Montana Savings Bank ATM:
Complete Report

Jake Balvin & Nicholas Girod
Software Engineering
Professor Ku

Table of Contents

Introduction

For this project, the client, Montana Savings Bank (MSB), a local bank in a small town in Montana, had requested new software for their ATMs (Automated Teller Machines) to streamline its operations for cost-cutting purposes. Specifically, MSB had sought to save labor costs, eliminate manual errors, and provide 24-hour service to its customers through the newly designed software.

After extensive dialog with MSB, the following requirements were drafted for the ATM:

- All the ATMs in the three branches and elsewhere (e.g., nearby malls) will be connected to a central database in MSB Headquarter. Every transaction of the ATMs will be pooled into this central database automatically. This central database will be maintained by an Oracle DBMS (Database Management System).
- All the customer information is stored in the central database. The central database in the headquarters is responsible for interacting with other banks.
- The ATM only handles transactions of five customer accounts: checking, savings, money market, consumer loan, and mortgage.
- For CDs (Certificate of Deposit), customers can use the ATM to check balance only, no transaction (i.e., deposit, withdrawal, etc.) is allowed.
- Customers access the ATM with an ATM card that is issued by the bank.
- Customers can deposit money to their checking, savings, or money market accounts.
- MSB has a group of preferred customers. There is no hold placed on their deposits (either cash or check) for preferred customers. Otherwise, a 3-day hold will be placed on their deposits (check only).
- Customers can withdraw money from their checking, savings, or money market accounts. However, non-preferred customers cannot withdraw money more than the account balance (i.e., no overdraft protection).
- Customers can withdraw money from their consumer loan account, up to the limit established by the loan. (The limit of the loan and the interest rate will be negotiated on an individual basis. This information is stored in the central database.)
- Preferred customers can withdraw more money than the balance from their checking, savings, or money market accounts. In this case, the money will automatically come from their consumer loan account, up to the limit established by the loan (overdraft protection).
- The ATM allows deposit of checks or cash. Cash consists of paper money only, no coins.
- Customers can deposit any amount but the withdrawal from all the accounts is limited to $500 per day. The withdrawal must be a multiple of $10.
- Customers can transfer money between accounts as follows:
    - Move money among checking, savings, and money market accounts.
    - Move money from checking, savings, or money market accounts to consumer loan accounts.
    - Move money from checking, savings, or money market accounts to mortgage.

- Customers can check the balance of their checking, savings, money market, CD, consumer loan, or mortgage account.
- A receipt can be printed out for a transaction. However, a customer can decline the printing of a receipt.
- MSB allows non-MSB customers to use their ATMs to deposit or withdraw money from their savings or checking accounts. A service fee of $3 will be charged to the non-MSB customers for each transaction.

Team Information

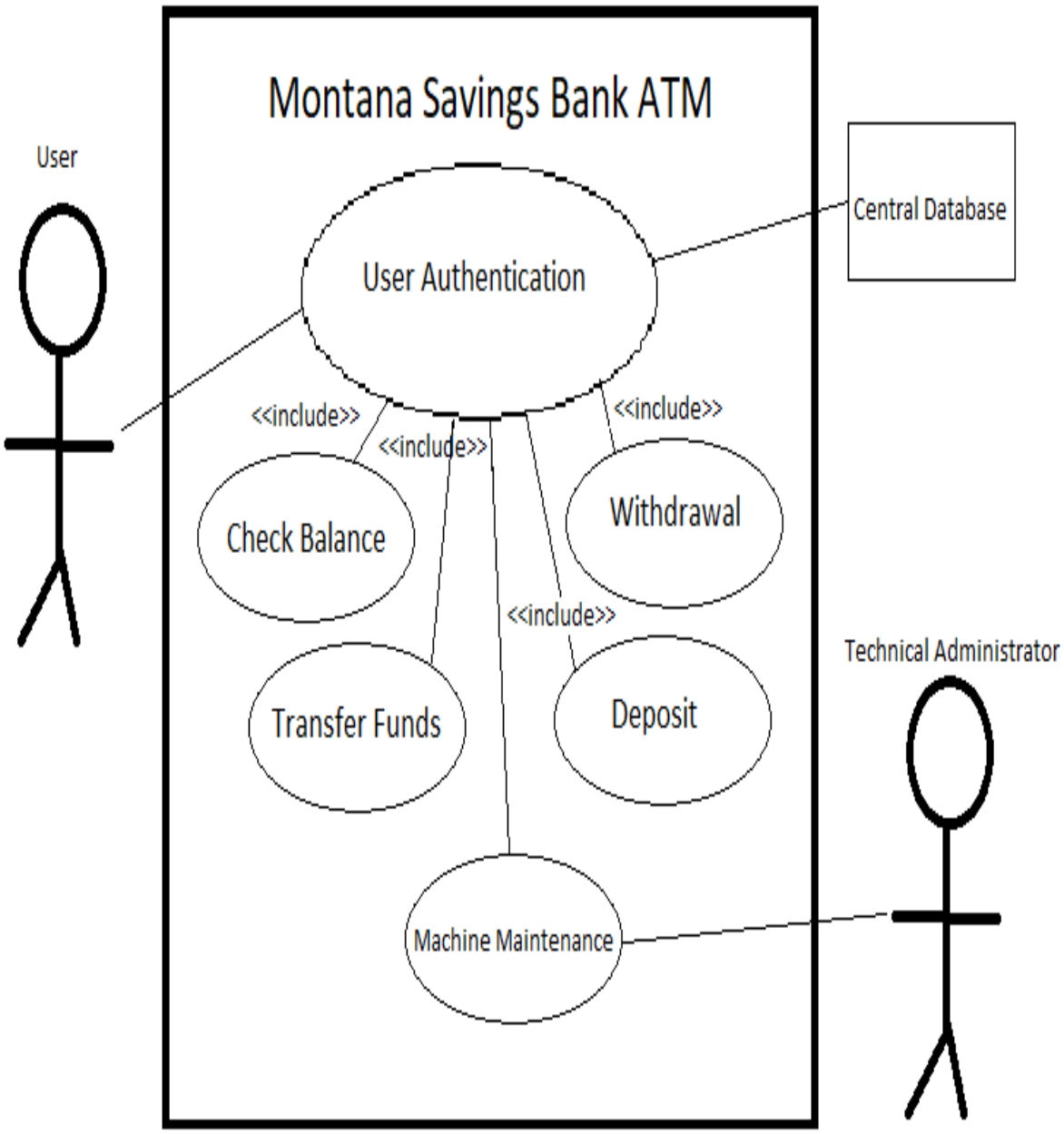
For this assignment, teams were either selected based on previous acquaintanceship or randomly selected based on alphabetical order of those remaining. This team, composed of Jake Balvin and Nicholas Girod, was formed by random selection.

As a whole, this team worked well and were able to complete the partitioned assignments with little effort. Communication is key to any partnership and both members were able to speak openly and effectively.

Nicholas Girod:
Working with Jake Balvin was easy; we had no problems working together to get the project finished on time and as envisioned. Jake was open for communication and reachable when collaboration was needed. Whenever I was unable to finish something, he was able to pick up any slack. An excellent partner with whom I have no qualms or issues to criticize.

Jake Balvin:
Nicholas Girod was an amazing partner. He was very easily reachable and was very knowledgeable about how to complete the project. He also always completed his end of the project very early, and never handed in his parts late. Whenever I was stuck on something, he was always available to ask questions to. I never had a problem working with him.

UML Model


Montana Savings Bank ATM
User
Central Database
User Authentication
<<include>>
<<include>>
<<include>>
Check Balance
Withdrawal
<<include>>
Technical Administrator
Transfer Funds
Deposit
Machine Maintenance

Use Case Descriptions

User Authentication

Author: Jake Balvin and Nicholas Girod

Last Update: 5/6/2022

Pre-Conditions:

- ➔ The user has an ATM card and bank account

Dialog:

- ➔ The user logs into the system by inserting their ATM card into the ATM **- display_logon_screen( )**
- ➔ The ATM checks to see if the card is authorized – **capture_ID_card(int x)**
- ➔ The ATM card is unauthorized - **authenticate_user(ID_card) = false**
    - ◆ The ATM displays a message that tells the user their card is unauthorized and ejects the card **– authenticate_error( ), eject_card( )**
- ➔ The card is authorized - **authenticate_user(ID_card) = true**
    - ◆ The ATM asks the user to insert their pin number **– capture_ID_pin(int y)**
- ➔ The pin code is incorrect – **authenticate_user(ID_pin) = false**
    - ◆ The ATM will display a message that informs the user the pin code they entered was incorrect and then prompts the user to reenter the pin code. The ATM will then allow the user to re enter their pin code **– authenticate_error( )**
    - ◆ The user fails to enter the correct pin code 3 times **– user_lock( )**
        - The card is locked for 24 hours and a message is displayed on the ATM that says, “You failed to enter the correct pin code 3 times. Your ATM

card is now blocked for 24 hours.” **- display_user_locked_message(error_handling: error_number x), eject_card( )**

➔ The user entered their correct pin number **– authenticate_user(ID_pin) = true**

➔ The ATM checks to see if the user is a Technical Administrator **–capture_user_status( )**

➔ The ATM will display a list of the possible transitions that the user can do **– display_user_options( )**

- User can check balance, go to CHECK BALANCE use case
- User can transfer funds, go to TRANSFER FUNDS use case
- User can deposit money, go to DEPOSIT use case
- User can withdraw money, go to WITHDRAW use case
- If the User is a technical Administrator **– if(capture_User_ID == “Technical_Administrator”)**
    - Technical Administrator can perform maintenance on the ATM, go to MACHINE MAINTENANCE

➔ A message is displayed on the bottom of the screen that says that non-MSB customers are charged three dollars for every transaction **– display_fee_notice ( )**

➔ The ATM has less than 100 dollars **– if(low_funds == true)**

◆ the screen will display a message that says the ATM is out of money and that the customer should use another ATM **- display_error_message(ATM_low_funds_error( )), logout( )**

➔ User logs off **– logout( )**

◆ ATM ejects ATM card **- eject_card( )**

Post-Conditions:

- ➔ The user has logged into their bank account on the ATM

Check Balance

Author: Jake Balvin and Nicholas Girod

Last Update: 5/6/2022

Pre-Conditions:

- ➔ the user has successfully logged into their bank account on the ATM

Dialog:

- ➔ The user has selected the option to check their balance on the ATM
- ➔ The user is given the option to choose one of their accounts, to go back to the list of transactions, or to logout – **display_accounts( ), display_return( ), logout( )**
- ➔ The user then chooses which of their accounts they want to check **- capture_user_choice( )**
    - ◆ The ATM will then extract the information of the desired account from the database – **communicate_DB( )**
- ➔ The ATM checks the database to confirm that the user is a MSB customer
- ➔ If the user is not a MSB customer – **if(user_identity == "Non-MSB_User")**
    - ◆ The ATM checks if there is more than three dollars in the checking account – **capture_NonMSB_fee( )**
    - ◆ The User has less than three dollars in their checking account – **user_NonMSB_fee = false**
        - The ATM displays a message that states that the user does not have the necessary funds to complete the transaction – **display_error_message(user_low_funds_error( ))**
        - The ATM ejects the ATM card **- eject_card( )**

- The user is logged off from their account **– logout( )**

◆ The User has more than three dollars in the checking account – **user_NonMSB_fee = true**

- The User is charged three dollars

➔ The ATM displays a message that says the user can check the balance of their checking, savings, money market, CD, consumer loan, or mortgage account. – **display_account_balance_options( ), capture_user_choice( )**

➔ The ATM displays a message that shows the user how much money is in the account – **display_account_amount( )**

➔ The ATM gives the user the option to either check the balance of a different account, to complete a different transaction, or to log off – **display_accounts( ), display_return( ), logout( )**

➔ The user selects the option to check the balance of a different account – **display_accounts( )**

◆ The ATM brings the user back to the option to choose which account they want to check the balance of

➔ The user selects the option to complete a different translation – **display_user_options( )**

◆ The user is brought back to the page after they successfully log in that lists the possible transitions that the user can do

➔ The user selects the option to log off

◆ The ATM card is ejected **- eject_card( )**

◆ The user is logged off from their bank account **- logout( )**

Post-Conditions:

