**Banking System**

*Software Design Specification*

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Revision** | **Description** | **Author** |
| 04/02/2025 | 1.00 | Initial Version | Harven Dhanota |
| 04/02/2025 | 1.01 | Revision: Added Additional ATM Module and Teller Module Use Cases | Harven Dhanota |
| 04/03/2025 | 1.02 | Revision: Added Teller Module Use Case | Harven Dhanota |
| 04/03/2025 | 1.03 | Revision: Added More Teller Module Use Cases | Harven Dhanota |
| 04/03/2025 | 1.04 | Revision: Added Even More Teller Module Use Cases | Harven Dhanota |
| 04/04/2025 | 1.05 | Revision: Added Central Server Use Cases | Harven Dhanota |
| 04/04/2025 | 1.06 | Revision: Clarified Distinctions Between User Account and Financial Account | Harven Dhanota |
| 04/04/2025 | 1.07 | Revision: Added Another ATM Module Use Case | Harven Dhanota |
| 04/05/2025 | 1.08 | Revision: Added Additional ATM Module and Teller Module Use Cases | Harven Dhanota |
| 04/07/2025 | 1.09 | Revision: Modified Use Cases 2.2.6, 2.2.7, and 2.2.12; Deleted “Check Deposit” Use Case from ATM Module | Harven Dhanota |
| 04/07/2025 | 1.10 | Revision: Added References to Design Diagrams; Added Additional Use Cases for ATM Module and Teller Module | Jaishnoor Kaur |
| 04/07/2025 | 1.11 | Revision: Cleaned Up Use Cases | Harven Dhanota |
| 04/07/2025 | 1.12 | Revision: Added Reference to Sequence Diagram | Zachary Wickliffe |
| 04/08/2025 | 1.13 | Revision: Added Relevant SRS Reqs.; Added Related Use Cases to each Use Case | Harven Dhanota |
| 04/09/2025 | 1.14 | Revision: Added Stubs for Central Module Use Cases; Fixed Some Use Cases (up to 2.2.12) | Jaishnoor Kaur |
| 04/09/2025 | 1.15 | Revision: Added Reference to Use Case Diagrams | Harven Dhanota |
| 04/09/2025 | 1.16 | Revision: Fixed Remaining Use Cases; Completed Central Module Use Case Stubs | Jaishnoor Kaur |
| 04/09/2025 | 1.17 | Revision: Cleaned Up Use Cases; Corrected Page Numbers in Table of Contents | Harven Dhanota |
| 05/05/2025 | 1.18 | Revision: Minor edits | Jaishnoor Kaur |
|  |  |  |  |
|  |  |  |  |

Table of Contents

**1.** **Diagrams** [**4**](#__RefHeading___Toc19440719)

1.1. Use Case Diagram [4](#__RefHeading___Toc19440720)

1.2. Class Diagram [4](#__RefHeading___Toc19440721)

1.3. Sequence Diagram [4](#__RefHeading___Toc19440722)

1.4. UI Prototype Diagrams [4](#__RefHeading___Toc19440722)

**2.** **Use Cases** [**5**](#__RefHeading___Toc19440724)

2.1. ATM Module Use Cases [5](#__RefHeading___Toc19440725)

2.2. Teller Module Use Cases [9](#__RefHeading___Toc19440726)

2.3. Central Server Module Use Cases 19

1. **Diagrams**
   1. **Use Case Diagrams**

See Attached File: UML\_UseCaseDiagrams.pdf

* 1. **Class Diagrams**

See Attached File: UML\_ClassDiagrams.png

* 1. **Sequence Diagram**

See Attached File: SequenceDiagram.pdf

* 1. **UI Prototype Diagrams**

See Attached Files: ATM\_GUI\_Prototype.png and Teller\_GUI\_Prototype.png

1. **Use Cases**
   1. **ATM Module Use Cases**

2.1.1. Customer Logging-In to ATM

Relevant Requirement(s) from SRS: 3.1.2.1

Precondition(s): the ATM Module is online and connected to the central server

Postcondition(s): the customer has access to their financial account(s)

Basic Flow:

(1) the customer initiates a log-in request

(2) the ATM Module requests the user for their full name, phone number, and password

