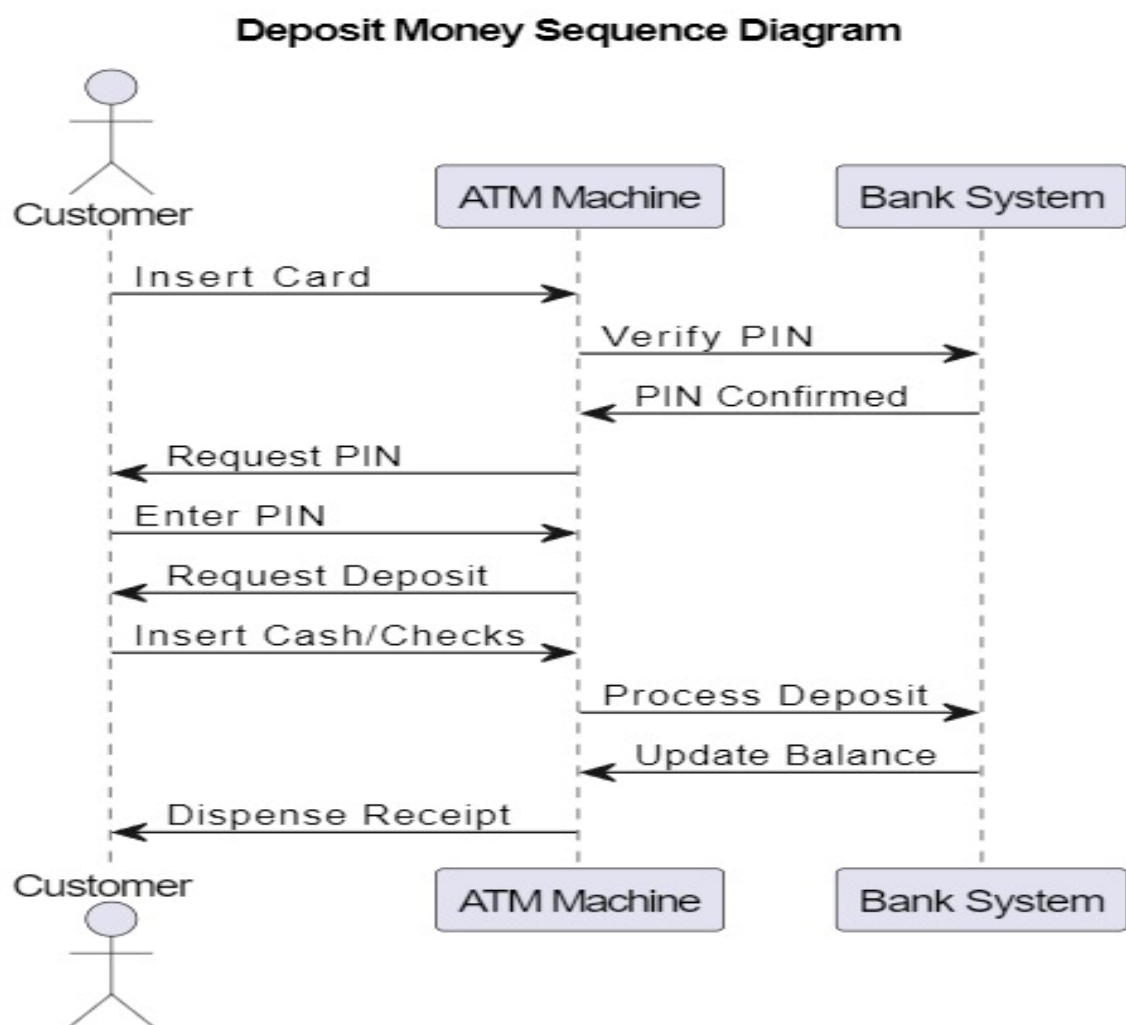


Project Component 3 – Design and UML Sequence Diagrams

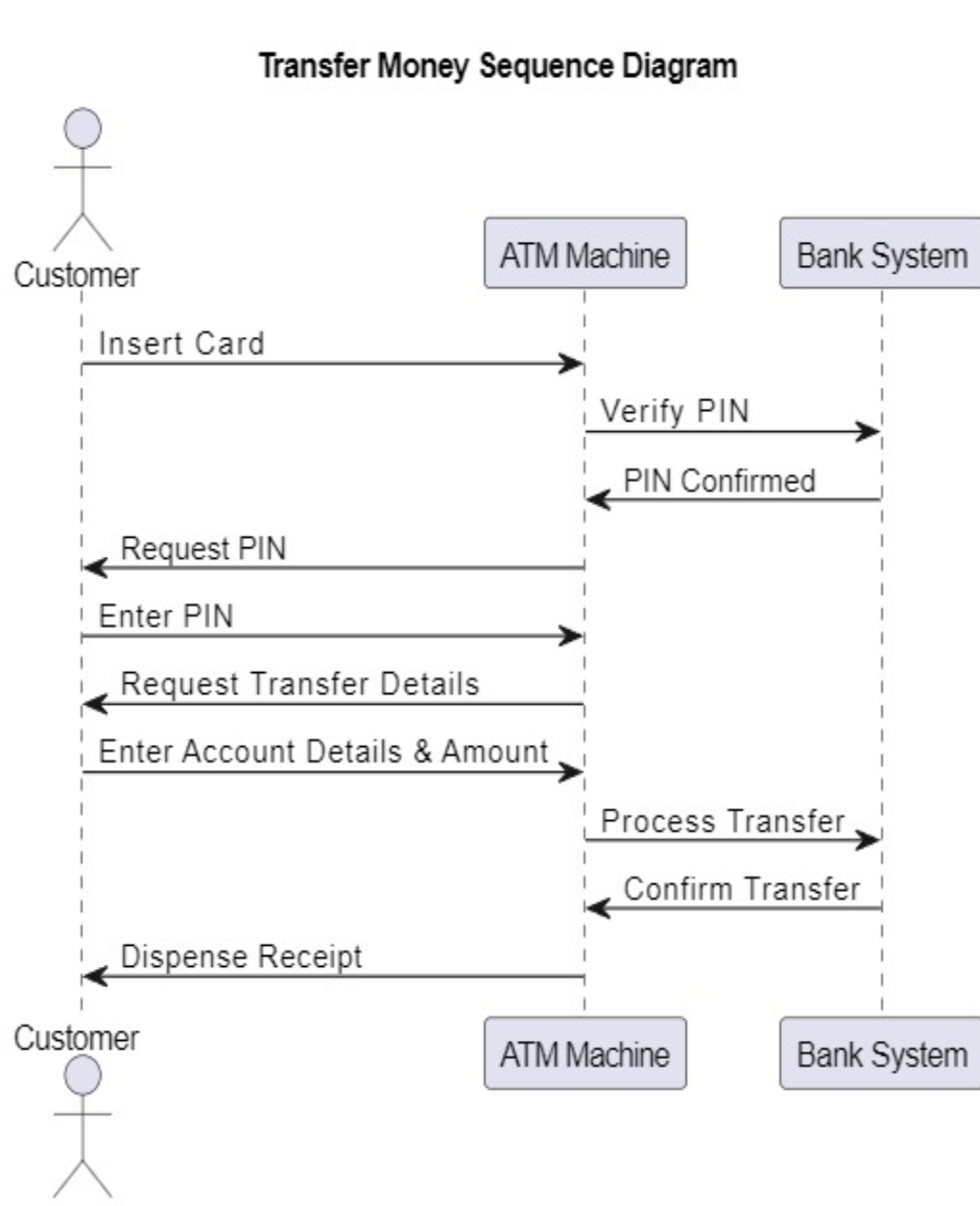
You may complete this in a group of size up to 3.

1. Using the high-level use cases that you created for Project Component 2, provide UML sequence diagrams for the most important used use cases. You may do these by hand or using an appropriate UML software diagramming tool.

