Individual Project: ATM Machine

CSC 340 Ethics & Software Engineering

Ivan Olivas

May 6, 2021

Table of Tables

Table of Contents

Table of Figures

I. Introduction

A. Problem Statement

A new local bank ZZZ asks for your help to develop a software system for their multiple ATM machines. They need an ATM machine system that can provide different banking services to their clients.

B. System Proposal

I propose a software system to help the bank with the desired functionality of the system. A software system would be the best approach to provide all the needs of the client.

II. System Description

The machines will provide basic operations to their customers, including depositing money, withdrawing money, checking balances, and transferring money from one account to another. Each machine will be refilled to hold $100,000 cash daily for possible withdrawals. For security reason, each account can have at most $3000 in total for all the transactions (except for checking balance) through ATM machines each day. Effectiveness and efficiency are their primary requirements.

III. System Requirements

A. Functional Requirements

R1. The system shall allow a customer to withdraw money from his/her banking accounts.

1.1. The system shall display a user menu to the customer.

1.2. The customer shall select "Withdraw Money" in the menu.

1.3. The system shall display a list of accounts of the customer to be selected for withdrawal.

1.3.1. The system shall retrieve the account information from the central database system with the customer's id.

1.3.2. The system shall format and display only account numbers to the customer.

1.4. The customer shall select one account from the list for withdrawal.

1.5. The system shall update the date of the daily transactions of the selected account.

1.5.1. If the last date of the transactions is not today's date, the system shall change the date of the transactions to today's date and reset the daily transaction total to $0.

1.6. The system shall check if the account's daily transaction total has reached $3000 (see NR1).

1.6.1. If the total has reached $3000 already, the system shall display an error message to
the customer and check if there is any other account to be selected for withdrawal.

1.6.1.1. If there are more accounts to be selected, return Step 1.3.

1.6.1.2. If there is not any other account to be selected, return to Step 1.1.

1.6.2. If they haven't reached $3000, the system shall continue Step 1.6.

1.7. The system shall request the customer to enter the amount to withdrawal.

1.8. The customer shall enter an amount.

1.9. The system shall check if the total of the amount plus the daily transaction total is greater
than $3000 (see NR1).

1.9.1. If the total is greater than $3000, the system shall display an error message to the
customer, and return to Step 1.5.

1.9.2. If the total is less than or equal to $3000, the system shall continue Step 1.9.

1.10. The system shall verify if the amount is less than the current balance of the selected account.

1.10.1. If the amount is greater than the balance, the system shall display an error message
to the customer and return to Step 1.5.

1.10.2. If the amount is less than or equal to the current balance, the system shall continue to Step 1.10.

1.11. The system shall check if the machine has enough cash for the withdrawal.

1.11.1. If there is not enough cash for the withdrawal, the system shall display an error message, and return to Step 1.6.

1.11.2. If there is enough cash for the withdrawal, the system shall continue Step1.

1.12. The system shall dispense the cash.

1.13. The system shall deduct the amount from the balance of the account.

1.14. The system shall add the amount to the daily transaction total.

1.15. The system shall store this transaction to the database.

1.16. The system shall update the account information in the database.

1.17. The system shall return to main menu.

R2. The system shall allow a customer to deposit money from his/her banking accounts.

2.1. The system shall display a user menu to the customer.

2.2. The customer shall select “Deposit Money” in the menu.

2.3. The system shall display a list of accounts of the customer to be selected for depositing
money.

2.3.1. The system shall retrieve the account information from the central database system with the customer’s id.

2.3.2. The system shall format and display only the account numbers to the customer.

2.4. The customer shall select one account from the list for depositing money.

2.5. The system shall request the customer to insert the amount of money he/she wishes to deposit.

2.6. The system will count the money inserted.

2.7. The system will display the amount of money the customer inserted, along with a “Confirm” button and a “Cancel” button.

2.8. The customer must verify, by selecting “Confirm”, that the amount displayed is the amount they meant to deposit.

2.8.1. If the amount displayed was not the correct amount, customer should select “Cancel”, and return to Step 2.4.

2.8.1.1. The system will return the money the customer inserted.

2.9. The system shall add the amount to the balance of the account selected.

2.10. The system shall store this transaction to the database.

2.11. The system shall update the account information in the database.

2.12. The system shall return to the main menu.

R3. The system shall allow a customer to transfer money from his/her banking accounts.

3.1. The system shall display a user menu to the customer.

3.2. The customer shall select “Transfer Money” in the menu.

3.3. The system will check if the customer has at least two accounts.

3.3.1. If customer has only one account, the system will prompt the user will

3.4. The system shall display a list of accounts of the customer to select which account to transfer money from.

3.4.1. The system shall retrieve the account information from the central database system with the customer’s ID.

3.4.2. The system shall format and display only the account numbers to the customer.

3.5. The customer shall select one account from the list to transfer money from.

3.6. The system shall update the date of the daily transactions of the selected account.

3.6.1. If the last date of the transactions is not today’s date, the system shall change the date of the transactions to today’s date and reset the daily transaction total to $0.

3.7. The system shall check if the account’s daily transaction total has reached $3000 (see NR1).

3.7.1. If the total has reached $3000 already, the system shall display an error message to the customer, and check if there is any other account to be selected for transfer.

3.7.1.1. If there are more accounts to be selected, return Step 3.7.

3.7.1.2. If there is not any other account to be selected, return to Step 3.1.

3.8. The system shall display a list of accounts of the customer to select which account to transfer money to.

3.8.1. The system shall retrieve the account information from the central database system with the customer’s ID.

3.8.2. The system shall format and display only the account numbers to the customer.

3.9. The customer shall select one account from the list to transfer money to.

3.10. The system shall update the date of the daily transactions of the selected account.

3.10.1. If the last date of the transactions is not today’s date, the system shall change the date of the transactions to today’s date and reset the daily transaction total to $0.

3.11. The system shall check if the account’s daily transaction total has reached $3000 (see NR1).

3.11.1. If the total has reached $3000 already, the system shall display an error message to the customer, and check if there is any other account to be selected for transfer.

3.11.1.1. If there are more accounts to be selected, return Step 3.8.

3.11.1.2. If there is not any other account to be selected, the customer will be presented with a screen, “No other accounts to transfer money to”, and return to Step 3.1.

3.12. The system shall request the customer to enter the amount they want to transfer from the first account selected in Step 3.5.

3.13. The customer shall enter an amount.

3.14. The system shall verify if the total amount entered is less than or equal to the current balance of the selected account in Step 3.5.

3.14.1. If the amount is greater than the balance, the system shall display an error message to the customer, and return to Step 3.13.

3.14.2. If the amount is less than or equal to the current balance, the system shall continue to step 3.15.

3.15. The system shall deduct the amount selected from the account (from Step 3.5) and add that same amount to the account selected (from Step 3.9).

3.16. The system shall store this transaction to the database.

3.17. The system shall update the account information in the database.

3.18. The system shall display “Transfer Successful” and return to the main menu.

R4. The system shall allow a customer to check their balances from his/her banking accounts.

4.1. The system shall display a user menu to the customer.

4.2. The customer shall select “Check Balance” in the menu.

4.3. The system shall display a list of accounts of the customer to be selected for checking balance.

4.3.1. The system shall retrieve the account information from the central database system with the customer’s ID.

4.3.2. The system shall format and display only account numbers to the customer.

4.4. The customer shall select one account from the list to check balance.

4.4.1. The system will retrieve the corresponding account balance from the database system with the customer’s ID.

4.4.2. The system shall format and display the account number along with the account balance, and check if there is any other account to be selected to check balance.

4.4.2.1. If there are more accounts to be selected, return to Step 4.3.

4.4.2.2. If there is not any other account to be selected, return to Step 4.1.

R5. The system shall allow a user to sign into the system to access the main menu.

5.1. The user will be prompted to insert their card or login into the ATM using username and password.

5.2. The customer shall enter the bank card or userID.

5.3. The machine shall validate the bank card or userID.

5.3.1. If the userID is found in the database, the customer will be prompted for the password or passcode. The machine will go to Step 5.4.

5.3.2. If login combination is not found in database, an error message (Invalid Login) shall be displayed with a “Continue” button. The machine will return to Step 5.1.

5.4. The customer shall ask the user to input their password or passcode.

5.5. The customer shall enter the password or passcode.

5.5.1. If the password is found in the database, the customer will be presented the main menu.

5.5.2. If password is not found in database, an error message (Invalid Password) shall be displayed with a “Continue” button. The machine will return to Step 5.4.

R6. The system shall allow a customer to sign out from ATM Machine.

6.1. The system shall display a user menu to the customer.

6.2. The system shall display an “Sign-Out” Button on the main menu to the customer.

6.3. The customer shall select “Sign Out” in the menu.

6.4. The system shall save any changes made by the customer.

6.5. The system shall update the account information in the database.

6.6. The system shall display the sign in screen from Step 5.1.

B. Non-functional Requirements:

NR1. The total of daily transactions (withdrawals and money transfers) shall not exceed $3000 from same account.

NR2. The cash in a machine will be reset to $100000 every day. This action will be done manually by the employees of the bank (The software system is not responsible for this part.)

NR3. The amount of a deposit will not be added to the balance of an account immediately. The system will show the deposit amount in another way. The action for adding the amount to the balance will be done manually by the employees of the bank (The software system is not responsible for this part.)

NR4. The ATM machine can only service one user or customer at a given time.

NR6. The minimum amount a customer can deposit is $1.00.

NR7. No coins are accepted at the ATM machine.

