Bank Automated Teller Machines

Software Engineering (CS385T) Project

ريم علي الغامدي
437004875

سارة خالد آل حسين
436006939

شهد الكناني
437004165

عبير عزت
436200058

لمياء القحطاني
437004164

Table 1: Member Roles

Name	ID	Responsibility
ريم علي الغامدي	437004875	Transfer
سارة خالد آل حسين	436006939	Withdraw
شهد الكناني	437004165	validate
عبير عزت	436200058	Deposit
لمياء القحطاني	437004164	Retrieve

Contents

1 Introduction

The ATM (Automated Teller Machine) is a machine that helps clients access their money thats in their bank accounts and do transactions on it. It does so when the client inserts a plastic ATM card with a verification PIN into this machine and the machine just retrieves the clients account information from the bank itself. The clients can do multiple transactions on the money they own in the bank including allowing them to specifically to see the amount of money they have, add their cash into their accounts, transfer money to some other account, and take some cash out, all with the use of that card. The system can print a receipt at the end of each transaction if the user wishes[1].

1.1 Purpose

1. Making transactions easier for clients
2. Making transactions more accessible to clients, since the ATMs are at remote locations closer to the client
3. Saving time so that clients dont have to have actual interactions with bank staff
4. Reducing the number of staff that have to be at a branch (clients can do the transactions themselves)
5. Allowing the clients to carry around their money with them in a safer way, without it being in cash form

1.2 Scope

The area that the system is concerned with is financial; it is monetary transactions that banks will allow clients to carry out on their own. This system shall help with the following transaction: deposit, withdraw, retrieving info and transfer. This system is not concerned with changing PIN, making new account or paying bills.

1.3 Generic Software Model

The model we are going to use to develop the ATM system is the waterfall model. It is the most suitable one because the requirements of the system were very clear from the beginning (allowing remote monetary transactions that included balance inquiry, money deposit, money transfer, and money withdrawal) and are not going to change or increase with time, and because we can finish from one phase and start the other without them overlapping them; i.e. we can finish collecting and verifying all the required functionalities, then designing each of them separately and finishing from all their design process, then implementing their designs as codes, integrating them, and finally deploying them [2].

2 Requirement Specifications

2.1 Functional Requirements

1. The ATM shall validate the user.
2. The user shall be able to retrieve account information.
3. The user shall be able to withdraw money.
4. The user shall be able to deposit money.
5. The user shall be able to transfer money to another account.

2.2 Non-functional Requirements

1. The system must be secure.
2. The system must be available 24H.

2.3 Functional Requirements Description

ID	Requirement	Description
1	The ATM shall validate the user.	ATM should ask for the user card and PIN code, then send these information to the bank to validate the user, if PIN is correct bank sends the user account to the ATM
2	The user shall be able to retrieve account information.	After the user is validated, the user should be able to get account information. Including current balance, debt, number of cards activated, and the last transactions.
3	The user shall be able to withdraw money.	After the user is validated, the user should be able to get withdraw money from account if withdrawn amount is less than the balance.
4	The user shall be able to deposit money.	After the user is validated, the user should be able to deposit money into account if money is validated to be real and undamaged.
5	The user shall be able to transfer money to another account.	After the user is validated, the user should be able to transfer money from account to any other account as long as IBAN entered is correct and amount transferred is less than the balance.

2.4 Non-functional Requirements Description

ID	Requirement	Description
1	The system must be secure.	The system must be impossible to hack, connection must be secure and data must be encrypted and layered.
2	The system must be available 24H.	The system must never fail or else the user or the bank might lose money.

3 Design

3.1 Validate User

3.2 Retrieve Account Information

3.2.1 Use Case Scenario

Use Case Name: Retrieve info.

Goal: to check account balance.

Actors: customer, ATM, Bank

Precondition: user is validated

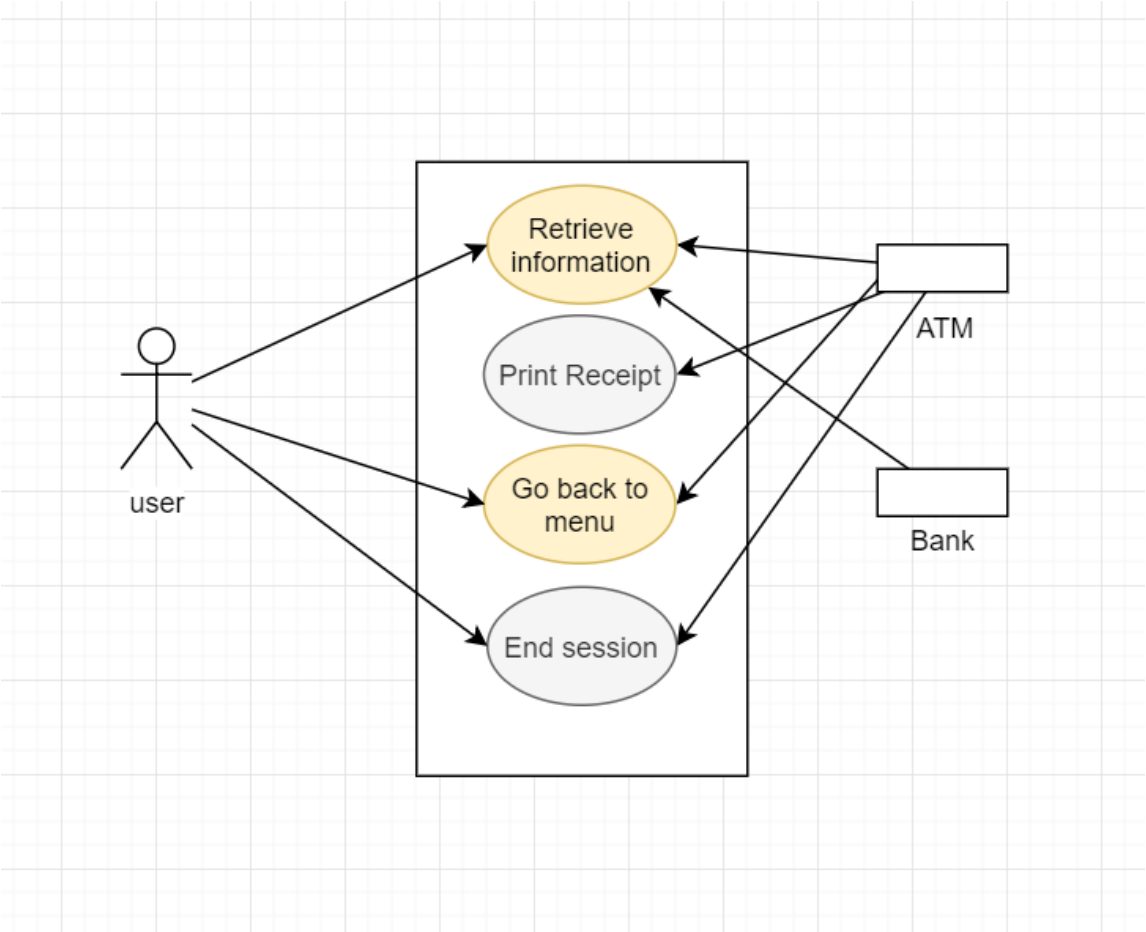
Primary Scenario:

1. ATM shows menu.
2. user chooses Balance Inquiry .
3. ATM prints receipt.
4. ATM go back to menu.
5. User quit.
6. ATM return the card to user.