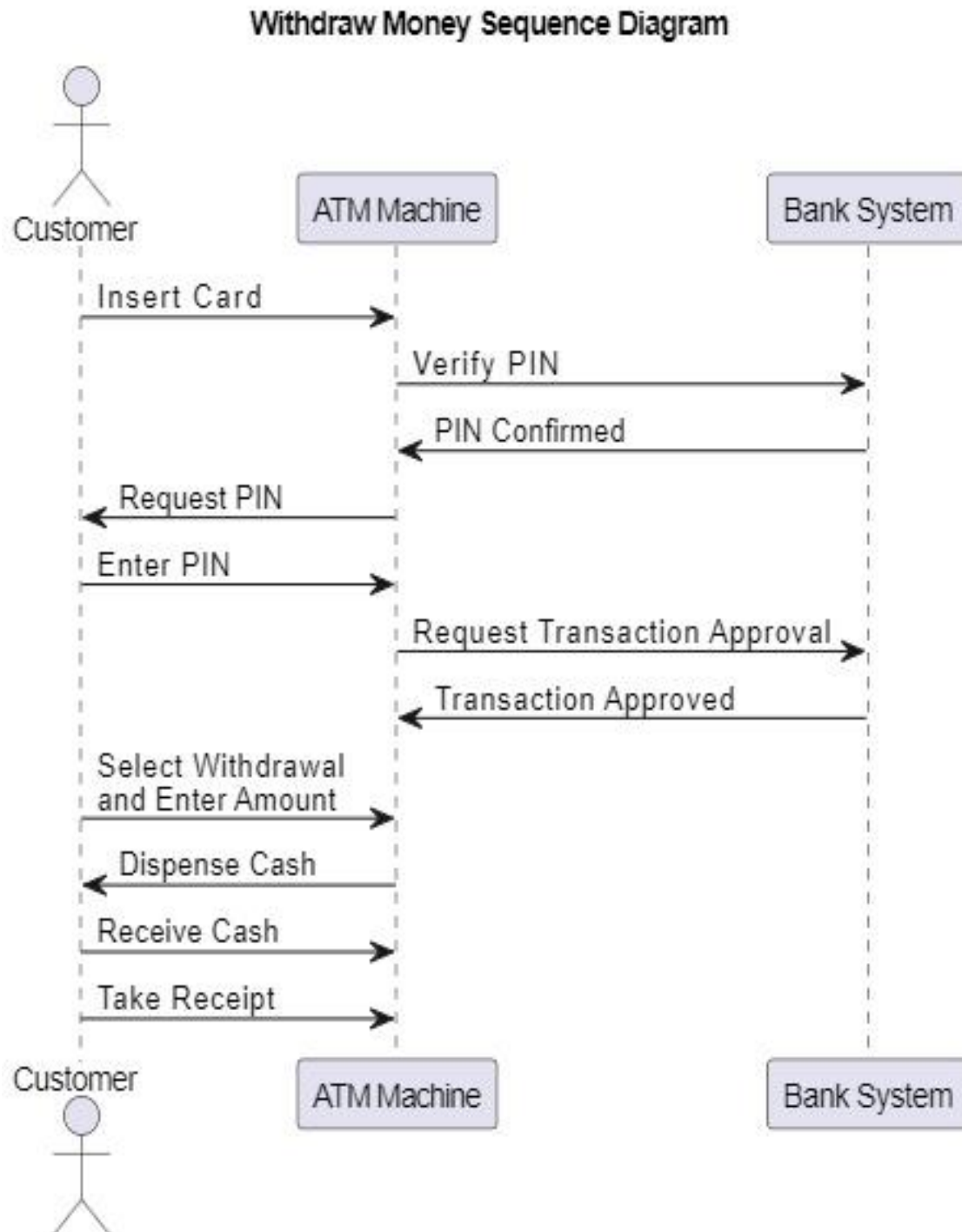
➔ Deposit Money Sequence Diagram



Transfer Money Sequence Diagram



Withdraw Money Sequence Diagram



2. Create a draft of the “testing plan” for your project (see Figure 25.8). It is recommended that you create a plan that includes testing of only the main components and any post-unit testing of your project at this point; if you can create a more detailed plan, so much the better.



1. What will we Test

We aim to ensure the ATM system handles:

- ➔ **Basic Transactions:** Withdrawals, deposits, and transfers.
- ➔ **Account Access:** Secure login functionality.
- ➔ **Account Management:** Checking balances and updating personal information.

2. Main Parts to test

- ➔ **ATM Interface:** Tests all user interactions including transactions, login, balance inquiries, and personal info updates.
- ➔ **Bank System:** Manages all backend processes like transaction processing, user authentication, balance updates, and personal info changes.

3.

➔ **ATM Interface:**

- **Transaction Tests:** Validate correct processing for withdrawals, deposits, and transfers.
- **Login Test:** Check for secure and accurate account access.
- **Balance Check Test:** Ensure accurate display of current account balance.

- **Update Info Test:** Verify that updates to personal details are handled correctly.

➔ **Bank System:**

- **Transaction Processing:** Ensure transactions are processed accurately and securely.
- **Login Verification:** Authenticate login details properly.
- **Balance Display:** Confirm that the balance updates correctly post-transaction.
- **Personal Info Processing:** Accurately process updates to personal information securely.

4. How We Test

Unit Testing: Individual testing of each functionality using automated tools.

Integration Testing: Combine and test functionalities to ensure they work well together.