➔ The user has received information about how much money is in their bank account

Transfer Funds

Author: Jake Balvin and Nicholas Girod

Last Update: 5/6/2022

Pre-Conditions:

- ➔ the user has successfully logged into their bank account on the ATM

Dialog:

- ➔ The user has selected the option to transfer funds on the ATM
- ➔ The ATM displays a message that tells the user that they can only take money out of the checking, savings, and money market accounts, and then gives the user the option to continue or to go back to the transaction selection page **– display_tranfser_notice( ), display_return( )**
- ➔ If the user selected to continue, the ATM displays a message that asks the user to choose an account to transfer money out of along with a list that consists of the checking, savings, and money market accounts **- display_account_CSMM( )**
- ➔ Once the user has selected which account to transfer money out of, the ATM displays a message that asks the user to select which account the user wants to transfer money into, and then gives the user a list that consists of the savings, checking, money market, consumer loan, and mortgage accounts **- display_accounts( )**
- ➔ The ATM then asks the user to enter the amount of money to transfer **- display_enter_money( )**
    - ◆ If the user enters more money then they have in that account, the ATM displays a message that states “you have entered more money than you have in the account” **- display_error_message(x), user_low_funds_error( )**

- The ATM then gives the user the option to either re enter the amount of money they wish to transfer or to go back to the list of transactions - **display_enter_money( ), display_return( )**

➔ The ATM checks the database to confirm that the user is a MSB customer

➔ If the user is not a MSB customer – **if(user_identity == "NonMSB")**

- The ATM checks if there will more than three dollars in the checking account after transferring the amount of money the user requested – **capture_NonMSB_fee( )**
  - The User will have more than three dollars in the checking account after the transaction – **user_NonMSB_fee = true**
    - The User is charged three dollars
  - The User will have less than three dollars in their checking account – **user_NonMSB_fee = false**
    - The ATM displays a message that states that the user does not have the necessary funds to complete the transaction – **display_error_message(x), user_low_funds_error( )**
    - The ATM ejects the ATM card **- eject_card( )**
    - The user is logged off from their account – **logout( )**

➔ The ATM transfers the specified amount of money from one account to the other and the amount of money in both accounts is updated in the central database - **transferFundsCSMM(account_number: int, account_number: int, amount: double), transferFundsCL(account_number: int, account_number: int, amount:**

double), transferFundsM(account_number: int, account_number: int, amount: double)

- ➔ The ATM then displays on the screen how much money is both of the accounts – **display_account_amount( )**
- ➔ The ATM displays a message that asks the user if they wish to print out a receipt of the transaction **- display_print_receipt_querry(answer: bool)**
  - ◆ If the user selects the option to print out a receipt the ATM prints out a receipt **- answer = true**
- ➔ The ATM asks the user if they wish to transfer more money or exit - **display_tranfser_notice( ), display_return( )**
  - ◆ If the user wishes to transfer more money the process repeats - **display_tranfser_notice( )**
  - ◆ If the user wishes to exit they are taken back to the transaction page - **display_user_options( )**

Post-Conditions:

- ➔ The user has transfer money from one account to the other

Withdrawal

Author: Jake Balvin and Nicholas Girod

Last Update: 5/6/2022

Pre-Conditions:

- ➔ The user has successfully logged into their bank account on the ATM

Dialog:

- ➔ The user has selected the option to withdraw money on the ATM
- ➔ The ATM displays a message that states that users can only withdraw $500 dollars a day, and then the ATM asks the user if they wish to continue or go back to the transaction list **- display_withdraw_notice( ), display_return( )**
- ➔ If the user selects to continue, the ATM asks the user which account they which to withdraw from and then gives the user the option to choose from their checking, savings, money market, or consumer loan accounts **- display_account_CSMMCL( )**
    - ◆ The user selects to withdraw money from their checking, savings, or money market accounts
        - The ATM checks the central database to see if the user is a preferred customer
        - The User is a preferred customer **– if(user_identity == “preferred”)**
            - A message is displayed that states that the user can withdraw more money than in their account due to being a preferred customer, and that the extra money will come from their consumer loan account **- display_preferred_withdraw_notice( )**

        - The ATM then receives the loan information of this user from the central database and displays the limit of how much more money they can withdraw **- capture_user_loan_interest_amount( )**

- The user selects to withdraw money from their consumer loan account
    - The ATM checks the central database to see if the user is a preferred customer
    - The User is a preferred customer **– if(user_identity == "preferred")**
        - A message is displayed that states that the user can withdraw more money than in their account due to being a preferred customer **- display_preferred_withdraw_notice( )**
        - The ATM then receives the loan information of this user from the central database and displays the limit of how much more money they can withdraw **- capture_user_loan_interest_amount( )**

➔ Once the user selects which account they wish to withdraw from, the ATM asks the user to input how much money they wish to withdraw, up to 500 in total dollars a day **- display_enter_money( )**

- The ATM can only dispense money in multiples of 10
- The ATM checks the central database to see how much money the user has withdrawn today **– communicate_DB( )**
    - If the user asks to withdraw more than 500 dollars in total, the ATM displays a message that tells the ussr that withdrawing that much from the ATM will result in withdrawing more than 500 dollars - **display_error_message(x), withdrew_more_than_500_error( ): int**

- The ATM then allows to user to insert a different amount to withdraw **- display_enter_money( )**

➔ The ATM checks the database to confirm that the user is a MSB customer

➔ If the user is not a MSB customer **– if(user_identity == "NonMSB")**

◆ The ATM checks if there will more than three dollars in the account they wish to withdraw from after the transaction is complete **– capture_NonMSB_fee( )**

- The User will have more than three dollars in the account after the transaction **– user_NonMSB_fee = true**
  - The user is charged three dollars
- The User will have less than three dollars in their account **– user_NonMSB_fee = false**
  - The ATM displays a message that states that the user does not have the necessary funds to complete the transaction **– display_error_message(x), user_low_funds_error( )**
  - The ATM ejects the ATM card **- eject_card( )**
  - The user is logged off from their account **– logout( )**

➔ The ATM checks to see if it has more than the amount of money requested

◆ The ATM does not have more money than requested - **if(low_funds == true)**

- A message is displayed that states that it does not have that much money, and to please go to another ATM - **display_error_message(x), ATM_low_funds_error( )**
- The ATM then logs the user off and ejects the ATM card **– logout( ), eject_card( )**

- ➔ If the ATM does have more money than requested, the ATM dispense the amount of money requested **- withdrawalFunds(account_number: int, funds: int%10)**
    - ◆ The amount of money in the ATM is then decreased by how much money the user has withdrawn
- ➔ The ATM releases the amount of money that was requested by the user
- ➔ The ATM updates the new account balance of the user in the central database, and then updates how much money the user has withdrawn from the ATM today in the central database **– communicate_DB( )**
- ➔ The ATM displays the new balance of the account, and asks the user if they wish to print a receipt **– display_account_amount( ), display_print_receipt_querry(answer: bool)**
    - ◆ The user selects yes **- answer = true**
        - A receipt is printed
    - ◆ The user selects no **- answer = false**
        - No receipt is printed
- ➔ The ATM then asks the user if they wish to withdraw more money from an account or to return to the list of transactions - **display_withdraw_notice( ), display_return( )**
    - ◆ The user requests to withdraw more money **display_withdraw_notice( )**
        - The process repeats from the start
    - ◆ The user request to return to the list of transactions - **– display_user_options( )**
        - The user is returned to the list of transactions

<u>Post-Conditions:</u>

- ➔ The user has withdrawn money from the ATM, and the balance of the user's account has decreased by the amount withdrawn

Deposit

Author: Jake Balvin and Nicholas Girod

Last Update: 5/6/2022

Pre-Conditions:

- ➔ The user has successfully logged into their bank account on the ATM

Dialog:

- ➔ The user has selected the option to deposit money into the ATM
- ➔ The ATM displays a message that states that the User can only deposit cash, and asks the user to click confirm **- display_deposit_message( )**
- ➔ The ATM asks which account they want to deposit money into and gives the user a list that consist of the checking, savings, and money market accounts **- display_account_CSMM()**
- ➔ Once the user selects which account they want to deposit into, the ATM asks the user if they wish to deposit a check or cash - **ask_cash_or_check( )**
- ➔ The user wishes to deposit a check **- user_choice == "check"**
    - ◆ The ATM checks the central database to see if the User is a preferred customer
        - The member is a preferred customer **if(user_identity == "preferred")**
            - The ATM displays a message thats thats that there is no 3 day hold for their deposits of checks **- display_nohold_notice( )**
        - The member is not a preferred customer **if(user_identity == "unpreferred")**

        - The ATM displays a message thats thats that there is a 3 day hold for their deposits of checks - **display_hold_notice( ), deposit_hold_remaining: int days**

- The ATM prompt the user to insert the check into the ATM's check slot - **display_deposit-check( )**
- The ATM reads the check and then checks to make sure that the account of the user who wrote the check has those funds in their account **– communicate_DB( )**
    - There is not enough money in the account
        - The ATM displays a message that there is not enough money in the writer of the check's account, and then returns to the user to the list of transactions **– display_error_message(x), user_low_funds_error( ), display_return( )**
- The ATM then displays the amount on the check on the ATM's screen, and then asks the user to confirm if this is the right amount **- display_check_amount( )**
    - If users selects that this is not the right amount, the check is ejected from the ATM and the user is taken back to the list of transactions **- eject_check(), display_return( )**
- The ATM checks the database to confirm that the user is a MSB customer - **communicate_DB( )**
- If the user is not a MSB customer **– if(user_identity == "NonMSB")**
    - The ATM checks if there will more than three dollars their account after the transaction is complete **– capture_NonMSB_fee( )**

- The User will have more than three dollars in the account after the transaction – **user_NonMSB_fee = true**
  - The User is charged three dollars
- The User will have less than three dollars in their account – **user_NonMSB_fee = false**
  - The ATM displays a message that states that the user does not have the necessary funds to complete the transaction – **display_error_message(x), user_low_funds_error( )**
  - The ATM ejects the ATM card **- eject_card( )**
  - The user is logged off from their account – **logout( )**

- The ATM updates the amount of money in the account **- depositFunds(account_number: int, amount: int)**

➔ The user wishes to deposit cash

- The ATM checks the database to confirm that the user is a MSB customer
- If the user is not a MSB customer – **if(user_identity == "NonMSB")**
  - The ATM checks if there will more than three dollars their account after the transaction is complete – **capture_NonMSB_fee( )**
    - The User will have more than three dollars in the account after the transaction – **user_NonMSB_fee = true**
      - The User is charged three dollars
    - The User will have less than three dollars in their account – **user_NonMSB_fee = false**

- The ATM displays a message that states that the user does not have the necessary funds to complete the transaction – **display_error_message(x), user_low_funds_error( )**
- The ATM ejects the ATM card **- eject_card( )**
- The user is logged off from their account **– logout( )**

- The ATM asks the user to insert the money they wish to deposit into the ATM's cash slot and then to select finished when they are finished **- display_deposit_cash( )**
- The ATM checks to see if each bill is real and adds up all the bills inserted into it
  - If the ATM cannot read a bill, it ejects it **- eject_cash( )**
  - If a bill is not real, the ATM ejects it **- eject_cash( )**
- The ATM puts each bill into the pile that matches its value so it can know which bill it is dispensing
- Once the user presses finished, the ATM displays how much money was inserted into the ATM **- display_amount_inserted( )**
- The ATM then updates the amount of money in the account and then updates how much money is in the ATM by increasing it by how much the user inserted into it **- depositFunds(account_number: int, amount: int)**

➔ The ATM displays the new balance of the account and asks the user if they wish to print a receipt - **display_account_amount( ), display_print_receipt_querry(answer: bool)**

- The user selects yes **- answer = true**
  - A receipt is printed
- The user selects no **- answer = false**

- No receipt is printed

➔ The ATM then asks the user if they wish to deposit more money or to return to the list of transactions

- ◆ The user requests to deposit more money **- display_deposit_message( )**
  - The process repeats from the start
- ◆ The user request to return to the list of transactions **– display_user_options( )**
  - The user is returned to the list of transactions

Post-Conditions:

➔ The user has deposited money into the ATM, and the balance of the user's account has increased by the amount deposited

Machine Maintenance

Author: Jake Balvin and Nicholas Girod

Last Update: 5/6/2022

Pre-Conditions:

- ➔ The has technical administrator successfully logged into their account

Dialog:

- ➔ The technical administrator has selected the option to perform maintenance on the ATM – **display_TA_logon_screen( )**
- ➔ The ATM asks the technical administrator for their employee ID – **authenticate_user(ID_employee_number)**
    - ◆ The technical administrator enters the wrong employee ID – **if(authenicate_user(ID_employee_number) == false)**
        - The ATM displays a message that states that the employee ID entered was incorrect and then prompts the technical administrator to re enter the ID. – **display_error_message(authenticate_error( ))**
        - The ATM will then allow the technical administrator to re-enter their ID
- ➔ The ATM displays a list of everything the technical administer can do - **display_TA_options( ), capture_user_choice( )**
    - ◆ Technical administrator can refill receipt paper
    - ◆ Technical administrator can refill cash
    - ◆ Technical administrator can fix paper jam
- ➔ Technical administrator chooses to refill receipt paper – **if(receipt_paper_low)**
    - ◆ The ATM opens up the compartment where the receipt paper is stored

- The technical administrator can replace the used up roll of receipt paper with a new one
- The technical administrator manually closes the compartment when they are done **- verify_TA_fix( ), receipt_paper_low = false**

➔ Technical administrator chooses to refill cash **- if(low_funds)**

- The ATM asks the technical administrator to deposit the money into the ATM **– display_TA_funds_request_message( )**
- The technical administrator places the money into the ATM
- The ATM counts how much money has been placed into the ATM, displays it, and asks the technical administrator to conform if the amount is correct **– display_TA_accuracy_message(int x)**
    - The amount is incorrect **– amount_inaccurate_error( ), entered_amount_displayed = false**
        - The ATM deposits all of the bills that were placed into it
        - The ATM asks the technical administrator to check the bills and then deposit the bills again – **display_TA_deposit_message( )**
- Once the technical administrator confirms the amount entered, the ATM adds that to the amount of money it holds **– entered_amount_displayed = true, display_ATM_total_funds( )**
- The ATM puts each bill into the pile that matches its value so it can know which bill it is dispensing
- The ATM verifies that its funds have been restored **– verify_TA_fix( ), low_funds = false**

- ➔ Technical administrator chooses to fix paper jam **– if(paper_jam)**
  - ◆ The ATM opens up its front, which allows the technical administrator to access and take out the ATM's money slots, receipt slot, and check slot
    - The technical administrator can then take out the object that is causing the jam
  - ◆ The technical administrator then closes the front of the ATM **- verify_TA_fix( )**
  - ◆ The ATM verifies that the jam has been resolved **– paper_jam = false**
- ➔ The ATM asks the technical administrator if they wish to fix something else **- display_TA_maintenance_message( ), capture_user_choice( )**
  - ◆ The technical administrator says yes **more_maintenance = true**
    - The ATM goes back to the list of everything the technical administrator can do **– display_TA_options( ), capture_user_choice( )**
    - The technical administrator says no **- more_maintenance = false**
      - The ATM takes the technical administrator back to the TA logon screen - **display_TA_logon_screen( )**
- ➔ Technical administrator chooses to logoff **- logout( )**

Post-Conditions:

- ➔ The machine has underwent maintenance

Class Diagrams

User
first_name: string # last_name: string # mailing_address: string # ID_card: int {12 digits} # ID_pin: int {4 digits} # user_identity: string {Preferred_User, Unpreferred_User, nonMSB_User)
+ getFName( ): string + setFName(first_name: string) + getLName( ): string + setLName(last_name: string) + getMailingAddress( ): string + setMailingAddress(street: string, city: string, stateCode: string = MT, zipCode: int {5 digits})

Preferred_User (Static Relationship to User)
- lock_status: bool - overdraft_protection: bool

Unpreferred_User (Static Relationship to User)
- lock_status: bool - deposit_hold_remaining: int

MSB_User (Static Relationship to User)
- lock_status: bool - overdraft_protection: bool

Non-MSB_User (Static Relationship to User)

- lock_status: bool - user_nonMSB_fee: bool

Technical_Administrator
- first_name: string - last_name: string - ID_employee_number: int {7 digits} - employment_status: string {active, suspended, inactive}
+ getFName( ): string + setFName(first_name: string) + getLName( ): string + setLName(last_name: string) + getEmployeeNumber( ): int + setEmployeeNumber(employee_number: int) + getStatus( ): string + setStatus(status: string)

Account
account_number: int {10 digits} # account_amount: double
+ checkAccountBalance(account_number: int)

Account_CSMM (Static Relationship to Account)
- account_type: string {checkings, savings, money market} - widthdrawal_amount_remaining_today: int {500 per day, multiple of 10}
+ withdrawalFunds(account_number: int, funds: int%10) + depositFunds(account_number: int, amount: int) + transferFundsCSMM(account_number: int, account_number: int, amount: double) + transferFundsCL(account_number: int, account_number: int, amount: double) + transferFundsM(account_number: int, account_number: int, amount: double)

Account_CL (Static Relationship to Account)

- account_type: string {consumer loan} - user_loan_allotted: int
+ withdrawalFundsCL(user_loan_allotted: int, funds: int)

Account_M (Static Relationship to Account)
- account_type: string {mortgage}

Screen
- x-coordinate: int - y-coordinate: int - color: hex - border: int - pixel: int - brightness: int - sharpness: int - more_maintenace: bool - entered_amount: int - entered_amount_displayed: bool - user_choice: string

- ask_cash_or_check( )

- capture_ID_card(user.ID_card: int {12 digits})

- capture_ID_pin(user.ID_pin: int {4 digits})

- capture_user_choice( )

+ display_accounts( )

+ display_account_amount(account.amount: double)

+ display_account_CSMM( )

+ display_account_CSMMCL( )

+ display_account_balance_options( )

+ display_amount_inserted( )

+ display_ATM_total_funds( ): string

+ display_check_amount( )

+ display_deposit_cash( )

+ display_deposit-check( )

+ display_deposit_message( )

+ display_error_message(error_handling.error_number x)

+ display_fee_notice( )

+display_hold_notice( )

+ display_logon_screen( )

+display_nohold_notice( )

+ display_print_receipt_querry(answer: bool)

+ display_remaining_loan( )

+ display_return( ) + display_tranfser_notice( ) + display_withdraw_notice( ) + display_preferred_withdraw_notice( ) + display_TA_accuracy_message(entered_amount: int) + display_TA_deposit_message( ) + display_TA_funds_request_message( ) + display_TA_logon_screen( ) + display_TA_maintenance_message( ) + display_TA_options( ) + display_user_options( ) + display_user_locked_message(error_handling: error_number x) + display_enter_money( ): int

ATM System
+ user_lock(Error_Handling.error_number: int): bool + employee_lock(Technical_Administrator.empoloyment_status: string): bool + eject_card( ) + eject_cash( )

+ eject_check( ) + verify_TA_fix( ) + logout( )

DB_Interface
- user_status: string (verified, invalid)
+ authenticate_user(ID_card: int || ID_pin: int || ID_employee_number: int): bool + communicate_DB( ) {connects to central database} + capture_user_loan_interest_amount( ): int + capture_NonMSB_fee( ): bool + capture_user_ID( ): string

Error_Handling
- error_number: int - error_count: int - receipt_paper_low: bool - low_funds: bool - paper_jam: bool

+ authenticate_error( ): int

+ ATM_low_funds_error( ): int

+ user_low_funds_error( ): int

+ receipt_paper_low_error( ): int

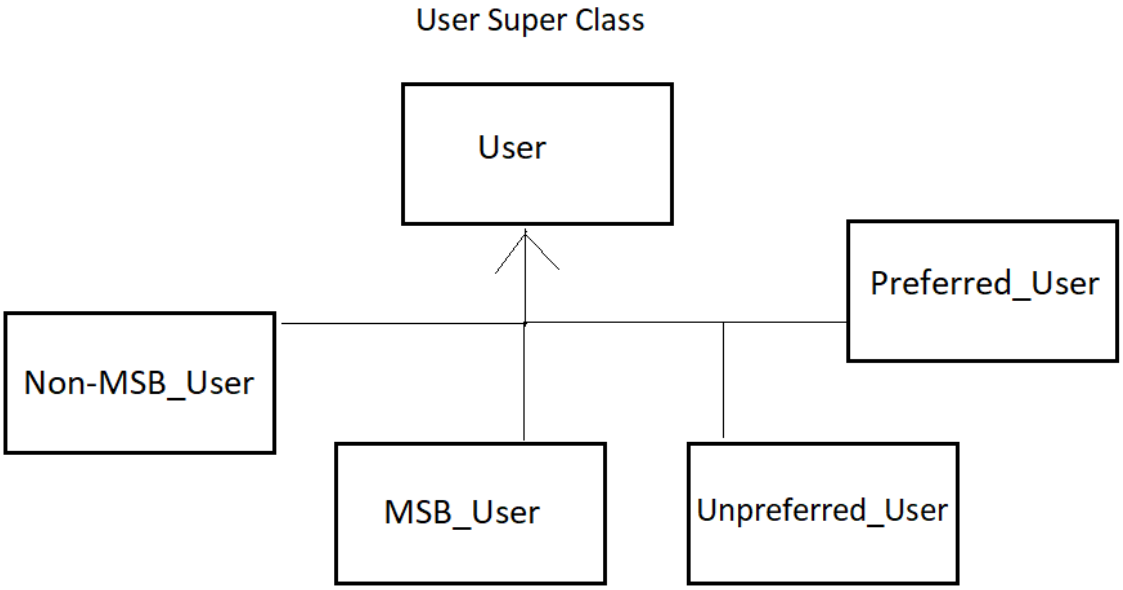
+ amount_inaccurate_error( ): int

+ paper_jam_error( ): int

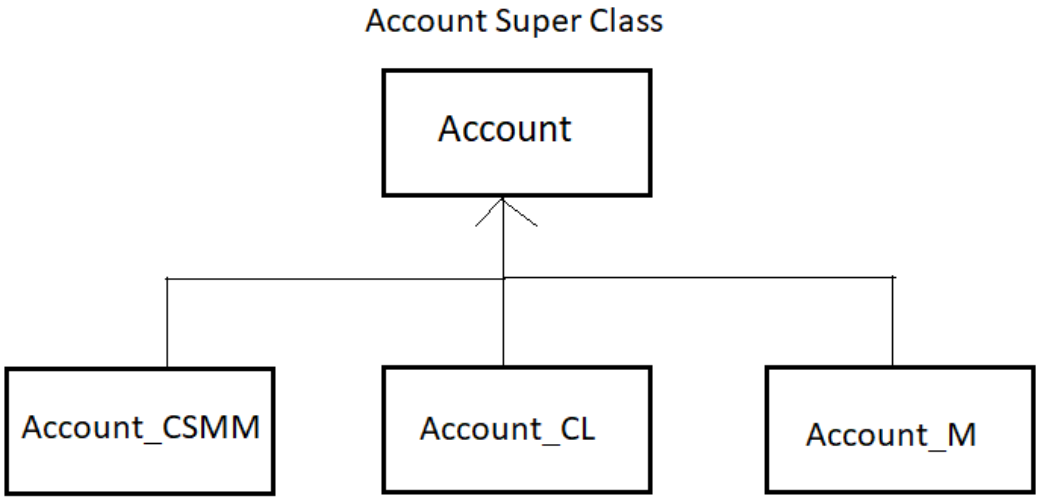
+ withdrew_more_than_500_error( ): int

Relationship Diagrams

User Class:



Account Class:



Class Diagram Descriptions

User Class (Super Class)

This class is used to get the User's name, mailing address, card number, and pin. The mailing address consists of the street, city, zip code and state.

Preferred User Class

This class is used to confirm that the user is a preferred user. Being a preferred user allows the user to withdraw more money from the ATM than is in their bank account. The amount of extra money they can withdraw is determined by the bank and retrieved from the central database.

Unpreferred User Class

This class is used to confirm that the user is not a preferred user. This means that they cannot withdraw more money than they have in their bank account. Unpreferred customers also have a three day hold when depositing their checks.

