ARCHITECTURE DESIGN

ATM Interface in Java

(Console Based Application)

The architecture of the ATM System involves several key classes and components that work together to facilitate banking operations, user interaction. The system follows a modular and layered architecture to ensure separation of concerns, maintainability, and scalability.

1. Presentation Layer:

Main Class: ATMInterface.java

This class serves as the entry point of the application and handles user interaction through the console.

It presents the Main Menu to users and directs control flow based on user input.

2. Application Layer:

Bank Class: Bank.java

Manages the bank's operations such as adding account holders, validating credentials, and facilitating fund transfers.

Provides methods for account management and transaction processing.

3. Business Logic Layer:

AccountHolder Class: AccountHolder.java

Represents an account holder with attributes such as User ID, PIN, balance, and transaction history.

Implements methods for depositing money, withdrawing money, adding transactions, and managing account details.

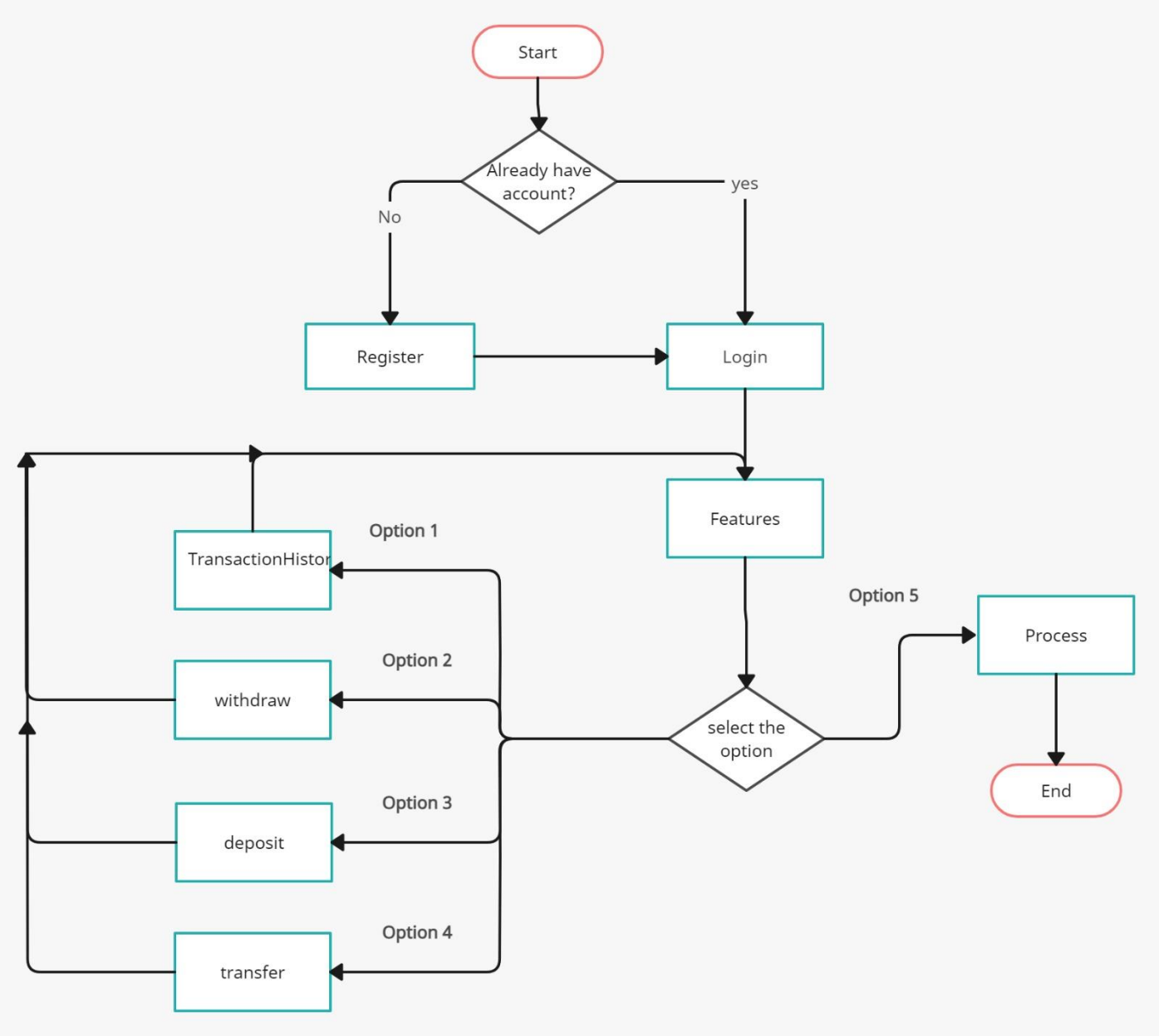
Transaction Class: Transaction.java

Represents a transaction with attributes such as amount, description, and timestamp.

Used for recording transaction details in the account holder's transaction history.

4. Control Flow:

Diagram:



Main Menu (ATMInterface.java):

Presents options for users to choose from (Show Transactions History, Withdraw Money, Deposit Money, Transfer Funds, Quit).

Based on user input, triggers corresponding methods in the Application Layer and Business Logic Layer.

Show Transactions History:

Calls displayTransactionHistory() method in ATMInterface.java.

This method fetches and displays the transaction history for the current account holder.

Control then returns to the Main Menu.

Withdraw Money:

Triggers withdraw() method in AccountHolder.java through withDraw() method in ATMInterface.java.

Validates withdrawal amount and updates account balance if successful.

Control returns to the Main Menu.

Deposit Money:

Triggers depositMoney() method in AccountHolder.java through depositMoney() method in ATMInterface.java.

Adds deposited amount to account balance.

Control returns to the Main Menu.

Transfer Funds:

Triggers transferFunds() method in Bank.java through transferFunds() method in ATMInterface.java.

Validates sender and recipient accounts, performs fund transfer, and updates transaction history.

Control returns to the Main Menu.

Quit:

Exits the program and stops the control flow.

Architecture Design Considerations:

Modularity: The system is designed with modular components (classes) that handle specific functionalities, promoting code reusability and ease of maintenance.

Layered Approach: The architecture follows a layered approach with clear separation of Presentation Layer, Application Layer, and Business Logic Layer, ensuring a structured and organized codebase.

Encapsulation: Classes encapsulate related data and behaviors, allowing for data hiding and abstraction of implementation details.

Controlled Flow: The control flow is managed through method calls and user inputs, directing the program's execution based on user choices while maintaining a structured flow of operations.

Scalability: The modular and layered architecture allows for easy scalability by adding new features, functionalities, or classes without affecting existing components.

In summary, the architecture design of the ATM System emphasizes modularity, layering, encapsulation, and controlled flow to create a well-structured, maintainable, and scalable application for managing banking operations effectively.