

Use Case Analysis - Orbit

System Actors

Actors represent the entities that interact with the system. In the Orbit context we logically distinguish:

- **Registered User (Player):** The main actor who uses the purchase and gaming services.
- **Publisher:** A user with special permissions to upload and sell their own games on the platform.
- **Administrator (Admin):** A super-user responsible for platform moderation.
- **Database (External System):** Passive entity that saves information (transactions, users, games, reviews).

Primary Use Cases

1 - Registration and Authentication (Login/Signup)

- **Actor:** User or New User.
- **Description:** The user creates a new account or accesses an existing one to unlock the platform's functionalities.
- **Flow:**
 1. The user enters nickname and password.
 2. The system verifies in the Database if the credentials are correct (Login) or if the nickname is available (Signup).
 3. In case of success, the system loads the user profile with the related roles (Admin/Publisher) and balance.
- **Exceptions:** UserNotFoundException, WrongPasswordException, PlayerAlreadyExistException.

2 - Game Purchase

- **Actor:** Registered User.
- **Description:** The user buys a game present in the store using their virtual balance.
- **Prerequisites:** The user is logged in and does not already own the game.

- **Flow:**

1. The user selects a game from the shop.
2. The user confirms the purchase intention.
3. The system verifies if the balance is sufficient (AmountNotValidException).
4. The system subtracts the amount from the user's balance.
5. The system adds the game to the user's library.
6. The system records the transaction in the history.

- **Consequences:** The balance is updated and the game is accessible in the library.

3 - Game Publication

- **Actor:** Publisher.
 - **Description:** A publisher inserts a new title in the Orbit catalog for sale.
 - **Flow:**
1. The publisher accesses the publication section.
 2. Inserts the game metadata: Title, Base Price, Genre/Tag and cover image path (cover_path).
 3. The system saves the new game in the Database.
- **Postconditions:** The game is immediately visible in the Shop for all users.

4 - Wallet Management (Gift Card Redemption)

- **Actor:** Registered User.

- **Description:** The user increases their balance by redeeming a unique code.

- **Flow:**

1. The user accesses the shop and clicks to buy a game.
2. Enters the Gift Card code.
3. The system verifies the code validity in the Database (checkGiftCard).
4. If valid, the system adds the value to the user balance and removes/invalidates the card from the Database.

- **Exceptions:** CodeNotFoundException.

Secondary Use Cases

Library Consultation

- **Actor:** Registered User.

- **Description:** The user views the list of purchased games.

- **Flow:** The system queries the Database retrieving all game IDs associated with the user ID and displays their details (Cover, Title).

Review Writing

- **Actor:** Registered User.

- **Description:** The user leaves feedback on a game they own.

- **Preconditions:** The user must own the game in their library.

- **Flow:**

1. The user selects a game from their library.
2. The user assigns a rating (1-5) and writes a textual comment.
3. The system saves the review and recalculates the game's average rating (score).

User Moderation (Ban)

- **Actor:** Administrator.
- **Description:** The administrator blocks access to a user or publisher who has violated the rules.
- **Flow:** The admin searches for the user and sets the ban flag. The user will no longer be able to log in.

Game Moderation (Game Ban)

- **Actor:** Administrator.
- **Description:** The administrator blocks the sale of a published game that violates the platform guidelines.
- **Flow:**
 1. The administrator searches for the game in the catalog.
 2. The system sets the `is_banned` flag to 1 in the Database for that game.
 3. The game is no longer visible in the Store for new purchases (but remains in the library of those who already purchased it).

Purchase via Credit Card

- **Actor:** Registered User.
- **Description:** The user buys a game by paying directly with an external payment method (e.g. Credit Card) instead of using the Orbit wallet.
- **Flow:**
 1. The user selects a game from the shop.
 2. Chooses to pay with Credit Card at Checkout.
 3. The system processes the payment through the external interface (simulated).
 4. The system adds the game to the library and records the transaction without affecting the user's balance.
- **Exceptions:** PaymentFailedException (if the card is declined).

Discount Management (Discount)

- **Actor:** Publisher.

- **Description:** A publisher decides to apply a temporary discount to their own game to incentivize sales.

- **Flow:**

1. The publisher selects one of their published games.
2. Enters a discount percentage (e.g. 20%).
3. The system calculates the new currentPrice through the DiscountManager.
4. The system updates the price in the Database.

- **Exceptions:** AmountNotValidException (if the percentage or price are not valid).