MSB User Class

This class is used to confirm that the user is a MSB customer.

Non-MSB User Class

This class is used to confirm that the user is a MSB customer. If the user is not a MSB customer, they must pay a three dollar fee everytime they do a transaction.

Technical Administrator Class

This class is used to confirm that the user is a Technical Administrator, and get their name, employee ID, and status.

Account Class (Super Class)

This account allows the user to check the amount of money that is located within their account.

Account CSMM (Checkings/Savings/Money Market) Class

This class allows Users to either transfer, withdraw, or deposit money into their accounts. It also changes the amount of money registered into the ATM if the user withdrawals or deposits money.

Account CL (Consumer Loan) Class

This class is used to allow the user to withdraw money from their consumer loan account.

Account M (Mortgage) Class

This class is used for the user's mortgage account.

Screen Class

This class is used to display the messages that the ATM must show to the user throughout its use.

ATM System Class

This class is used to allow the ATM to lock users out after failing to input their passwords, eject cards, cash and checks, and log the user out.

DB (Database) Interface Class

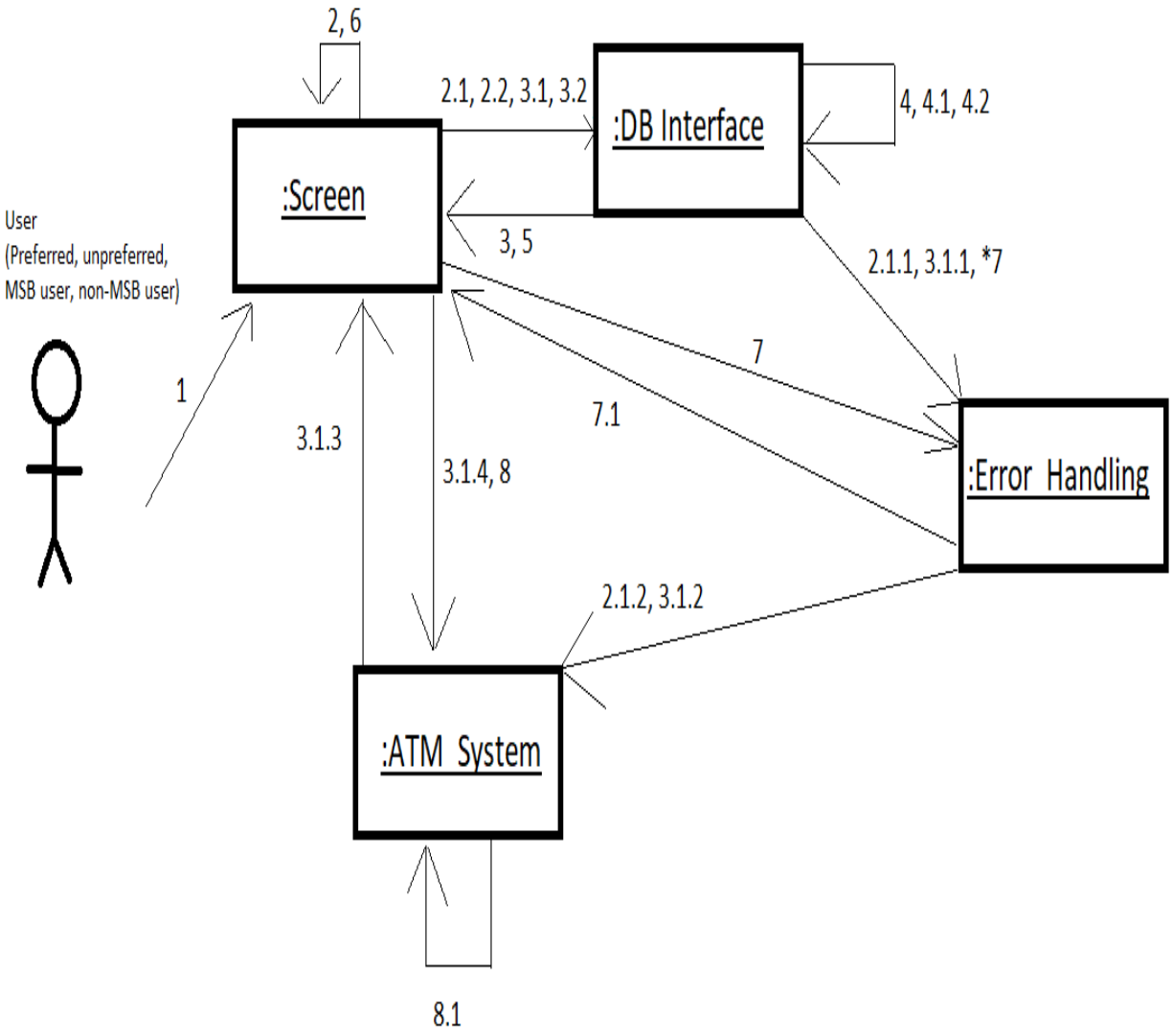
This class is used to connect to the central database and retrieve information from it. It also authenticates the user, charges non MSB users a fee, and determines how much money the bank has loaned the user.

Error Handling Class

This class is used to determine the error that the user has run into and tells the user how to fix the error.

Communication Diagrams: User Authentication

Diagram:



Operation Sequence:

1: display_logon_screen( )

2: capture_ID_card( )

    2.1: authenticate_user(ID_card) = false

        2.1.1: authenticate_error( )

        2.1.2: eject_card( )

    2.2: authenticate_user(ID_card) = true

3: capture_ID_pin( )

    3.1: authenticate_user(ID_pin) = false

        3.1.1: authenticate_error( )

        3.1.2: user_lock( )

        3.1.3: display_user_locked_message( )

        3.1.4: eject_card( )

    3.2: authenticate_user(ID_pin) = true

4: capture_user_ID( )

    4.1: user_status == “Technical_Administrator”

        4.1.1: go to MACHINE MAINTENANCE

    4.2: user_status == “MSB_User” || “non-MSB_User”

5: display_user_options( )

6: display_fee_notice( )

*7: if(low_funds == true)

    7.1: display_error_message(ATM_low_funds_error( ))

8: logout( )

    8.1: eject_card( )

Scenario Sequence:

1, 2, 2.1, 2.1.1, 2.1.2 - Invalid card

1, 2, 2.2, 3, 3.1, 3.1.1, 3.1.2, 3.1.3, 3.1.4 - Invalid pin

1, 2, 2.2, 3, 3.2, 4, 4.1 - Employee status confirmed, go to MACHINE MAINTENANCE

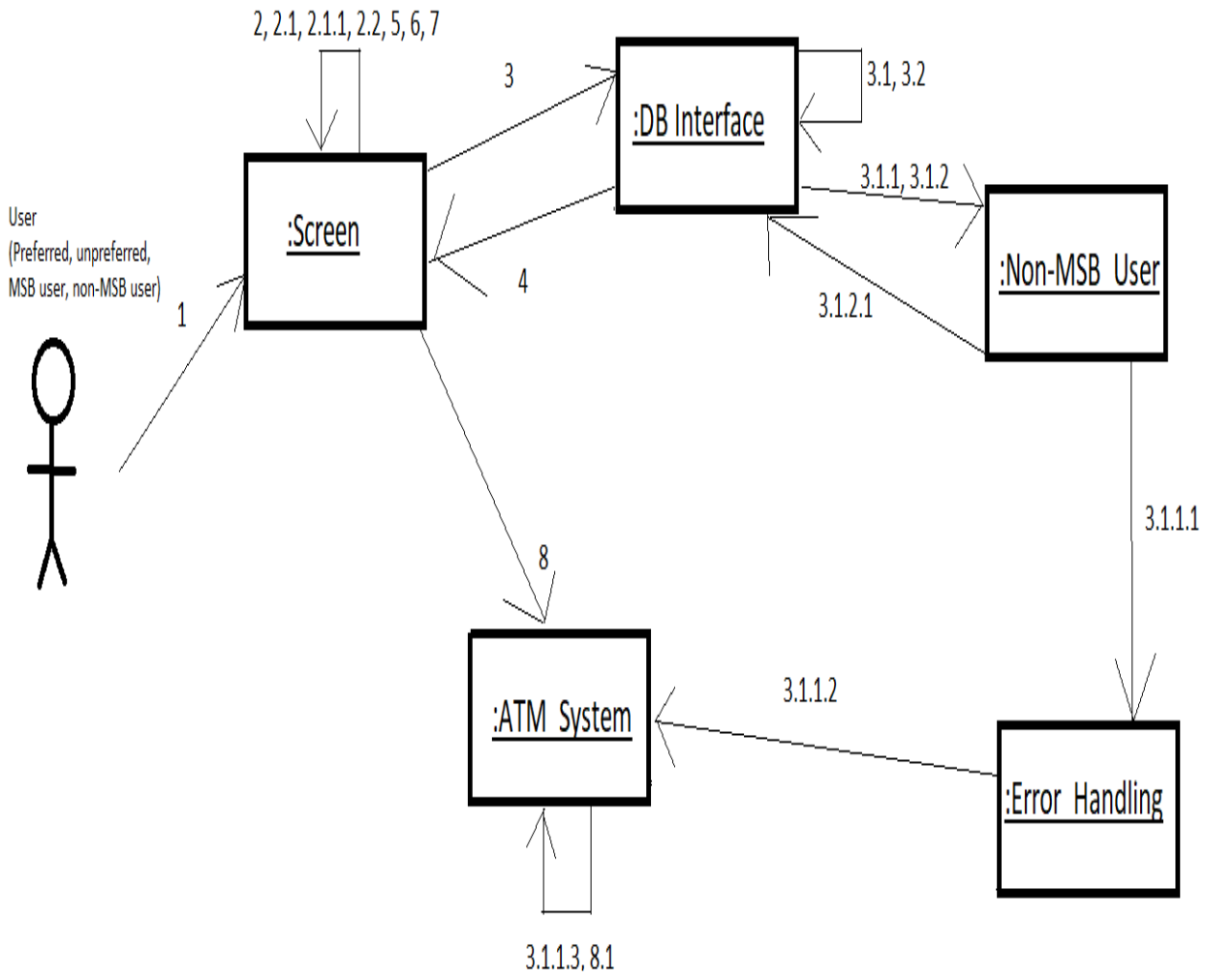
1, 2, 2.2, 3, 3.2, 4, 4.2, 5 - Customer status confirmed, go to other use cases (display other screens)

1, 2, 2.2, 3, 3.2, 4, 4.2, 5, 6, 7, 7.1, 8, 8.1 - User logon and other use cases displayed, but low funds in machine determined, logoff of user

1, 2, 2.2, 3, 3.2, 8, 8.1 - User logon then logoff, or user logon then system timed out

Communication Diagrams: Check Balance

Diagram:



Operation Sequence:

1: display_accounts( )

1: display_return( )

2: capture_user_choice( )

2.1: user_choice == “return”

2.1.1: display_user_options( )

2.2: user_choice == “accounts”

3: communicate_DB( )

3.1: user_identity == “Non-MSB_User”

3.2: capture_nonMSB_fee( )

3.1.1: user_NonMSB_fee = false

3.1.1.1: display_error_message(user_low_funds_error( ))

3.1.1.2: eject_card( )

3.1.1.3: logout( )

3.1.2: user_NonMSB_fee = true

3.1.2.1: capture_NonMSB_fee( )

4: display_account_balance_options( )

5: capture_user_choice( )

6: display_account_amount(account.amount: double)

7: display_return( )

8: logout( )

8.1: eject_card( )

Scenario Sequence:

1, 2, 2.1, 2.1.1 - User has decided not to check account balances and to view other options, go to other use cases (display other screens)