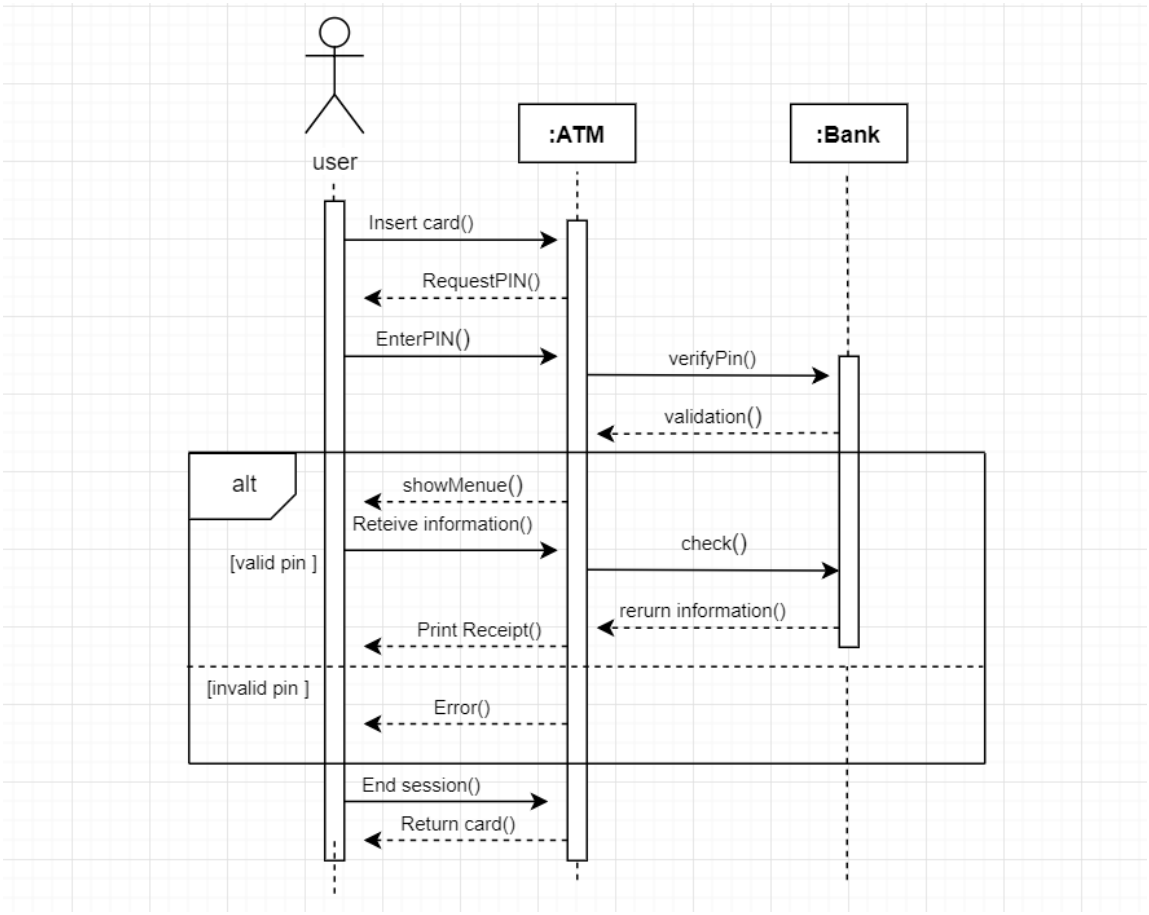
Variant:

*. user may cancel the session ;ATM return card.

