DELIVERABLE 1

BY: ALEXANDER NIKOPOULOS

SCENARIO

The Game Library Management System (GLMS) is an application whose purpose is to keep a collection of games that can be organized and managed for regular users to purchase and add to their library. Managers can add or remove available games, view all available games, and edit game details such as title, genre, platform, and release year. Regular users can browse and search the games by title, genre, or platform. After purchasing a game, it is added to the user's game library, where they can view all their games, rate them, or remove games. The system keeps separate information for the available games and each user's owned library.

DESIGN PARADIGM

There are different available services depending on the user type:

1. **Manager:**

- Add an available game
- Remove an available game
- Edit game details (title, genre, platform, release year)
- View all available games
- Control over sorting and rating features

2. Regular User

- Browse available games
- Search games by title
- Search games by genre
- Search games by platform
- Purchase a game to add to library
- View all owned games in library
- Remove a game from their library
- Rate a game from their library

EXPECTED OUTPUT

The GLMS will allow regular users to view the available game catalog and their own game libraries. They will be able to browse, search, purchase, and rate games. After a purchase, the game will be added to their library where they can view or remove it. If they choose to rate their game, it will be presented on the game's overall rating.

Managers will be able to modify the game catalog by adding, removing, or updating game details. They can also change how games are sorted or rated.

These actions will result in the console outputting confirmations or indications for the actions taken such as successful purchases, search results, added games, removed games, updated ratings, etc.

IMPLEMENTATION DETAILS

Hierarchies: 1. User -> RegularUser 2. Game -> OwnedGame
-> Manager

Interface: Rateable

This interface is implemented by OwnedGame so that only games that are purchased and in the user's library can be rated by the regular users.

Methods that apply runtime-polymorphism:

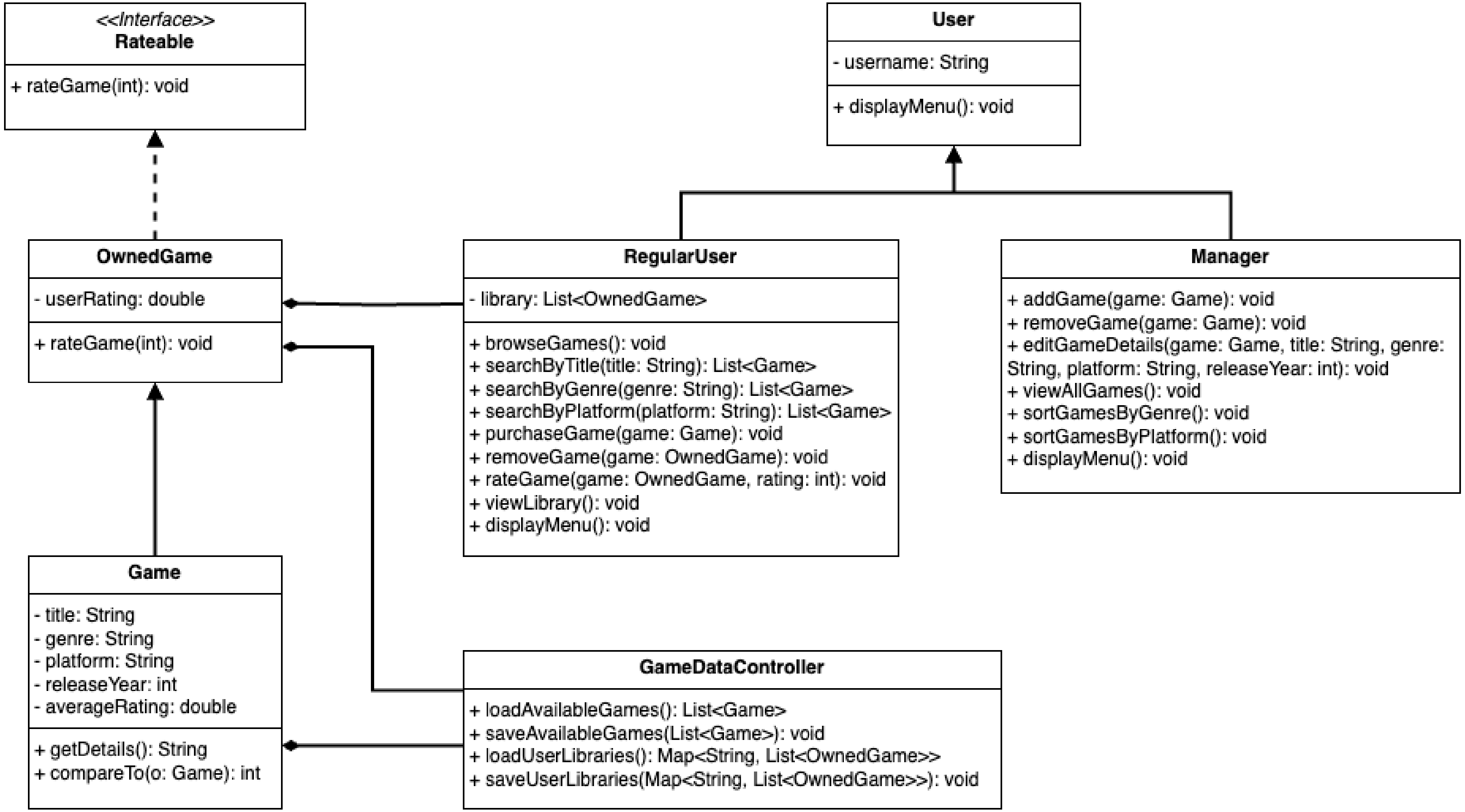
displayMenu(): Displays all available options depending on the user.

