*XP Course Project*

*Game Flow Software Backend Design Document*

*Ammar Hatiya*

*Sammak Ahmed – 100747439*

*Fahad Fauzan*

*March 15, 2024*

# Table of Contents

[Table of Contents 2](#_Toc161421259)

[**Introduction:** 3](#_Toc161421260)

[**Architecture Overview:** 3](#_Toc161421261)

[**Back End Design:** 3](#_Toc161421262)

[**Class and Method Intentions:** 4](#_Toc161421263)

[**Inputs/Outputs:** 5](#_Toc161421264)

# **Introduction:**

The goal of this design document is to outline the purpose of the backend as well as visualize all of its functions/classes and how they work. To begin, the backend is designed to read through several files; Current User Accounts, Available Games, Game Collection, and the Daily Transaction Files. After reading through these files, it will merge all the Daily Transaction files and then process each transaction that occurs line by line. The transactions are essentially, adding credit, refunding purchases, buying and selling games, and creating and deleting accounts. Once these transactions are processed, it will then replace the existing files with new ones that contain the updated information.

# **Architecture Overview:**

There is one main focus:

The Back End which will be written in Python and testing using PyTest.

# **Back End Design:**

Will consist of various functions in relation to each command:

**User Module:** Focuses on handling reading the Current User Accounts file and handling that information.

**Available Games Module:** Will handle reading the Available Games file and handling that information.

**Game Collection Module:** Reads through the Game Collection file and stores and records that information

**Transactions Module:** Focuses on reading the merged transaction files and going through and processing each command; AddCredit, Refund, Buy, Sell, Create, and Delete.

**Daily Transaction Module:** Handles reading the transaction files and creating a new file that is the merged version.

**Replace Module:** Focuses on getting all the information changed by the Transactions Module and creating/replacing the old files.

# **Class and Method Intentions:**

|  |  |  |
| --- | --- | --- |
| Class | Method | Intention |
| UserController |  | Class meant to handle user account information |
|  | \_\_init\_\_(self, userPath): requires a string input of file path | Constructor to set the file path based on *input* given |
|  | getUserAccounts(self) | Function to create a list and add the appropriate information acquired from the text file |
| AvailableGamesController |  | Class for handling available games information |
|  | \_\_init\_\_(self, availableGamesPath): requires a string input of file path | Constructor to set file path based on input given |
|  | getAvailableGames(self) | Function for reading through text file and storing available games information into a list. |
| GameCollectionController |  | Class meant to handle information regarding the game collection file |
|  | \_\_init\_\_(self, gameCollectionPath) : requires a string input of file path | Constructor for setting the file path based on input given |
|  | getOwnedGames(self) | Function intended to store information acquired from game collection file into a list |
| TransactionController |  | Class meant to handle reading through inputted file and processing transactions |
|  | \_\_init\_\_(self, transactionPath) : requires a string input of file path | Constructor to set file path based on string input |
|  | processTransactions(self, allUsers, allAvailableGames, allGameCollection): requires the appropriate lists | Function meant to read through the merged transaction file and update lists based on the transaction type |
|  | processBuy(self, line, allUsers, allGameCollection) : requires the appropriate lists | Function meant to process buy command by adjusting appropriate credit amounts and updating collection list |
|  | processAddCredit(self, line, allUsers) : requires the appropriate lists | Function designed to update user credit based on input given from file |
|  | processSell(self, line, allAvailableGames, allGameCollection) : requires the appropriate lists | Function meant to process sell command by adding to available games and owned games. |
|  | processRefund(self, line, allUsers, allAvailableGames, allGameCollection) : requires the appropriate lists | Function that will process Refund command by adjusting buyer and seller credits, and updating game collections. |
|  | processCreate(self, line, allUsers) : requires the appropriate lists | Function meant to process create command by adding to allUsers list |
|  | processDelete(self, line, allUsers, allAvailableGames, allGameCollection) : requires the appropriate lists | Function intending to process Delete command by removing appropriate items from the list |
| dailyTransactionController |  | Class meant to handle merging transaction files. |
|  | \_\_init\_\_(self, transactionPath) : requires a string input of file path | Constructor to set file path based on input given |
|  | mergedDailyTransaction(self) | Function designed to merge all information in given text files into one new text file |
|  |  |  |

# **Inputs/Outputs:**

For the classes, there should be appropriate text files in the respective folders. Current User Accounts, Available Games, Game Collection, and the Daily Transaction files should be placed in the folders if not already existing. Terminal should, at this moment, only need to run python backend.py in order to run completely

Output will be either replace the current existing files with new files containing the correct information, or update the existing files with the new and correct information.