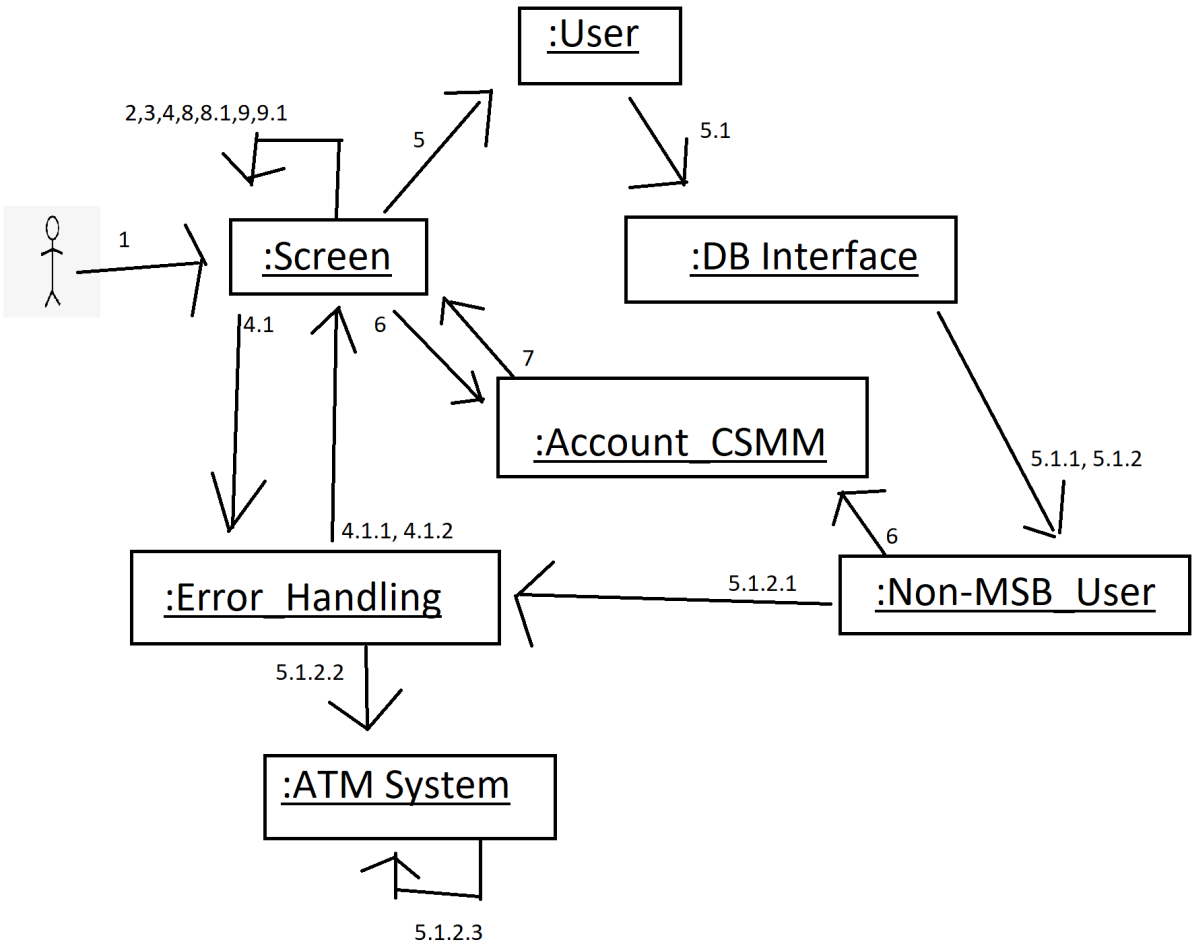
1, 2, 2.2, 3, 3.1, 3.2, 3.1.1, 3.1.1.1, 3.1.1.2, 3.1.1.3 - Non-MSB user does not have funds for ATM use fee

1, 2, 2.2, 3, 3.1, 3.1.2, 3.1.2.1, 4, 5, 6, 7, 8, 8.1 - Non-MSB user has funds for ATM use fee, checks balance, and logoffs

1, 2, 2.2, 3, 4, 5, 6, 7, 2, 2.1, 2.1.1 - MSB user has checked an account balance, then decided to view other options, go to other use cases (display other screens)

Communication Diagrams: Transfer Funds

Diagram:



Operation Sequence:

1: display_tranfser_notice( )

1: display_return( )

2: display_account_CSMM( )

3: display_accounts( )

4: display_enter_money( )

4.1: display_error_message(x), user_low_funds_error( )

4.1.1:display_enter_money( )

4.1.2: display_return( )

5: user_identity == “Non-MSB_User”

5.1: capture_nonMSB_fee( )

5.1.1: user_NonMSB_fee = true

5.1.2 user_NonMSB_fee = false

5.1.2.1: display_error_message(user_low_funds_error( ))

5.1.2.2: eject_card( )

5.1.2.3: logout( )

6: transferFundsCSMM(account_number: int, account_number: int, amount: double)

6: transferFundsCL(account_number: int, account_number: int, amount: double)

6:transferFundsM(account_number: int, account_number: int, amount: double)

7: display_account_amount( )

8: display_print_receipt_querry(answer: bool)

8.1 answer = true

9: display_tranfser_notice( )

9: display_return( )

9.1: display_user_options( )

Scenario Sequence:

1 - User has decided to return to the transaction selection page

1, 2, 3, 4, 4.1, 4.1.1, 5, 5.1, 5.1.1, 6, 7, 8, 8.1, 9 - A non MSB user who has enough money for the transaction has selected to transfer money from an account into another, entered entered more money than they have in that account, decided to reenter the money, transferred funds from one account to another, is shown the amount of money in both accounts, wants to print a receipt, and decides to transfer more money

1, 2, 3, 4, 4.1, 4.1.1, 5, 5.1, 5.1.2, 5.1.2.1, 5.1.2.2, 5.1.2.3 - A non MSB user requests to make a transaction, but does not have enough money to do so, so their ATm card is ejected and they are logged off

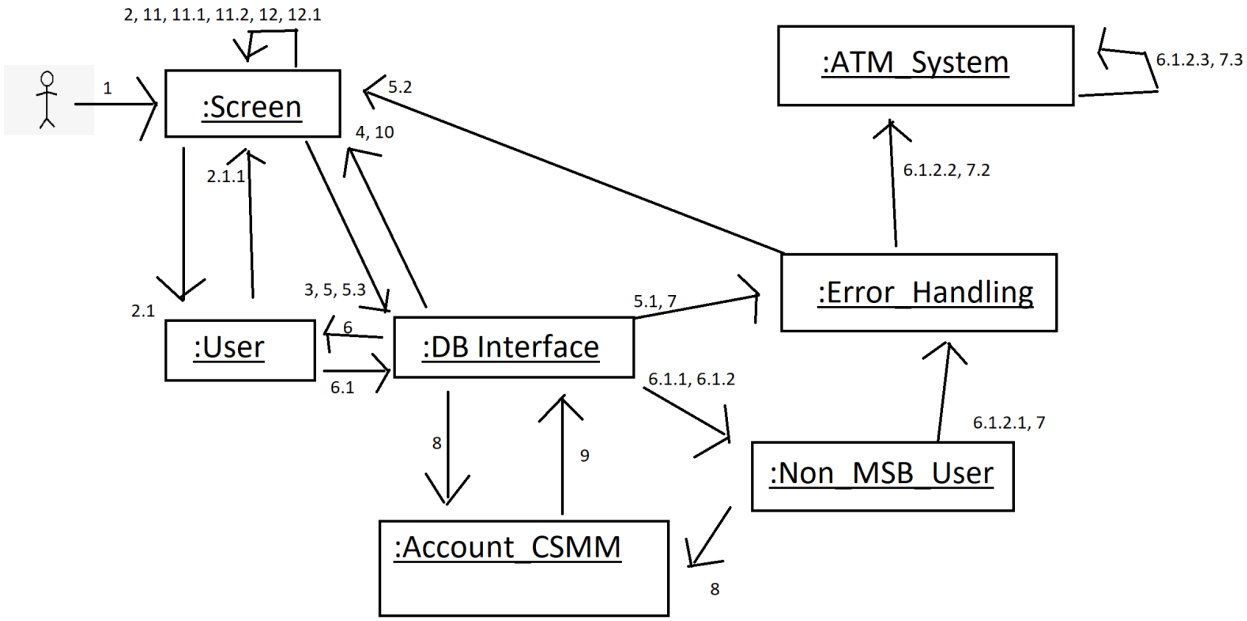
1, 2, 3, 4, 4.1, 4.1.1, 6, 7, 8, 8.1, 9 - A MSB user has selected to transfer money from an account into another, entered entered more money than they have in that account, decided to reenter the money, transferred funds from one account to another, is shown the amount of money in both accounts, wants to print a receipt, and decides to transfer more money

1, 2, 3, 4, 4.1, 4.1.1, 6, 7, 8, 8.1, 9, 9.1 - A MSB user has selected to transfer money from an account into another, entered entered more money than they have in that account, decided to reenter the money, transferred funds from one account to another, is shown the amount of money in both accounts, wants to print a receipt, and decides to return to the transactions page

1, 2, 3, 4, 4.1, 4.1.2 - the user has selected to transfer money from an account into another, entered entered more money than they have in that account, and decides to return to the transaction page

Communication Diagrams: Withdrawal

Diagram:



Operation Sequence:

1: display_tranfser_notice( )

1: display_return( )

2: display_account_CSMMCL( )

    2.1:user_identity == “preferred”

        2.1.1: display_preferred_withdraw_notice( )

3: capture_user_loan_interest_amount( )

4: display_enter_money( )

5: communicate_DB( )

    5.1: display_error_message(x), withdrew_more_than_500_error( ): int

    5.2: display_enter_money( )

    5.3: communicate_DB( )

6: user_identity == “Non-MSB_User”

    6.1: capture_nonMSB_fee( )

        6.1.1: user_NonMSB_fee = true

        6.1.2 user_NonMSB_fee = false

            6.1.2.1: display_error_message(user_low_funds_error( ))

            6.1.2.2: eject_card( )

            6.1.2.3: logout( )

7: low_funds == true

    7.1: display_error_message(x), ATM_low_funds_error( )

    7.2: eject_card( )

    7.3 : logout( )

8: withdrawalFunds(account_number: int, funds: int%10)

9: communicate_DB( )

10: display_account_amount( )

11: display_print_receipt_querry(answer: bool)

    11.1: answer = true

    11.2: answer = false

12: display_withdraw_notice( )

12: display_return( )

    12.1: display_user_options( )

Scenario Sequence:

1 - User has decided to return to the transaction selection page

1, 2, 2.1, 2.1.1, 3, 4, 5, 5.1, 5.2, 5.3, 6, 6.1, 6.1.1, 7, 7.1, 7.2, 7.3 - The user is a preferred customer but not a MSB member and attempts to withdraw more than 500 dollars, then tries to withdraw money from their account, but the ATM does not have enough money

1, 2, 2.1, 2.1.1, 3, 4, 5, 5.1, 5.2, 5.3, 6, 6.1, 6.1.1, 8, 9, 10, 11, 11.1, 12 - The user is a preferred customer but not a MSB member and attempts to withdraw more than 500 dollars, then successfully withdraws money and prints a receipt, and chooses to withdraw more from their account

1, 2, 2.1, 2.1.1, 3, 4, 5, 5.1, 5.2, 5.3, 6, 6.1, 6.1.1, 8, 9, 10, 11, 11.1, 12, 12.1 - The user is a preferred customer but not a MSB member and attempts to withdraw more than 500 dollars, then successfully withdraws money, prints a receipt, and chooses to return to the transaction list

1, 2, 2.1, 2.1.1, 3, 4, 5, 5.1, 5.2, 5.3, 6, 6.1, 6.1.1, 8, 9, 10, 11, 11.2, 12 - The user is a preferred customer but not a MSB member and attempts to withdraw more than 500 dollars, then successfully withdraws money, doesn't print a receipt, and chooses to withdraw more from their account

1, 2, 2.1, 2.1.1, 3, 4, 5, 5.1, 5.2, 5.3, 6, 6.1, 6.1.1, 8, 9, 10, 11, 11.2, 12.1 - The user is a preferred customer but not a MSB member and attempts to withdraw more than 500 dollars, then successfully withdraws money, doesn’t print a receipt, and chooses to return to the transaction list

1, 2, 2.1, 2.1.1, 3, 4, 5, 5.1, 5.2, 5.3, 6, 6.1, 6.1.2, 6.1.2.1, 6.1.2.2, 6.1.2.3 - The user is a preferred customer but not a MSB member and attempts to withdraw more than 500 dollars, then tries to withdraw money from their account, but the user does not have enough money to do the transaction