TextIO: GameDataController class will be able to read/write the available games from a file and the user libraries.

Classes implementing Comparable: Game class to sort games by title for default.

Classes using Comparators: Manager class to sort games by genre and platform.

CLASS DIAGRAM



<<Interface>>
Rateable

+ rateGame(int): void

User

- username: String

+ displayMenu(): void

OwnedGame

- userRating: double

+ rateGame(int): void

RegularUser

- library: List<OwnedGame>

+ browseGames(): void
+ searchByTitle(title: String): List<Game>
+ searchByGenre(genre: String): List<Game>
+ searchByPlatform(platform: String): List<Game>
+ purchaseGame(game: Game): void
+ removeGame(game: OwnedGame): void
+ rateGame(game: OwnedGame, rating: int): void
+ viewLibrary(): void
+ displayMenu(): void

Manager

+ addGame(game: Game): void
+ removeGame(game: Game): void
+ editGameDetails(game: Game, title: String, genre: String, platform: String, releaseYear: int): void
+ viewAllGames(): void
+ sortGamesByGenre(): void
+ sortGamesByPlatform(): void
+ displayMenu(): void

Game

- title: String
- genre: String
- platform: String
- releaseYear: int
- averageRating: double

+ getDetails(): String
+ compareTo(o: Game): int

GameDataController

+ loadAvailableGames(): List<Game>
+ saveAvailableGames(List<Game>): void
+ loadUserLibraries(): Map<String, List<OwnedGame>>
+ saveUserLibraries(Map<String, List<OwnedGame>>): void

IMPLEMENTATION FOR DELIVERABLE 2

For Deliverable 2, the features that will be implemented are:

Classes: User, RegularUser, Manager, Game

Interface: Rateable

Methods: For User: displayMenu()

For RegularUser: browseGames(), purchaseGame(game: Game), removeGame(game: OwnedGame), viewLibrary(), rateGame(game: Game, rating: int), displayMenu()

For Manager: addGame(game: Game), removeGame(game: Game), viewAllGames(), displayMenu()

For Game: getDetails(), compareTo(o: Game)