(3) the customer enters their name, phone number, and password

(4) the ATM Module sends a LOGINREQATM message with the customer’s credentials to the central server

(5) the central server validates the customer’s credentials

(6) the central server grants access and sends the customer’s financial accounts list back to the ATM Module

(7) the ATM Module displays the customer’s list of financial accounts to the customer

Alternate Flows:

(1) if the customer enters invalid credentials, the central server sends a failure message to the ATM Module, rather than sending the customer’s financial account information

Exceptions:

(1) the central server does not receive any credentials

(2) the customer’s financial account information is not properly sent back to the ATM Module after the central server validates the customer’s credentials

(3) messages time out

Related Use Case(s): 2.1.2, 2.1.7, 2.1.8

2.1.2. Customer Selects Financial Account

Relevant Requirement(s) from SRS: [none]

Precondition(s): the ATM Module is online and connected to the central server, and the customer is logged-in to their user account

Postcondition(s): the customer can view all menu options for the selected financial account

Basic Flow:

(1) the customer selects one of their financial accounts

(2) the ATM Module sends a request to the central server to request access

(3) central module checks to see if the financial account is currently being accessed

(3) if the financial account is not being accessed, then the central server sets a flag on the financial account to mark it as being currently in access and allows requesting ATM access to said account

(4) the ATM Module displays a menu of options for the selected financial account

Alternate Flows:

(1) if a financial account is already being accessed and is attempted to be accessed again from another ATM Module or from a Teller Module, the central server sends a failure message to the ATM Module, rather than allowing the customer to view the menu options for the selected financial account

Exceptions:

(1) the central server does not properly verify that the financial account is not currently being accessed

(2) the financial account status is not set to “currently in access”

Related Use Case(s): 2.1.3, 2.1.4, 2.1.5, 2.1.6, 2.1.8, 2.1.9

2.1.3. Customer Exits Financial Account

Relevant Requirement(s) from SRS: [none]

Precondition(s): the ATM Module is online and connected to the central server, and the customer is currently accessing one of their financial accounts

Postcondition(s): the customer has exited the financial account and returned to the main menu, which contains a list of financial accounts

Basic Flow:

(1) the customer selects to exit the financial account

(2) the ATM Module sends a request to the central server to flag the financial account as no longer being accessed

(3) the ATM Module displays a list of the customer’s financial accounts

Alternate Flows:

[none]

Exceptions:

(1) the financial account status is not set to “no longer in access”

Related Use Case(s): 2.1.2, 2.1.8

2.1.4. Deposit into ATM

Relevant Requirement(s) from SRS: 3.1.2.5

Precondition(s): the ATM Module is online and connected to the central server, and the customer is logged-in to their user account

Postcondition(s): the customer’s financial account balance has increased by the amount deposited

Basic Flow:

(1) the customer selects a financial account

(2) the customer selects to deposit into that financial account

(3) the ATM Module prompts the customer to input how much they will deposit

(4) the ATM Module sends a message to the central server with deposit amount

(5) the Central Module verifies that the amount to be deposited does not exceed the $4,000 daily limit for ATM deposits

(6) if the amount to be deposited is valid, the Central Module sends a message to the ATM granting request

(7) ATM Module prompts the customer to insert their cash and then sends a confirmation message to server

(8) the central server updates the amount deposited by that customer, so that they cannot exceed the daily limit

(9) the central server updates the customer’s financial account balance

(10) the central server records the transaction into the user account’s transaction history

Alternate Flows:

[none]

Exceptions:

(1) the central server does not properly update the customer’s financial account balance

(2) the central server does not properly update the amount of cash that the customer can deposit before they exceed their daily limit

(3) the central server does not properly record the transaction into the transaction history

Related Use Case(s): 2.1.2, 2.1.8

2.1.5. Withdrawing from ATM

Relevant Requirement(s) from SRS: 3.1.2.4

Precondition(s): the ATM Module is online and connected to the central server, and the customer is logged-in to their user account

Postcondition(s): the customer’s financial account balance has decreased by the amount withdrawn, and the customer now has cash equal to the amount withdrawn

Basic Flow:

(1) the customer selects a financial account

(2) the customer selects to withdraw from that financial account

(3) the ATM Module prompts the user to input the amount they want to withdraw

(4) the ATM Module verifies that the ATM has enough cash within its reserves to perform the withdrawal

(5) if the ATM has enough cash, the ATM Module sends the withdrawal request to the central server

(6) the central server verifies that ATM withdrawal limit for this account has not been reached and that the account has sufficient funds

(7) the central module temporarily updates the customer’s financial account balance and sends a request granted message to the ATM, which gives the amount of cash requested to the customer

(8) the ATM Module sends a confirmation to the central server, acknowledging that the cash was successfully withdrawn

(9) the central server permanently updates the customer’s financial account balance

(10) the central server records the transaction into the user account’s transaction history and also updates withdrawal limit

Alternate Flows:

(1) if the customer tries to withdraw an amount greater than their current financial account balance or above Atm withdrawal limit, the central server sends a failure message to the ATM Module, rather than temporarily updating the customer’s financial account balance

(2) if the ATM does not have enough cash in its reserves to perform the withdrawal, the ATM Module displays a failure message, rather than sending a request to the central server

Exceptions:

(1) the central server does not properly update the customer’s financial account balance

(2) the central server does not properly record the transaction into the transaction history

(3) the central server does not receive the withdrawal request from the ATM Module

(4) the ATM Module does not properly verify that there is enough cash in the ATM to perform the withdrawal

(5) the central module does not properly verify that account has sufficient remaining withdrawal limit and funds

Related Use Case(s): 2.1.2, 2.1.8

2.1.6. Checking Balance in Financial Accounts

Relevant Requirement(s) from SRS: 3.1.2.2

Precondition(s): the ATM Module is online and connected to the central server, and the customer is logged-in to their user account

Postcondition(s): the customer can view their financial account balance

Basic Flow:

(1) the customer selects a financial account

(2) the customer selects to view the account balance

(3) the ATM sends a request to the central server

(4) the central server sends the requested account balance back to the ATM Module

(5) the ATM Module displays the balance

Alternate Flows:

[none]

Exceptions:

(1) the central server does not receive the request from the ATM Module

Related Use Case(s): 2.1.2, 2.1.8

2.1.7. Customer Logging-Out of ATM

Relevant Requirement(s) from SRS: [none]

Precondition(s): the ATM Module is online and connected to the central server, and the customer is logged-in to their user account

Postcondition(s): the customer is logged out of their user account

Basic Flow:

(1) the customer initiates a log-out request

(2) the ATM Module prompts the customer to confirm that they want to log out of the ATM

(3) if the customer confirms the log-out request, then the ATM Module sends the request to the central server to log the user out

(3) the central server marks the user account as no longer being in access

(4) the central server sends a confirmation back to the ATM Module to notify the customer that they are no longer logged in

(5) the ATM Module returns to the log-in page, ready for the next customer to log in

Alternate Flows:

(1) if the customer does not confirm that they want to log out, then the log-out request is canceled and nothing is sent to the central server

Exceptions:

(1) the central server does not receive the log-out request

(2) the confirmation message is not properly sent back to the ATM Module after the user account is marked as not being in access

(3) the customer’s user account status is not set to “no longer in access”

Related Use Case(s): 2.1.1

2.1.8. Automatic Log-Out from ATM

Relevant Requirement(s) from SRS: 3.1.2.6

Precondition(s): the ATM Module is online and connected to the central server, the customer is logged-in to their user account, and there is no activity or input from the customer for 60 seconds

Postcondition(s): the customer is logged out of their user account

Basic Flow:

(1) there is no input or activity from the customer for 50 seconds

(2) the ATM Module prompts the customer to confirm that they are still present at the ATM, and a visible 10-second timer is shown to the user

(3) if the customer does not confirm that they are still present before the 10-second timer runs out, then the ATM Module sends a request to the central server to automatically log out the user

(4) the central server marks the user account as no longer being in access

(5) the ATM Module returns to the log-in page, ready for the next customer to log in

Alternate Flows:

(1) if the customer does confirm that they are still present at the ATM before the 10-second timer runs out, then nothing is sent to the central server and the automatic log-out process is canceled altogether

