Name: Michael Hage
Student ID: 500765523
Section Number : 08

The Use Case Diagram depicts two types of users accessing various parts of the program. The users being the average user and the manager. The user accesses the MainMenu class, to log in his account through the Authentication class and enter to the UserMainMenu class with his account info. The UserMainMenu allows the user to deposit money into the account, withdraw money, and also do transactions. Also, the UserMainMenu updates the user's account information after any transaction. The AccountInforamtion class interacts with the UserMainMenu class by retrieving the account information stored in the user's corresponding text file, updating any changes to the account information, money, status, and names, and also saving the data to the text file when the user is finished. The AccountState interface changes the state of the user's membership status when the user exceeds certain milestones with the account's balance. The AccountBalance changes from one of the three states to the other. The states include Silver, Gold and Platinum State. Furthermore, the Manager actor can access the MainMenu class to log in and access the ManagerMainMenu class. The ManagerMainMenu class is able to perform two function, adding accounts and deleting accounts. The Manager class interacts with those two functions and performs the functions by interacting with the text files.

The Class Diagram provides a much more rich background to how the classes interact with one another. The BankGUIMain provides the template of the GUIs dimensions and characteristics. The BankGUIMain sends the window to the MainMenu. The MainMenu provides the GUI to the user and manager. The mainMenuScene interacts with the Authentication class to check if the user entered in the correct information for his account or if a manager is accessing the account. If the user entered in the correct information, then the user will enter the UserMainMenu, where the user will access the deposit, withdrawal, and online transaction functions. These functions perform a single function that influence the user's account information. The account informaton is stored in the AccountInformation class. The AccountInformation class retreives the

information from text files, interacts with the information by providing updates from the transactions throughout the various methods, such as checkState, addAccountBalance, and also saves the updated information to the user's specific text file. The user's AccountInformation also has a variable that keeps track of its membership status and changes states that are dependent on the amount of money within the account. The membership changes states within the AccountInformation class by the checkState and setState methods in the class. The AccountState interface checks for the user's accountBalance and changes the account state to Silver, Gold or Platinum, if the user acquires a certain amount of money. The AccountState is stored within the AccountInformation class as an instance variable and is updated constantly. Moreover, the ManagerMainMenu, accessed by the MainMenu authentication process, is used to manage the users within the banking software. The manager can create or delete the users by accessing and sending the inputs to the Manager class. Within the Manager class, the user's are checked to see if they exist before performing any functions. This is done through the isValidUser method and sends the result to the other two methods that create the user or delete the user based on the function chosen by the manager.

The AccountInformation class houses the user's information to be manipulated and changed throughout the various interactions done by the user. The AccountInformation instance variables describe the user's names (userFirstName and userLastName), the account balance (accountBalance), the user's membership status (accountState), and the user's file location (userFile). The AccountInformation class is a valid class if the user has a name and the value isn't a null value, if the accountBalance is greater than 0, if the accountStaet isn't null, and if the user's file exists. The constructor initializes the user's information by reading from a text file that is user specific and inputting the information into the instance variables, userFirstName, userLastName, and accountBalance. The instance variable accountState is set by sending the AccountInformation into the checkState method of the accountState variable to set the proper membership status. The addAccountBalance method takes an input to the method and adds the input, regardless if it's positive or negative, to the accountBalance. This is used for every transaction the user does with depositing, withdrawing, and transactions. The getAccountState

method updates the account state by using the checkAccountState method and returning the AccountState variable. The setAccountState method sets the new AccountState variable sent from one of the States. The rest of the getters and setters perform the basic functionality of those functions. The toString method returns the account information in the form of a String variable. The toString method implements the abstraction function and checks for the rep invariant through the repOk method. The repOk method checks for the rep invariant by checking the instance variables for their conditions. The saveAccountInformation method saves the user's updated information to the user specific text file.

The State design pattern is displayed in the AccountState interface. The AccountState revolves around three States that act as bank membership levels. Once a user acquire a certain balance, the user's membership status changes to the appropriate status. The AccountState also controls the fee for the customer's online purchases. The Silver State Membership is used when the customer's balance is less than $10000. When the user reaches more than $10000 but has less than $20000, then the user is granted a Gold State Membership. Finally, when the user reaches a Platinum State Membership, the user is given a Platinum State Membership. The user's state is stored in the AccountInformation class as an instance variable. The State is updated whenever a change is made to the account's balance.