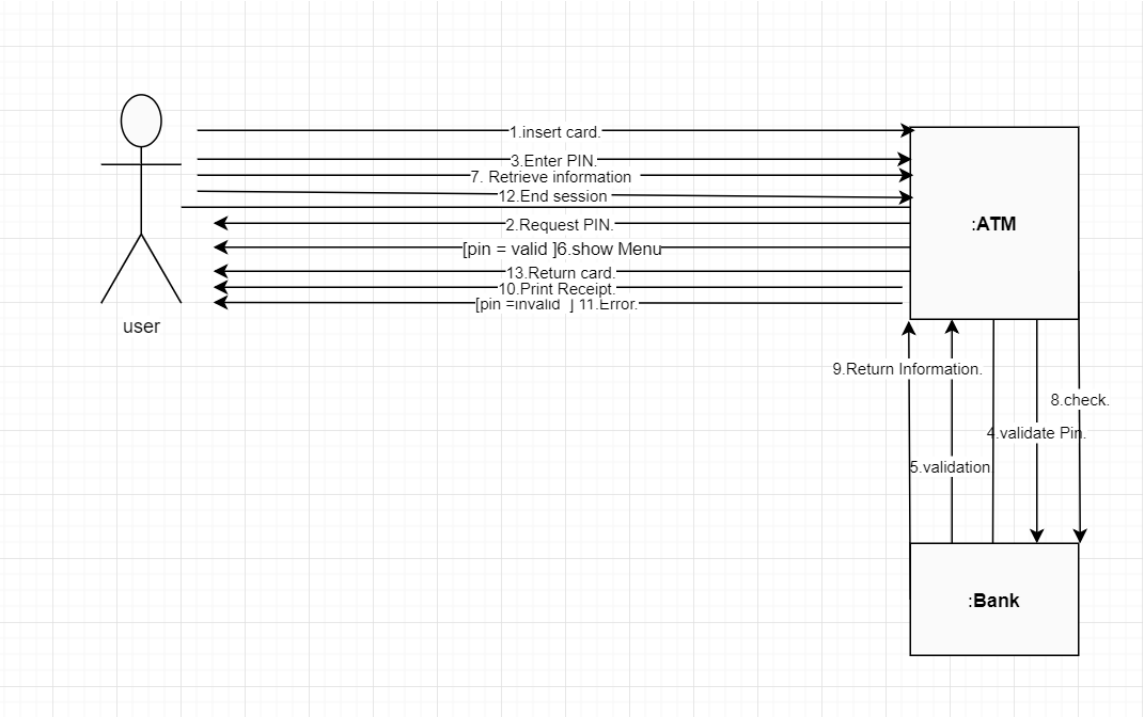
3.2.2 Use Case Diagram



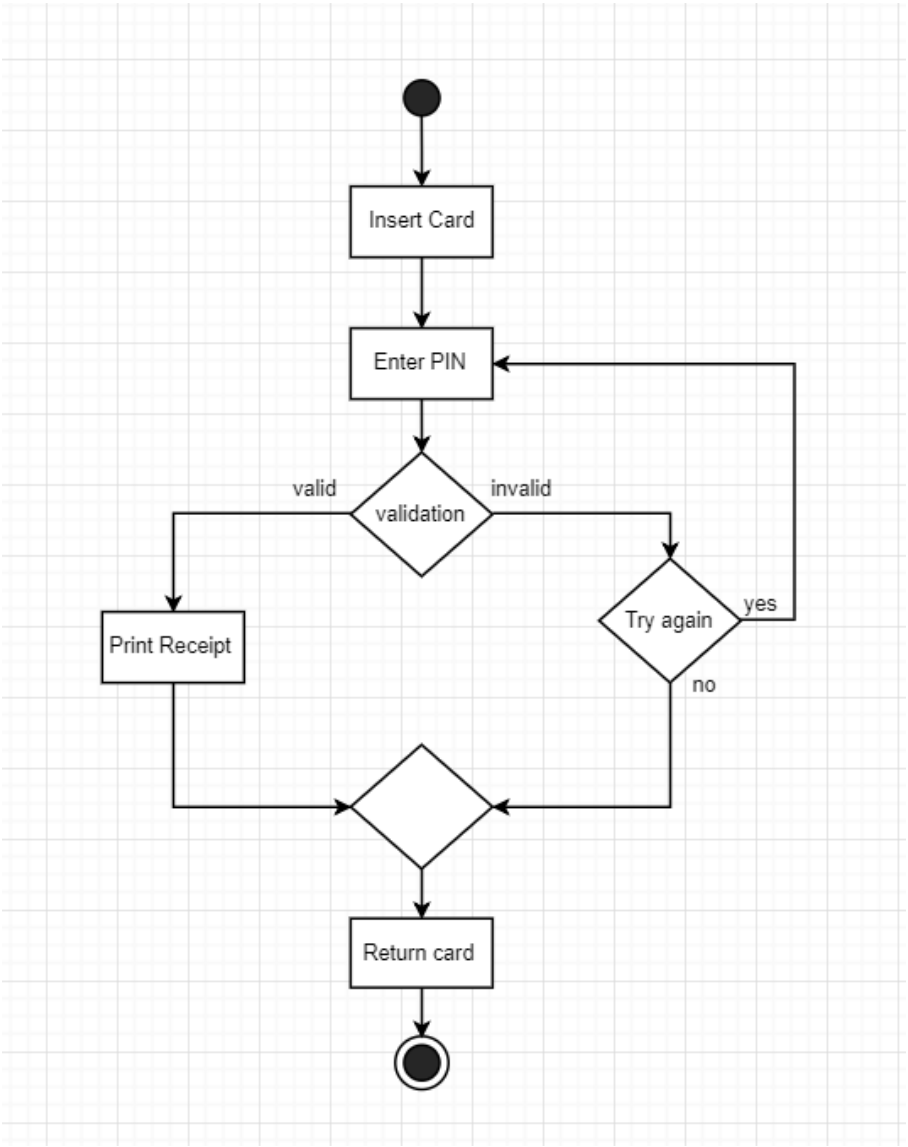
3.2.3 Sequence Diagrams



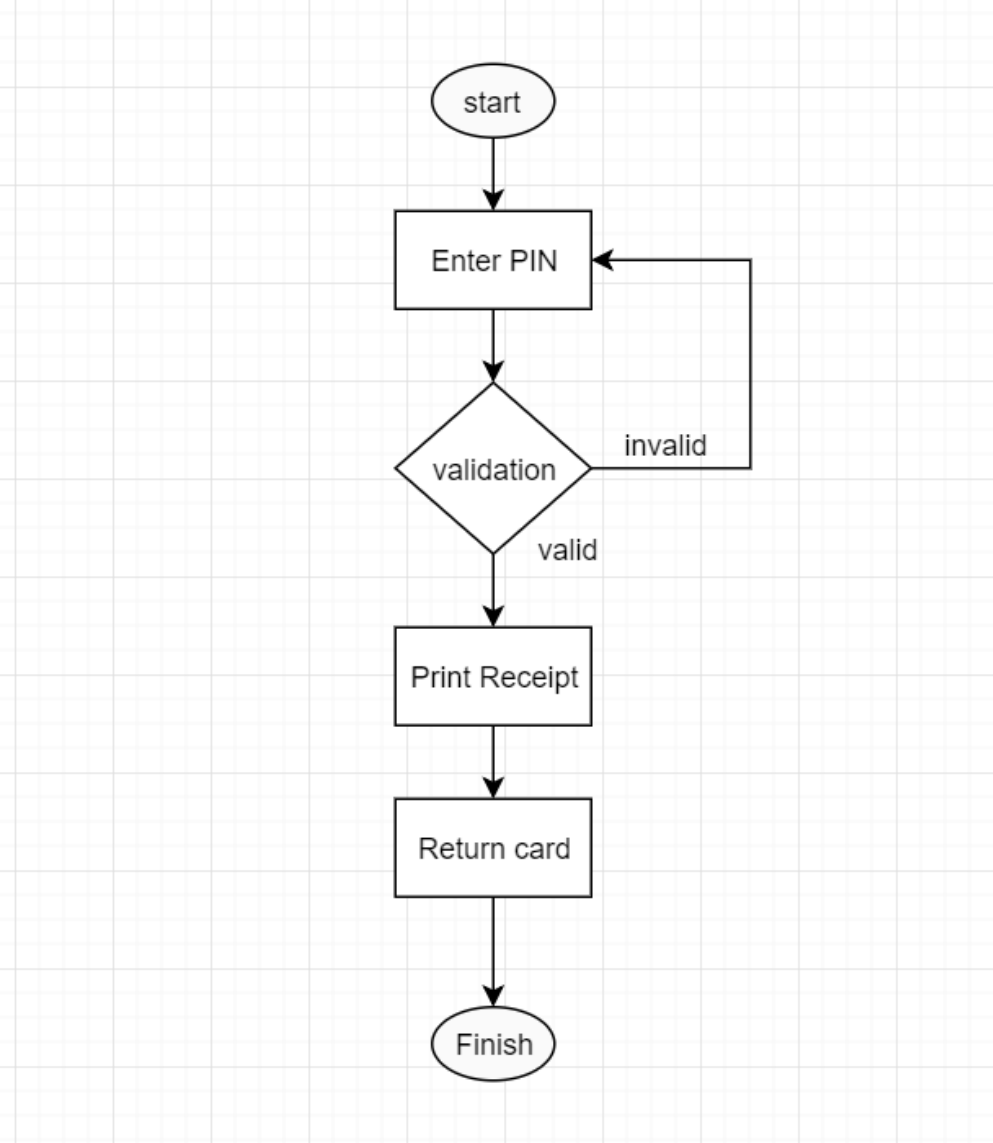
3.2.4 Collaboration Diagram



3.2.5 Activity Diagram



3.2.6 Flow Chart



3.3 Withdraw Money

3.4 Deposit Money

3.4.1 Use Case Scenario

Use Case Name: Deposit money into account.

Goal: To deposit money (cash notes) into account.

Actors: user, ATM, Bank

Precondition: user is validated

Primary Scenario:

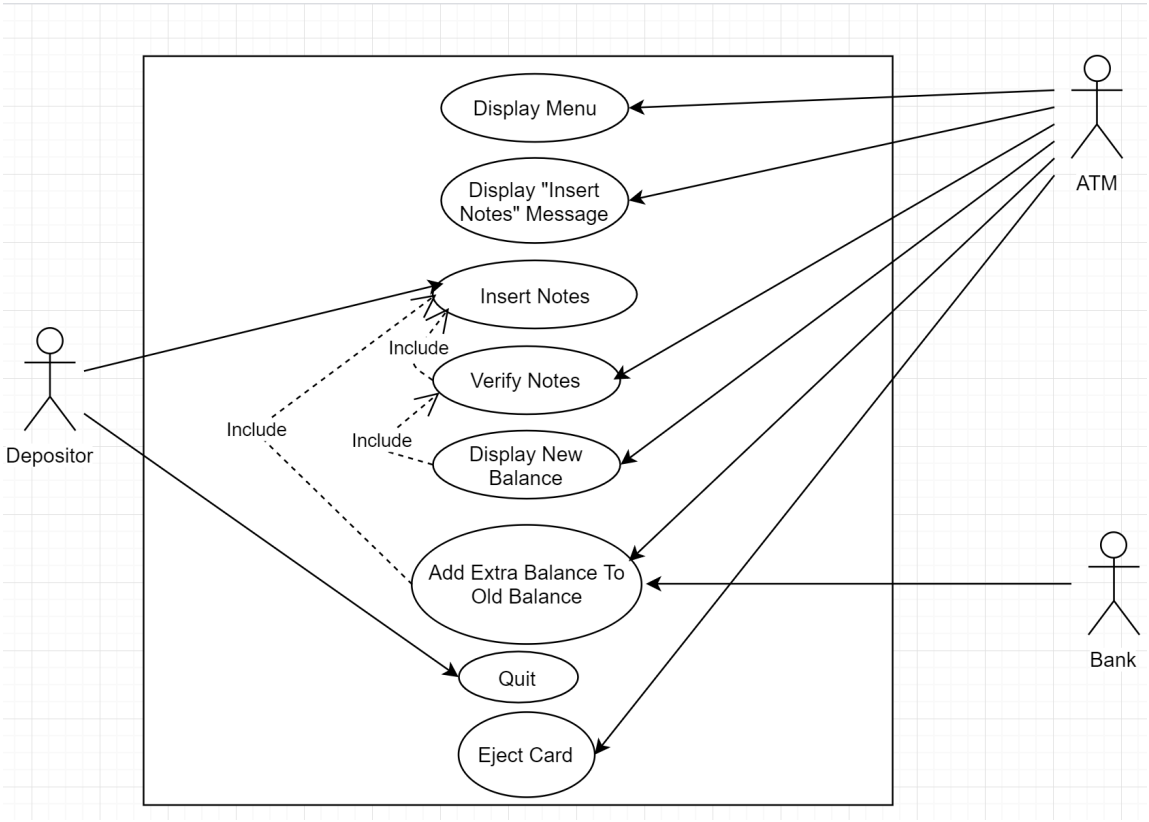