Exceptions:

(1) the central server does not receive the automatic log-out request

(2) the customer’s user account status is not set to “no longer in access”

Related Use Case(s): 2.1.1

2.1.9. View Transaction History

Relevant Requirement(s) from SRS: 3.1.2.2

Precondition(s): the ATM Module is online and connected to the central server, and the customer is logged-in to their user account

Postcondition(s): the customer can view all transactions that have occurred on one of their financial accounts

Basic Flow:

(1) the customer selects a financial account

(2) the customer selects to view the transaction history

(3) a request is sent to the central server to get the transaction history associated with the financial account that the customer selected

(4) the transaction history is sent back to the ATM Module from the central server

(5) the transaction history associated with the financial account selected is shown to the customer

Alternate Flows:

[none]

Exceptions:

(1) the central server does not receive the request to access financial account’s transaction history

(2) the transaction history is not properly sent back to the ATM Module

Related Use Case(s): 2.1.2, 2.1.8

* 1. **Teller Module Use Cases**

2.2.1. Teller Logging-In to Teller Module

Relevant Requirement(s) from SRS: 3.1.3.1

Precondition(s): the Teller Module is online and connected to the central server

Postcondition(s): the bank employee has access to the Teller Module

Basic Flow:

(1) the bank employee initiates a log-in request

(2) the Teller Module requests the user for their bank-issued Employee ID and their password

(3) the bank employee enters their Employee ID and password

(4) the Teller Module sends the employee’s credentials to the central server

(5) the central server validates the employee’s credentials

(6) if the credentials are correct, the central server sets the teller’s status to logged-in

(7) the central server sends list of all user accounts back to the Teller Module where the log-in request was initiated

(8) the Teller Module displays initial teller options

Alternate Flows:

(1) if the bank employee enters invalid credentials, the central server sends a failure message to the Teller Module, rather than sending all customer data

Exceptions:

(1) the central server does not receive any credentials

(2) the user account data is not properly sent back to the Teller Module after the central server validates the employee’s credentials

(3) the teller’s account status is not set to “currently logged in”

Related Use Case(s): 2.2.2, 2.2.3, 2.2.18

2.2.2. Teller Creates New User Account for a Customer

Relevant Requirement(s) from SRS: 3.1.3.4

Precondition(s): the Teller Module is online and connected to the central server, and the teller is logged-in to the Teller Module

Postcondition(s): the customer now has a user account

Basic Flow:

(1) the teller initiates a user account creation request

(2) the Teller Module prompts the teller to input the full name, phone number, and address of the customer

(3) the Teller Module prompts the teller for a password for the customer’s user account

(4) the customer themselves inputs a password for their user account

(5) the Teller Module sends the customer’s details and password to the central server

(6) the central server checks if there already exists a user account with the provided name and phone number

(7) if no user account exists, then the central server creates a new user account with the provided credentials

(8) the central server sends a confirmation message back to the Teller Module to notify the teller that a new user account was successfully created

Alternate Flows:

(1) if a user account already exists with the provided name and phone number, then the central server sends a message back to the Teller Module to notify the teller than a user account already exists, rather than creating a new user account with the provided credentials

Exceptions:

(1) the central server does not properly receive the customer’s credentials

(2) the central server does not find a customer’s existing user account before creating a new user account, assuming the customer does have an existing user account

(3) the confirmation message is not properly sent back to the Teller Module after the new user account is created

Related Use Case(s): 2.2.3, 2.2.7, 2.2.15

2.2.3. Teller Selects User Account

Relevant Requirement(s) from SRS: [none]

Precondition(s): the Teller Module is online and connected to the central server, and the teller has verified the customer’s identity

Postcondition(s): the teller has access to a list of the customer’s financial accounts

Basic Flow:

(1) the teller selects to access a customer’s user account

(2) the teller selects a user account by clicking on provided list of user accounts or manually entering user details and clicking select

(2) the Teller Module sends a ACCESSCAREQ message containing full name and phone number to central module

(3) the central server searches for a user account with the provided credentials

(4) central module verifies that user account is not currently in access

(5) central module sends ACCESSCAREQGRANTED message and updates in\_access to true

