C-DAC Mumbai
OOPJ Lab Assignment

Capstone Project – Game Hub; Game Collection Manager

START - Developer Requirements

---

1) Project Overview

Project Name: Game Hub
Project Type: Java Console Application
Duration: 120 minutes
Objective: Develop a **menu-driven game collection manager** using **Java OOP concepts** and **Collection Framework**.

---

2) Functional Requirements

2.1 Game Management

- **Add Game**

➔ Input: Name, Genre, Rating (1-5), Platform (Console/PC)
➔ Validation: Rating must be 1-5
➔ Action: Add game to allGames list
➔ Output: Success message with game ID

- **Remove Game**

➔ Input: Game ID
➔ Validation: Cannot remove if game is borrowed
➔ Output: Success/Error message

- **View Games**

➔ Options: Sort by ID, Name, Rating
➔ Output: List of games with all details (platform-specific info included)

- **Search Game**

➔ Input: Name or Genre
➔ Output: Display matching games

2.2 User Management

- **Register User**

➔ Input: Username, Email
➔ Validation: Username must be unique
➔ Output: Confirmation message

- **View Users**

➔ Output: List of all registered users

C-DAC Mumbai

2.3 Borrowing System

- **Issue Game**

➔ Input: Game ID, Username
➔ Validation: Game must be available, User must exist
➔ Action: Move game to borrowedQueue, add game to user's borrowed list
➔ Output: Success/Error message

- **Return Game**

➔ Input: Game ID, Username
➔ Validation: Only borrower can return the game
➔ Action: Move game back to available list
➔ Output: Success/Error message

- **Borrowed Games Queue**

➔ Output: Display current borrowed games in queue order

2.4 Reporting & Statistics

- **Collection Statistics**

➔ Total Games, Available Games, Borrowed Games

- **User Statistics**

➔ Total Registered Users

- **Borrowed Queue Display**

➔ Current games on loan

---

3) Technical Requirements

3.1 OOP Concepts to Implement

- **Classes & Objects:** Game, ConsoleGame, PCGame, User, GameHubManager
- **Encapsulation:** Private fields with public getters/setters
- **Inheritance:** ConsoleGame and PCGame extend Game
- **Polymorphism:** Method overriding (displayDetails())
- **Abstract Classes:** Game as base abstract class
- **Interfaces:** GameActions for managing operations
- **Constructor Chaining:** Default + parameterized constructors
- **Static Variables:** Track totalGames and totalUsers
- **toString() Override:** Clean display for objects

3.2 Collections & Advanced Features

- **ArrayList allGames** – Main game collection
- **LinkedList borrowedQueue** – Queue of borrowed games
- **HashSet users** – Maintain unique users
- **HashMap<Integer, Game> gameMap** – Quick lookup by game ID
- **Comparable Interface** – Sort games by ID
- **Comparator Interface** – Sort games by Name or Rating

CDAC Mumbai

3.3 Exception Handling

- Custom Exceptions:

➔ GameNotFoundException
➔ UserNotFoundException
➔ InvalidRatingException

END – Developer Requirements

User Stories & Acceptance Criteria (User POV)

User Story 1 – Add Game

- **As a user**, I want to add a new game to my collection so that I can track it.
- **Acceptance Criteria:**

➔ Game ID auto-generated
➔ Rating validated (1–5)
➔ Confirmation displayed

User Story 2 – Borrow Game

- **As a user**, I want to borrow a game so that I can play it.
- **Acceptance Criteria:**

➔ Game must exist and be available
➔ User must be registered
➔ Borrowed queue updates automatically

User Story 3 – Return Game

- **As a user**, I want to return a borrowed game so it becomes available for others.
- **Acceptance Criteria:**

➔ Only the original borrower can return
➔ Borrowed queue updates automatically

User Story 4 – View & Sort Games

- **As a user**, I want to view my collection sorted by ID, Name, or Rating.
- **Acceptance Criteria:**

➔ Sort options selectable
➔ Output includes platform-specific info

User Story 5 – User Management

- **As an admin**, I want to register new users and prevent duplicates.
- **Acceptance Criteria:**

➔ Duplicate usernames rejected
➔ Success message on registration

CDAC Mumbai