# CSE202 - Object Oriented Analysis & Design with Java

## Part A: Banking System - Requirements & Behavioural UML Modelling

Student Name: \_Lebopo Mogorosi\_ Student Number: \_cse23-126\_

### 1. Requirements Elicitation

Banking System Requirements Elicitation – Interview Questions. 18/09/2025

Hello, thank you for agreeing to this interview. I am gathering requirements for a proposed Banking System as part of my assignment. I will be asking you some questions about the services your bank offers, the challenges you face, and the qualities you expect from the new system. Please feel free to explain in as much detail as possible.

Section A: Functional Requirements

1. Account Management

• Can you describe how customers currently open and manage accounts?

• What types of accounts do you provide (Savings, Investment, Cheque)?

• What conditions or restrictions apply to each account type (e.g., minimum deposits, withdrawal rules, employment details)?

2. Transactions

• What are the most common transactions your customers perform?

• How do customers currently deposit, withdraw, or transfer funds?

• What challenges do they face when performing these transactions?

• Should every transaction be recorded? If yes, what details must be stored (amount, type, date, balance)?

3. BankCard Services

• Do you provide BankCards to customers?

• Should one BankCard be able to access all of a customer’s accounts, or only one account?

4. Interest Management

• How is interest currently applied to Savings and Investment accounts?

• Should interest calculation and crediting be automatic in the new system?

• Are there any exceptions or special rules for interest payments?

Section B: Non-Functional Requirements

1. Security

• What level of security should the system enforce for login and transactions?

• What security features should be included for BankCards (e.g., PIN, OTP, card blocking, alerts)?

2. Performance

• What is an acceptable transaction processing time (e.g., under 2 seconds)?

• Are there specific services that must always be fast?

3. Usability

• Who will be the main users of the system (bank staff, customers, both)?

4. Reliability & Availability

• What is the minimum acceptable system uptime (e.g., 99%)?

• How should the system behave in case of errors or downtime?

5. Scalability

• Should the system be designed to handle more customers and accounts in the future?

• Do you foresee integration with other systems (e.g., mobile banking, payment gateways)?

Thank you very much for your input. I will use your responses to document the requirements and prepare UML models of the system.

#### Functional Requirements

1. Customer Registration & Authentication:  
- Register new customers with personal details.  
- Authenticate customers during login.  
  
2. Account Management:  
- Support Savings, Investment, and Cheque accounts.  
- Savings: deposits only, 0.05% monthly interest, no withdrawals.  
- Investment: deposits and withdrawals, 5% monthly interest, minimum opening BWP 500.  
- Cheque: deposits and withdrawals, requires employment details.  
  
3. Bank Card Management (Optional):  
- Customers may request BankCards linked to their accounts.  
- BankCard allows ATM withdrawals, POS/auto transactions, and balance inquiries.  
  
4. Transactions:  
- Deposits, withdrawals (where allowed), transfers, and transaction history.  
  
5. Interest Management:  
- Automatic interest calculation and application for applicable accounts.

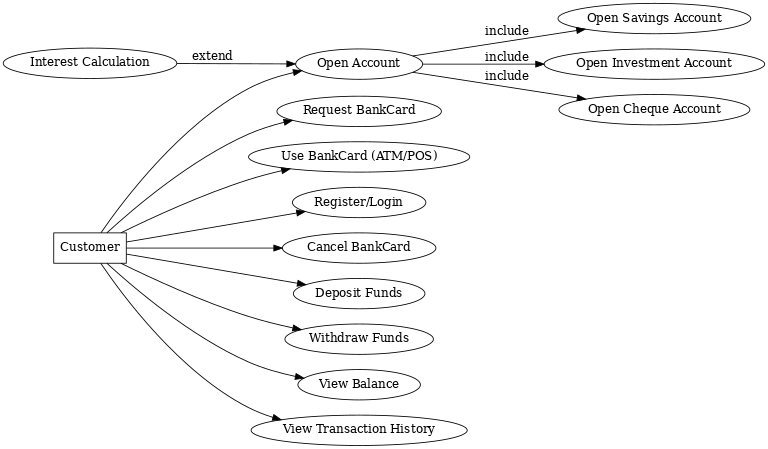
#### Non-Functional Requirements

- Security: PIN protection for BankCards and encryption for data.  
- Performance: Transactions processed promptly (target < 2s).  
- Usability: Intuitive UI and clear feedback.  
- Reliability: High uptime and robust error handling.  
- Scalability: Support growth in users and accounts.