1, 2, 2.1, 2.1.1, 3, 4, 5, 5.1, 5.2, 5.3, 8, 9, 10, 11, 11.1, 12 - The user is a preferred customer and a MSB member and attempts to withdraw more than 500 dollars, then successfully withdraws money and prints a receipt, and chooses to withdraw more from their account

1, 2, 2.1, 2.1.1, 3, 4, 5, 5.1, 5.2, 5.3, 8, 9, 10, 11, 11.1, 12.1 - The user is a preferred customer and a MSB member and attempts to withdraw more than 500 dollars, then successfully withdraws money, prints a receipt, and chooses to return to the transaction list

1, 2, 2.1, 2.1.1, 3, 4, 5, 5.1, 5.2, 5.3, 8, 9, 10, 11, 11.2, 12 - The user is a preferred customer and a MSB member and attempts to withdraw more than 500 dollars, then successfully withdraws money, doesn’t print a receipt, and chooses to withdraw more from their account

1, 2, 2.1, 2.1.1, 3, 4, 5, 5.1, 5.2, 5.3, 8, 9, 10, 11, 11.2, 12.1 - The user is a preferred customer and a MSB member and attempts to withdraw more than 500 dollars, then successfully withdraws money, doesn’t print a receipt, and chooses to return to the transaction list

1, 2, 2.1, 2.1.1, 3, 4, 5, 6, 6.1, 6.1.1, 7, 7.1, 7.2, 7.3 - The user is a preferred customer but not a MSB member tries to withdraw money from their account, but the ATMdoes not have enough money

1, 2, 2.1, 2.1.1, 3, 4, 5, 6, 6.1, 6.1.1, 8, 9, 10, 11, 11.1, 12 - The user is a preferred customer but not a MSB member and successfully withdraws money and prints a receipt, and chooses to withdraw more from their account

1, 2, 2.1, 2.1.1, 3, 4, 5, 6, 6.1, 6.1.1, 8, 9, 10, 11, 11.1, 12, 12.1 - The user is a preferred customer but not a MSB member successfully withdraws money, prints a receipt, and chooses to return to the transaction list

1, 2, 2.1, 2.1.1, 3, 4, 5, 6, 6.1, 6.1.1, 8, 9, 10, 11, 11.2, 12 - The user is a preferred customer but not a MSB member successfully withdraws money, doesn't print a receipt, and chooses to withdraw more from their account

1, 2, 2.1, 2.1.1, 3, 4, 5, 6, 6.1, 6.1.1, 8, 9, 10, 11, 11.2, 12.1 - The user is a preferred customer but not a MSB member successfully withdraws money, doesn’t print a receipt, and chooses to return to the transaction list

1, 2, 2.1, 2.1.1, 3, 4, 5 , 6, 6.1, 6.1.2, 6.1.2.1, 6.1.2.2, 6.1.2.3 - The user is a preferred customer but not a MSB member tries to withdraw money from their account, but the user does not have enough money to do the transaction

1, 2, 2.1, 2.1.1, 3, 4, 5, 8, 9, 10, 11, 11.1, 12 - The user is a preferred customer and a MSB member successfully withdraws money and prints a receipt, and chooses to withdraw more from their account

1, 2, 2.1, 2.1.1, 3, 4, 5, 8, 9, 10, 11, 11.1, 12.1 - The user is a preferred customer and a MSB member successfully withdraws money, prints a receipt, and chooses to return to the transaction list

1, 2, 2.1, 2.1.1, 3, 4, 5, 8, 9, 10, 11, 11.2, 12 - The user is a preferred customer and a MSB member successfully withdraws money, doesn’t print a receipt, and chooses to withdraw more from their account

1, 2, 2.1, 2.1.1, 3, 4, 5, 8, 9, 10, 11, 11.2, 12.1 - The user is a preferred customer and a MSB member successfully withdraws money, doesn’t print a receipt, and chooses to return to the transaction list

1, 2, 3, 4, 5, 5.1, 5.2, 5.3, 6, 6.1, 6.1.1, 7, 7.1, 7.2, 7.3 - The user is not a preferred customer nor a MSB member and attempts to withdraw more than 500 dollars, then tries to withdraw money from their account, but the ATM does not have enough money

1, 2, 3, 4, 5, 5.1, 5.2, 5.3, 6, 6.1, 6.1.1, 8, 9, 10, 11, 11.1, 12 - The user is not a preferred customer nor a MSB member, attempts to withdraw more than 500 dollars, then successfully withdraws money and prints a receipt, and chooses to withdraw more from their account

1, 2, 3, 4, 5, 5.1, 5.2, 5.3, 6, 6.1, 6.1.1, 8, 9, 10, 11, 11.1, 12, 12.1 - The user is not a preferred customer nor a MSB member, attempts to withdraw more than 500 dollars, then successfully withdraws money, prints a receipt, and chooses to return to the transaction list

1, 2, 3, 4, 5, 5.1, 5.2, 5.3, 6, 6.1, 6.1.1, 8, 9, 10, 11, 11.2, 12 - The user isn’t a preferred customer and not a MSB member successfully, attempts to withdraw more than 500 dollars, then withdraws money, doesn't print a receipt, and chooses to withdraw more from their account

1, 2, 3, 4, 5, 5.1, 5.2, 5.3, 6, 6.1, 6.1.1, 8, 9, 10, 11, 11.2, 12.1 - The user is not a preferred customer nor a MSB member and attempts to withdraw more than 500 dollars, then successfully withdraws money, doesn’t print a receipt, and chooses to return to the transaction list

1, 2, 3, 4, 5, 5.1, 5.2, 5.3, 6, 6.1, 6.1.2, 6.1.2.1, 6.1.2.2, 6.1.2.3 - The user is not a preferred customer nor a MSB member and attempts to withdraw more than 500 dollars, then tries to withdraw money from their account, but the user does not have enough money to do the transaction

1, 2, 3, 4, 5, 5.1, 5.2, 5.3, 7, 7.1, 7.2, 7.3 - The user is not a preferred customer but is a MSB member and attempts to withdraw more than 500 dollars, then tries to withdraw money from their account, but the ATM does not have enough money

1, 2, 3, 4, 5, 5.1, 5.2, 5.3, 8, 9, 10, 11, 11.1, 12 - The user is not a preferred customer but is a MSB member and attempts to withdraw more than 500 dollars, then successfully withdraws money and prints a receipt, and chooses to withdraw more from their account

1, 2, 3, 4, 5, 5.1, 5.2, 5.3, 8, 9, 10, 11, 11.1, 12.1 - The user is not a preferred customer but is a MSB member and attempts to withdraw more than 500 dollars, then successfully withdraws money, prints a receipt, and chooses to return to the transaction list

1, 2, 3, 4, 5, 5.1, 5.2, 5.3, 8, 9, 10, 11, 11.2, 12 - The user is not a preferred customer but is a MSB member and attempts to withdraw more than 500 dollars, then successfully withdraws money, doesn't print a receipt, and chooses to withdraw more from their account

1, 2, 3, 4, 5, 5.1, 5.2, 5.3, 8, 9, 10, 11, 11.2, 12.1 - The user is not a preferred customer but is a MSB member and attempts to withdraw more than 500 dollars, then successfully withdraws money, doesn’t print a receipt, and chooses to return to the transaction list

1, 2, 3, 4, 5, 6, 6.1, 6.1.1, 7, 7.1, 7.2, 7.3 -The user, who is not a preferred customer and not a MSB member, tries to withdraw money, but The ATM does not have enough money

1, 2, 3, 4, 5, 6, 6.1, 6.1.1, 8, 9, 10, 11, 11.1, 12 - The user is not a preferred customer nor a MSB member, successfully withdraws money and prints a receipt, and chooses to withdraw more from their account

1, 2, 3, 4, 5, 6, 6.1, 6.1.1, 8, 9, 10, 11, 11.1, 12, 12.1 - The user is not a preferred customer nor a MSB member, successfully withdraws money, prints a receipt, and chooses to return to the transaction list

1, 2, 3, 4, 5, 6, 6.1, 6.1.1, 8, 9, 10, 11, 11.2, 12 - The user isn’t a preferred customer and not a MSB member successfully, withdraws money, doesn't print a receipt, and chooses to withdraw more from their account

1, 2, 3, 4, 5, 6, 6.1, 6.1.1, 8, 9, 10, 11, 11.2, 12.1 - The user is not a preferred customer nor a MSB member successfully withdraws money, doesn’t print a receipt, and chooses to return to the transaction list

1, 2, 3, 4, 5, 6, 6.1, 6.1.2, 6.1.2.1, 6.1.2.2, 6.1.2.3 - The user is not a preferred customer nor a MSB member tries to withdraw money, but doesn’t have enough money in their account to do the transaction

1, 2, 3, 4, 5, 7, 7.1, 7.2, 7.3 - The user is not a preferred customer but is a MSB member tries to withdraw money butt he ATm doesn’t have enough money

1, 2, 3, 4, 5, 8, 9, 10, 11, 11.1, 12 - The user is not a preferred customer but is a MSB member successfully withdraws money and prints a receipt, and chooses to withdraw more from their account

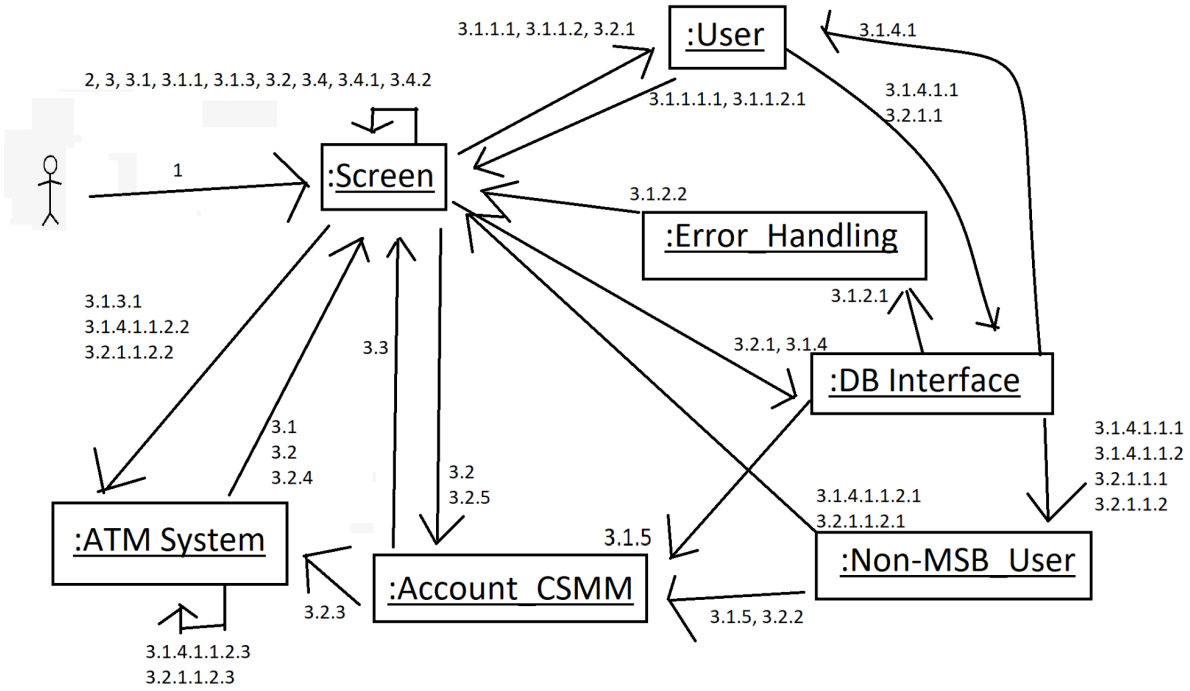
1, 2, 3, 4, 5, 8, 9, 10, 11, 11.1, 12.1 - The user is not a preferred customer but is a MSB member and successfully withdraws money, prints a receipt, and chooses to return to the transaction list

1, 2, 3, 4, 5, 8, 9, 10, 11, 11.2, 12 - The user is not a preferred customer but is a MSB member and successfully withdraws money, doesn't print a receipt, and chooses to withdraw more from their account

1, 2, 3, 4, 5, 8, 9, 10, 11, 11.2, 12.1 - The user is not a preferred customer but is a MSB member successfully withdraws money, doesn’t print a receipt, and chooses to return to the transaction list