Use Case Diagram

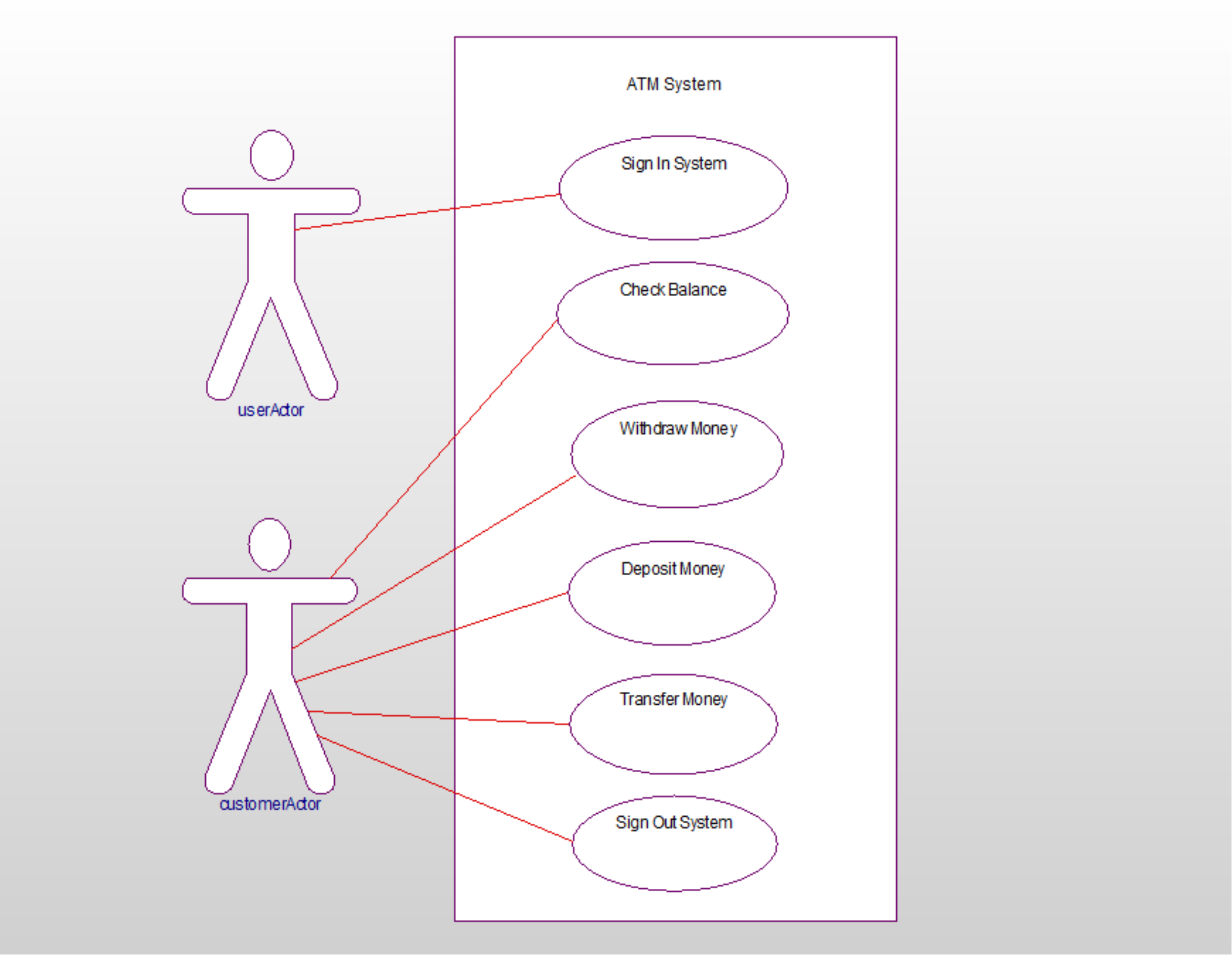


Figure 1 - Use Case Diagram for ATM Machine

This is a diagram to show the functionality of the system with use cases. It has all the different kinds of services the system will offer to the clients. As you can see, the diagram differentiates between a user and a customer. A user is someone who is attempting to access the system but is not a verified customer. While a customer is someone who has been verified and has an account with ZZZ bank.

IV. Class Diagram

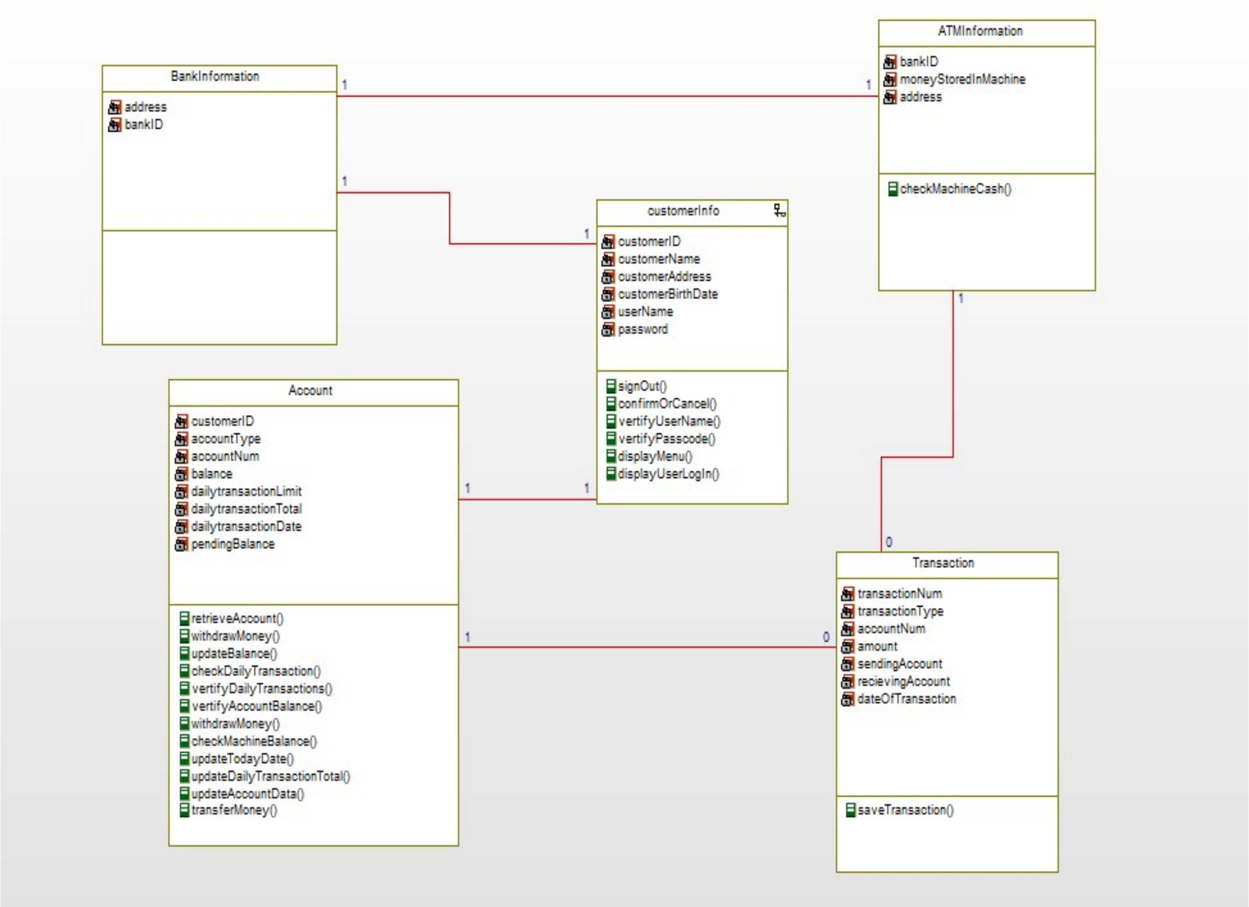


Figure 2 Class Diagram for ATM Machine

This is a diagram to show the related attributes and operations within each class that the system will use to perform the functionalities needed. The account class will have most of the functions in the system. The information about the customer will be in the customerInfo class and that is where the system will check to see if someone is a verified customer.

V. Sequence Diagrams

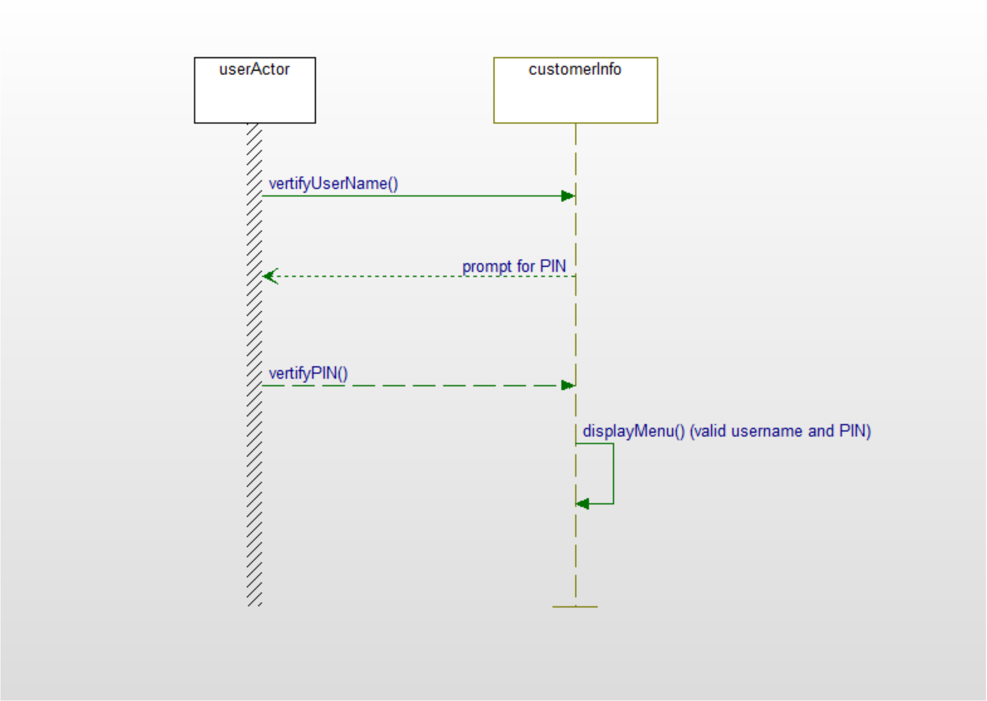


Figure 3 Sequence Diagram for logging in the ATM

This is diagram shows the steps of the "logging in" functionality. The system will verify the username and passcode of the user to display the menu, if the combination is valid.

