Starting with importing library and class definition

import java.sql.\*; The code enables access to classes and interfaces in the Java Database Connectivity (JDBC)

import java.util.ArrayList; is imported to use dynamic arrays that can grow or shrink in size as needed. It is often used to store and manage collections of objects.

**import java.util.Scanner**; is imported to read user input from the console. It provides methods for reading different types of input, such as integers, doubles, and strings.

This code is a simplified banking management system implemented in Java. It consists of two classes:

**BankAccount** and **BankingManagementSystem**.

The **BankAccount** class represents a bank account. It has private instance variables such as accountNumber (to store the account number), password (to store the password), balance (to store the account balance), and transaction-history (to store the transaction history as an array list of strings).

The **BankAccount** class has two constructors. The first constructor takes accountNumber, password, and lastName as parameters and initializes the instance variables. The *second constructo*r is an overloaded version that takes accountNumber and password (as an integer) and initializes the instance variables accordingly.

The BankAccount class **provides various methods to interact with the bank account,** such as:

getAccountNumber() --> to retrieve the account number,

validatePassword() --> to validate the provided password,

getBalance() --> t o retrieve the account balance,

deposit() --> to deposit money into the account,

withdraw() --> to withdraw money from the account, and

showTransactionHistory() --> to display the transaction history.

The BankingManagementSystem class represents the banking management system. It has private instance variables such as connection (to establish a connection with the database), scanner (to read user input), and currentUser (to store the current user's bank account).

The BankingManagementSystem class has a **constructor** that initializes the scanner and establishes connection with the database by calling the establishDatabaseConnection() method.

The BankingManagementSystem class **provides private methods** such as establishDatabaseConnection() --> to establish a connection with the database, closeDatabaseConnection() --> to close the database connection), and

performTransaction() --> to perform a transaction, such as deposit or withdrawal.

**The establishDatabaseConnection()**

method uses the DriverManager.getConnection() method to establish a connection with the MySQL database. It uses the URL, username, and password to connect to the database. If an error occurs during the connection process, an error message is displayed.

**The closeDatabaseConnection()**

method checks if the database connection is not null and then closes the connection. If an error occurs during the closing process, an error message is displayed.

**The performTransaction()**

method is responsible for performing a transaction, such as a deposit or withdrawal. It prompts the user to enter the amount and description of the transaction. It then updates the account balance in the database by executing an SQL UPDATE statement and inserts a record of the transaction into the database by executing an SQL INSERT statement.

**User Interaction Methods:**

**showMainMenu():** Displays the main menu for user interaction and handles user choices.

**initiateTransaction(String accountNumber)**: Displays options for deposit, withdrawal, or returning to the main menu.

**isLoggedIn():** Checks if a user is logged in.

**createAccount():** Prompts users to input account details to create a new account and adds it to the database.

**login():** Handles the login process, validating account number and password.

**performTransactions(BankAccount account)**: Manages user transactions after successful login.

**Main Class:**

main(String[] args): The entry point of the application. It creates an instance of BankingManagementSystem and starts the main menu.

**Program Flow:**

Here's an overview of the program flow based on our project

1.The **program starts** in the main method of the Main class.

2.An instance of BankingManagementSystem named bankingSystem is created.

3.The showMainMenu method of the BankingManagementSystem class is called, initiating the main menu loop for user interaction.

4. The main menu is displayed, showing options for creating an account, logging in, or exiting the system.

5.The user is prompted to enter their choice.

6. Depending on the user's choice, one of the following actions is performed:

If the user chooses to create an account (option 1), the createAccount method of the BankingManagementSystem class is called. This prompts the user to input account details and creates a new account in the database.

If the user chooses to log in (option 2), the login method of the BankingManagementSystem class is called. This prompts the user to enter their account number and password, validates the credentials, and sets the currentUser attribute to the logged-in user's account.

If the user chooses to exit the system (option 3), the showMainMenu method returns, and the program execution ends.

1. If the user successfully logs in, the performTransactions method of the BankingManagementSystem class is called.
2. This method manages user transactions after a successful login.

The performTransactions method allows the user to initiate transactions such as deposit or withdrawal by calling the initiateTransaction method. The user is presented with a transaction menu and prompted to choose an action.

Depending on the user's choice in the transaction menu, one of the following actions is performed:

If the user chooses to deposit (option 1), the program performs the deposit operation for the logged-in user's account.

If the user chooses to withdraw (option 2), the program performs the withdrawal operation for the logged-in user's account.

If the user chooses to return to the main menu (option 3), the control is returned to the main menu loop.

The program continues to loop through the main menu until the user chooses to exit the system.