(6) the teller Module displays options (update user information, create new bank account, select financial account)

Alternate Flows:

(1) if no user account can be found with the provided credentials, the central server sends a failure message back to the Teller Module, and the process of accessing a user account is canceled altogether

Exceptions:

(1) the credentials are not properly sent to the central server

(2) the central server fails to find the user account

(3) the central server sends incorrect financial accounts back to the Teller Module

Related Use Case(s): 2.2.4, 2.2.5, 2.2.7, 2.2.13, 2.2.14, 2.2.15

2.2.4. Teller Exits User Account

Relevant Requirement(s) from SRS: [none]

Precondition(s): the Teller Module is online and connected to the central server, and the teller is currently accessing a customer’s user account

Postcondition(s): the teller is no longer accessing the customer’s user account

Basic Flow:

(1) the teller selects to exit a customer’s user account

(2) the Teller Module sends a message to central server, which marks user account not in access and sends a message back to teller module

(3) exits the customer’s user account and returns to the main menu, which displays a list of all user accounts

Alternate Flows:

[none]

Exceptions:

1. Account does not correctly gets marked not in access

Related Use Case(s): 2.2.3

2.2.5. Teller Selects Customer’s Financial Account

Relevant Requirement(s) from SRS: [none]

Precondition(s): the Teller Module is online and connected to the central server, and the teller has selected the customer’s user account

Postcondition(s): the customer can view all menu options for the selected financial account

Basic Flow:

(1) the teller selects one of the customer’s financial accounts

(2) the Teller Module sends a request to the central server to see if the financial account is currently being accessed

(3) if the financial account is not being accessed, then the central server sets a flag on the financial account to mark it as being currently in access and sends an access granted message

(4) the Teller Module displays a menu of options for the selected financial account

Alternate Flows:

(1) if a financial account is already being accessed and is attempted to be accessed again from another Teller Module or from an ATM Module, the central server sends a failure message to the Teller Module, rather than allowing the teller to view the menu options for the selected financial account

Exceptions:

(1) the central server does not properly verify that the financial account is not currently being accessed

(2) the financial account status is not set to “currently in access”

Related Use Case(s): 2.2.3, 2.2.6, 2.2.8, 2.2.9, 2.2.10, 2.2.11, 2.2.12, 2.2.16, 2.2.17

2.2.6. Teller Exits Customer’s Financial Account

Relevant Requirement(s) from SRS: [none]

Precondition(s): the Teller Module is online and connected to the central server, and the teller is currently accessing one of the customer’s financial accounts

Postcondition(s): the teller has exited the financial account and returned to the main menu, which contains a list of the customer’s financial accounts

Basic Flow:

(1) the teller selects to exit the customer’s financial account

(2) the Teller Module sends a request to the central server

(3) the central module flags the financial account as no longer being accessed

(4) the Teller Module goes back to displaying a list of the customer’s financial accounts

Alternate Flows:

[none]

Exceptions:

(1) the financial account status is not set to “no longer in access”

Related Use Case(s): 2.2.5

2.2.7. Teller Creates Financial Account for a Customer

Relevant Requirement(s) from SRS: 3.1.3.4

Precondition(s): the Teller Module is online and connected to the central server, and the teller has verified the customer’s identity and is accessing customer’s user account

Postcondition(s): the customer now has one additional financial account

Basic Flow:

(1) the teller selects to create new financial account

(2) the Teller module prompts teller to select if the financial account will be a checking account or a savings account, and it prompts the teller to input the initial deposit amount

(3) outside this system, the teller takes the money for the initial deposit from the customer

(4) the teller sends the information from the Teller Module to the central server in a create bank account request message

(5) the central server creates the financial account and associated transaction history file

(6) the financial account’s creation and initial deposit are recorded in the financial account’s transaction history

(7) the central server sends a confirmation message back to the Teller Module to notify the teller that the financial account was successfully created

(8) the teller returns to the options pane for the customer’s user account

Alternate Flows:

(1) if the initial deposit for the financial account is zero, then the central server sends a message notifying the teller that the initial deposit amount must be non-zero, and the process of creating a financial account is then canceled altogether