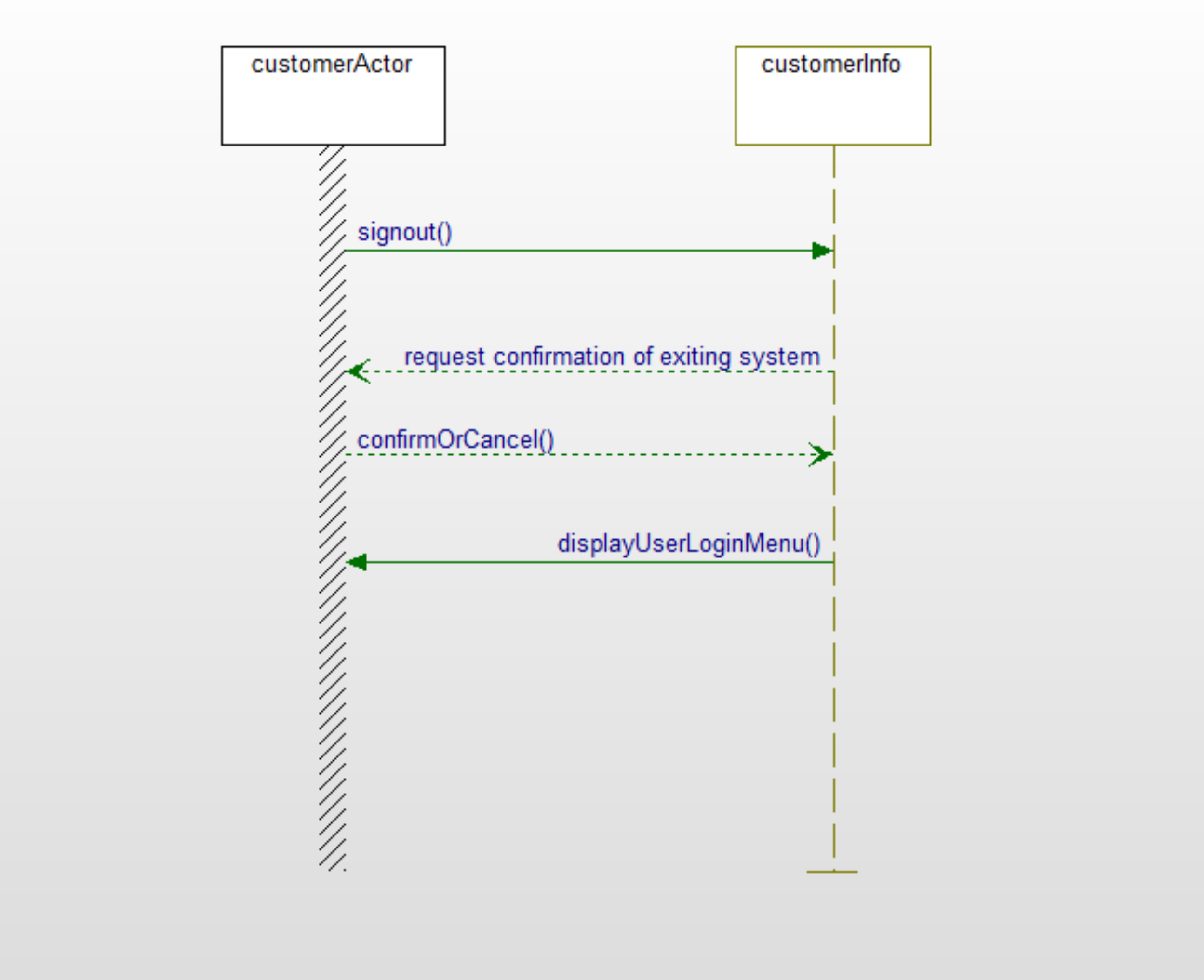


Figure 4 - Sequence Diagram for logging out of the system

This diagram shows the steps the system will take when signing out of the system. After signing out, the system will display the login menu.

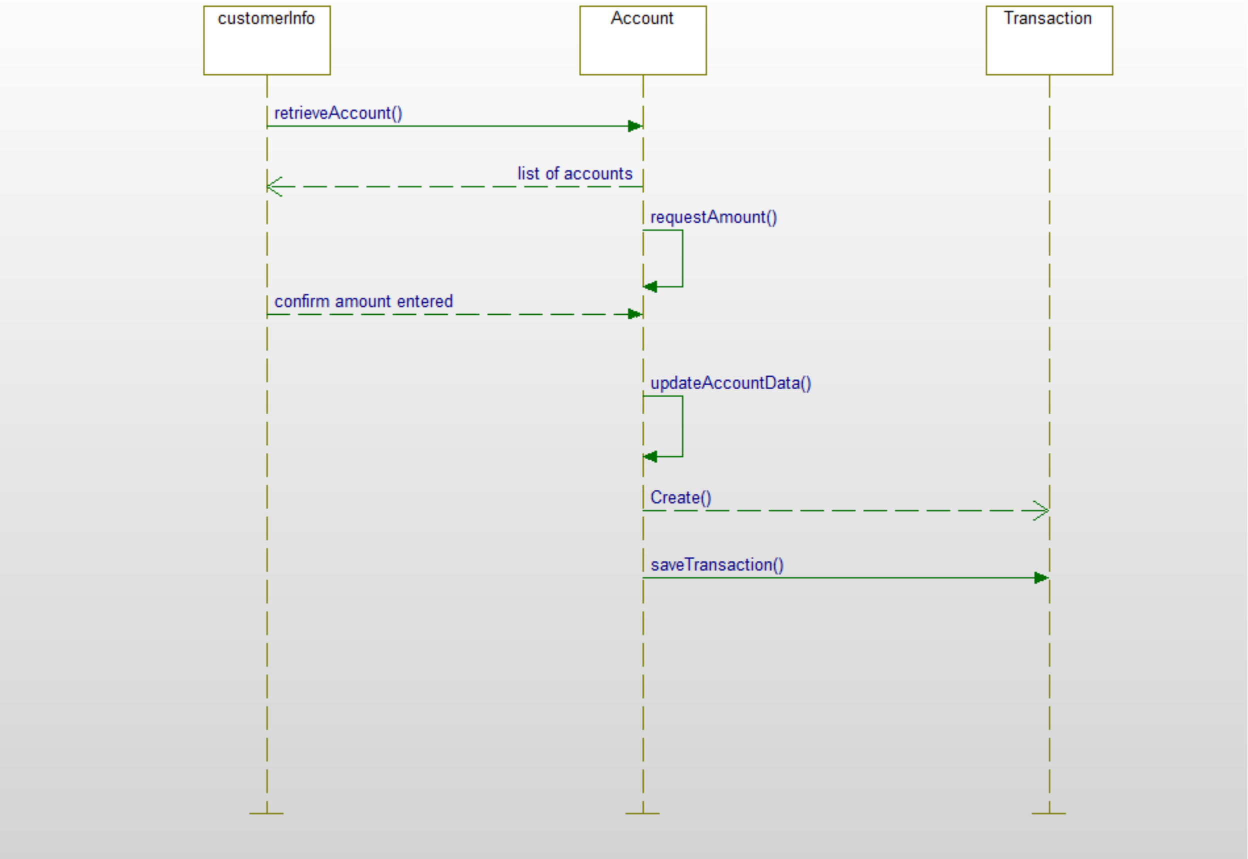


Figure 5 - Sequence Diagram for depositing money

This diagram shows the steps the system will take when depositing money in an account. The system will request for the amount they want to enter, and the system will update. When depositing money, the pending balance attribute in the account will be updated. The bank should manually add that amount to the regular balance in the respective account.

