Banking System

Requirement:

Group 25

Author: 王子瑜 15355832

Main requirements:

R1 : Open Account

The customer can open a account at the bank counter providing his personID and newPassword. The cashier must be able to handle the open accounts requirement of the customer. After the account has been created successfully, the cashier can see the accountID generated by the banking system and give it to the customer.

R2 : Close Account

The customer can close his account at the bank counter providing his personID, accountID, password. The cashier must be able to handle the close accounts requirement of the customer.

R3 : Change Password

The customer can change his account password at the bank counter providing his accountID, his password and the newPassword. The cashier must be able to handle the change password requirement of the customer.

R4 : Desposit

The customer can deposit his money at the bank counter providing his accountID. No password is needed. The cashier must be able to handle the deposit requirement of the customer.

R5 : Withdraw

The customer can withdraw his money at the bank counter providing his accountID and password. The cashier must be able to handle the withdraw requirement of the customer.

R6 : Transfer

The customer can withdraw his money at the bank counter providing his accountID and password and the receiver’s accountID. The cashier must be able to handle the transfer requirement of the customer.

R7: Get a ticket

The customer can get a ticket from the ticket machine, It will print a ticket with the ticket number.

R8: Display next customer

When the current customer finishs his requirement at the bank counter, the cashier can display the next customer’ s ticket number, prompt the next customer to come to the bank counter.

R9 : ATM deposit

The customer can deposit his money at the bank ATM providing his accountID. No password is needed.

R10 : ATM withdraw

The customer can withdraw his money at the bank ATM providing his accountID and password.

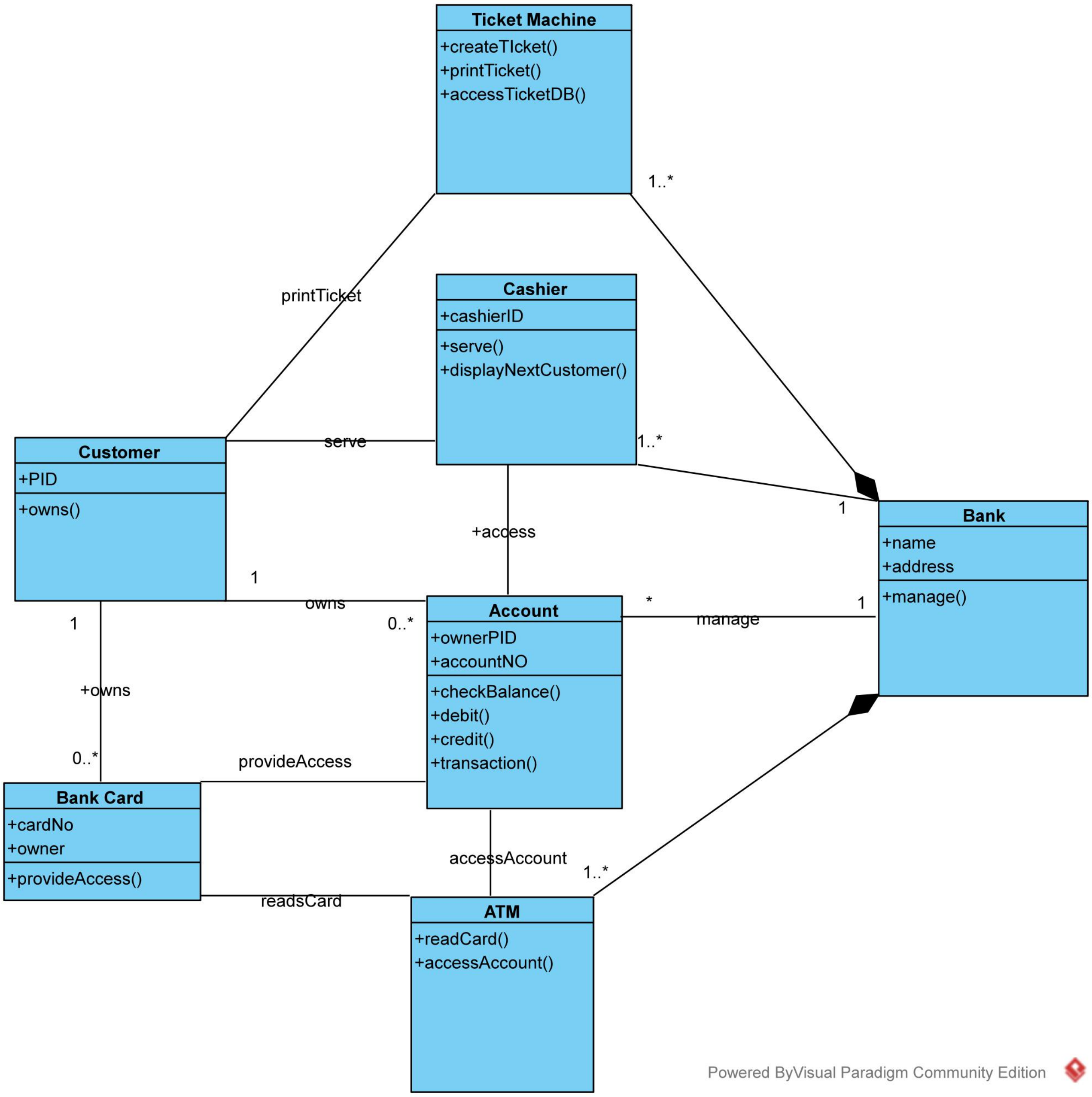
Part 1: Domain analysis

Main class:

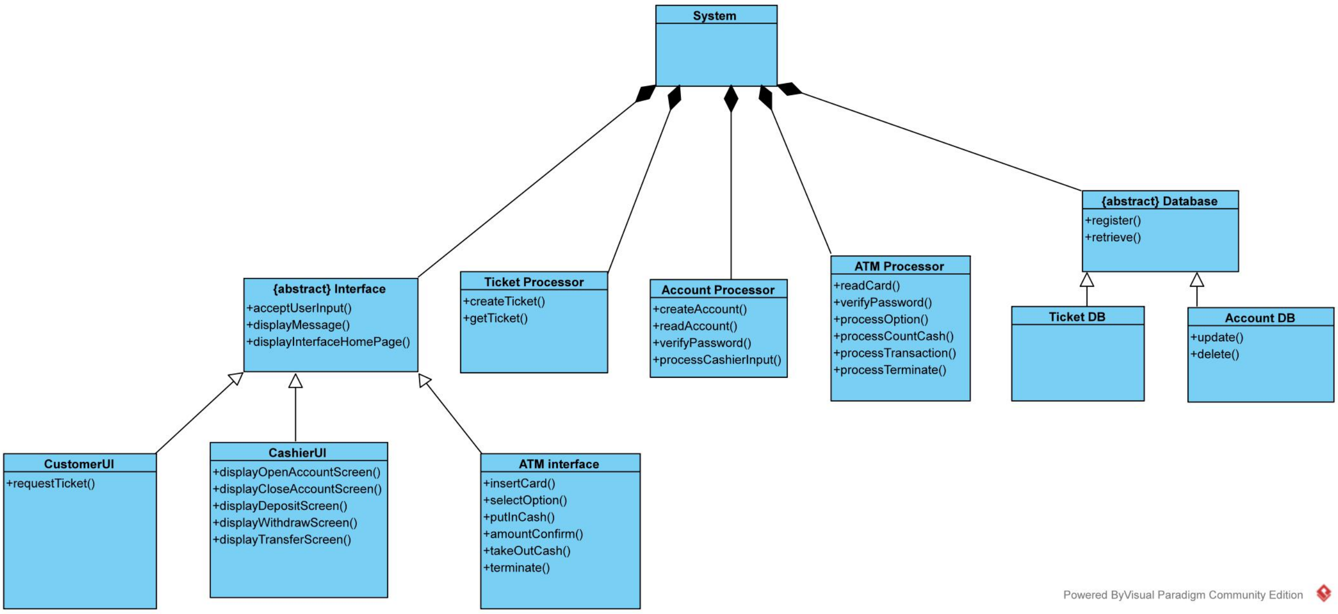
1. Customer: each customer has a unique personID, and can have several accounts and bank cards. Basically, the customers use the banking system for deposit, withdrawal and transfer the money.
2. Cashier: the person who serves the customer to deposit or withdraw in person, transfer money to other people, and open or close the account.
3. Account: Including checking account and saving account, can be access via the bank cards. Have several transactions such as withdraw, deposit and transfer.
4. Bank: a bank has a acount database which contains all account data and some bank machines. Such as ticket machine, used for customer to get the ticket and wait in line; ATM, allows the customers to deposit or withdraw without the cashiers.

Main interface (in part2):

1. Customer UI for geting the ticket when waiting in bank.
2. Cashier UI for displaying next customer, depositing & withdrawing in person, transforing money to other people. And open or close the account.
3. ATM interface: for customers to deposit or withdraw cash.