1. ATM screen displays menu.
2. The user presses on Deposit.
3. The machine displays a message to insert the notes into the allocated slot.
4. The user inserts the notes.
5. The machine verifies the notes.
6. The system displays the new balance value and sends it to the bank.
7. The bank adds the extra balance value to the old balance.
8. The user quits the session.
9. The ATM ejects the users card.

Variant:

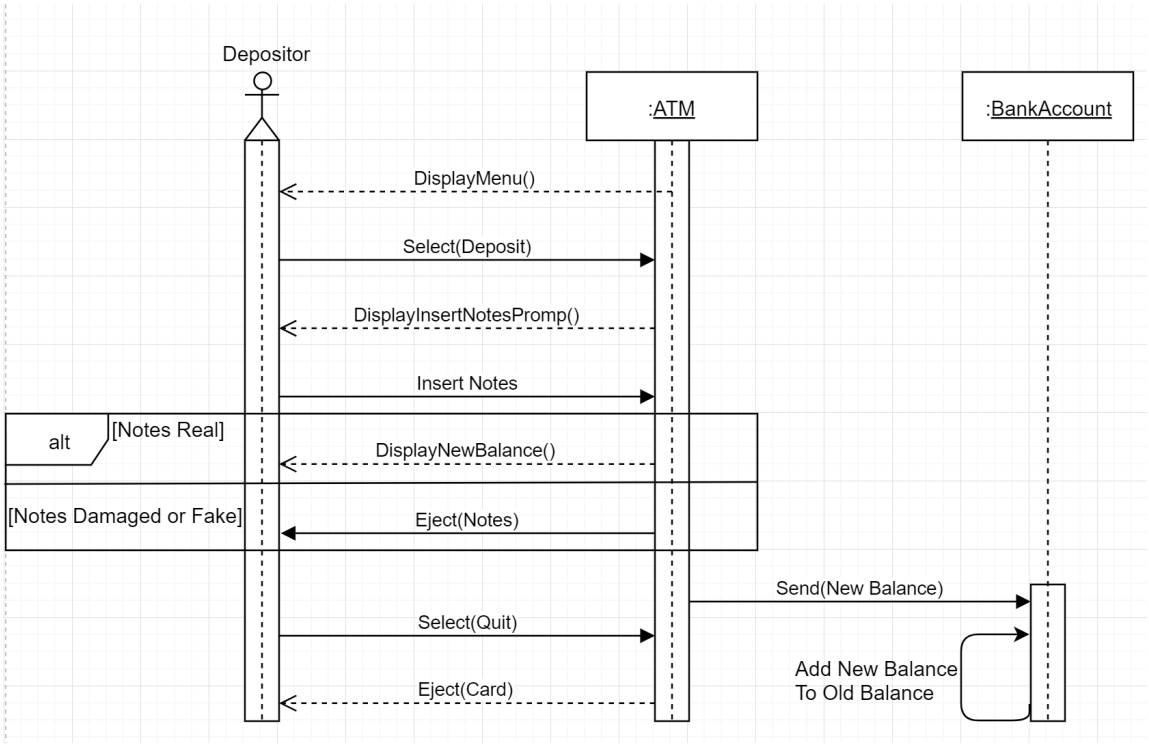
5a. The notes are not verified or accepted; the ATM ejects them.

