**AC51002 – Software Development**

**Course Work 2**

**Name: Akachukwu Adiele**

**CLASS DESIGN**

**account**

accountId

customerLoginId

accountType

currency

accountBalance

creationDate

accountStatus

deposit(depositAmount)

withdraw(withdrawalAmount)

saveAccountState(accountsDirectory)

**customer**

firstName

lastName

dateOfBirth

phone

email

occupation

customerLoginId

customerPassword

creationDate

status

customerAccountsList

customerId

generateCustomerId(customersDirectory)

saveCustomerState(customersDirectory)

**mortgageAccount**

mortgagePrincipal

mortgageTerm

repaymentAccount

calculateInterestAmount()

calculateRepaymentAmount()

saveAccountState(accountsDirectory)

**currentAccount**

applyInterest

self.interestRate

saveAccountState(accountsDirectory)

calculateInterestAmount()

**savingsAccount**

applyInterest

self.interestRate

saveAccountState(accountsDirectory)

calculateInterestAmount()

**DEFINITIONS:**

1. **‘customer’ Class**

This class is used to represent a client of the bank. An instance of this class would have the below attributes and functions:

Attributes:

* firstName – A variable to hold the first name.
* lastName – A variable to hold the last name.
* dateOfBirth – A variable to hold the date of birth.
* phone – A variable to hold the phone number.
* email – A variable to hold the email address.
* occupation – A variable to hold the occupation.
* customerLoginId – A variable to hold the login id for the customer.
* customerPassword – A variable to hold the login password for the customer.
* creationDate – A variable to hold the date the customer record was created.
* status – A variable to hold the status of the customer (e.g. Created).
* customerAccountsList – A list to hold the account ids owned by the customer.
* customerId – A variable to hold the generated id for the customer record.

Functions:

* generateCustomerId(customersDirectory) – A method to implement a logic to generate the unique customer id.
* saveCustomerState(customersDirectory) – A method to update the customer text file for the instance of customer in question.

1. **‘account’ Class**

This class is used to represent a client’s account with the bank. An instance of this class would have the below attributes and functions:

Attributes:

* accountId – A variable to hold the unique account id.
* customerLoginId – A variable to hold the login id for the customer that owns this account.
* accountType – A variable to hold the type of account (savings, current, mortgage accounts).
* currency – A variable to hold the currency for this account.
* accountBalance – A variable to hold the account’s balance.
* creationDate – A variable to hold the date the account was created.
* accountStatus – A variable to hold the status of the account.

Functions:

* deposit(depositAmount) – A method to trigger a deposit into the account
* withdraw(withdrawalAmount) – A method to trigger a withdrawal from the account
* saveAccountState(accountsDirectory) – A method to update the account text file for the instance of account in question.

The ‘account’ class serves as a super class for the ‘savingsAccount’, ‘currentAccount’ and ‘mortgageAccount’

1. **‘savingsAccount’ Class**

This class is used to represent a client’s savings account with the bank. It is created as a subclass of the ‘account’ superclass.

An instance of this class would have the below attributes and functions:

Attributes:

* applyInterest – A variable to hold the decision marker to determine if interest is applied or not
* self.interestRate – A variable to hold the interest rate for savings account

\*\*\* All other attributes of the parent class (account) are inherited.

Functions:

* saveAccountState(accountsDirectory) – A method to update the account text file for the instance of the savings account in question.
* calculateInterestAmount() – A method to calculate the interest for the current month.

\*\*\* All other functions of the parent class (account) are inherited.

1. **‘currentAccount’ Class**

This class is used to represent a client’s current account with the bank. It is created as a subclass of the ‘account’ superclass.

An instance of this class would have the below attributes and functions:

Attributes:

* applyInterest – A variable to hold the decision marker to determine if interest is applied or not
* self.interestRate – A variable to hold the interest rate for current account

\*\*\* All other attributes of the parent class (account) are inherited.

Functions:

* saveAccountState(accountsDirectory) – A method to update the account text file for the instance of the current account in question.
* calculateInterestAmount() – A method to calculate the interest for the current month.

\*\*\* All other functions of the parent class (account) are inherited.

1. **‘mortgageAccount’ Class**

This class is used to represent a client’s mortgage account with the bank. It is created as a subclass of the ‘account’ superclass.

An instance of this class would have the below attributes and functions:

Attributes:

* mortgagePrincipal – A variable to hold the mortgage principal amount.
* mortgageTerm – A variable to hold the mortgage term
* repaymentAccount – A variable to hold the mortgage repayment account.

\*\*\* All other attributes of the parent class (account) are inherited.

Functions:

* calculateInterestAmount() – A method to calculate the mortgage interest amount for the full tenure.
* calculateRepaymentAmount() – A method to calculate the mortgage repayment amount.
* saveAccountState(accountsDirectory) – A method to update the account text file for the instance of the mortgage account in question.

