Queen Mary University of London

ECS505U Software Engineering Coursework I

03/11/2025 v2.3

This coursework makes up 18% of the total marks for the module. Answer all the questions to achieve a maximum score of 100 (Q1 has 35 marks and Q2 has 65 marks).

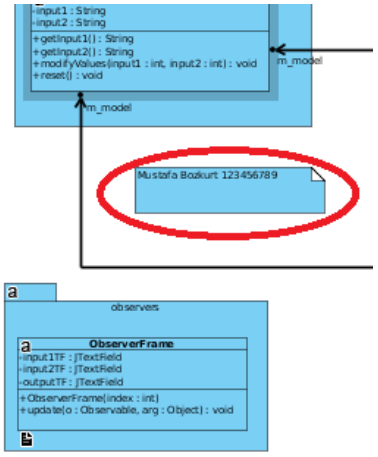
Submit your solution using the submission link on QMplus course webpage by the submission deadline. You should be able to see the details of the coursework submission on QMplus course page under assessed coursework section.

Please submit one PDF file for all questions with one diagram on each page. It is your responsibility to make sure the images are readable. We will not mark diagrams that are not readable. You will be penalized harshly if you do not submit your coursework with the required format: a two-page pdf file with each diagram in a separate page.

This coursework should take you about 5-6 hours to complete. **You are expected to complete it without collaborating with other students.**

It is recommended that you use Visual Paradigm UML tool to create diagrams as marks will be deducted if the tool you choose uses a non-standard UML notation.

Please add a note with your name and student ID on every diagram as depicted below to avoid any marking issues as we mark each diagram separately.



Overview

You are given the task of designing a software that handles day to day activities of a financial institution. Beware, the following requirements are a subset of an actual system. The requirements in this document are selected with the aim of enabling you to demonstrate your system design ability using UML. Thus, some of the requirements were omitted. However, you are not expected to improve the requirements of the software. Please, do not make assumptions and follow only the instructions.

If you have any questions regarding requirements, please ask it through the QMplus forum. I might not respond to questions via email.

Glossary:

The system: The software system designed to help staff with their day-to-day activities.
System User: A person who is using the system. Includes all types of users described below.
Staff: Employees who deals with day-to-day activities of the institution.
Customer: A person or a business who holds an account with the institution.
Personal customer (or person): A customer with a personal account.
Business customer (or business): A customer with a business account.

Question 1 (35 marks)

Draw a use case diagram following the set of requirements below.

1. Staff must be able to find customer accounts by the following:
    a. Customer name
    b. Account number
2. Staff must be able to order a new card for an existing account.
3. Staff must be able to cancel an existing card.
4. Staff must be able to print statement of an existing account.
5. Staff must be able to withdraw money from a customer account.
6. Staff must be able to deposit money to a customer account.
7. Staff must be able to open new accounts.
8. All customers must be able to deposit, withdraw and transfer money from their account.
9. Persons must be able to apply for personal loan.
10. Businesses must be able to apply for business loan.

HINTS:

- Staff must first search for an account before performing any activity for an existing account. Finding the customer account is required for **all existing account related activities** such as cancelling a card or printing a statement.

You are expected to use one of the names below for each use case and actor in your use case diagram.
Staff, Customer, Business, Person, Order New Card, Cancel Card, Withdraw from Customer Account, Deposit to Customer Account, Print Statement, Find Account, Find by Customer Name, Find by Account Number, Open New Account, Deposit, Withdraw, Transfer Money, Apply for Personal Loan, Apply for Business Loan

Question 2 (65 marks)

Draw a class diagram using the following set of requirements.

11. System must keep first and last name of each user.
12. Each user is identified by a unique username.
13. There are two types of users: staff and customer.
14. There are two types of accounts: personal and business.
15. There are two types of personal accounts: checking and savings.
16. Each account is identified by a unique account number.
17. The system must be able to track the balance of each account.
18. All accounts can be deposited and withdrawn money.
19. Each account belongs to a branch (the branch where the account was opened).
20. Business and checking accounts have an overdraft allowance.
21. All accounts are kept in the registry.
22. Bank cards are identified by a unique number.
23. Bank cards have the name of the owner (might be different than the account holder name).
24. The system must allow setting limits on bank cards.
25. All bank cards are kept in the registry.
26. There are two types of bank cards: personal card and business card.
27. Each bank card must be assigned to one account.
28. Checking accounts can be assigned one personal card.
29. Savings accounts cannot have assigned bank cards.
30. Business accounts can be assigned up to seven business cards.
31. The system must be able to add and remove cards to/from business and checking accounts.
32. There are two types of users in the system: staff and customer.
33. There are two types of customers: person and business.
34. All persons in the system must have a checking account.
35. A checking account is for one person.
36. A person can have one savings account.
37. A person is not required to have a savings account.
38. Savings accounts can be shared by four persons.
39. Savings accounts belong to at least one person.
40. Businesses can have more than one business account.
41. Branches have one or more staff.
42. Staff can work only in one branch any given time.

HINTS:

- **The requirements in question one** are also part of the requirements of this question.
- A savings account can have up to four personal account holders.
- Certain entities are represented as abstract classes as a part of my design choice so draw these elements as specified in your diagrams.

You are expected to use one of the names below for the class names. You can use them more than once.

Abstract Classes: User (both staff and customers are users), Customer, Account, BankCard, PersonalAccount
Classes: Registry, BusinessAccount, CheckingAccount, SavingsAccount, BusinessCard, PersonalCard, Branch, Staff, Business, Person

Place following methods into the classes/interfaces in your diagram

- +findAccountByNumber(accountNumber : String) : Account (finds accounts in registry)
- +findAccountByName(name : String) : Account (finds accounts in registry)
- -accountNumber : String
- -balance : float
- +withdraw(amount : float) : boolean
- +deposit(amount : float) : boolean
- -overdraftAllowance : float
- +getCards() : List<BusinessCard>
- +addCard(card : BusinessCard) : boolean
- +removeCard(card : BusinessCard) : void
- -overdraftAllowance : float
- +addCard(card : PersonalCard) : boolean
- +removeCard() : void
- +getCard() : PersonalCard
- -cardNumber : String
- -nameOnCard : String
- -limit : float
- +BankCard(number : String, nameOnCard : String)
- +getCardNumber() : String
- +getNameOnCard() : String
- +searchByAccountNumber(accountNumber : String) : Account
- +searchByName(name : String) : Account
- +orderNewCard(account : AccountI) : boolean
- +cancelCard(card : BankCard) : boolean
- +depositToAccount(account : Account, amount : float) : boolean
- +withdrawFromAccount(account : Account, amount : float) : boolean
- +openNewAccount() : Account
- -firstName : String
- -lastName : String
- -userName : String
- +deposit(account : Account, amount : float) : boolean
- +withdraw(account : Account, amount : float) : boolean
- +transferMoney(accountNo : String, amount : float) : boolean
- +applyForPersonalLoan() : boolean
- +applyForBusinesslLoan() : boolean

END OF CW INSTRUCTIONS