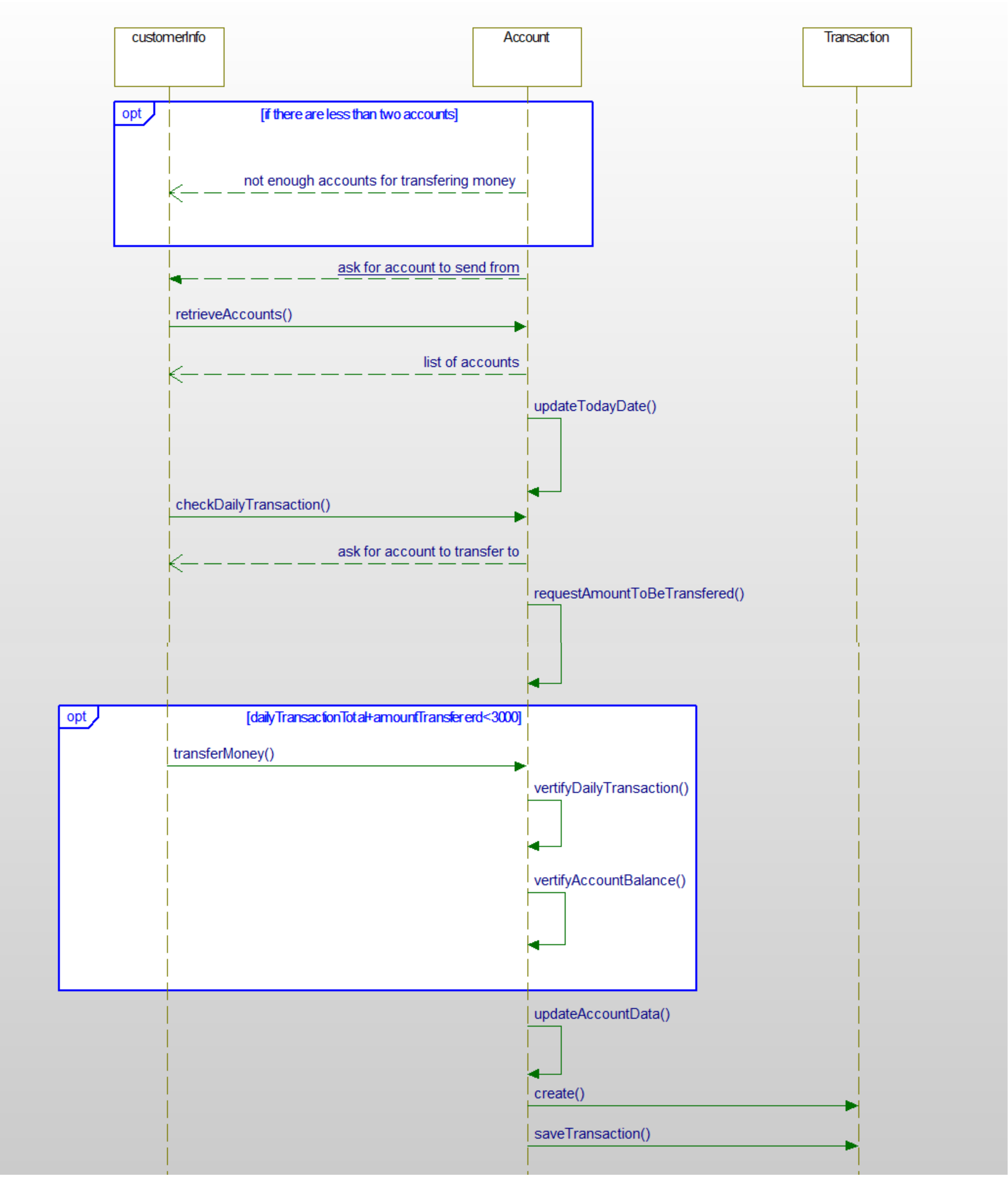
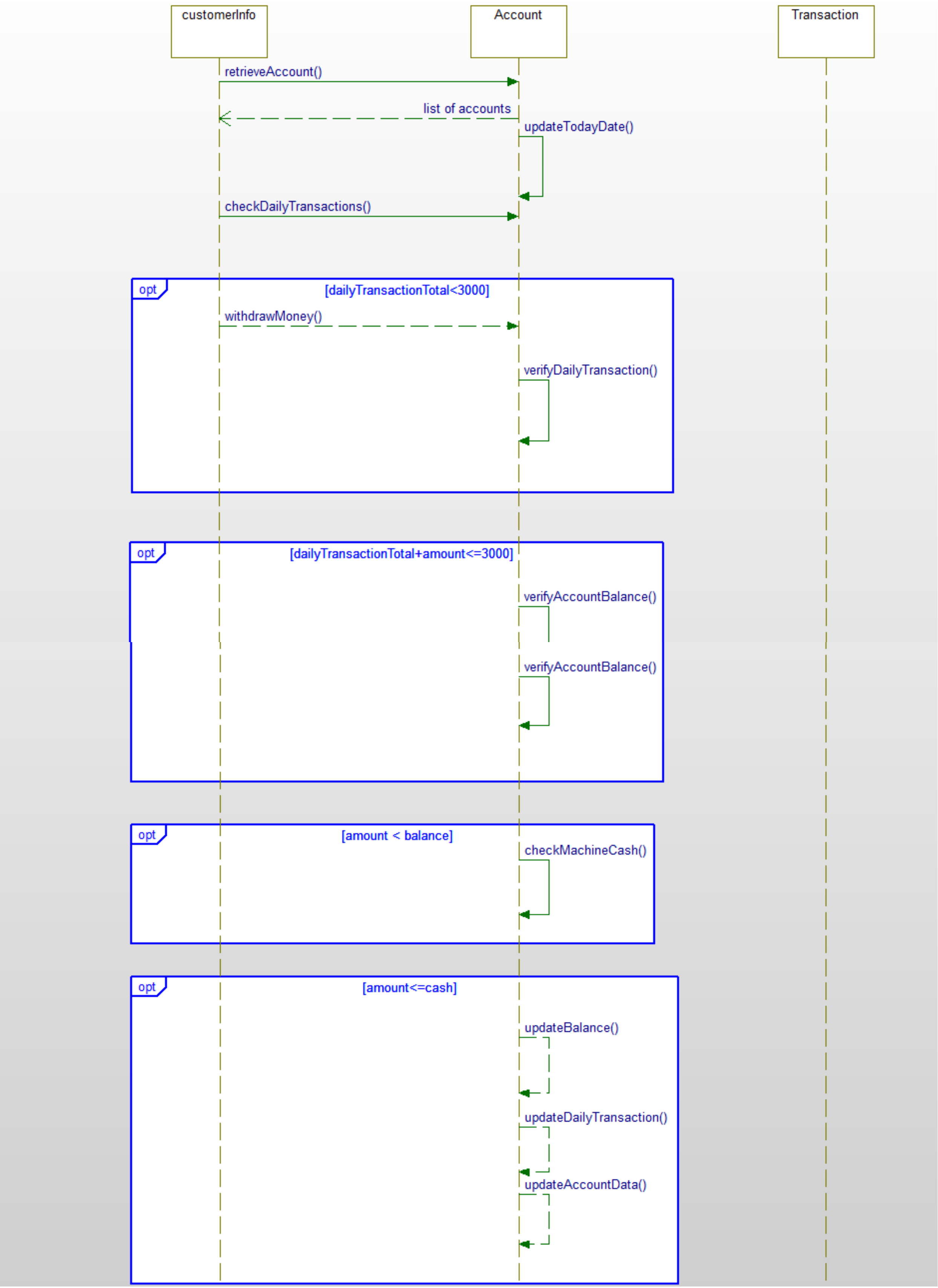

customerInfo
Account
Transaction
opt
[if there are less than two accounts]
not enough accounts for transfering money
ask for account to send from
retrieveAccounts()
list of accounts
updateTodayDate()
checkDailyTransaction()
ask for account to transfer to
requestAmountToBeTransfered()
opt
[dailyTransactionTotal+amountTransfererd<3000]
transferMoney()
vertifyDailyTransaction()
vertifyAccountBalance()
updateAccountData()
create()
saveTransaction()

Figure 6 Sequence Diagram for transferring money

This diagram shows the steps the system will take when transferring money. The daily transaction limit will also apply to transfer. This means a customer cannot either transfer or withdraw more than 3, 000 in one day.


customerInfo
Account
Transaction
retrieveAccount()
list of accounts
updateTodayDate()
checkDailyTransactions()
opt
[dailyTransactionTotal<3000]
withdrawMoney()
verifyDailyTransaction()
opt
[dailyTransactionTotal+amount<=3000]
verifyAccountBalance()
verifyAccountBalance()
opt
[amount < balance]
checkMachineCash()
opt
[amount<=cash]
updateBalance()
updateDailyTransaction()
updateAccountData()

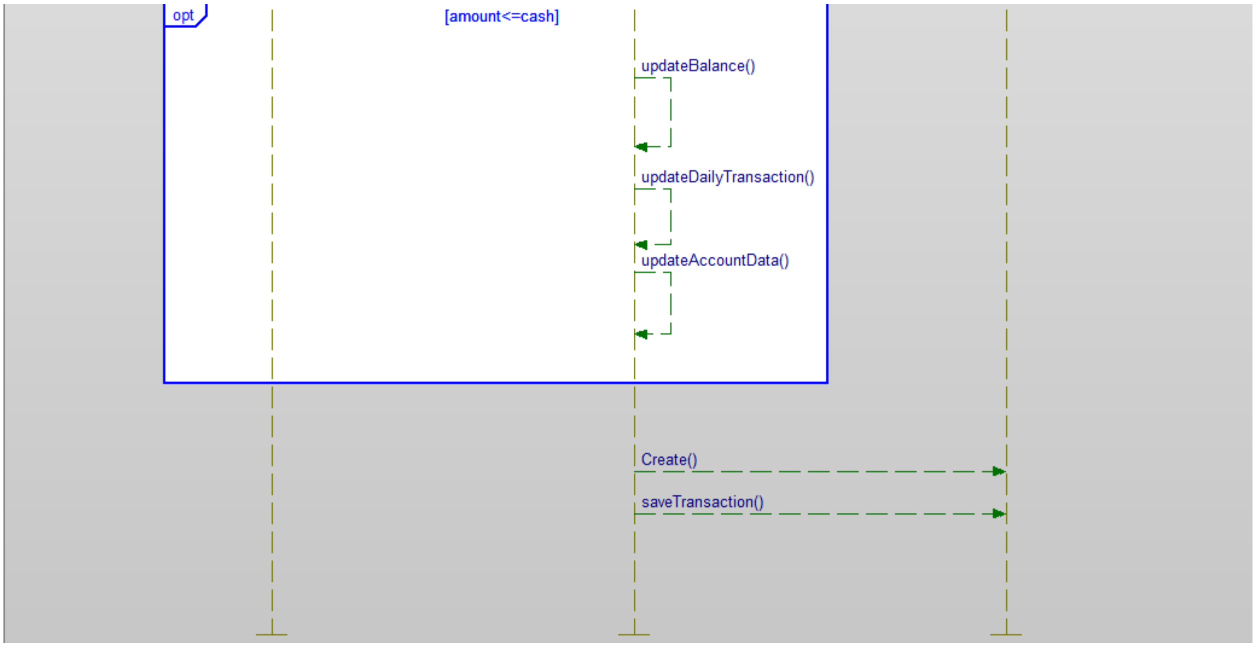


Figure 7 - Sequence Diagram for withdraw

This diagram shows the steps the system will take to withdraw money. The system will perform various error handling check in order to make sure the customer does not exceed the transaction limit for that day, makes sure that they ATM has enough cash and that their balance has enough money.

