Software Requirement Document

Project Title:

Banking Application CLI

Project Overview:

The Banking Application CLI is a command line interface application developed using Java SE 17. The application aims to provide a basic banking management system that allows users to perform various operations such as account creation, listing all accounts, viewing individual account details, deleting an account, and performing deposit and withdrawal transactions. The application is designed to be simple, user friendly, and efficient, using core Java concepts like ArrayLists, loops, conditions, and packages.

Functional Requirements:

1. Account Creation

Description: The system should allow users to create a new bank account.
Input:

- Account holder's name
- NRIC
- Initial deposit amount

Process:
Prompt the user for the account holder's name and initial deposit.
Generate a unique account number for the new account.
Add the new account to the account list.
Output: Confirmation message displaying the new account number and details.

2. Accounts Listing

Description: The system should display a list of all bank accounts.
Process:
Retrieve all accounts from the account list.
Display account numbers and the names of account holders.
Output: List of all accounts with account numbers and account holder names.

3. Individual Account Detail Display

Description: The system should display detailed information for a specific account.
Input:

- Account number

Process:
Prompt the user for an account number.
Retrieve and display the account holder's name, balance, and transaction history (if applicable).
Output: Detailed information of the specified account.

4. Deletion of an Account

Description: The system should allow users to delete an account from the system.
Input:

- Account number

Process:

Prompt the user for the account number to delete.
Verify if the account exists.
Remove the account from the account list.
Output: Confirmation message stating that the account has been deleted.

5. Deposit

Description: The system should allow users to deposit money into a specific account.
Input:

- Account number
- Deposit amount

Process:
Prompt the user for the account number and deposit amount.
Verify if the account exists.
Add the deposit amount to the account balance.
Output: Confirmation message displaying the updated account balance.

6. Withdrawal

Description: The system should allow users to withdraw money from a specific account.
Input:

- Account number
- Withdrawal amount

Process:
Prompt the user for the account number and withdrawal amount.
Verify if the account exists.
Check if the account has sufficient balance.
Deduct the withdrawal amount from the account balance.
Output: Confirmation message displaying the updated account balance.

Nonfunctional Requirements:

1. User Interface:

The application will be a Cli based interface that interacts with the user through the console.
The interface should be intuitive and easy to navigate.

2. Performance:

The application should respond to user commands promptly.
Operations like account creation, deletion, and transactions should be processed efficiently.

3. Reliability:

The application should handle invalid inputs gracefully.
It should ensure data integrity during transactions and other operations.

Java Concepts to be Used:

1. User Input Handling:

Use the `Scanner` class to capture and handle user inputs for various operations.
Validate the inputs for correctness and completeness before processing.

2. Data Storage with ArrayList:

Use `ArrayList` to store and manage a dynamic list of bank accounts.
Each account will be represented as an object and stored in the `ArrayList`.

3. Packages:

Organize the application into different packages such as `model`, `service`, and `utility` for better code organization and maintainability.

4. Control Flow:

Use loops (such as `while` or `for` loops) to present the menu repeatedly until the user chooses to exit.
Use conditional statements (`if`, `switchcase`) to control the program flow based on user choices.

5. String Formatting:

Use `String.format()` or `System.out.printf()` for displaying output in a formatted manner.
Properly format account details, balances, and other information displayed to the user.

Sample Menu:

Welcome to the Banking Application

1. Create a new account
2. List all accounts
3. View account details
4. Delete an account
5. Deposit
6. Withdraw
7. Exit

Please enter your choice:

Assumptions:

Each account has a unique account number.
Basic validation is performed for each input (e.g., negative deposits or withdrawals are not allowed).
The application does not persist data after the program terminates (data is stored in memory during the session).