***EXERCISE 4***🡪 <https://cmps-people.ok.ubc.ca/bowenhui/310/8-UML.pdf>

**PROMPT:** The following should be textually analyzed and a use case diagram created containing several use cases. Identify the actors, use cases and associations. Please, use the user goal level approach. Also, please consider any possible generalization relationship between use cases or between actors, and any possible "include" or "extend" relationship between use cases. Please give me the PlantUML code for the use case diagram corresponding to the following text:

***This case study concerns a simplified system of the automatic teller machine (ATM). The ATM offers the following services: 1. Distribution of money to every holder of a smartcard via a card reader and a cash dispenser. 2. Consultation of account balance, cash and cheque deposit facilities for bank customers who hold a smartcard from their bank. Do not forget either: 3. All transactions are made secure. 4. It is sometimes necessary to refill the dispenser, etc. From these four sentences, we will work through the following activities: • Identify the actors, • Identify the use cases, •Construct a use case diagram, •Organize and structure the use cases.***

***What are the external entities that interact directly with the ATM? Let’s look at each of the sentences of the exposition in turn. Sentence 1 allows us to identify an obvious initial actor straight away: every “holder of a smartcard”. He or she will be able to use the ATM to withdraw money using his or her smartcard. However, be careful: the card reader and cash dispenser constitute part of the ATM. They can therefore not be considered as actors! You can note down that the identification of actors requires the boundary between the system being studied and its environment to be set out exactly. If we restrict the study to the control/ command system of physical elements of the ATM, the card reader and cash dispenser then become actors. Another trap: is the smartcard itself an actor? The card is certainly external to the ATM, and it interacts with it... Yet, we do not recommend that you list it as an actor, as we are putting into practice the following principle: eliminate “physical” actors as much as possible to the advantage of “logical” actors. The actor is the who or what that benefits from using the system. It is the card holder who withdraws money to spend it, not the card itself! Sentence 2 identifies additional services that are only offered to bank customers who hold a smartcard from this bank. This is therefore a different profile from the previous one, which we will realize by a second actor called Bank customer. Sentence 3 encourages us to take into account the fact that all transactions are made secure. But who makes them secure? There are therefore other external entities, which play the role of authorisation system and with which the ATM communicates directly. An interview with the domain expert is necessary to allow us to identify two different actors: • the Visa authorisation system (VISA AS) for withdrawal transactions carried out using a Visa smartcard (we restrict the ATM to Visa smartcards for reasons of simplification); • the information system of the bank (Bank IS) to authorize all transactions carried out by a customer using his or her bank smartcard, but also to access the account balance. Finally, sentence 4 reminds us that an ATM also requires maintenance work, such as refilling the dispenser with bank notes, retrieving cards that have been swallowed, etc. These maintenance tasks are carried out by a new actor, which – to simplify matters – we will call the Maintenance operator.***

ChatGPT

**ChatGPT**

Based on the information provided, here's the textual analysis and the corresponding PlantUML code for the use case diagram:

**Actors:**

1. Holder of a Smartcard
2. Bank Customer
3. Visa Authorization System (VISA AS)
4. Bank Information System (Bank IS)
5. Maintenance Operator

**Use Cases:**

1. Withdraw Money
2. Consult Account Balance
3. Deposit Cash
4. Deposit Cheque
5. Secure Transactions
6. Refill Dispenser
7. Retrieve Swallowed Card

**Associations:**

* Holder of a Smartcard (Actor) interacts with:
  + Withdraw Money (Use Case)
  + Consult Account Balance (Use Case)
  + Deposit Cash (Use Case)
  + Deposit Cheque (Use Case)
  + Secure Transactions (Use Case)
* Bank Customer (Actor) interacts with:
  + Consult Account Balance (Use Case)
* Visa Authorization System (VISA AS) (Actor) interacts with:
  + Withdraw Money (Use Case)
* Bank Information System (Bank IS) (Actor) interacts with:
  + Consult Account Balance (Use Case)
  + Secure Transactions (Use Case)
* Maintenance Operator (Actor) interacts with:
  + Refill Dispenser (Use Case)
  + Retrieve Swallowed Card (Use Case)

**Generalization Relationships:**

* Withdraw Money (Use Case) and Consult Account Balance (Use Case) can be generalized under a common Use Case named "Basic Transaction."

