OOP CA2 Individual

C20483514

TU856/2

Bank Management System Report

**Project Classes and Methods**

**Classes:**

* **Account –** A general account parent class for account types. Its attributes are protected so only itself and its subclasses can access them. Some of the attributes have get methods as required by a method from the Customer class or a function. It has general templates for **withdrawal, deposit, transfer** and **view transactions**. Some additional methods are **get\_transactionlist** ,**receive\_transfer** and **update\_details.** Get\_transactionlist is a method for converting the transactions list attribute into a string. Each element is separated by a comma. The string is used for writing/updating to the accounts.txt file. Receive\_transfer is a method for a payee account object that adds the transaction id to its transactions attribute. Since there is 2 accounts associated with a transaction, the transferer and the payee, the payee account must also record the transaction as well. Update\_details is a method that writes all the accounts attributes to the account.txt updating the appropriate account line
* **SavingsAccount –** A subclass of account which defines a savings account type. As it is a subclass, most of the method are the same as Account. **Withdrawal** and **transfer** methods include the error validation for a 1 transfer/withdrawal transaction per month as according to the specification. **Update\_details** has the specific format for the class
* **CheckingAccount –** A subclass of account which defiens a checking account type. Similar to SavingsAccount there is not many differences with the parent class. **Withdrawal** and **transfer** methods are the same except the funds limit also includes the credit limit. **Update\_details** has the specific format for the class
* **Customer –** A class defining a general bank customer. All the attributes are private for security concerns. Get methods are in place for appropriate functions. Other methods are **new\_account** and **delete\_account.** New\_account is a method for creating a new account. It links the new account object to the customer accounts attribute and updates the accounts.txt file. Delete\_account does the opposite by severing the link between the account and customer. However, the account does not get deleted from accounts.txt file in the case of funds tracking/logistics reasons in a “real world” application. It is also good for archiving.

**User Manual**

The program was made to be very simple to read and input. On startup, user must input their customer id along with the corresponding PIN to login (test customer has an id of 1 and a pin of 1234, can be seen in customer.txt). Once logged in, various menu options are available. User should input the menu option number corresponding to the menu option they would like to choose. Follow the prompts and look out for any error/incorrect messages which will indicate if an action is disallowed/cannot be done or an input was wrong.

**Difficulties and more challenging parts**

The biggest difficulty with this assignment was finding the motivation to start it. Procrastination again got the better of me and here I am 2 days late writing this report with a 16% penalty. Needless to say, I am very disappointed with myself. I think I did a good job implementing everything and I thoroughly enjoyed the process of building up the code and ironing out the bugs. However, obviously my final grade will not reflect that and all the unnecessary stress has been draining. I definitely do not want to go through this again and will try to keep in mind in the future of deadlines and unforeseen circumstances/delays that can happen when working on a project.

Onto the actual code and debugging. The specification is deceptively simple and I did not expect the depth of my classes I had to code and all the methods and workarounds. It was a lot of thinking and code to figure out. For some sections of the code, I used a test python file and txt file that used a similar system but with a much simpler and smaller sample set. Once I got that working, I implemented it appropriately into the project. Nothing gave me too much hardship but it was a hefty enough assignment. The most challenging part of the assignment has to be figuring out a good format for the txt files in order to save objects and then implementing it. There was a lot of bugs and unexpected writes to the files but they were all eventually sorted out. Overall, It was enjoyable but very stressful given my stupidity. I’d have liked to develop it further but with the looming deadline, this is the best I could do.