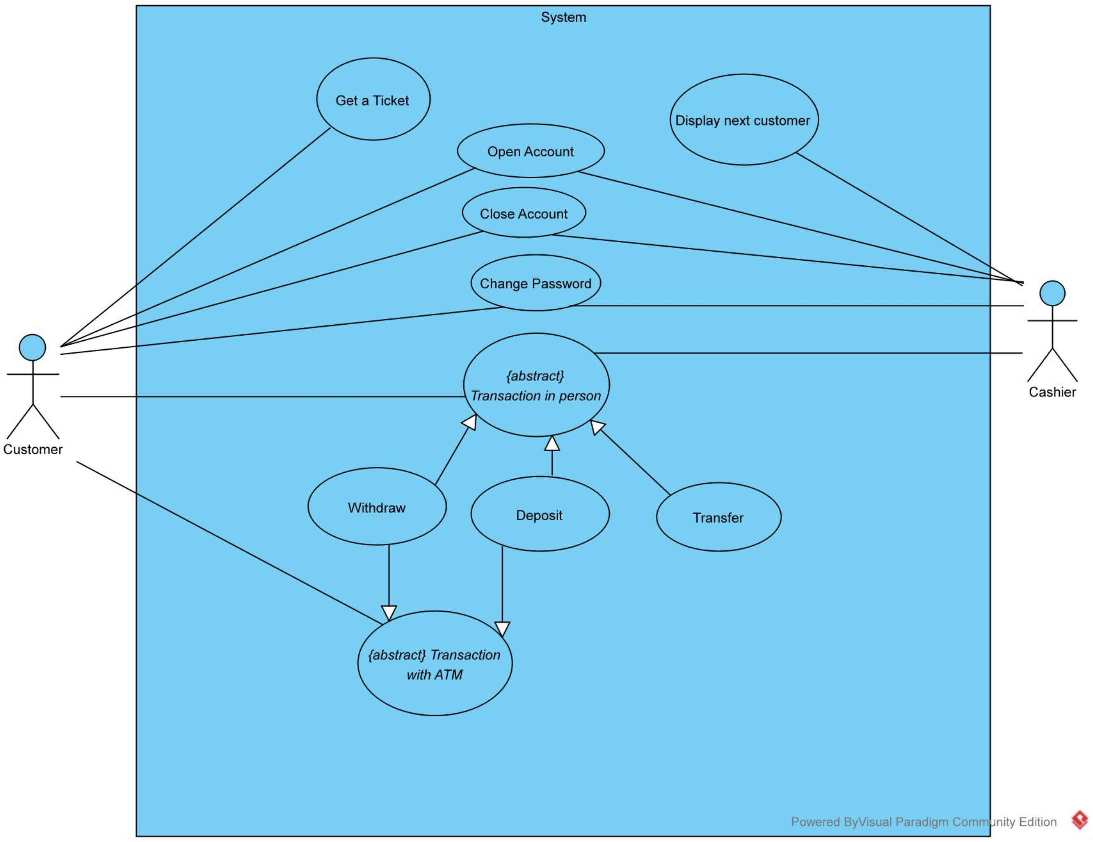

Ticket Machine
+createTIcket()
+printTicket()
+accessTicketDB()
1..*
printTicket
Cashier
+cashierID
+serve()
+displayNextCustomer()
1..*
Customer
+PID
+owns()
serve
1
+access
Bank
+name
+address
+manage()
1
owns
0..*
Account
+ownerPID
+accountNO
+checkBalance()
+debit()
+credit()
+transaction()
*
manage
1
1
+owns
0..*
provideAccess
Bank Card
+cardNo
+owner
+provideAccess()
accessAccount
1..*
readsCard
ATM
+readCard()
+accessAccount()

Part 2 : System Composition



The System is consist of three inferface, three processors and two databases. The following use case diagrams and sequence Diagrams will show how this system works for the requirements. Note that, in the CustomerUI and ATM interface, all the operations are also screen displays.

Here we list the basic use cases. Then we will show each use case with the system classes.



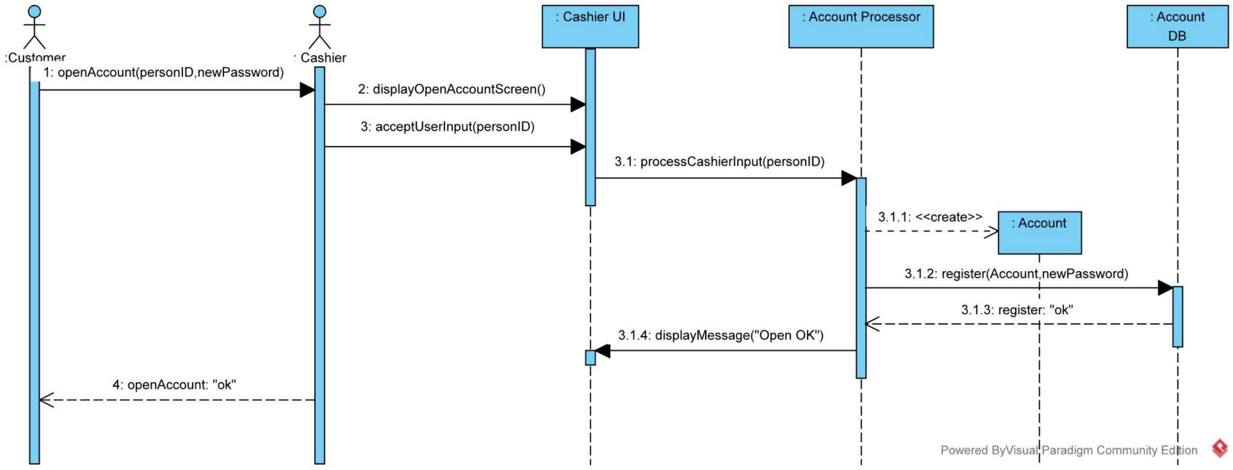
In our banking system, one customer can have many accounts in the bank, each account has one bank card to access. To simplify the relationship between account and bank card, we set the accountID is equal to the bank cardID, such that most of the operations in the bank can be done with the accountID.

R1: Open Account:

When the customer wants to open a account, he must request the cashier in person. His personID is needed for creating the account by the account processor. After the account is create, the account processor will register the account to the account database with the newPassword.