VI. Activity Diagrams

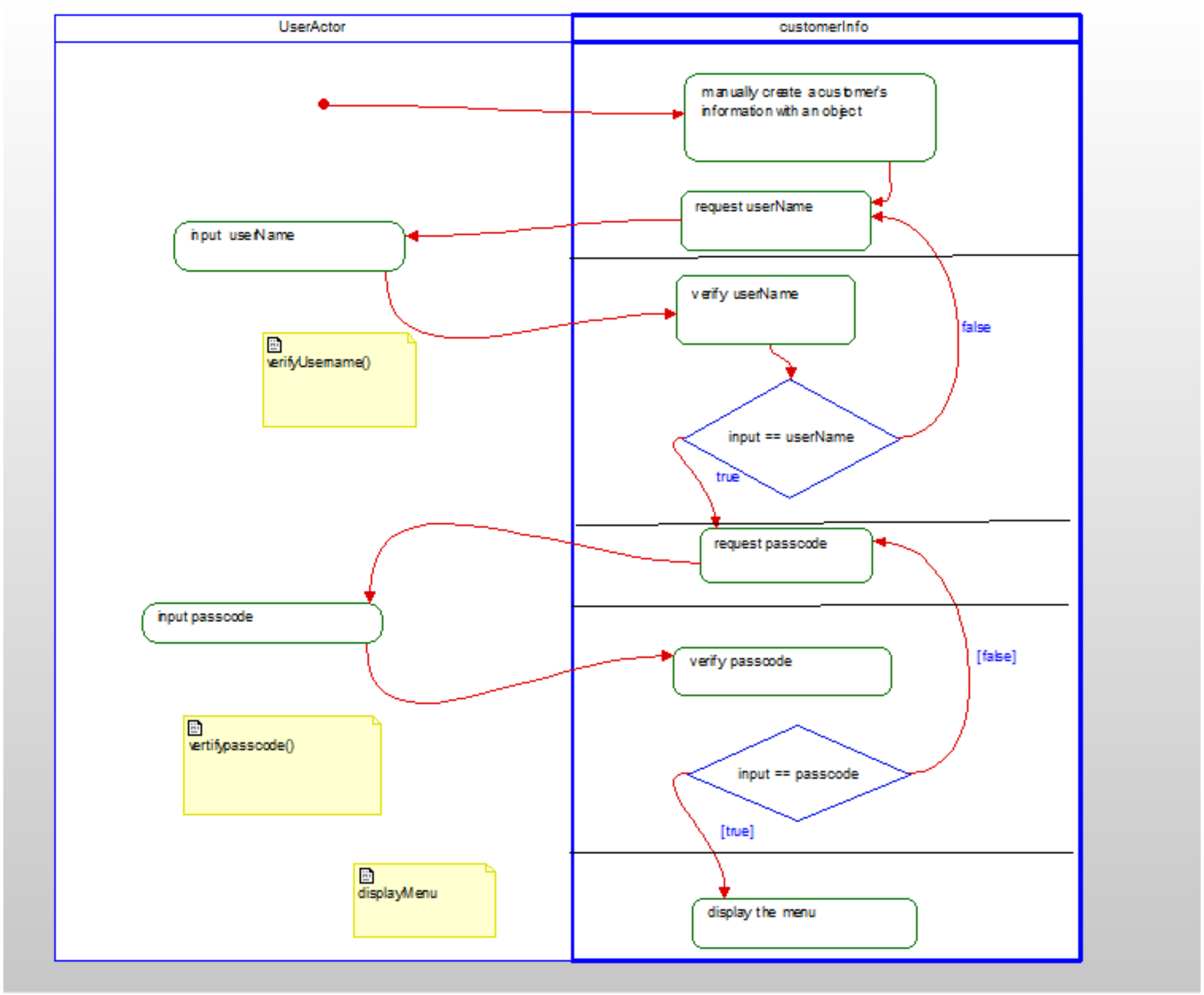


Figure 8 - Activity Diagram for logging in

This diagram shows the various functions the system will call when the user logs in. The system will check to see if the login combination is associated with an account.

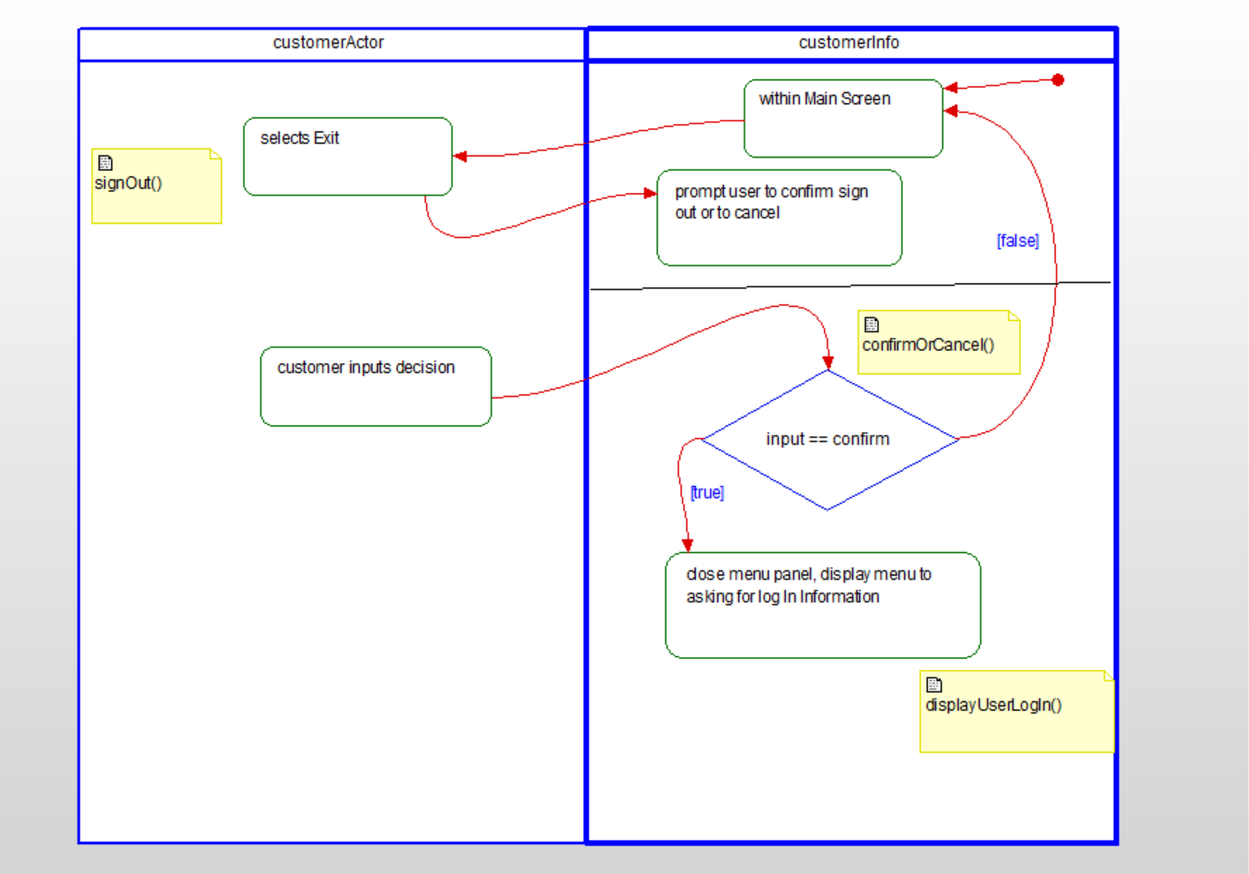


Figure 9 - Activity Diagram for logging out of the system

The diagram shows the methods used to log out of the system. After signing out, the system will display the user login menu.