Communication Diagrams: Deposit

Diagram:



Operation Sequence:

1: display_deposit_message( )

2: display_account_CSMM()

3: ask_cash_or_check( )

3.1: user_choice == “check”

3.1.1: display_nohold_notice( )

3.1.1.1: user_identity == “preferred”

3.1.1.1.1: display_nohold_notice( )

3.1.1.2: user_identity == “unpreferred”

3.1.1.2.1: display_hold_notice( ), deposit_hold_remaining: int days

3.1.2: communicate_DB( )

3.1.2.1: display_error_message(x), user_low_funds_error( )

3.1.2.2: display_return( )

3.1.3: display_check_amount( )

3.1.3.1: eject_check()

3.1.3.2: display_return( )

3.1.4: communicate_DB( )

3.1.4.1: user_identity == “NonMSB”

3.1.4.1.1: capture_nonMSB_fee( )

3.1.4.1.1.1: user_NonMSB_fee = true

3.1.4.1.1.2: user_NonMSB_fee = false

3.1.4.1.1.2.1: display_error_message (user_low_funds_error( ))

3.1.4.1.1.2.2: eject_card( )

3.1.4.1.1.2.3: logout( )

3.1.5: depositFunds(account_number: int, amount: int)

3.2: user_choice == “cash”

3.2.1: user_identity == “NonMSB”

3.2.1.1: capture_nonMSB_fee( )

3.2.1.1.1: user_NonMSB_fee = true

3.2.1.1.2: user_NonMSB_fee = false

3.2.1.1.2.1: display_error_message(user_low_funds_error( ))

3.2.1.1.2.2: eject_card( )

3.2.1.1.2.3: logout( )

3.2.2: display_deposit_cash( )

3.2.3: eject_cash( )

3.2.4: display_amount_inserted( )

3.2.5: depositFunds(account_number: int, amount: int)

3.3: display_account_amount( )

3.4: display_print_receipt_querry(answer: bool)

3.4.1: answer = true

3.4.2: answer = false

4: display_deposit_message( )

4: display_user_options( )

Scenario Sequence:

1, 2, 3, 3.1, 3.1.1, 3.1.1.1, 3.1.1.1.1, 3.1.2, 3.1.2.1, 3.1.2.2 - A preferred user wants to cash a check, but the person who wrote the check does not have that much money in their account.

1, 2, 3, 3.1, 3.1.1, 3.1.1.1, 3.1.1.1.1, 3.1.2, 3.1.3, 3.1.3.1, 3.1.3.2 - A preferred user wants to cash a check, but the ATM does not show the correct amount, the check is ejected, and the user is returned to the transaction page

1, 2, 3, 3.1, 3.1.1, 3.1.1.1, 3.1.1.1.1, 3.1.2, 3.1.3, 3.1.4, 3.1.4.1, 3.1.4.1.1, 3.1.4.1.1.1, 3.1.5, 3.3, 3.4, 3.4.1, 4 - A preferred user who is not a MSB member cashes a check, prints a receipt, and then decides to return to the transaction page

1, 2, 3, 3.1, 3.1.1, 3.1.1.1, 3.1.1.1.1, 3.1.2, 3.1.3, 3.1.4, 3.1.4.1, 3.1.4.1.1, 3.1.4.1.1.1, 3.1.5, 3.3, 3.4, 3.4.1, 4 - A preferred user who is not a MSB member cashes a check, prints a receipt, and then decides to deposit something else.

1, 2, 3, 3.1, 3.1.1, 3.1.1.1, 3.1.1.1.1, 3.1.2, 3.1.3, 3.1.4, 3.1.4.1, 3.1.4.1.1, 3.1.4.1.1.1, 3.1.5, 3.3, 3.4, 3.4.2, 4 - A preferred user who is not a MSB member cashes a check, does not print a receipt, and decides to deposit something else

1, 2, 3, 3.1, 3.1.1, 3.1.1.1, 3.1.1.1.1, 3.1.2, 3.1.3, 3.1.4, 3.1.4.1, 3.1.4.1.1, 3.1.4.1.1.1, 3.1.5, 3.3, 3.4, 3.4.2, 4 - A preferred user who is not a MSB member cashes a check, does not print a receipt, and then decides to return to the transaction page

1, 2, 3, 3.1, 3.1.1, 3.1.1.1, 3.1.1.1.1, 3.1.2, 3.1.3, 3.1.4, 3.1.4.1, 3.1.4.1.1.2, 3.1.4.1.1.2.1, 3.1.4.1.1.2.2, 3.1.4.1.1.2.3 - A preferred user who is not a MSB member wants to cash a check but doesn't have enough money for the transaction

1, 2, 3, 3.1, 3.1.1, 3.1.1.2, 3.1.1.2.1, 3.1.2, 3.1.2.1, 3.1.2.2 - A non-preferred user wants to cash a check, but the person who wrote the check does not have that much money in their account.

1, 2, 3, 3.1, 3.1.1, 3.1.1.2, 3.1.1.2.1, 3.1.2, 3.1.3, 3.1.3.1, 3.1.3.2 - A non-preferred user wants to cash a check, but the ATM does not show the correct amount, the check is ejected, and the user is returned to the transaction page

1, 2, 3, 3.1, 3.1.1, 3.1.1.2, 3.1.1.2.1, 3.1.2, 3.1.3, 3.1.4, 3.1.4.1, 3.1.4.1.1, 3.1.4.1.1.1, 3.1.5, 3.3, 3.4, 3.4.1, 4 - A non-preferred user who is not a MSB member cashes a check, prints a receipt, and decides to return to the transaction page

1, 2, 3, 3.1, 3.1.1, 3.1.1.2, 3.1.1.2.1, 3.1.2, 3.1.3, 3.1.4, 3.1.4.1, 3.1.4.1.1, 3.1.4.1.1.1, 3.1.5, 3.3, 3.4, 3.4.1, 4 - A non-preferred user who is not a MSB member cashes a check, prints a receipt, and decides to deposit something else.

1, 2, 3, 3.1, 3.1.1, 3.1.1.2, 3.1.1.2.1, 3.1.2, 3.1.3, 3.1.4, 3.1.4.1, 3.1.4.1.1, 3.1.4.1.1.1, 3.1.5, 3.3, 3.4, 3.4.2, 4 - A non-preferred user who is not a MSB member cashes a check, does not print a receipt, and decides to deposit something else

1, 2, 3, 3.1, 3.1.1, 3.1.1.2, 3.1.1.2.1, 3.1.2, 3.1.3, 3.1.4, 3.1.4.1, 3.1.4.1.1, 3.1.4.1.1.1, 3.1.5, 3.3, 3.4, 3.4.2, 4 - A non-preferred user who is not a MSB member cashes does not print a receipt, and decides to return to the transaction page

1, 2, 3, 3.1, 3.1.1, 3.1.1.2, 3.1.1.2.1, 3.1.2, 3.1.3, 3.1.4, 3.1.4.1, 3.1.4.1.1.2, 3.1.4.1.1.2.1, 3.1.4.1.1.2.2, 3.1.4.1.1.2.3 - A non-preferred user who is not a MSB member wants to cash a check, and does not have enough money to do the transaction. The user is logged out and their card is ejected.

1, 2, 3, 3.2, 3.2.1, 3.2.1.1, 3.2.1.1.1, 3.2.2, 3.2.3, 3.2.4, 3.2.5, 3.3, 3.4, 3.4.1, 4 - A user who is not a MSB member deposits cash, the ATM rejects some cash as it is either fake or it cannot be read, the user prints a receipt, and then decides to return to the transaction list

1, 2, 3, 3.2, 3.2.1, 3.2.1.1, 3.2.1.1.1, 3.2.2, 3.2.3, 3.2.4, 3.2.5, 3.3, 3.4, 3.4.1, 4 - A user who is not a MSB member deposits cash, the ATM rejects some cash as it is either fake or it cannot be read, the user prints a receipt, and then decides to deposit something else

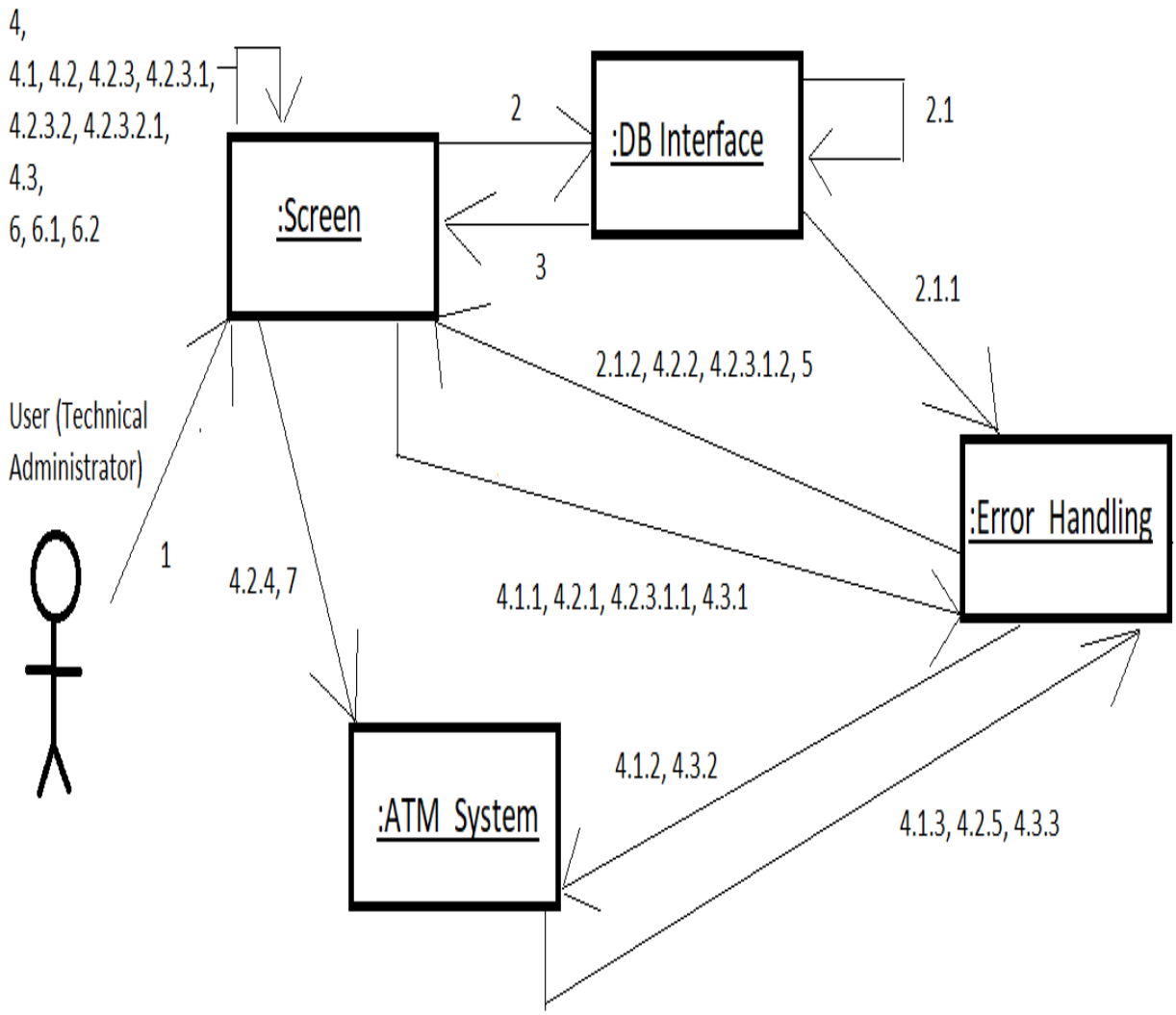
1, 2, 3, 3.2, 3.2.1, 3.2.1.1, 3.2.1.1.1, 3.2.2, 3.2.3, 3.2.4, 3.2.5, 3.3, 3.4, 3.4.2, 4 - A user who is not a MSB member deposits cash, the ATM rejects some cash as it is either fake or it cannot be read, the user doesn't print a receipt, and then decides to return to the transaction list