Exceptions:

(1) the central server does not properly create the financial account files

(2) the confirmation message is not properly sent back to the Teller Module after the financial account is created

Related Use Case(s): 2.2.8, 2.2.9, 2.2.10, 2.2.11, 2.2.16, 2.2.17

2.2.8. Teller Deletes Financial Account for a Customer

Relevant Requirement(s) from SRS: 3.1.3.4

Precondition(s): the Teller Module is online and connected to the central server, and the teller has verified the customer’s identity and is accessing customer’s financial account

Postcondition(s): the customer now has one less financial account

Basic Flow:

(1) the teller selects to delete that financial account

(2) the Teller Module prompts the teller to confirm that they want to delete the user account

(3) if the teller confirms that they want to delete the user account, then the request is sent to the central server

(4) if the financial account has a balance of zero, then the central server deletes the files associated with that financial account and creates an account deletion log in account level history

(5) the central server sends a confirmation message back to the Teller Module to notify the teller that the financial account was successfully deleted

(6) the teller module returns to screen showing list of financial accounts (this list no longer includes deleted account)

Alternate Flows:

(1) if the teller does not confirm that they want to delete the financial account, then nothing is sent to the central server and the process of deleting a financial account is allowed to time out and is canceled altogether

(2) if the financial account does not have a balance of zero, then the central server sends a message notifying the teller that the financial account has a non-zero balance, and that they must withdraw that money before the financial account can be deleted

Exceptions:

(1) the central server does not properly delete the financial account files

(2) the central server does not verify that the financial account has a balance of zero before deleting the files

(3) the confirmation message is not properly sent back to the Teller Module after the financial account is deleted

Related Use Case(s): 2.2.7

2.2.9. Deposit by Teller

Relevant Requirement(s) from SRS: 3.1.3.3

Precondition(s): the Teller Module is online and connected to the central server, and the teller has verified the customer’s identity and is accessing the financial account

Postcondition(s): the customer’s financial account balance has increased by the amount deposited

Basic Flow:

(1) the teller selects to deposit into that financial account

(2) the teller takes the money from the customer (outside our system)

(3) the Teller Module prompts the teller to input the amount being deposited

(4) the Teller Module sends the deposit request to the central server

(5) the central module updates the customer’s financial account balance, increasing it by the amount deposited

(6) the central module records the transaction into the user account’s transaction history

(7) the central module sends a confirmation message to teller module

(8) the teller module returns to account’s options pane

Alternate Flows:

(1)

Exceptions:

(1) the central server does not properly update the customer’s financial account balance

(2) the central server does not receive the deposit request from the Teller Module

(3) the central server does not properly record the transaction into the transaction history

Related Use Case(s): 2.2.5

2.2.10. Withdrawing by Teller

Relevant Requirement(s) from SRS: 3.1.3.2

Precondition(s): the Teller Module is online and connected to the central server, and the teller has verified the customer’s identity and is accessing the financial account

Postcondition(s): the customer’s financial account balance has decreased by the amount withdrawn, and the customer now has cash equal to the amount withdrawn

Basic Flow:

(1) the teller selects to withdraw cash from that financial account

(2) the Teller Module prompts the teller to input the amount being withdrawn and once teller confirms, it sends the withdrawal request to the central server

(3) the central server verifies that input amount is valid and temporarily updates the customer’s financial account balance, and waits for the teller to confirm that they have given the cash to the customer

(4) once the cash has been given to the customer, the teller sends a confirmation message (via the Teller Module) to the central server

(5) the central server permanently updates the customer’s financial account balance

(6) the central server records the transaction into the user account’s transaction history

(7) the teller module returns to account’s options pane

Alternate Flows:

(1) if the teller tries to withdraw an amount greater than the customer’s current financial account balance, the central server sends a failure message to the Teller Module, rather than temporarily updating the customer’s financial account balance

Exceptions:

(1) the central server does not properly update the customer’s financial account balance

(2) the central server does not properly record the transaction into the transaction history

(3) the central server does not receive the withdrawal request from the Teller Module