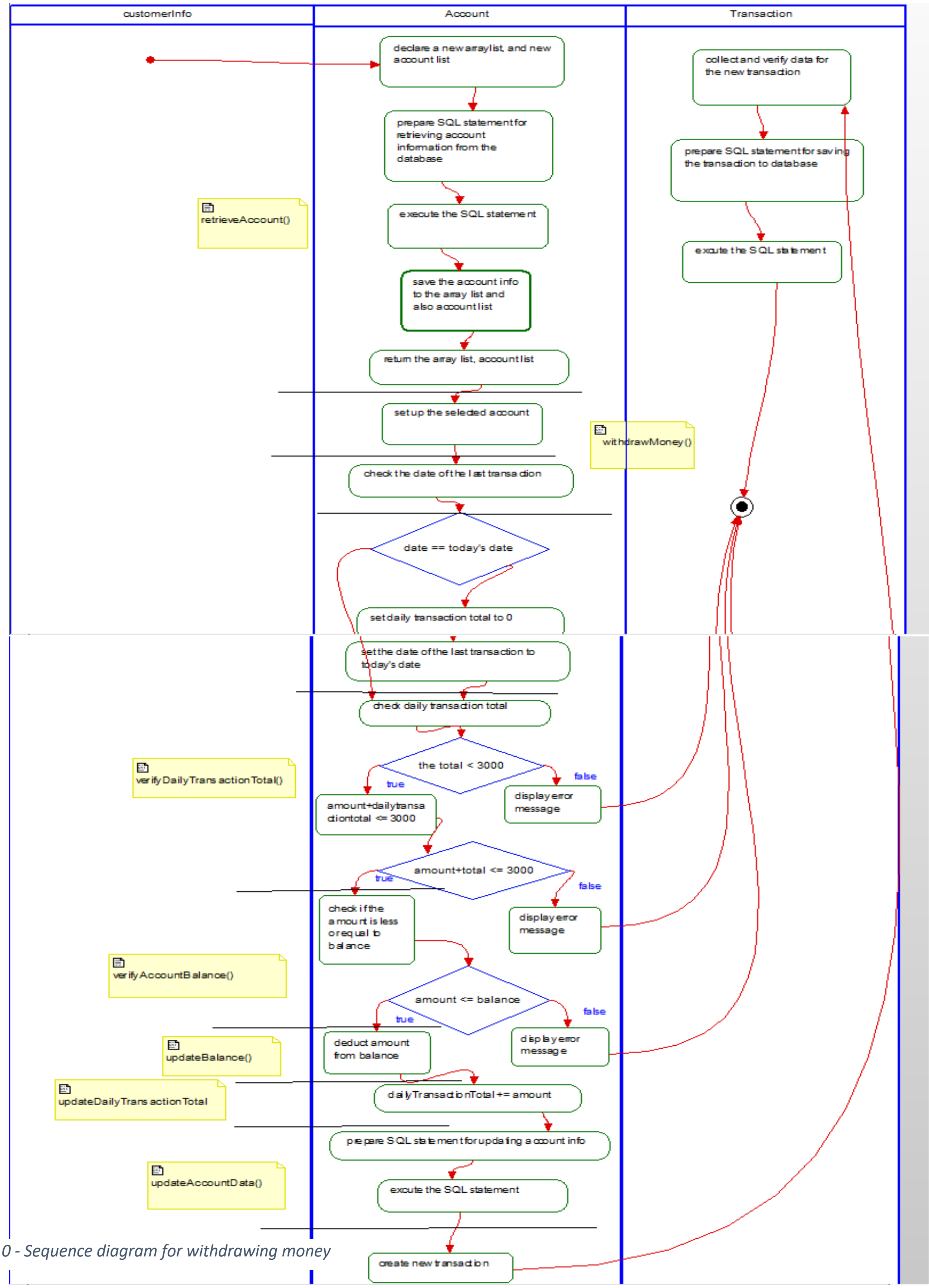

customerInfo
Account
Transaction
declare a new arraylist, and new account list
prepare SQL statement for retrieving account information from the database
execute the SQL statement
save the account info to the array list and also account list
return the array list, account list
set up the selected account
check the date of the last transaction
date == today's date
set daily transaction total to 0
set the date of the last transaction to today's date
check daily transaction total
the total < 3000
true
false
amount+dailytransactiontotal <= 3000
display error message
amount+total <= 3000
true
false
check if the amount is less or equal to balance
display error message
amount <= balance
true
false
deduct amount from balance
display error message
dailyTransactionTotal += amount
prepare SQL statement for updating account info
excute the SQL statement
create new transaction
collect and verify data for the new transaction
prepare SQL statement for saving the transaction to database
excute the SQL statement
retrieveAccount()
withdrawMoney()
verifyDailyTransactionTotal()
verifyAccountBalance()
updateBalance()
updateDailyTransactionTotal
updateAccountData()

Figure 10 - Sequence diagram for withdrawing money

This diagram shows the functions that will be used for the withdraw use case.

VII. State Diagrams

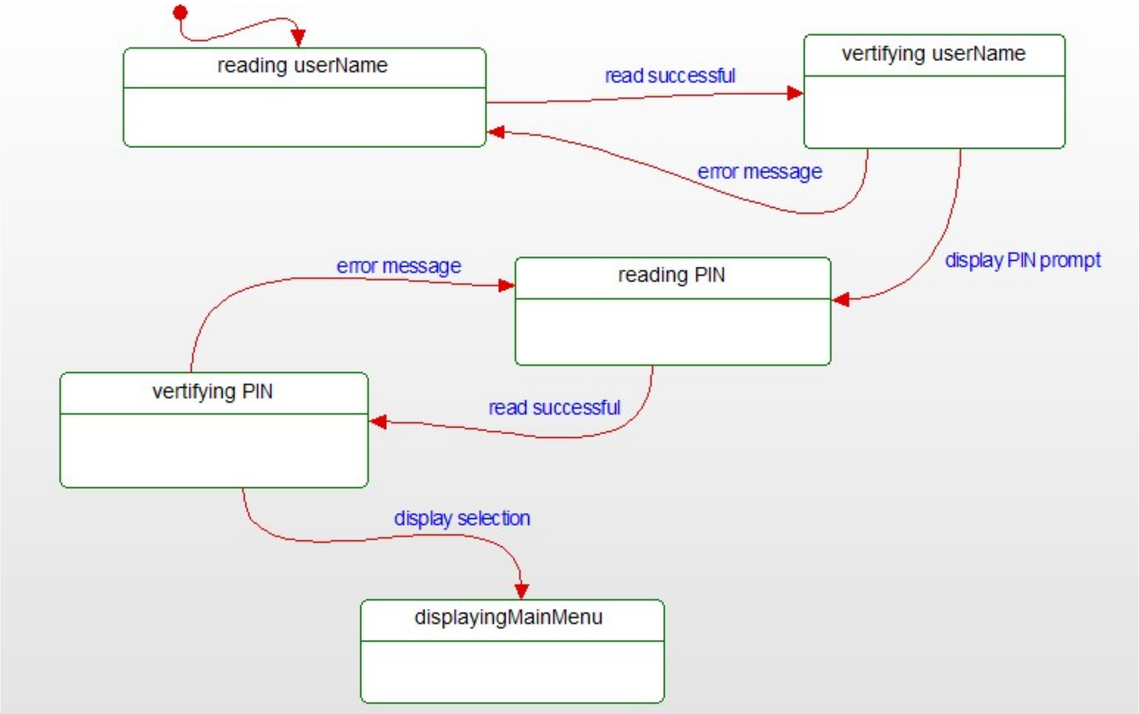


Figure 11 - State diagram for signing in

This diagram shows the various states the system is in when a user signs in. If the user does not enter a correct login, they will have the option to re-enter the credentials.

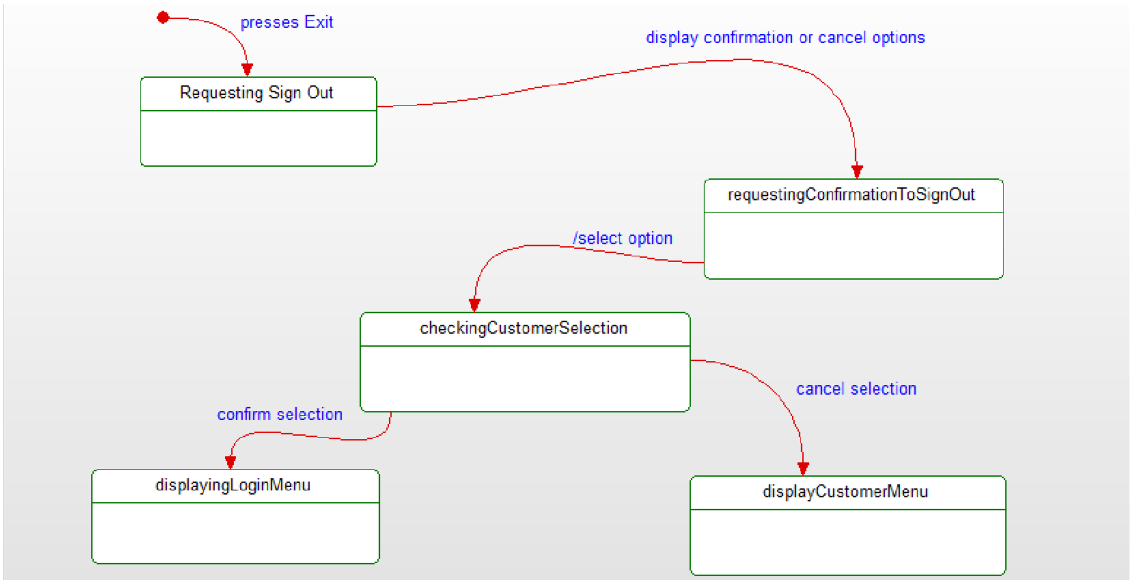


Figure 12 - State diagram for signing out

This diagram shows the states of the system when signing out. The system will return to the login menu after this.