Here's the PlantUML code:

@startuml

left to right direction

actor "Holder of a Smartcard" as Holder

actor "Bank Customer" as BankCustomer

actor "Visa Authorization System (VISA AS)" as VisaAS

actor "Bank Information System (Bank IS)" as BankIS

actor "Maintenance Operator" as Operator

rectangle "ATM" {

usecase "Withdraw Money" as Withdraw

usecase "Consult Account Balance" as ConsultBalance

usecase "Deposit Cash" as DepositCash

usecase "Deposit Cheque" as DepositCheque

usecase "Secure Transactions" as SecureTransactions

usecase "Refill Dispenser" as RefillDispenser

usecase "Retrieve Swallowed Card" as RetrieveCard

}

Holder --> Withdraw

Holder --> ConsultBalance

Holder --> DepositCash

Holder --> DepositCheque

Holder --> SecureTransactions

BankCustomer --> ConsultBalance

VisaAS --> Withdraw

BankIS --> ConsultBalance

BankIS --> SecureTransactions

Operator --> RefillDispenser

Operator --> RetrieveCard

Withdraw <|.. ConsultBalance : <<generalization>>

@enduml

**PROMPT:** Thank you. but I think that Visa AS and Bank IS should be clearly identified as secondary actors, and you should consider that a cardholder can only withdraw money, cannot deposit and cannot consult balance, on the other hand, the bank customer is able to do everything (deposit, consult balance and withdraw cash). Also, the use case "secure transactions'' should not exist, since you are confusing it with the fact that they are secured by the visa AS and bank IS.

ChatGPT

**ChatGPT**

Certainly, I appreciate your feedback. Here's the revised PlantUML code considering the provided clarifications:

@startuml

left to right direction

actor "Cardholder" as Cardholder

actor "Bank Customer" as BankCustomer

actor "Maintenance Operator" as Operator

rectangle "ATM" {

usecase "Withdraw Money" as Withdraw

usecase "Deposit Cash" as DepositCash

usecase "Deposit Cheque" as DepositCheque

}

rectangle "Authorization System" {

usecase "Authorize Visa Transaction" as AuthorizeVisa

usecase "Authorize Bank Transaction" as AuthorizeBank

}

rectangle "Maintenance" {

usecase "Refill Dispenser" as RefillDispenser

usecase "Retrieve Swallowed Card" as RetrieveCard

}

Cardholder --> Withdraw

Cardholder --> DepositCash

Cardholder --> DepositCheque

BankCustomer --> Withdraw

BankCustomer --> DepositCash

BankCustomer --> DepositCheque

AuthorizeVisa --> Withdraw

AuthorizeBank --> Withdraw

Operator --> RefillDispenser

Operator --> RetrieveCard

@enduml

**PROMPT:** You are still considering that a cardholder can deposit, but I told you that the cardholder is only able to withdraw cash, while the bank customer is the one who is able to do everything. Apart from this, the bank IS is a secondary actor that should support deposits and consulting balance, while the visa AS is a secondary actor that only supports withdrawing money.

ChatGPT

**ChatGPT**

I appreciate the clarification. Here's the revised PlantUML code based on your feedback:

@startuml

left to right direction

actor "Cardholder" as Cardholder

actor "Bank Customer" as BankCustomer

actor "Maintenance Operator" as Operator

actor "Bank Information System" as BankIS

actor "Visa Authorization System" as VisaAS

rectangle "ATM" {

usecase "Withdraw Cash" as Withdraw

}

rectangle "Bank Services" {

usecase "Deposit Cash" as DepositCash

usecase "Deposit Cheque" as DepositCheque

usecase "Consult Balance" as ConsultBalance

}

rectangle "Authorization Systems" {

usecase "Authorize Visa Transaction" as AuthorizeVisa

usecase "Authorize Bank Transaction" as AuthorizeBank

}

rectangle "Maintenance" {

usecase "Refill Dispenser" as RefillDispenser

usecase "Retrieve Swallowed Card" as RetrieveCard

}

Cardholder --> Withdraw

BankCustomer --> Withdraw

BankCustomer --> DepositCash

BankCustomer --> DepositCheque

BankCustomer --> ConsultBalance

BankIS --> DepositCash

BankIS --> ConsultBalance

VisaAS --> Withdraw

Operator --> RefillDispenser

Operator --> RetrieveCard

@enduml

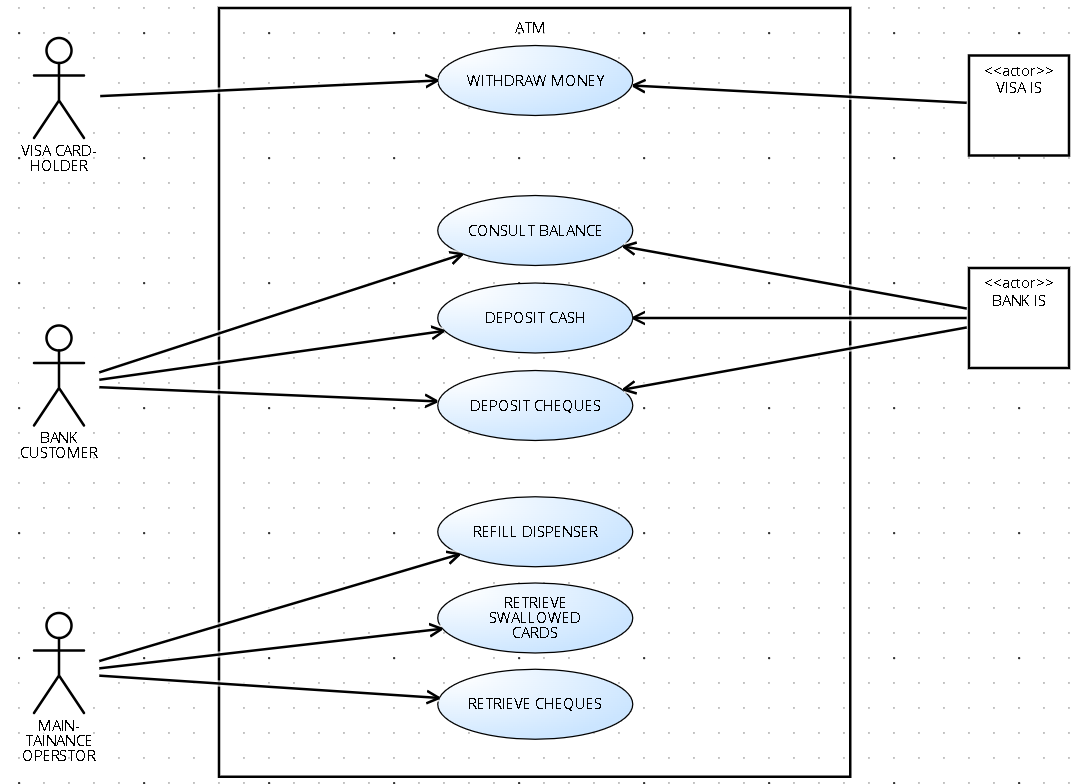
**Analysis:**

This exercise, featuring an extensive textual description, required three prompts before achieving a refined outcome. The initial code generated exhibited a few undesirable issues, despite the Use Case Diagram (UCD) being fairly satisfactory. One notable concern was that ChatGPT confused the use cases applicable to cardholders and bank customers, erroneously attributing capabilities such as deposits and balance inquiries to cardholders, which were intended only for bank customers. While this error is relatively minor, given the length of the exercise, it could introduce confusion. Fortunately, this discrepancy was rectified in the final code.