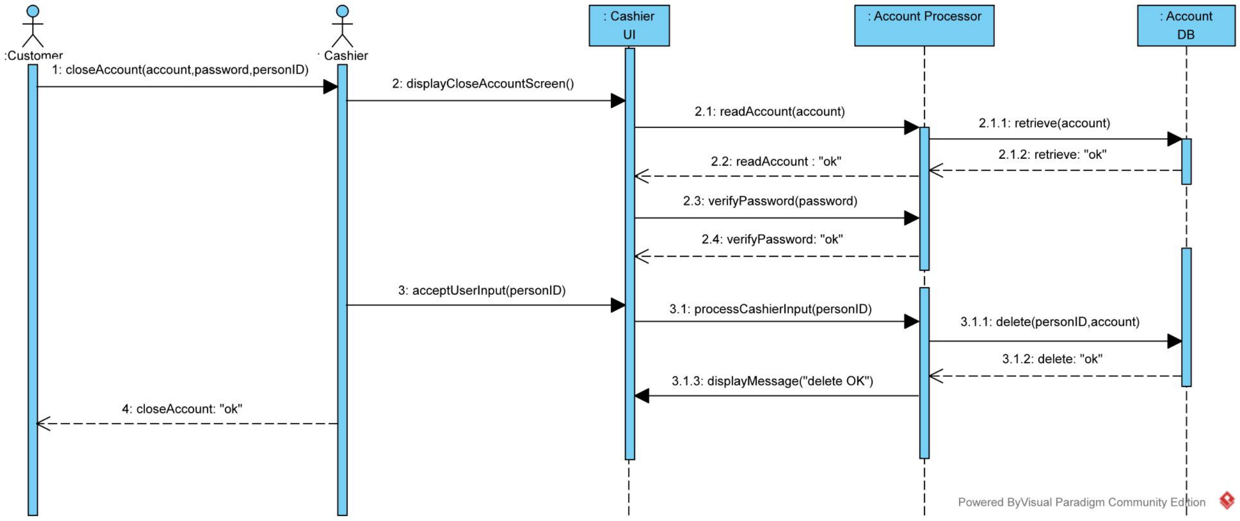
The customer can give the cashier a bank card actually. But to simpfy the graph, we use account to represent the bank card or the bank book since they just provides the access to the actual account.

The newPassword is entered by the customer actually. But to simpfy the graph, we just show that the newPassword is tranfered via the cashier. Since the graph is mainly to show the use case via the cashier UI. The following graphs are also the same.


:Customer
: Cashier
: Cashier UI
: Account Processor
: Account DB
1: openAccount(personID,newPassword)
2: displayOpenAccountScreen()
3: acceptUserInput(personID)
3.1: processCashierInput(personID)
3.1.1: <<create>>
: Account
3.1.2: register(Account,newPassword)
3.1.3: register: "ok"
3.1.4: displayMessage("Open OK")
4: openAccount: "ok"
Powered ByVisual Paradigm Community Edition

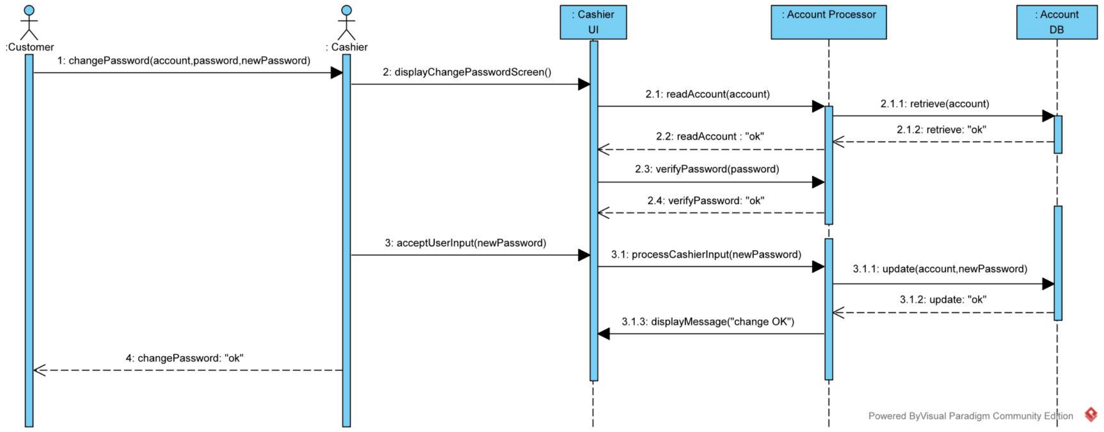
R2: Close Account:

When the customer wants to close the account , the account processor will first verify the password the account, then the personID is needed for the account Database to delete the according record. If is not matching, delete will fail.