*. The user might quit during any logical step; the ATM will eject the card.

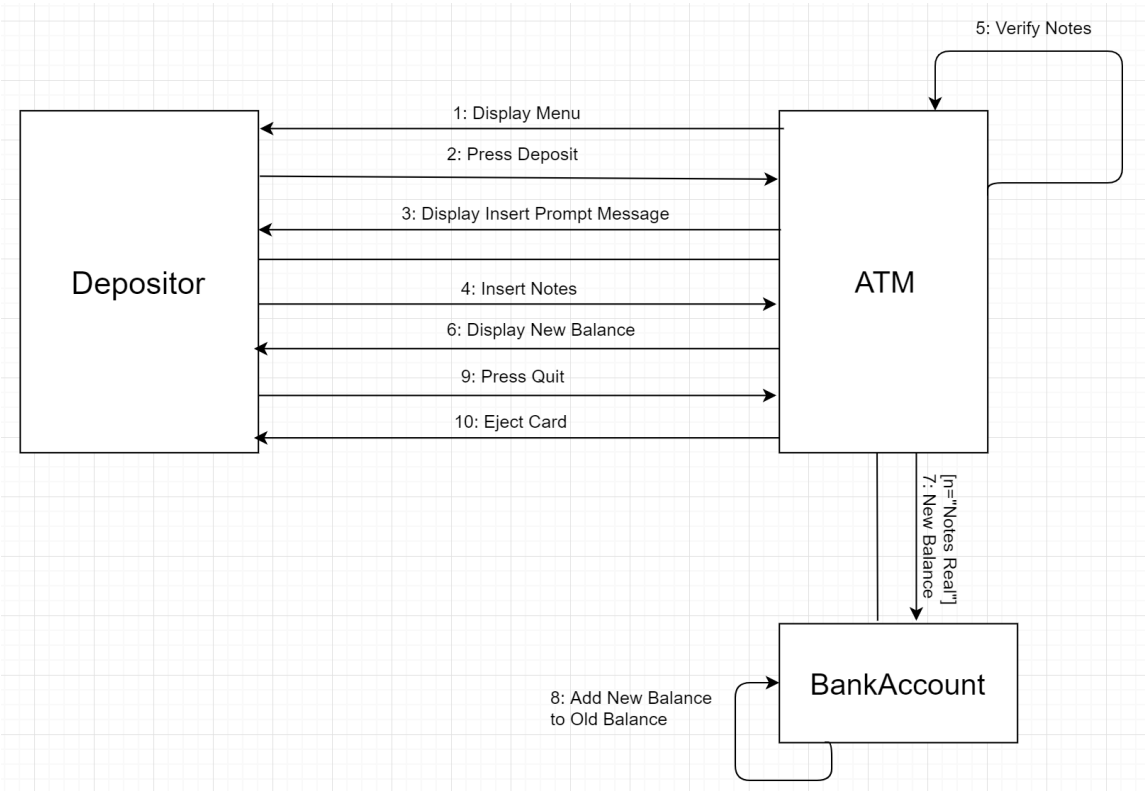
3.4.2 Use Case Diagram



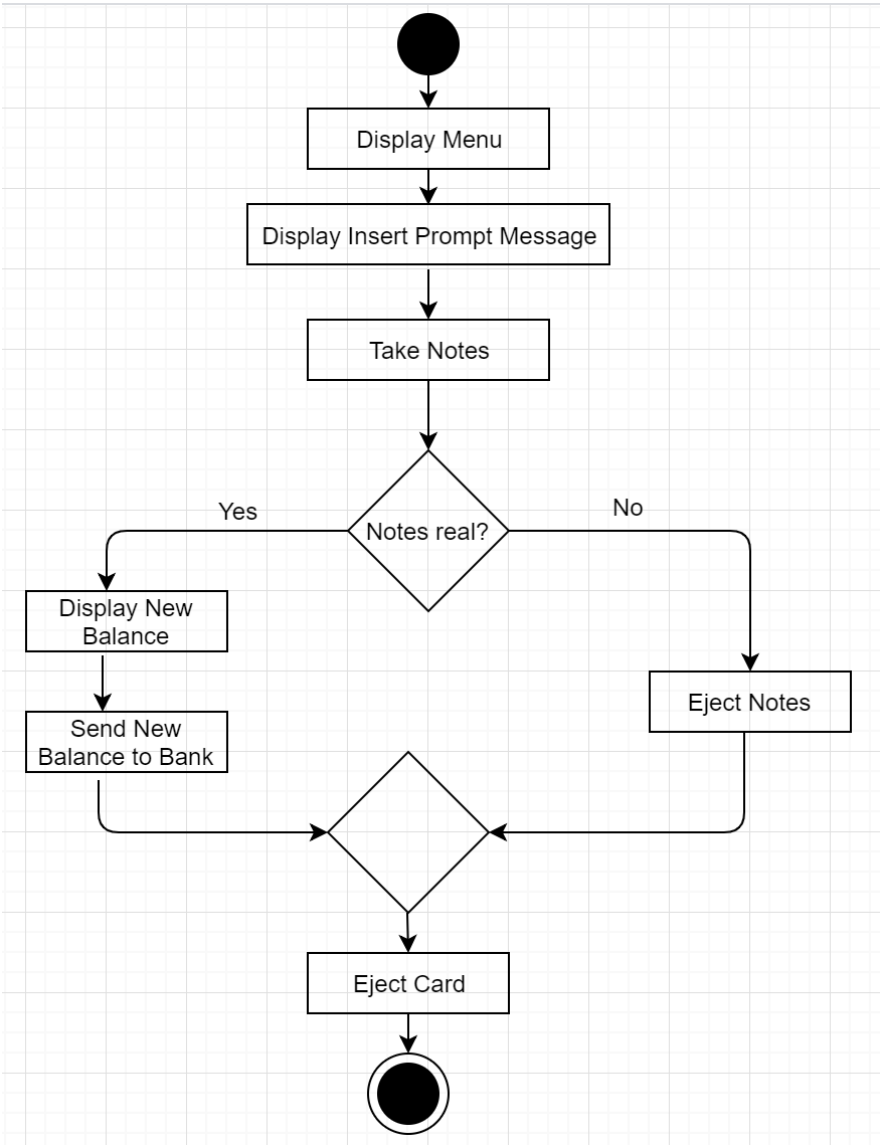
3.4.3 Sequence Diagrams



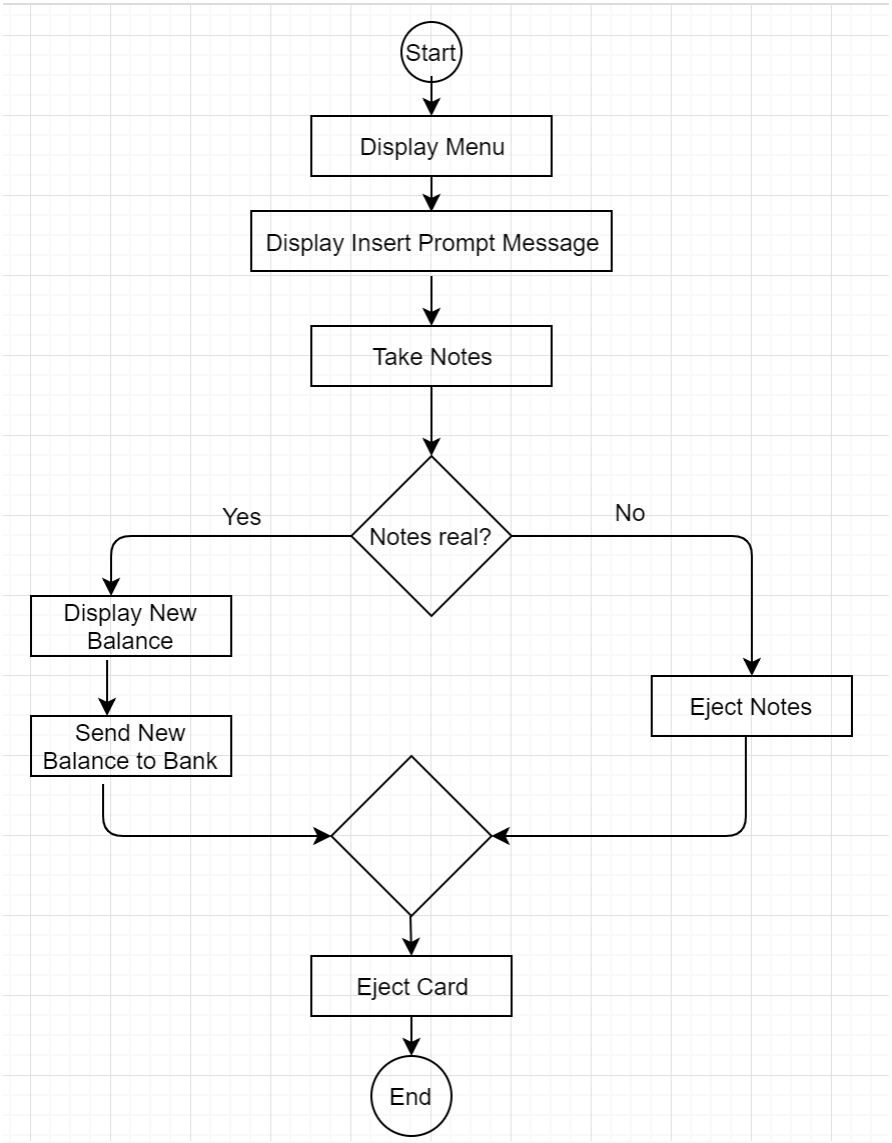
3.4.4 Collaboration Diagram



3.4.5 Activity Diagram



3.4.6 Flow Chart



3.5 Transfer

3.5.1 Use Case Scenario

Use Case Name: transfer money to another account.

Goal: to transfer money.

Actors: user, receiver, ATM, Bank