### Structural UML Modelling

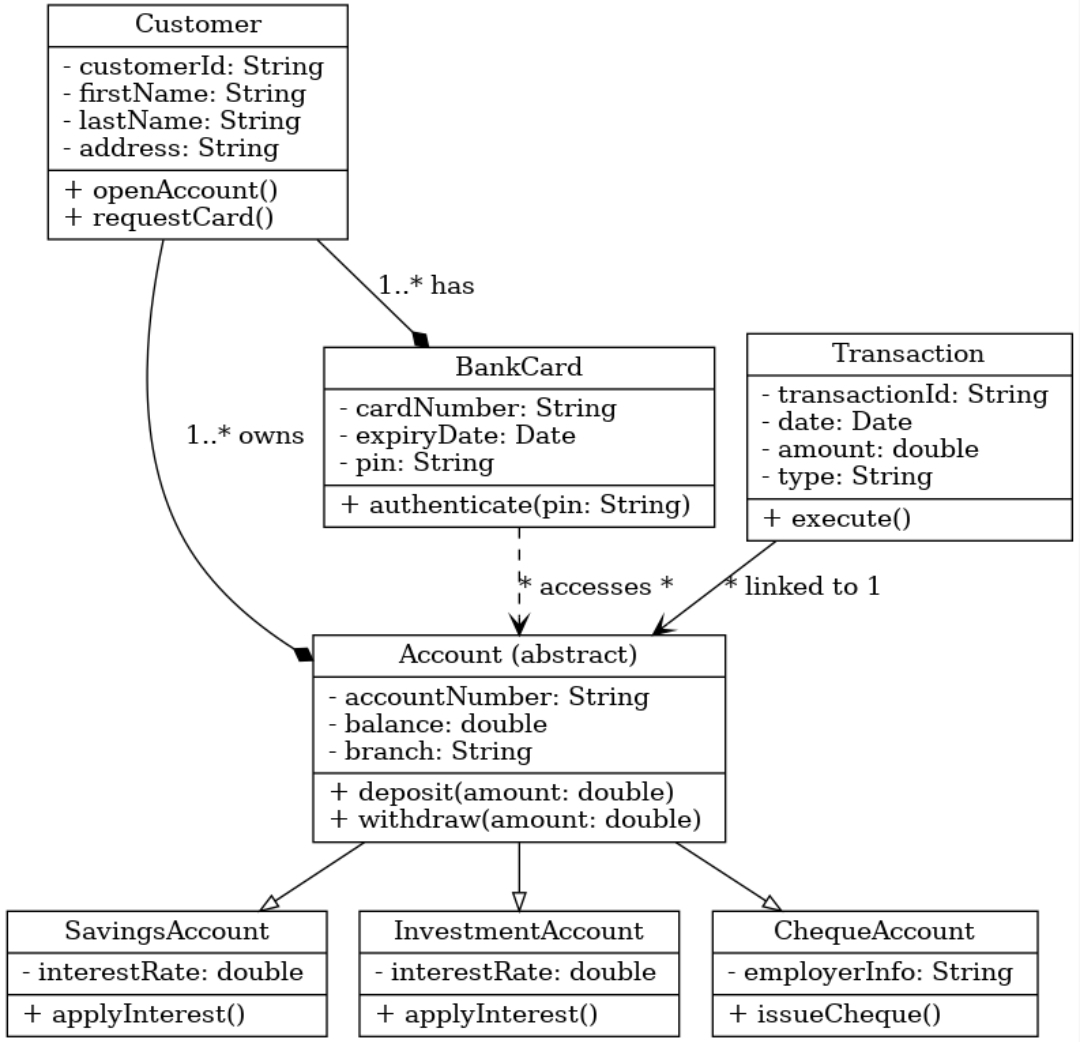
#### Use Case Diagram

Explanation: The Use Case Diagram shows external actors and the main system use cases. Customers can register/login, open accounts, request/use/cancel BankCards, deposit and withdraw funds, and view balances and history. Interest calculation extends the Open Account use case for accounts that accrue interest.



#### Class Diagram

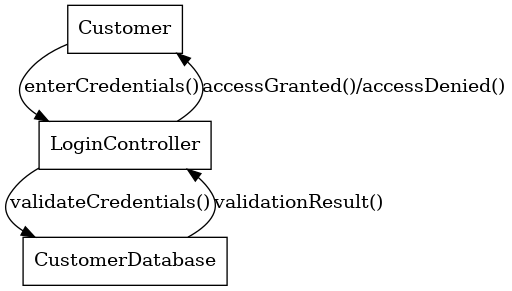
Explanation: The Class Diagram shows the core domain model. Customer can have multiple accounts and bank cards. Account is abstract with concrete subclasses (SavingsAccount, InvestmentAccount, ChequeAccount). Transactions are linked to accounts. BankCard belongs to a customer and can access one or more accounts of that customer.



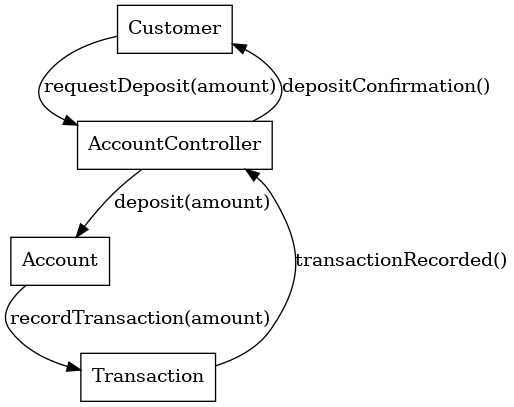
### Behavioural UML Modelling

#### Sequence Diagrams

Sequence Diagram 1: Login - shows interaction between Customer, LoginController, and CustomerDatabase.



Sequence Diagram 2: Deposit Funds - shows the flow from Customer to AccountController to Account and Transaction.



#### State Diagram

State Diagram for InvestmentAccount - shows lifecycle states (Inactive, Active, Depositing, Withdrawing, Closed) and transitions triggered by events such as openAccount, deposit, withdraw, and closeAccount.