:Customer
: Cashier
: Cashier UI
: Account Processor
: Account DB
1: closeAccount(account,password,personID)
2: displayCloseAccountScreen()
2.1: readAccount(account)
2.1.1: retrieve(account)
2.1.2: retrieve: "ok"
2.2: readAccount : "ok"
2.3: verifyPassword(password)
2.4: verifyPassword: "ok"
3: acceptUserInput(personID)
3.1: processCashierInput(personID)
3.1.1: delete(personID,account)
3.1.2: delete: "ok"
3.1.3: displayMessage("delete OK")
4: closeAccount: "ok"
Powered ByVisual Paradigm Community Edition

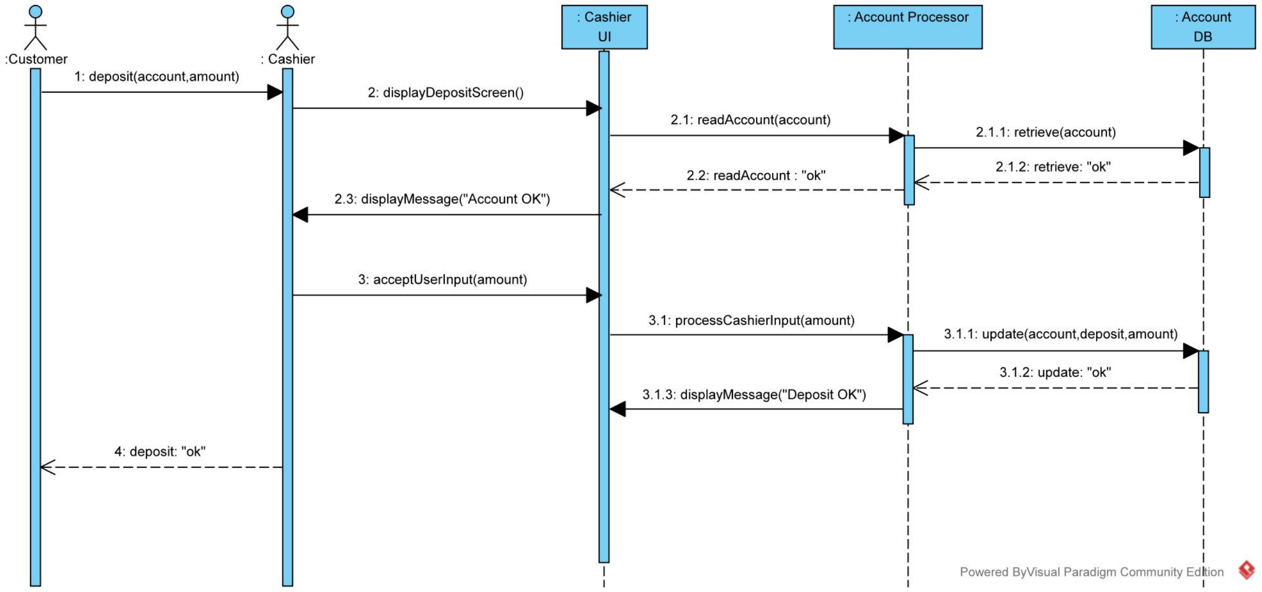
R3: Change password:

The password(old) is needed for change password in person. After verifying that the password(old) is correct, the cashierUI accepts the userInput of the newPassword, account process it , then update the Account DB.


:Customer
: Cashier
: Cashier UI
: Account Processor
: Account DB
1: changePassword(account,password,newPassword)
2: displayChangePasswordScreen()
2.1: readAccount(account)
2.1.1: retrieve(account)
2.1.2: retrieve: "ok"
2.2: readAccount : "ok"
2.3: verifyPassword(password)
2.4: verifyPassword: "ok"
3: acceptUserInput(newPassword)
3.1: processCashierInput(newPassword)
3.1.1: update(account,newPassword)
3.1.2: update: "ok"
3.1.3: displayMessage("change OK")
4: changePassword: "ok"
Powered ByVisual Paradigm Community Edition

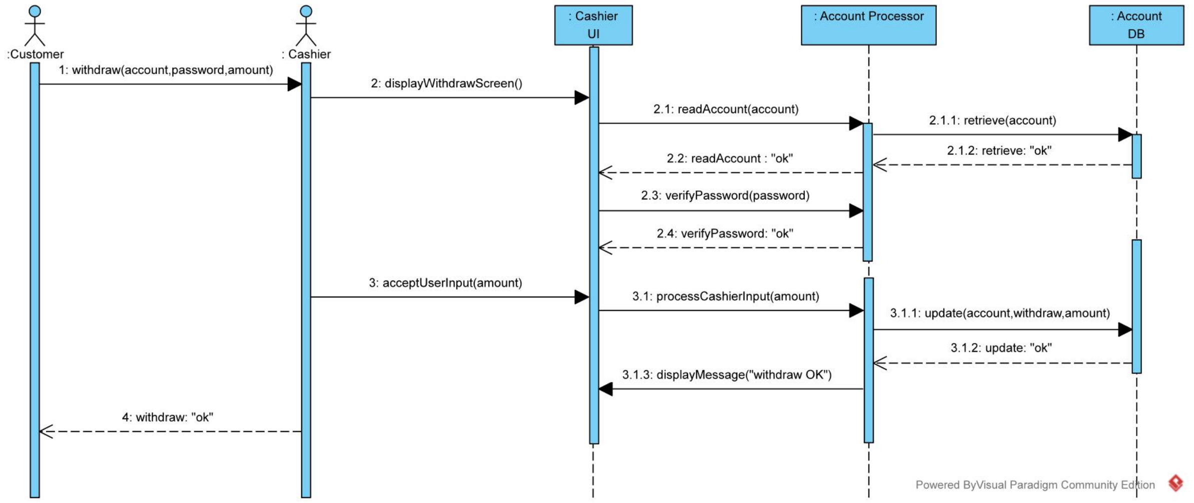
R4: Deposit in person:

The password is not needed for deposit in person, the customer can just give the account and the money to cashier for depositing. Here we use amount to represent the amount of money is given to the cashier.


