 1. Admin navigates to the user management panel  2. Admin searches or selects a user  3. Admin clicks “Ban”  4. System marks the user as banned and restricts their access |
| **Alternative Flows** | **Alternative flow 1: Invalid ban attempt**  1. From #3, when admin bans a non-existent user  2. System shows error message |
| **Pre-conditions** | Admin is logged in |
| **Post-conditions** | Targeted user is banned and access is revoked |

## Use Case 16 – Admin: Manage site content

| **Use case Name** | **Manage site content** |
| --- | --- |
| **Brief description** | This use case describes how an admin manages reviews, comments, and user reports. |
| **Actors** | Admin |
| **Basic Flow** | 1. Admin accesses moderation dashboard  2. Admin views pending items or flagged content  3. Admin reviews each item and chooses to approve, delete, or ignore  4. System updates the database accordingly |
| **Alternative Flows** | **Alternative flow 1: Content already deleted**  1. From #3, when admin moderates an item that has already been removed  2. System informs admin and refreshes the list |
| **Pre-conditions** | Admin is logged in with proper privileges |
| **Post-conditions** | Content is moderated successfully and platform remains clean |

## 

## Use Case 17 – Admin: Manage user passwords

| **Use case Name** | **Manage user passwords** |
| --- | --- |
| **Brief description** | This use case describes how an admin helps a user reset their password when the user  cannot recover it via the standard method. |
| **Actors** | Admin |
| **Basic Flow** | 1. User contacts admin for help with password  2. Admin verifies user identity manually (e.g. via email or user info)  3. Admin resets password or sends reset link manually  4. User receives new credentials or resets instructions  5. User logs in with the new password |
| **Alternative Flows** | **Alternative flow 1: Identity verification fails**  - From #2, admin informs the user of failure and requests more proof |
| **Pre-conditions** | User cannot recover password through automatic means |
| **Post-conditions** | User is able to log in again with new or reset credentials |

## 