Precondition: user is validated

Primary Scenario:

1. ATM shows menu.
2. user chooses "transfer money".
3. ATM asks user to enter receiver's IBAN.
4. If IBAN is correct, ATM asks for the amount transferred.
5. If the amount is available in the user's account, money is transferred to receiver's account.
6. User quits.
7. ATM ejects the card.

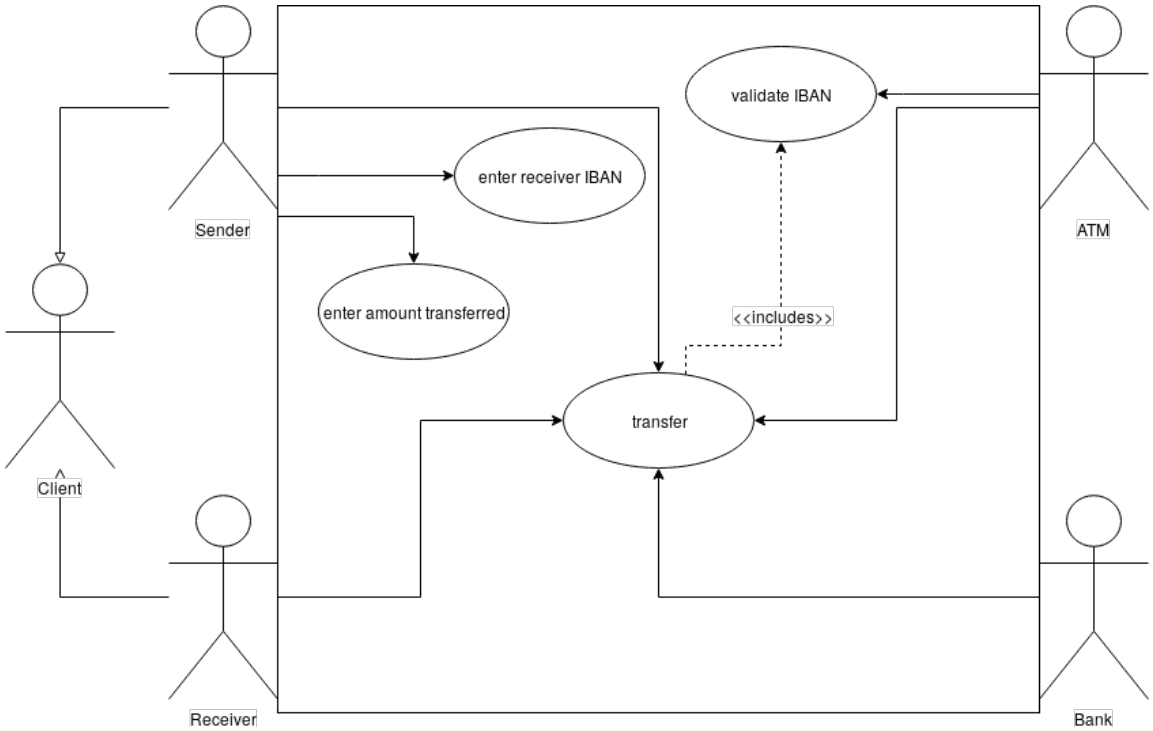
Variant:

4a. If IBAN is incorrect, banks asks the user to enter it again.