:Customer
: Cashier
: Cashier UI
: Account Processor
: Account DB
1: deposit(account,amount)
2: displayDepositScreen()
2.1: readAccount(account)
2.1.1: retrieve(account)
2.1.2: retrieve: "ok"
2.2: readAccount : "ok"
2.3: displayMessage("Account OK")
3: acceptUserInput(amount)
3.1: processCashierInput(amount)
3.1.1: update(account,deposit,amount)
3.1.2: update: "ok"
3.1.3: displayMessage("Deposit OK")
4: deposit: "ok"
Powered ByVisual Paradigm Community Edition

R5: Withdraw

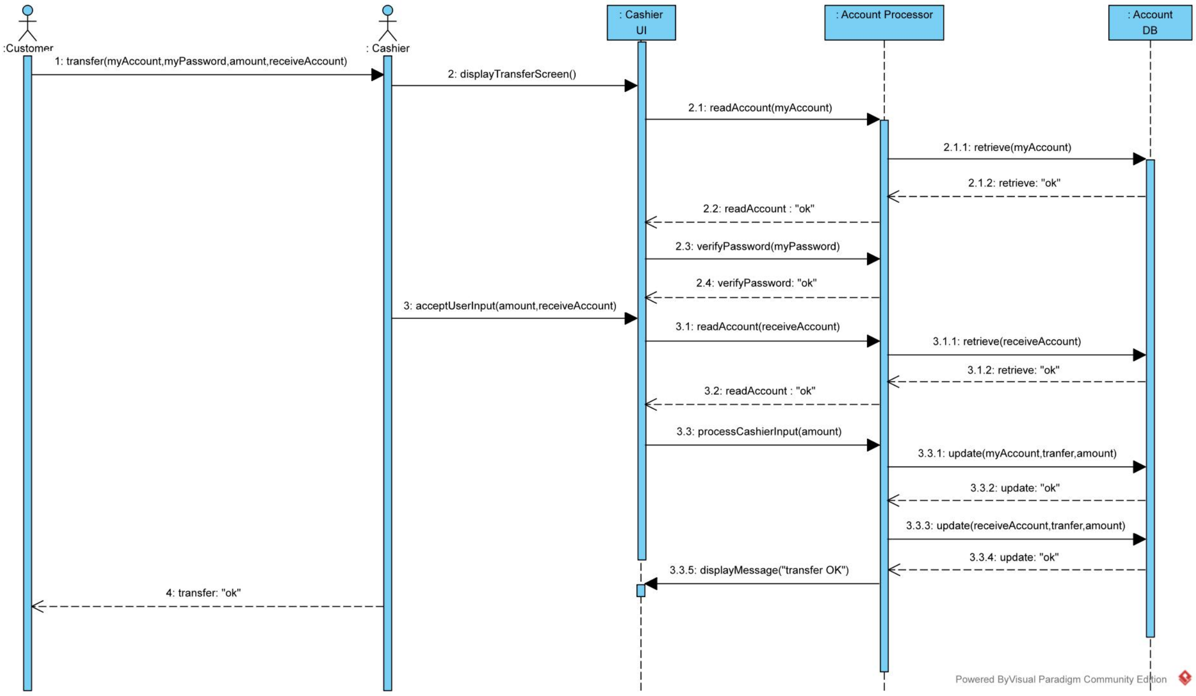
The password is needed for withdraw in person. And if there is no enough money in the account to withdraw , the account update() will fail.


:Customer
: Cashier
: Cashier UI
: Account Processor
: Account DB
1: withdraw(account,password,amount)
2: displayWithdrawScreen()
2.1: readAccount(account)
2.1.1: retrieve(account)
2.1.2: retrieve: "ok"
2.2: readAccount : "ok"
2.3: verifyPassword(password)
2.4: verifyPassword: "ok"
3: acceptUserInput(amount)
3.1: processCashierInput(amount)
3.1.1: update(account,withdraw,amount)
3.1.2: update: "ok"
3.1.3: displayMessage("withdraw OK")
4: withdraw: "ok"
Powered ByVisual Paradigm Community Edition