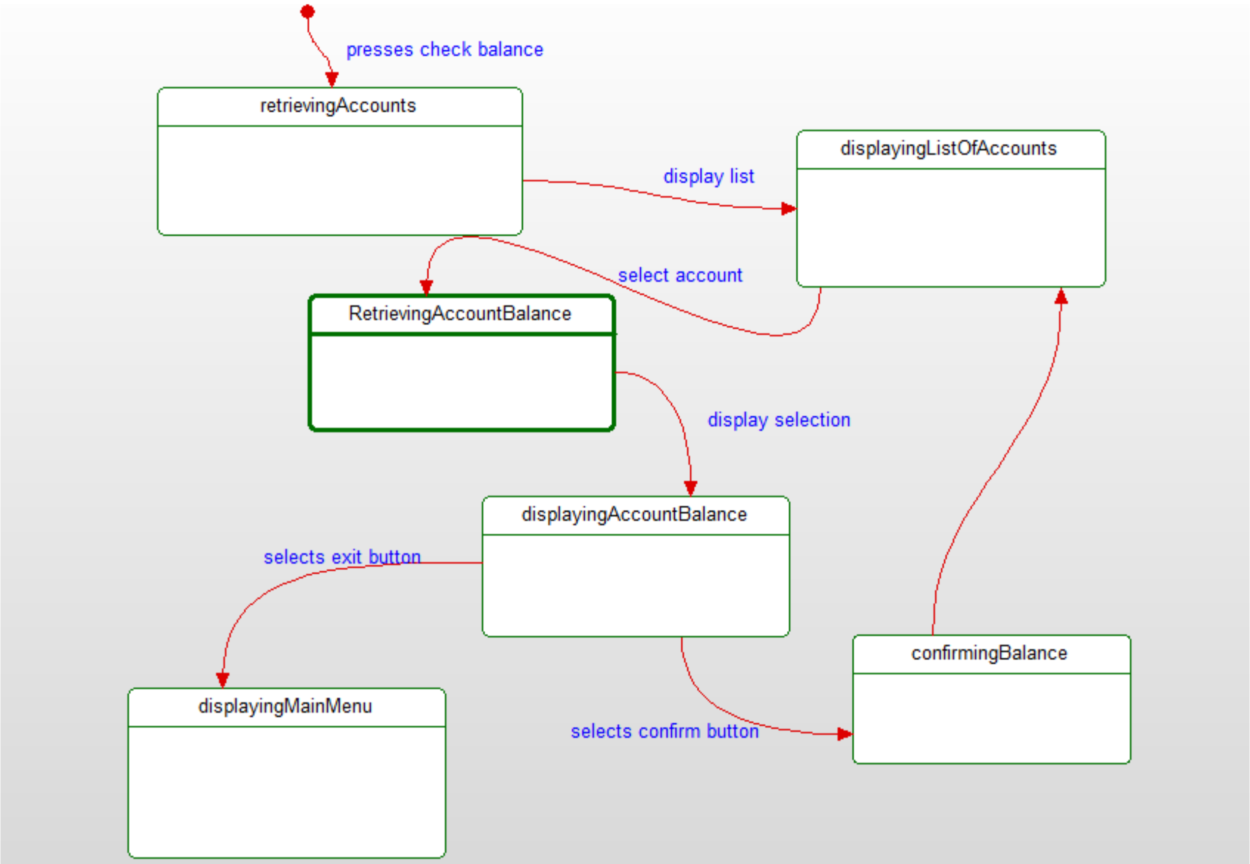


Figure 13 - State diagram for checking the balance

This diagram shows the states of the system when a customer checks their balance. There are six different states for this use case.

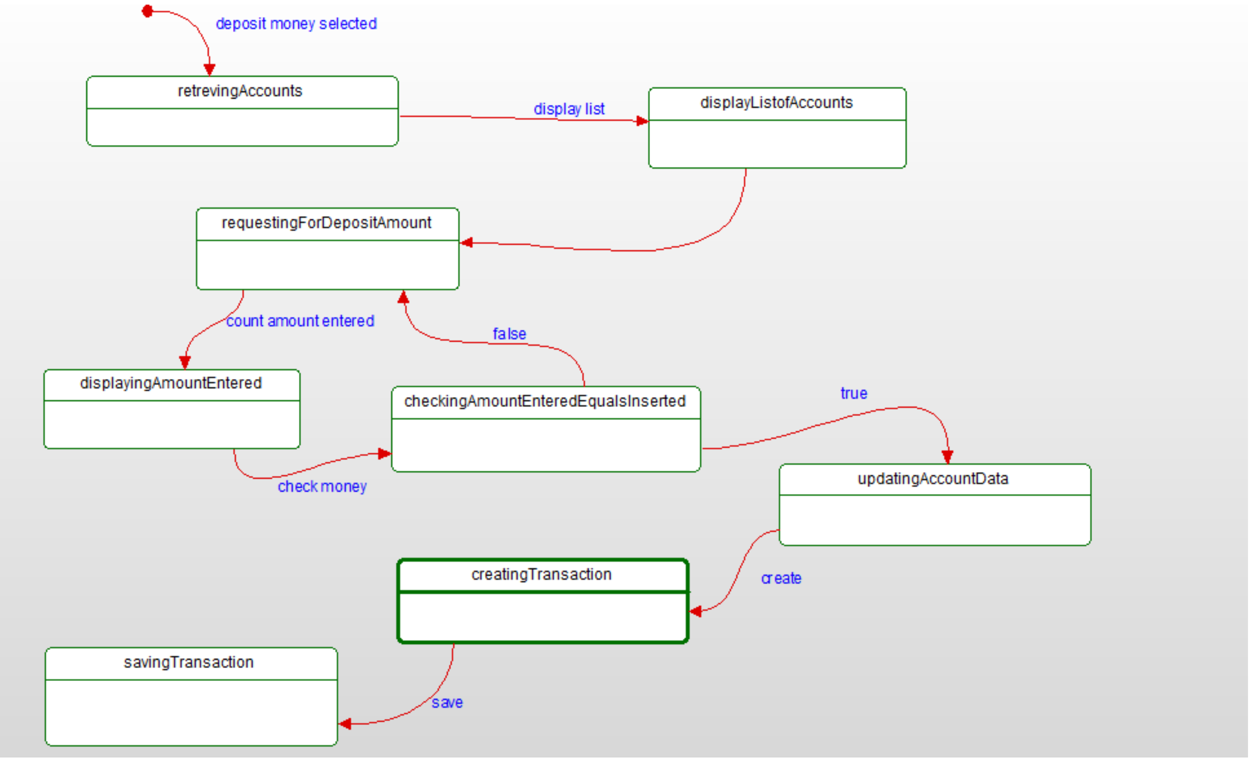


Figure 14 - State diagram for depositing money

This diagram shows the states of the system for depositing money. There are eight different states the machine is in for this use case.

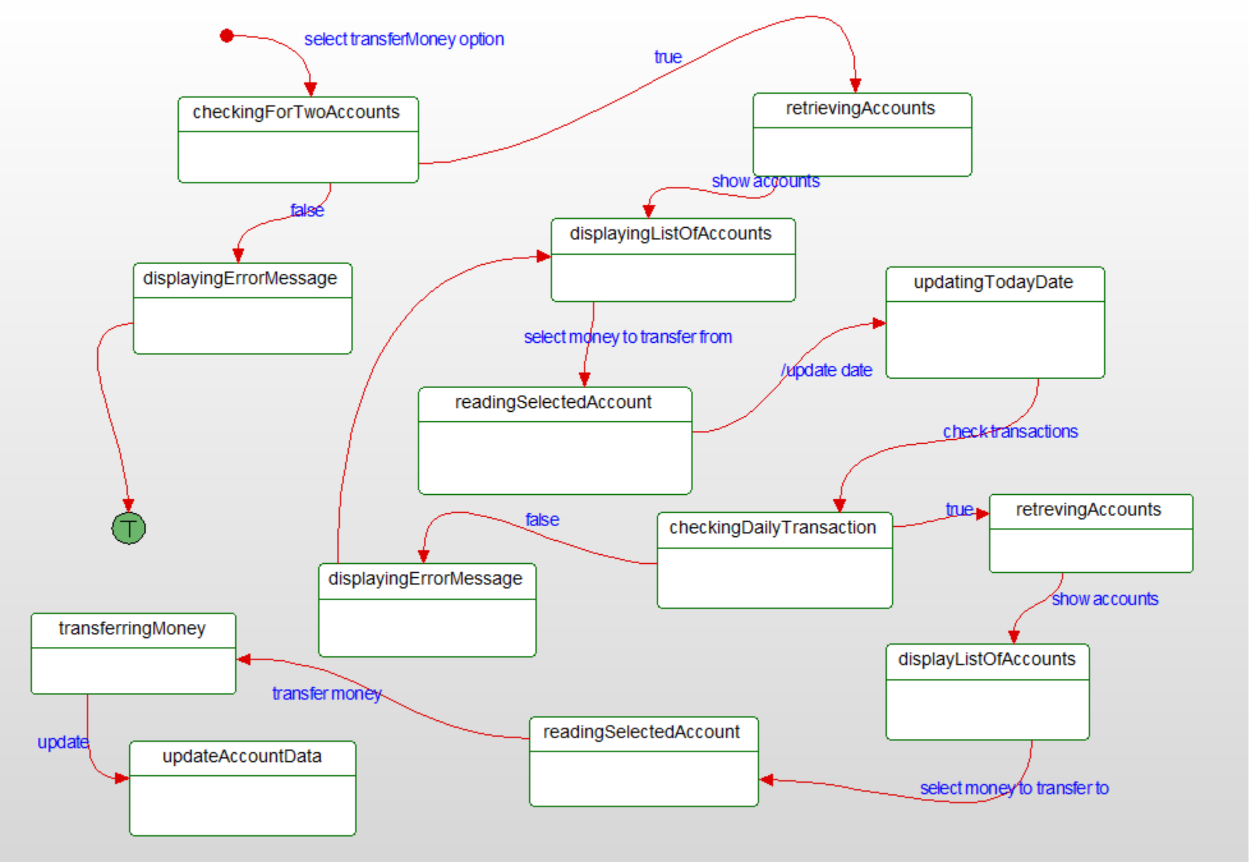


Figure 15 - State diagram for transferring money

This is the diagram that shows the different states of the system when the customer is transferring money. The system will also check for the daily transaction limit in this use case. There are 13 different system states associated with this use case.

