**UML Diagram – Table description of classes**

Jude AntonyRajan

Abhiram Sinnarajah

2019-02-09

# User Class

|  |  |
| --- | --- |
| **Name** | **Description** |
| username | Username of the currently active user. |
| userAccountType | Account type of the currently active user. |
| availableCredit | Available credit of the currently active user. |
| User() | Creates an object of the class for user functions. |
| stringRepresentation() | Creates a string that will be added to User Accounts File. |
| getAvailableCredit() | Retrieves the currently active user’s available credit. |
| updateAvailableCredit() | Updates the currently active user’s available credit. |
| createUser() | Checks if active user is Admin and proceeds with new user creation. |
| deleteUser() | Checks if active user is Admin and proceeds with deletion of existing a user. |
| sellTickets() | Checks if active user is not buy standard and proceeds with actions for selling tickets. |
| buyTickets() | Checks if active user is not sell standard and proceeds with actions for purchasing tickets. |
| refund() | Checks if active user is Admin and proceeds with transferring credits from seller to buyer. |
| addCreditStandardMode() | Checks if active user is not an Admin and requests system for extra credit. |
| addCreditAdminMode() | Checks if active user is Admin and prompts the admin information about user to receive credit. |
| terminateSession() | Ends session and returns a string indicating termination |
| resetInput() | Resets getline(cin) input method to prevent bugs |

# Event Class

|  |  |
| --- | --- |
| **Name** | **Description** |
| eventName | Name of the Event the ticket is for |
| hostname | Name of the seller of the tickets |
| ticketsAvailable | Amount of available tickets for sale for this event. |
| Event() | Creates an object of the Event class, for Event. |

# Daily Transaction Data Class

|  |  |
| --- | --- |
| **Name** | **Description** |
| transactionCode | Code which represents a type of transaction |
| username | Username of the current user |
| userType | Privilege type of the user |
| availableCredits | The amount of credits the account holds |
| sellerUsername | Username of the seller |
| refundCredits | Amount of credits to be refunded |
| eventName | Name of the event |
| numTicketsForSale | The amount of tickets released for sale |
| ticketPrice | Price each ticket is listed for |
| CategoryOne{}  CategoryTwo{} | Struct to overload or differentiate between different transaction types |
| DailyTransactionData() | Creates an empty DTD object |
| DailyTransactionData(CategoryOne) | Creates a DTD object for transactions create & delete user, add credits, end session |
| DailyTransactionData(CategoryTwo) | Creates a DTD object for refund transaction |
| DailyTransactionData(CategoryThree) | Creates a DTD object for sell & buy transactions |
| toStrCodeTypeOne() | Creates the string representation for CategoryOne |
| toStrCodeTypeTwo() | Creates the string representation for CategoryTwo |
| toStrCodeTypeThree() | Creates the string representation for CategoryThree |
| stringRepresentation() | Calls the appropriate category and its a string |
| floatToPrecision() | Returns a string with float value stripped to the decimal places taken as input |

# Login Interface Class

|  |  |
| --- | --- |
| **Name** | **Description** |
| username | Username inputted by the user |
| userAccountsFilePath | The file path/location of the accounts file which contains user information |
| userExist | A Boolean statement which determines if a matching user is discovered |
| currentUserData | String array which stores the user information to be transferred to main interface |
| LoginInterface() | Constructor which creates a new Login Interface object |
| isExistingUser() | Validates the existence of a user by user input |
| scanForExistingUser() | Scans user accounts file to check for existing user |
| displayLogin() | Displays the entrance to the program/ login screen |
| retrieveUserData() | A getter method which sends the user information to main system |
| exitProgram() | Exits the program upon request |

# Main Interface Class

|  |  |
| --- | --- |
| **Name** | **Description** |
| loginInterface | A new Login Interface object |
| dailyTransactionLog | Vector which stores each transaction string data |
| MainInterface() | Constructs a MainInterface object in executable main function – int main() |
| displayLoginInterface() | Calls displaylogin() method from LoginInterface class and then calls displayUserProfile() |
| displayUserProfile() | Displays the user’s details on the terminal – user view |
| displayScreen() | Displays transaction options to the user |
| runSelection() | Retrieves user input and runs the corresponding selection |
| endSession() | Completes session and calls writeToDailyTransaction() method |
| writeToDailyTransaction() | Opens daily transaction output file or creates a new file, and writes the transaction data stored in transaction log in each line |