\*\*\* All other functions of the parent class (account) are inherited.

**TEST REPORT**

* Test Case 1 – Registering a new customer

The ‘Bank Maximus’ is launched and the functionality to ‘register a new customer’ is tested.

A screenshot of a computer screen

Description automatically generated Select option ‘2’

A screenshot of a computer

Description automatically generated The required details are filled and submitted

A screenshot of a computer screen

Description automatically generated Customer is created successfully

Customer data is then saved in a text file, with the file name as the customer login id.

A black screen with white text

Description automatically generated

* Test Case 2 – Customer logging in

A screenshot of a computer

Description automatically generated Option 1 is selected

A screen shot of a computer screen

Description automatically generated Required input for login details are entered.

A screenshot of a computer

Description automatically generated Login successful

* Test Case 3 – Opening a new account

A screenshot of a computer screen

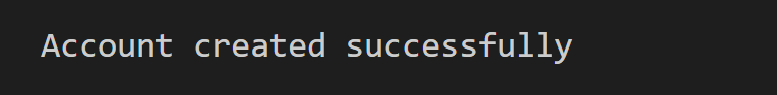
Description automatically generated Select option ‘2’ on customer menu

A screenshot of a computer

Description automatically generated Select account type

A black background with white text

Description automatically generated choose if interest should be applied or not

 Account created succesfully

Account data is then saved in a text file, with the file name as the customer login id.

A screenshot of a computer

Description automatically generated

* Test Case 4 – View account state for one account

A screenshot of a computer screen

Description automatically generated Select option ‘1’ on customer menu

A black background with white text

Description automatically generated Input the corresponding number code for account

A screen shot of a computer

Description automatically generated Select option ‘4’ on account menu

A screen shot of a computer screen

Description automatically generated State of the account is displayed

* Test Case 5 – View account state for one account

A screenshot of a computer

Description automatically generated Select option ‘3’ on customer menu

A screenshot of a computer screen

Description automatically generated State of all the accounts for the customer is displayed

* Test Case 6 – Make deposit

A screenshot of a computer program

Description automatically generated Select option ‘3’ on customer menu

A black background with white text

Description automatically generated Select an account

A screen shot of a computer

Description automatically generated Select option ‘2’ on account menu

A screen shot of a computer

Description automatically generated Deposit is completed

* Test Case 7 – Make withdrawal

A screenshot of a computer program

Description automatically generated Select option ‘3’ on customer menu

A black background with white text

Description automatically generated Select an account

A screen shot of a computer

Description automatically generated Select option ‘3’ on account menu

A screen shot of a computer

Description automatically generated Withdrawal is completed

* Test Case 8 – View Account Balance

A screenshot of a computer program

Description automatically generated Select option ‘3’ on customer menu

A black background with white text

Description automatically generated Select an account

A screen shot of a computer

Description automatically generated Select option ‘3’ on account menu

A black background with white text

Description automatically generated Account balance is displayed

* Test Case 9 – Using the admin menu to customer data

A screenshot of a computer program

Description automatically generated Select option ‘3’ to switch to admin menu

A black background with white text

Description automatically generated Input log in details

A screen shot of a computer

Description automatically generated Select option ‘1’ to view customers

A black screen with white text

Description automatically generated Select customer to view

A computer screen with white text

Description automatically generated Customer data is displayed

* Test Case 10 – Using the admin menu to accounts data

A screenshot of a computer program

Description automatically generated Select option ‘3’ to switch to admin menu

A black background with white text

Description automatically generated Input log in details

A screen shot of a computer screen

Description automatically generated Select option ‘2’ to view accounts

A screenshot of a computer screen

Description automatically generated Select account to view

A screen shot of a computer

Description automatically generated Account data is displayed

**REFLECTION**

For this assessment on the banking program for ‘Bank Maximus’ I was able to implement all the mandatory requirements stated in the assessment brief which included the following:

* Defining classes for data used withing the application.
* Providing menus to navigate for customers and admin users
* Providing options to open current accounts and savings accounts
* Providing customers the option to add interest to the accounts created.
* Providing features in the program to facilitate saving to and loading data from text files.

In addition, I implemented the optional requirements for providing a sub-menu for the bank admin and functionality for opening a mortgage account where a customer’s current account is used as the repayment account.

I also put in informative comments within the program to enable ease in understanding the code and also made use of exceptions to cater for possible errors that may rise from inputs.

With respect to extra features, I did the below:

* Added redundancy functionality to check for admin profile setup and create the default login to allow access on the subsequent login attempt with the default log in
* Included overdraft feature for current accounts, hence withdrawals that exceed the account balance can still proceed if it is within the available overdraft

For usage purpose, here are default the login details for the admin section:

* Username - admin
* Password - admin123