VIII. Database Design

A. ER Diagram

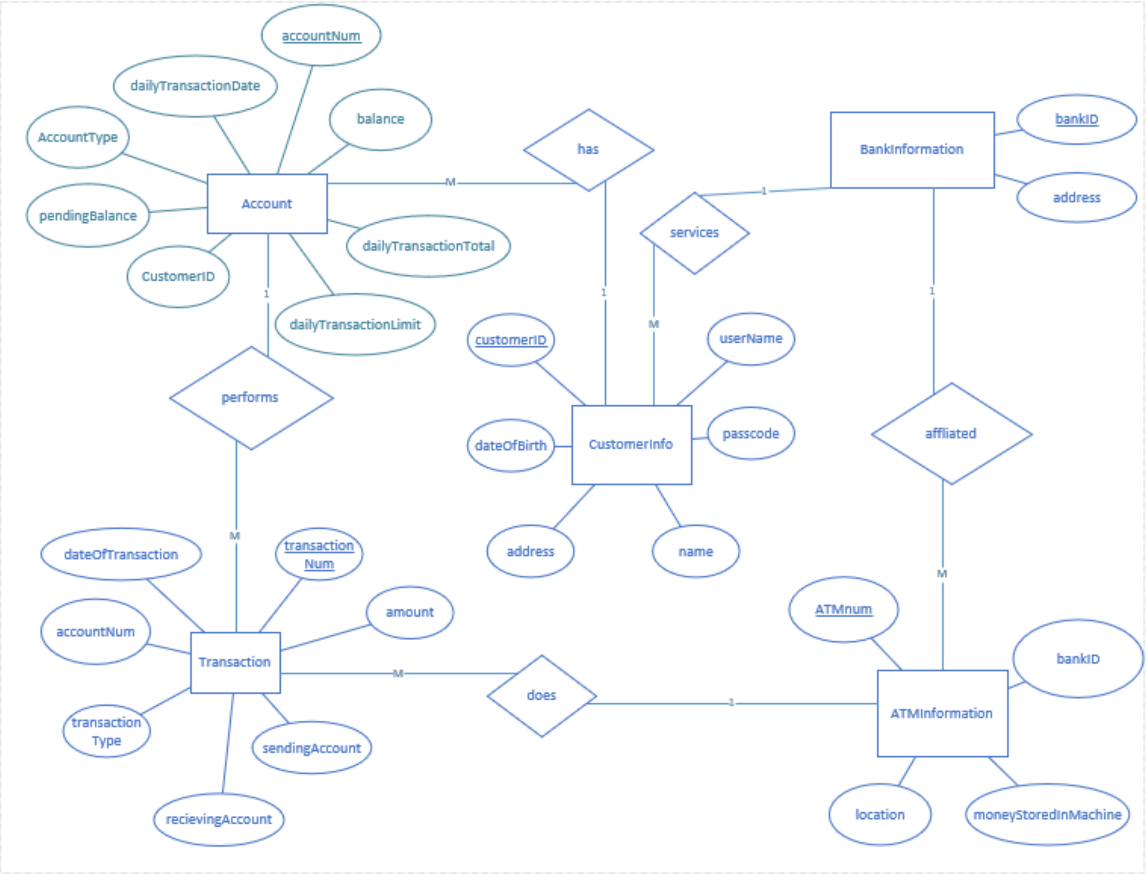
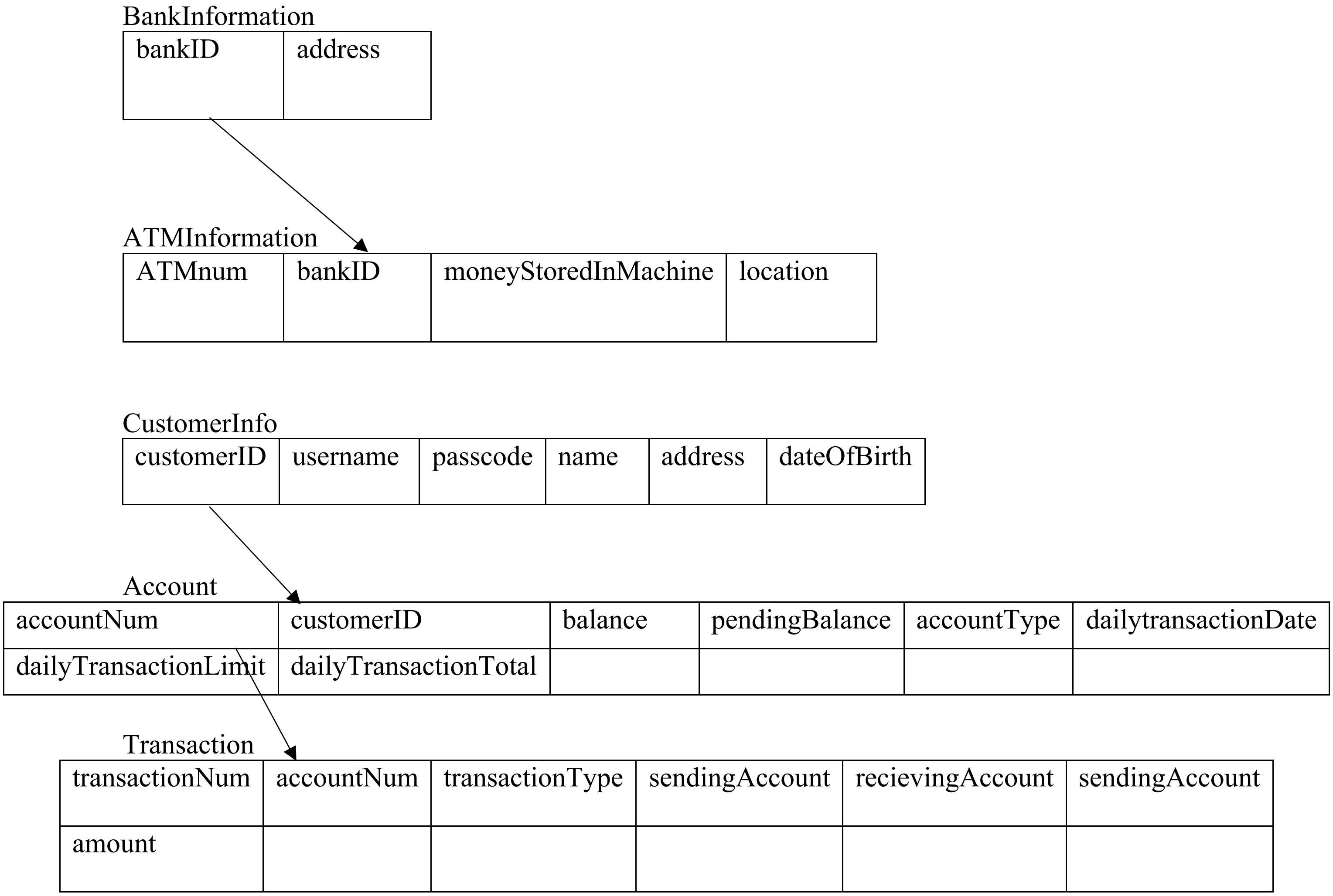


Figure 16 - ER Diagram for structuring the database

This is the ER diagram that is used to design the database. The primary entities for storing the information that the ATM will use will be in account, transaction, and customerInfo. The entities of ATMinformation and BankInformation are used in case the bank decides to add another ATM in the future. This will make it easier to add the next system and track it.

B. Table Schema



BankInformation

bankID	address

ATMInformation

ATMnum	bankID	moneyStoredInMachine	location

CustomerInfo

customerID	username	passcode	name	address	dateOfBirth

Account

accountNum	customerID	balance	pendingBalance	accountType	dailytransactionDate
dailyTransactionLimit	dailyTransactionTotal				

Transaction

transactionNum	accountNum	transactionType	sendingAccount	recievingAccount	sendingAccount
amount					

This is a visualization of the table schema within the database. The account table uses the accountNum to track the transactions that are done on each account. The account uses the customerID attribute of customerInfo to keep track of each account for each customer.

IX. Conclusion

This project consisted of being very through with all the needs of the client. The functionality of the system needed to be very detailed in order to set up and implement the logic and the code. The system needed to conduct a lot of error handling and also a lot of retrieving of information. There were many scenarios that had to be taken into account in order to provide the correct functionality. The sequence, state, and activity diagrams help us plan for these kinds of scenarios and how to go about testing for them.

The graphical user interface was designed so the user could easily see what services the ATM provided. It was also set up in a way where most interaction and alerts would be on the right side of the page given that most people are right-handed.

The database implementation was done using heidiSQL, and the front-end, along with the communication to the server was done in c#.

X. Data Dictionary

Activity Diagram – describes parallel and conditional activities, use cases, and system functions at a detailed level.
Attribute – a specification that defines a property of an object, element, or file.
Class – blueprint or a set of instructions to build a specific type of object
Entity (in database) – is a thing, person, place, unit, object or any item about which the data should be captured and stored in the form of properties, workflows and tables.
ER Diagram – An entity relationship diagram shows the relationships of entity sets stored in a database
Function – part of a program that has its own name. This name can be used in the program as a command. When a function is called, the commands of which it consists are executed.
Operation – an action that is carried out to accomplish a given task
Sequence Diagram – type of interaction diagram because it describes how and in what order a group of objects works together
Use Case – a description of how users will perform tasks on a system or software
State diagram – describes the behavior or status that the system may be in. It consists of states, transitions, events, and activities the system may perform