1, 2, 3, 3.2, 3.2.1, 3.2.1.1, 3.2.1.1.1, 3.2.2, 3.2.3, 3.2.4, 3.2.5, 3.3, 3.4, 3.4.2, 4 - A user who is not a MSB member deposits cash, the ATM rejects some cash as it is either fake or it cannot be read, the user doesn't print a receipt, and then decides to deposit something else

1, 2, 3, 3.2, 3.2.1, 3.2.1.1, 3.2.1.1.2, 3.2.1.1.2.1, 3.2.1.1.2.2, 3.2.1.1.2.3 - A user wants to deposit cash, but they are a non MSB member and don't have enough money to complete the transaction, so they have their card ejected and they get logged out

Communication Diagrams: Machine Maintenance

Diagram:



Operation Sequence:

1: display_TA_logon_screen( )

2: authenticate_user(ID_employee_number)

    2.1: authenticate_user(ID_employee_number) = false

        2.1.1: authenticate_error( )

        2.1.2: display_error_message(authenticate_error( ))

3: display_TA_options( )

4: capture_user_choice( )

    4.1: capture_user_choice == “Refill receipt paper”

        4.1.1: receipt_paper_low = true

        4.1.2: verify_TA_fix( )

        4.1.3: receipt_paper_low = false

    4.2: capture_user_choice == “Refill funds”

        4.2.1: low_funds = true

        4.2.2: display_TA_funds_request_message( )

        4.2.3: display_TA_accuracy_message(int x)

            4.2.3.1: entered_amount_displayed = false

                4.2.3.1.1: amount_inaccuracte_error( )

                4.2.3.1.2: display_TA_deposit_message( )

            4.2.3.2: entered_amount_displayed = true

                4.2.3.2.1: display_ATM_total_funds( )

        4.2.4: verify_TA_fix( )

        4.2.5: low_funds = false

    4.3: capture_user_choice == “Fix paper jam”

4.3.1: paper_jam = true

4.3.2: verify_TA_fix( )

4.3.3: paper_jam = false

5: display_TA_maintenance_message( )

6: capture_user_choice( )

6.1: more_maintenance = true

6.2: more_maintenance = false

7: logout( )

Scenario Sequence:

1, 2, 2.1, 2.1.1, 2.1.2 - Invalid employee ID

1, 2, 3, 4, 4.1, 4.1.1, 4.1.2, 4.1.3, 5, 6, 6.1, 3 - User is brought to menu where they may choose which maintenance work they wish to perform; after performing the selected maintenance, the user chooses to perform more maintenance and are brought back to the options menu

1, 2, 3, 4, 4.1, 4.1.1, 4.1.2, 4.1.3, 5, 6, 6.2, 7 - User is brought to menu where they may choose which maintenance work they wish to perform; after performing the selected maintenance, the user chooses not to continue maintenance and logouts

1, 7 - User enters Technical Administrator logon screen and logouts or user enters Technical Administrator logon screen and the system timed out

Class Diagram Implementations

User	class User{
	public:
	string getFName( );
	void setFName(string first_name);
	string getLName( );
	void setLName(string last_name);
	string getMailingAddress( );
	void setMailingAddress (string street,
	string city, string stateCode, int zipCode);
	private:
	string first_name;
	string last_name;
	string mailing_address; // stateCode = MT,
	// zipCode{5 digits}
	int ID_card; // 12 digits
	int ID_pin; // 4 digits
	string user_identity; // Preferred_User,
	// Unpreferred_User
	// nonMSB_User
	};

# first_name: string # last_name: string # mailing_address: string # ID_card: int {12 digits} # ID_pin: int {4 digits} # user_identity: string {Preferred_User, Unpreferred_User, nonMSB_User)	
+ getFName( ): string + setFName(first_name: string) + getLName( ): string + setLName(last_name: string) + getMailingAddress( ): string + setMailingAddress(street: string, city: string, stateCode: string = MT, zipCode: int {5 digits})	

Preferred_User (Static Relationship to User)	class Preferred_User{
	private:
	bool lock_status;
	bool overdraft_protection;
	};

- lock_status: bool - overdraft_protection: bool	

Unpreferred_User (Static Relationship to User)	class Unpreferred_User{
	private:
	bool lock_status;
	int deposit_hold_remaining;
	};

- lock_status: bool - deposit_hold_remaining: int	

MSB_User (Static Relationship to User)	class MSB_User{
	private:
	bool lock_status;
	bool overdraft_protection;
	};

- lock_status: bool - overdraft_protection: bool	

Non-MSB_User (Static Relationship to User)	class Non-MSB_User{
	private:
	bool lock_status;
	bool user_nonMSB_fee;
	};

- lock_status: bool - user_nonMSB_fee: bool	

Technical_Administrator	class Technical_Administrator{
	public:
	string getFName( );
	void setFName(string first_name);
	string getLName( );
	void setLName(string last_name);
	int getEmployeeNumber( );
	void setEmployeeNumber(int employee_number);
	string getStatus( );
	void setStatus(string status);
	private:
	string first_name;
	string last_name:
	int ID_employee_status; // 7 digits
	string employment_status; // active, suspended,
	// inactive
	};

- first_name: string - last_name: string - ID_employee_number: int {7 digits} - employment_status: string {active, suspended, inactive}	

+ getFName( ): string + setFName(first_name: string) + getLName( ): string + setLName(last_name: string) + getEmployeeNumber( ): int + setEmployeeNumber(employee_number: int) + getStatus( ): string + setStatus(status: string)	

Account	class Account{
	public:
	void checkAccountBalance(int account_number);
	private:
	int account_numer; //10 digits
	double account_amount;
	};

# account_number: int {10 digits} # account_amount: double	
+ checkAccountBalance(account_number: int)	

Account_CSMM (Static Relationship to Account)

- account_type: string {checkings, savings, money market}

- widthdrawal_amount_remaining_today: int {500 per day, multiple of 10}

+ withdrawalFunds(account_number: int, funds: int%10)

+ depositFunds(account_number: int, amount: int)

+ transferFundsCSMM(account_number: int, account_number: int, amount: double)

+ transferFundsCL(account_number: int, account_number: int, amount: double)

+ transferFundsM(account_number: int, account_number: int, amount: double)

```
class Technical_Administrator{
public:
   void withdrawalFunds ( int account_number, int funds);       //multiples of 10
   void depositFunds( int account_number, int amount);
   void transferFundsCSMM( int account_number,  int account_number, double amount);
   void  transferFundsCL( int account_number,  int account_number, double amount);
   void  transferFundsM( int account_number,  int account_number, double amount);
private:
   string account_type;                                  //checkings, savings, money market
   int widthdrawal_amount_remaining_today;     //500 per day, multiples of 10
};
```

Account_CL (Static Relationship to Account)

- account_type: string {consumer loan}

- user_loan_allotted: int

+ withdrawalFundsCL(user_loan_allotted: int, funds: int)

class Account_CL{
public:
void withdrawalFundsCL(int user_loan_allotted, int funds);
private:
string account_type; //consumer loan
int user_loan_allotted;
};

Account_M (Static Relationship to Account)	class Account_M{
	private:
	string account_type; //mortgage
	};

- account_type: string {mortgage}	

Screen	class Screen{
	public:
	void ask_cash_or_check( );
	void capture_ID_card(int user.ID_card);
	//12 digits
	void capture_ID_pin(int uder.ID_card);
	//4 digits
	void capture_user_chopice( );
	void display_accounts( );
	void display_accounts_amount(double
	account.amount);
	void display_account_CSMM( );
	void display_accountCSMMCL( );
	void display_accountbalance_options( );

- x-coordinate: int - y-coordinate: int - color: hex - border: int - pixel: int - brightness: int - sharpness: int - more_maintenace: bool - entered_amount: int - entered_amount_displayed: bool - user_choice: string	

- ask_cash_or_check( )
 	
- capture_ID_card(user.ID_card: int {12 digits})
 	
- capture_ID_pin(user.ID_pin: int {4 digits})
 	
- capture_user_choice( )
 	
+ display_accounts( )
 	
+ display_account_amount(account.amount: double)
 	
+ display_account_CSMM( )
 	
+ display_account_CSMMCL( )
 	
+ display_account_balance_options( )
 	
+ display_amount_inserted( )
 	
+ display_ATM_total_funds( ): string
 	
+ display_check_amount( )
 	
+ display_deposit_cash( )
 	
+ display_deposit-check( )
 	
+ display_deposit_message( )
 	
+ display_error_message(error_handling.error_number x)
 	
+ display_fee_notice( )
 	
+display_hold_notice( )
 	
+ display_logon_screen( )
 	
+display_nohold_notice( )
 	
+ display_print_receipt_querry(answer: bool)
	void display_amount_inserted( );
	string display_ATM_total_funds( );
	void display_check-amount( );
	void display_deposit_amount( );
	void display_deposit_check( );
	void display_deposit-check( );
	void display_deposit_message( );
	void display_error_message (
	error_handling.error_number x);
	void display_fee_notice( );
	void display_hold_notice( );
	void display_logon_screen( );
	void display_nohold_notice( );
	void display_print_reciept_querry (bool
	answer);
	void display_remaining_loan( );
	void display_return( );
	void display_transfer_notice( );
	void display_withdrawal_notice( );
	void display_preferred_withdraw_notice();
	void display_TA_accuracy_message(
	int entered_amount);
	void display_TA_deposit_message( );
	void display_TA_funds_request_message();
	void display_TA_logon_screen( );
	void display_TA_maintenance_message( );
	void display_TA_options( );
	void display_user_options( );
	void display_user_locked_message(
	error_handling: error_number x);
	int display_enter_money( );
	private:
	int x-coordinate;
	int y-coordinate;
	hex color;
	int border;
	int pixel;
	int brightness;
	int sharpness;
	bool more_maintenance;

+ display_remaining_loan( ) + display_return( ) + display_tranfser_notice( ) + display_withdraw_notice( ) + display_preferred_withdraw_notice( ) + display_TA_accuracy_message(entered_amount: int) + display_TA_deposit_message( ) + display_TA_funds_request_message( ) + display_TA_logon_screen( ) + display_TA_maintenance_message( ) + display_TA_options( ) + display_user_options( ) + display_user_locked_message(error_handling: error_number x) + display_enter_money( ): int	int entered_amount; bool entered_amount_displayed; string user_choice; };

ATM System	class ATM System{
	public:
	bool user_lock(int Error_handling.error_number);
	bool employee_lock(string Technical_Administrator.empoloyment_status);
	void eject_card( );
	void eject_cash( );
	void eject_check( );
	void verify_TA_fix( );
	void logout( );

+ user_lock(Error_Handling.error_number: int): bool + employee_lock(Technical_Administrator.empoloyment_status: string): bool + eject_card( ) + eject_cash( ) + eject_check( ) + verify_TA_fix( ) + logout( )	};

DB_Interface
- user_status: string (verified, invalid)
+ authenticate_user(ID_card: int || ID_pin: int || ID_employee_number: int): bool + communicate_DB( ) {connects to central database} + capture_user_loan_interest_amount( ): int + capture_NonMSB_fee( ): bool + capture_user_ID( ): string

```
class DB_Interface{
public:
    bool authenticate_user( int ID_card, int ID_pint, int ID_employee_number);
    void communicate_DB( )  //connects to central database
    int capture_user_loan_interest_amount( );
    bool capture_NonMSB_fee( );
    string capture_user_ID( );
private:
    string user_status;  //verified, invalid
};
```

Error_Handling

- error_number: int
- error_count: int
- receipt_paper_low: bool
- low_funds: bool
- paper_jam: bool

+ authenticate_error( ): int

+ ATM_low_funds_error( ): int

+ user_low_funds_error( ): int

+ receipt_paper_low_error( ): int

+ amount_inaccurate_error( ): int

+ paper_jam_error( ): int

+ withdrew_more_than_500_error( ): int

```
class DB_Interface{
public:
    int authenticate_error( );
    int ATM_low_funds_error( );
    int user_low_funds_error( );
    int receipt_paper_low_error( );
    int amount_inaccurate_error( );
    int paper_jam_error( );
    int withdrew_more_than_500_error( );
private:
    int error_number;
    int error_count;
    bool receipt_paper_low;
    bool low_funds;
    bool paper_jam;
};
```