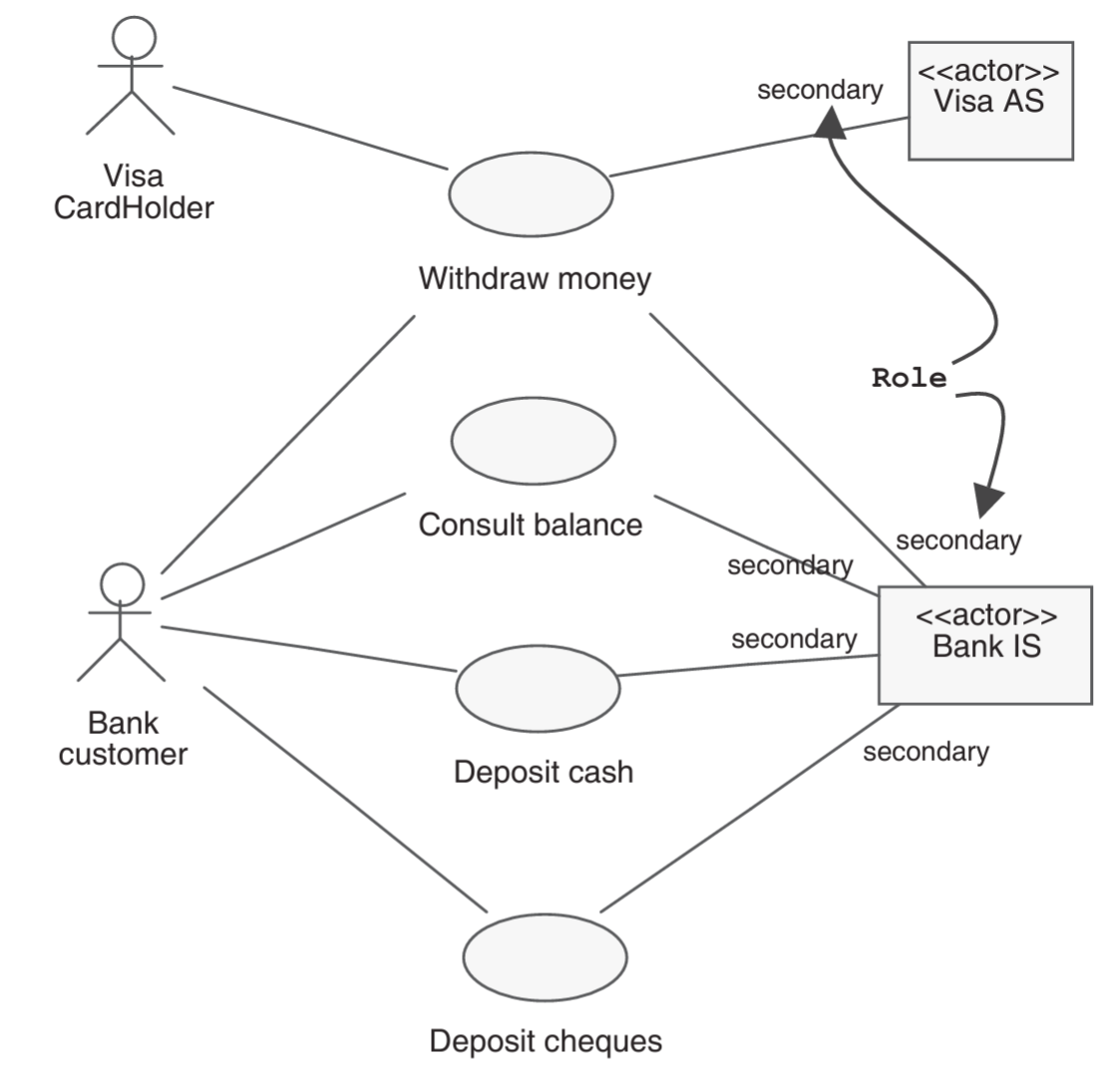
Another inconvenience arose when ChatGPT included a use case for "secure transactions" associated with the bank customer, which was unnecessary and inconsistent with the text's description. The text explicitly indicated that transactions were secured through the VISA Authorization System and Bank Information System. In the final code, this discrepancy was appropriately addressed.

The last notable point is that both the VISA Authorization System and the Bank Information System are secondary actors. In the initial code generated, this distinction wasn't clarified. Nevertheless, considering the extensive description, it can be acknowledged that if the user possesses an understanding of Use Case Diagram (UCD) design, ChatGPT proves to be highly beneficial in drafting an initial code approach. This initial draft can then be refined further by the user.

**MY SOLUTION:**

****

**REFERENCE SOLUTION:**

****

**LLM SOLUTION:**