R6: Transfer

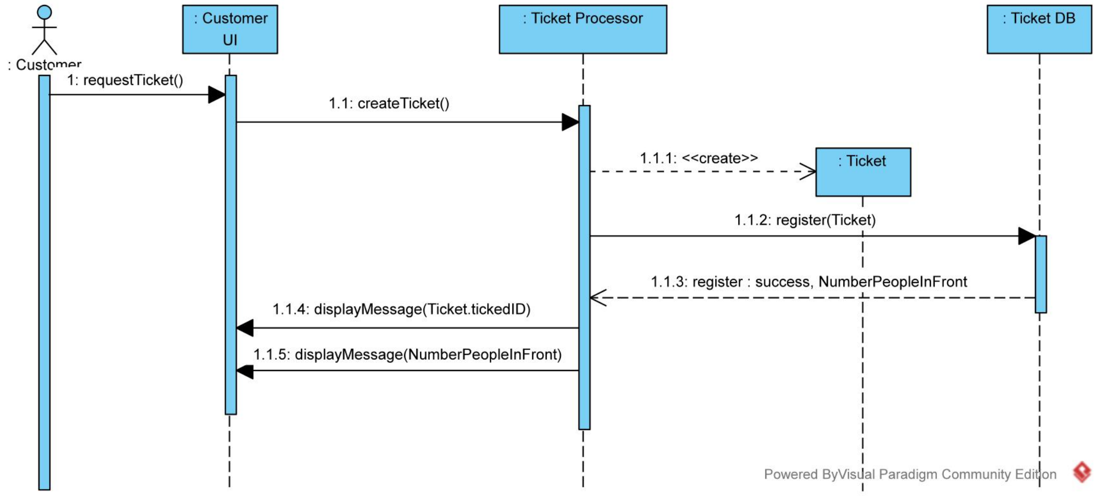
Transfer to other people in person needs the customer's account and the receiver's account. After verify the password of the customer, retrieve(receriveAccount) here will justify the existence of the receiver's account.

Then we deduct the tranfer money from the customer's account, add it to the receiver's account. If customer's account don't have the enough money, the update() will fail.


:Customer
: Cashier
: Cashier UI
: Account Processor
: Account DB
1: transfer(myAccount,myPassword,amount,receiveAccount)
2: displayTransferScreen()
2.1: readAccount(myAccount)
2.1.1: retrieve(myAccount)
2.1.2: retrieve: "ok"
2.2: readAccount : "ok"
2.3: verifyPassword(myPassword)
2.4: verifyPassword: "ok"
3: acceptUserInput(amount,receiveAccount)
3.1: readAccount(receiveAccount)
3.1.1: retrieve(receiveAccount)
3.1.2: retrieve: "ok"
3.2: readAccount : "ok"
3.3: processCashierInput(amount)
3.3.1: update(myAccount,tranfer,amount)
3.3.2: update: "ok"
3.3.3: update(receiveAccount,tranfer,amount)
3.3.4: update: "ok"
3.3.5: displayMessage("transfer OK")
4: transfer: "ok"
Powered ByVisual Paradigm Community Edition

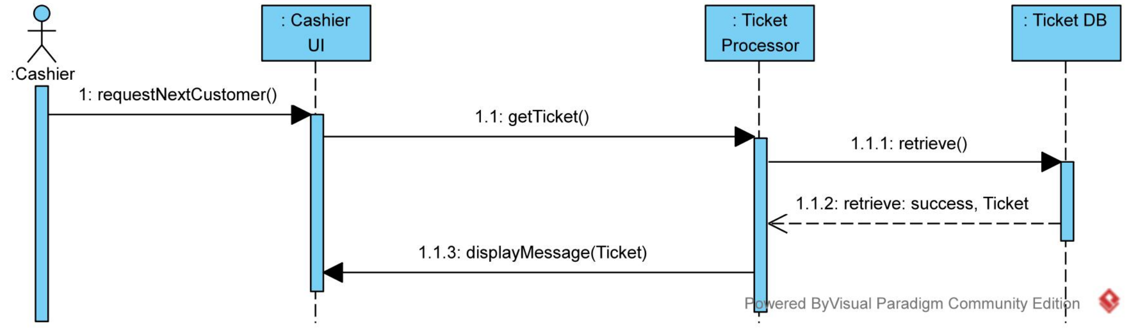
R7: Get a ticket

A customer request a ticket when waiting in the bank. After he press the requestTicket() in the customer UI, the ticket processor will create a ticket based on the current time, then register the ticket to the ticket database, then display the ticketID and the number of people are waiting in front the customer.


: Customer
: Customer UI
: Ticket Processor
: Ticket DB
: Ticket
1: requestTicket()
1.1: createTicket()
1.1.1: <<create>>
1.1.2: register(Ticket)
1.1.3: register : success, NumberPeopleInFront
1.1.4: displayMessage(Ticket.tickedID)
1.1.5: displayMessage(NumberPeopleInFront)
Powered ByVisual Paradigm Community Edition