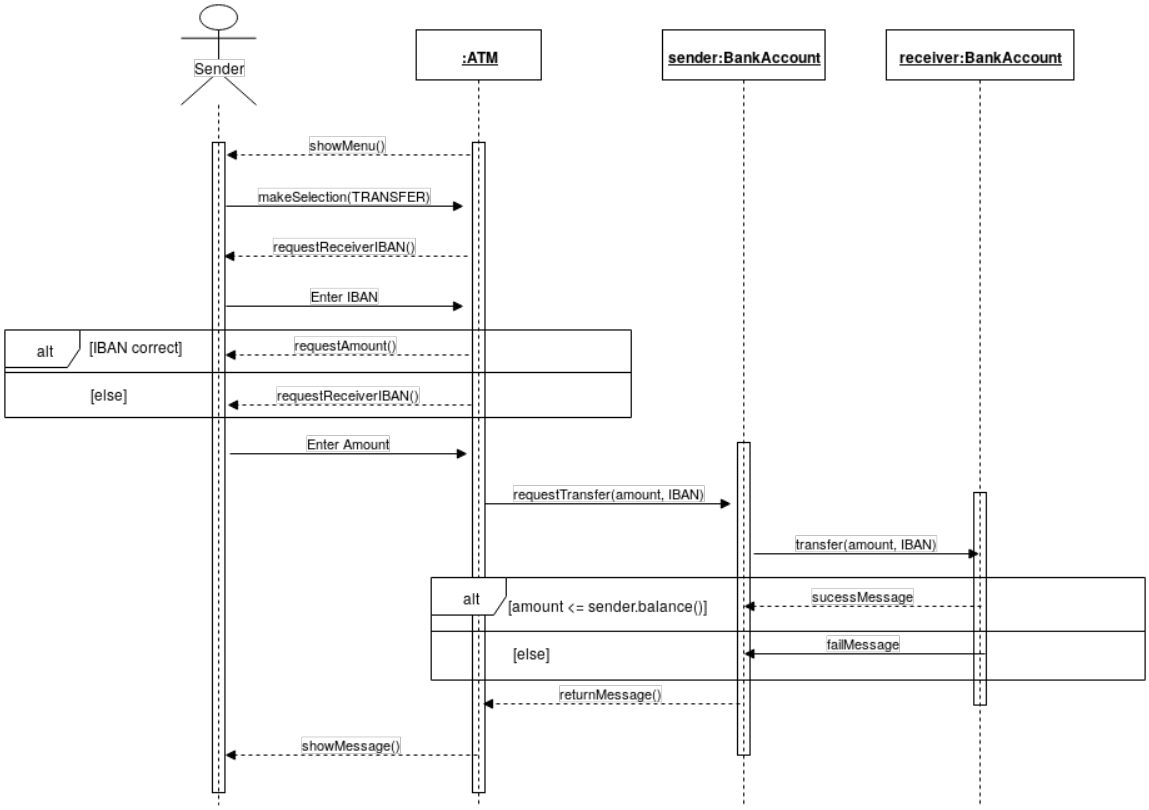
5a. If the amount transferred is less than the user's current balance, the ATM shows an error message.

*. The user may cancel the session; the ATM ejects the card.