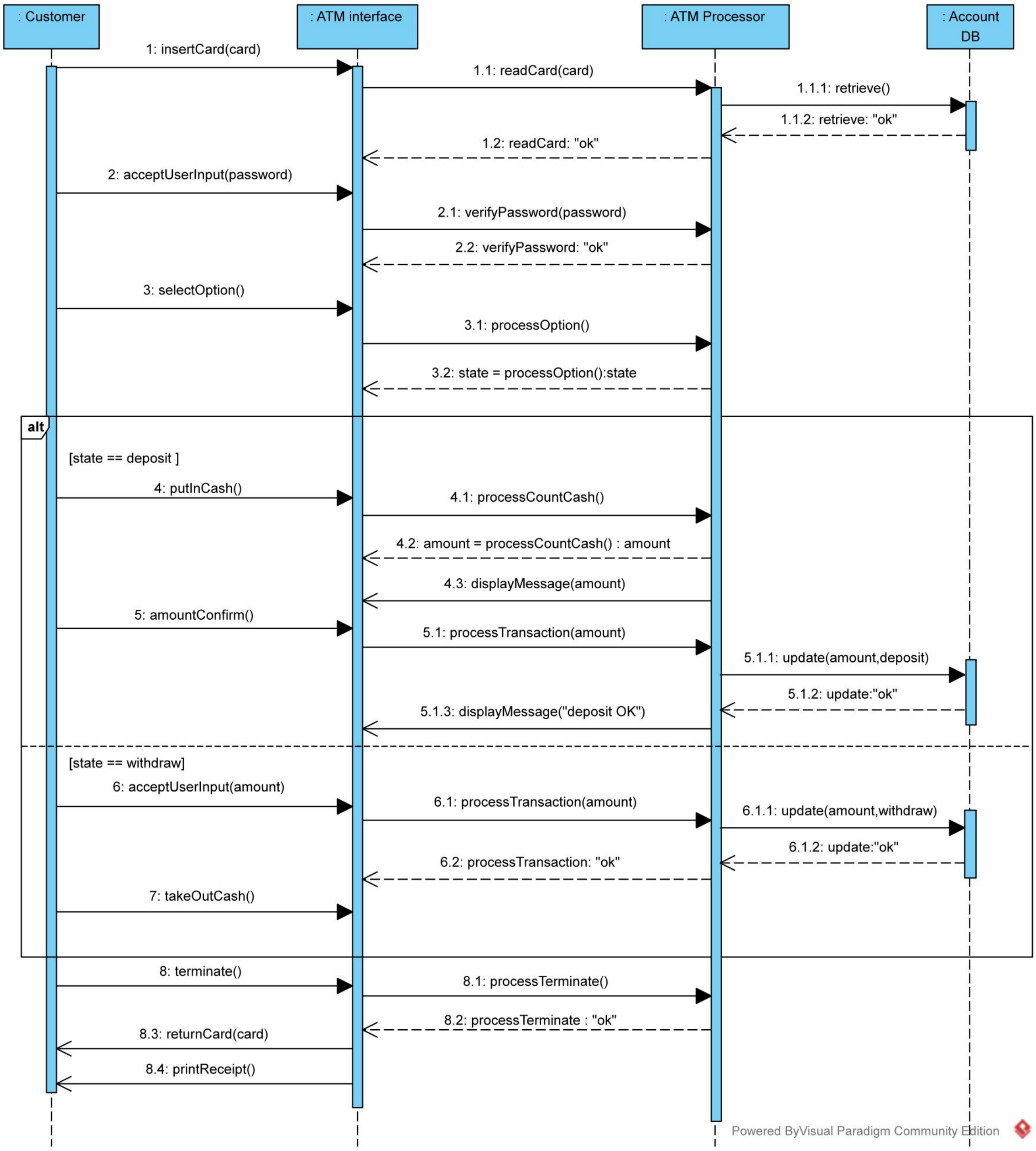
R8: Display next customer

When the cashier wants to display next customer, he can requestNextCustomer() on the cashier UI, then the ticket process will retrieve the next customer's ticket from the database, then displays the ticket message to the cashier.



R9,10 : ATM deposit and tranfer

When the customer use the ATM to deposit or withdraw the money, the bank card and the password are needed. After verifying the password, the customer can select the option to deposit or withdraw. After that, the customer can continue to select the option on the ATM interface(here we omit it in the graph). If the customer wants to finish, he have to use the terminate() on the ATM interface, then the ATM will return his bank card and print the receipt.


: Customer
: ATM interface
: ATM Processor
: Account DB
1: insertCard(card)
1.1: readCard(card)
1.1.1: retrieve()
1.1.2: retrieve: "ok"
1.2: readCard: "ok"
2: acceptUserInput(password)
2.1: verifyPassword(password)
2.2: verifyPassword: "ok"
3: selectOption()
3.1: processOption()
3.2: state = processOption():state
alt
[state == deposit]
4: putInCash()
4.1: processCountCash()
4.2: amount = processCountCash() : amount
4.3: displayMessage(amount)
5: amountConfirm()
5.1: processTransaction(amount)
5.1.1: update(amount,deposit)
5.1.2: update:"ok"
5.1.3: displayMessage("deposit OK")
[state == withdraw]
6: acceptUserInput(amount)
6.1: processTransaction(amount)
6.1.1: update(amount,withdraw)
6.1.2: update:"ok"
6.2: processTransaction: "ok"
7: takeOutCash()
8: terminate()
8.1: processTerminate()
8.2: processTerminate : "ok"
8.3: returnCard(card)
8.4: printReceipt()
Powered ByVisual Paradigm Community Edition