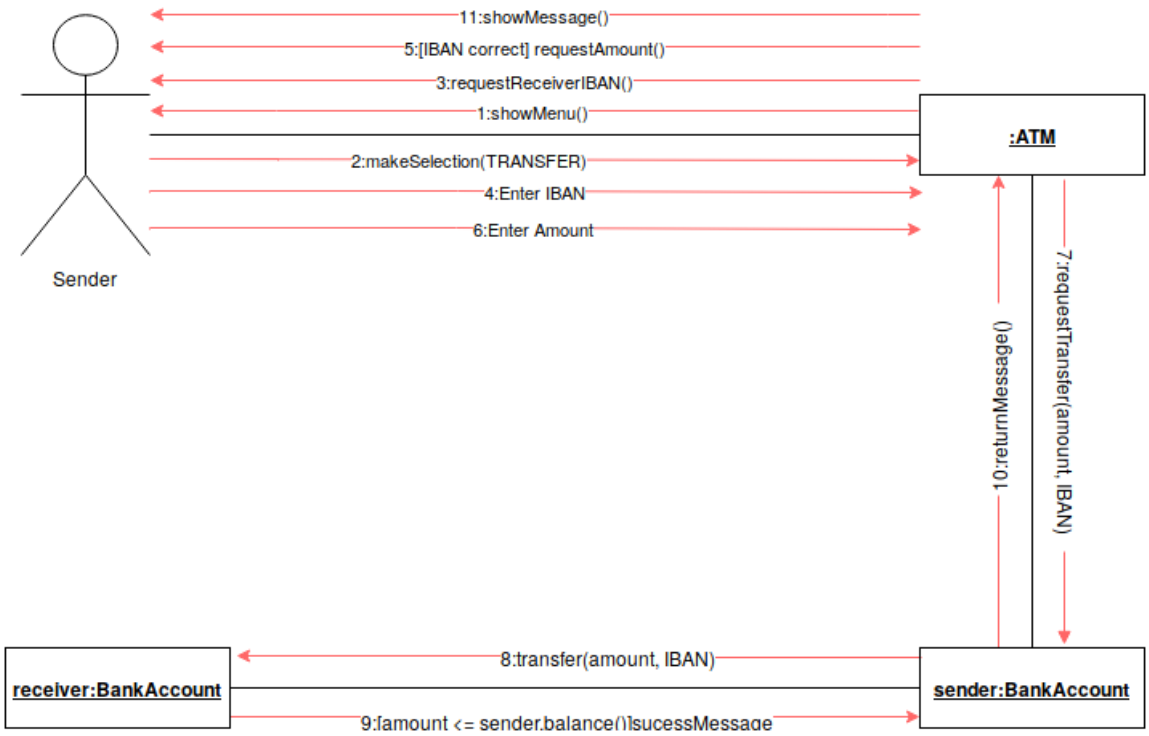
3.5.2 Use Case Diagram



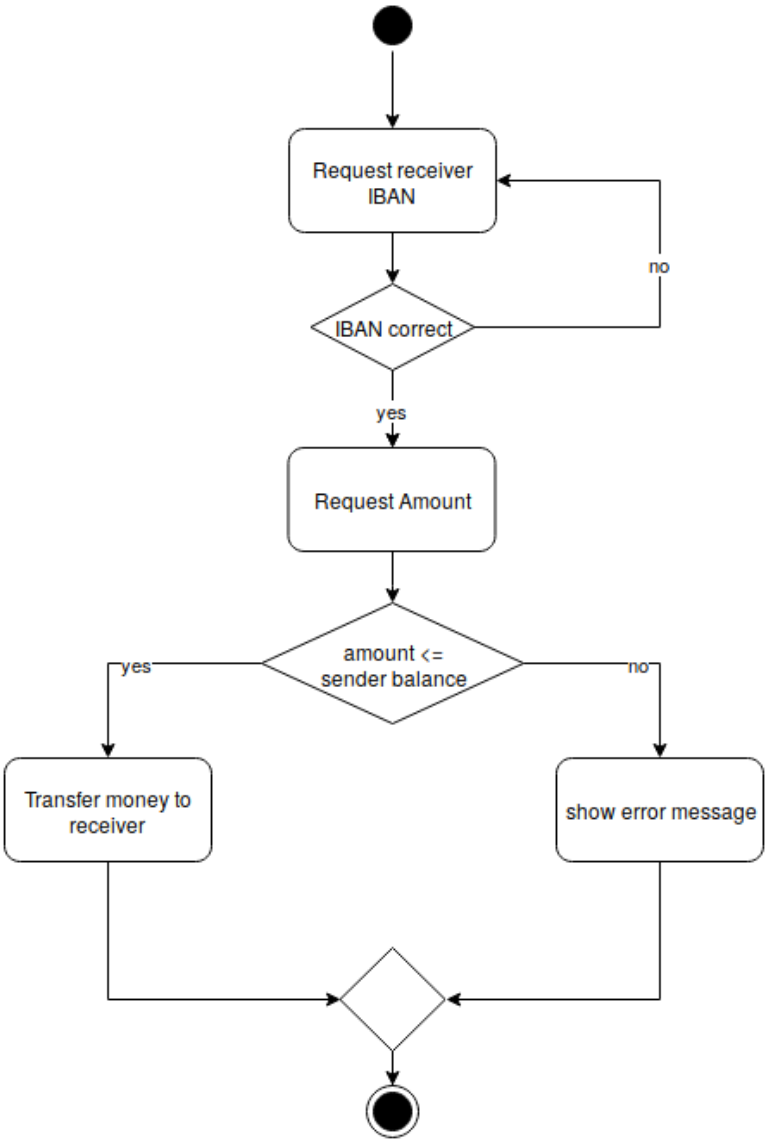
3.5.3 Sequence Diagrams



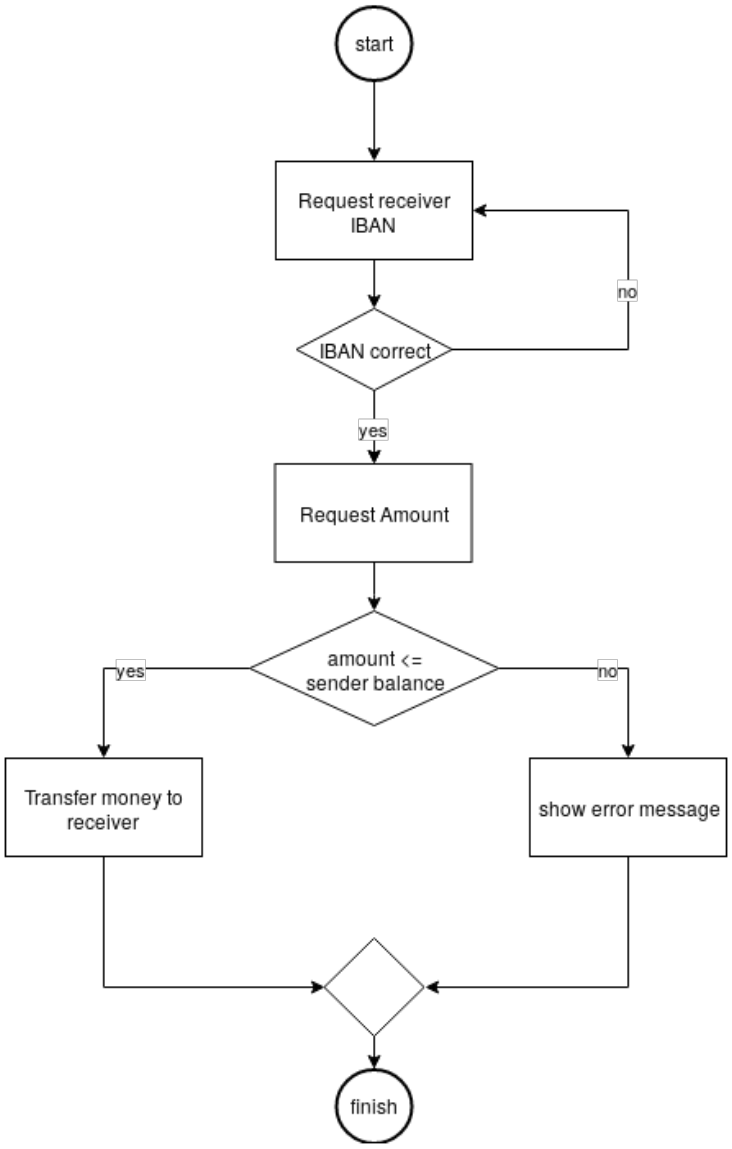
3.5.4 Collaboration Diagram



3.5.5 Activity Diagram



3.5.6 Flow Chart



3.6 For The Whole System

3.6.1 Context Diagram

3.6.2 Use-case Diagram

3.6.3 Component diagram

3.6.4 Deployment diagram

3.6.5 Data flow Diagram

3.6.6 Class Diagram

3.6.7 Object Diagram

3.6.8 State Chart Diagram

3.6.9 Architectural patterns

References

[1] "An example of object-oriented design: An atm simulation." http://www.math-cs.gordon.edu/courses/cs211/ATMExample/UseCases.html, Dec 2004. Accessed on 2019-03-09.

[2] "Sdlc waterfall model." https://www.tutorialspoint.com/sdlc/sdlc_waterfall_model.htm, Mar 2019. Accessed on 2019-03-09.