Related Use Case(s): 2.2.5

2.2.11. Adding Additional Users to a Financial Account

Relevant Requirement(s) from SRS: 3.1.3.5

Precondition(s): the Teller Module is online and connected to the central server, and the teller has verified the identities of both the customer who owns the financial account and the customer that wants to be added to that financial account

Postcondition(s): an additional user is added to the financial account, and that user now has full access to the financial account they were added to

Basic Flow:

(1) the teller selects to add an additional user to that financial account

(2) the Teller Module prompts the teller to input the full name and phone number of the user that wants to be added to the financial account

(3) the Teller Module sends the credentials to the central server

(4) the central module verifies that the user being added has a user account

(5) if yes, then the financial account they are being added to is added to their user account and the suer account is added to financial account’s user list

(6) the central server sends a confirmation message back to the Teller Module to notify the teller that the user was successfully added to the financial account

(7) the central module adds the event to history logs

(8) the teller module returns to account’s options pane

Alternate Flows:

(1) if the user being added does not have a user account, then the central server sends a message back to the Teller Module to notify the teller that the user being added does not have a user account, and the process of adding a user to the financial account is canceled altogether

Exceptions:

(1) the central server does not properly receive the credentials of the user that wants to be added to the financial account

(2) the central server does not properly add the new user to the financial account

(3) the confirmation message is not properly sent back to the Teller Module after the user is added to the financial account

Related Use Case(s): 2.2.7, 2.2.12, 2.2.17

2.2.12. Removing Additional Users from a Financial Account

Relevant Requirement(s) from SRS: 3.1.3.5

Precondition(s): the Teller Module is online and connected to the central server, and the teller has verified the identity of the customer and is accessing financial account

Postcondition(s): the additional user is removed from the financial account, and that user can no longer access the financial account they were removed from

Basic Flow:

(1) the teller selects to remove a user from that financial account

(2) the Teller Module prompts the teller to select the name of the user that they want to remove from the financial account (current user is not displayed as an option)

(3) the Teller Module prompts the teller to confirm that they want to remove the selected user from the financial account

(4) if the teller confirms they want the user removed, then the Teller Module sends a request to the central server

(5) the central server removes that user from the financial account’s users list and removes the financial account from the user’s account list

(6) the central server sends a confirmation message back to the Teller Module to notify the teller that the user was successfully removed from the financial account

(7) the event is logged by central module

(8) the teller module returns to account’s options pane

Alternate Flows:

(1) if the teller does not confirm that they want the user removed, then nothing is sent to the central server and the process of removing a user is canceled altogether

Exceptions:

(1) the central server does not properly receive the name of the user that needs to be removed from the financial account

(2) the confirmation message is not properly sent back to the Teller Module after the is removed from the financial account

(3) the central server does not properly remove the user from the financial account

Related Use Case(s): 2.2.7, 2.2.11, 2.2.17

2.2.13. Update User Account Information

Relevant Requirement(s) from SRS: [none]

Precondition(s): the Teller Module is online and connected to the central server, and the teller has verified the customer’s identity

Postcondition(s): the customer’s user account information is updated

Basic Flow:

(1) the teller selects to edit/modify the customer’s user account information

(2) the teller updates the user account information, such as full name, mailing address, phone number, etc.

(3) once the teller has finished inputting the updates the user account information, the Teller Module sends the information to the central server

(4) the central server updates the user account information

(5) if name or phone number are modified, file names and associated accounts’ list of users are modified recursively

(6) the central server sends a confirmation message back to the Teller Module to notify the teller that the user account information was successfully updated

(7) the event is logged in account level history by central module

(8) the teller module returns to user account options pane

Alternate Flows:

[none]

Exceptions:

(1) the central server does not properly update the user account information

(2) the central server does not send the confirmation message back to the Teller Module after the user account information is updated

Related Use Case(s): 2.2.3

2.2.15. View Transaction History

Relevant Requirement(s) from SRS: 3.1.3.6, 3.1.3.7

Precondition(s): the Teller Module is online and connected to the central server, and the teller has verified the customer’s identity and is accessing the financial account

Postcondition(s): the customer can view all transactions that have occurred on one of their financial accounts

Basic Flow:

(1) the teller selects to view the transaction history of that financial account

(2) a request is sent to the central server to get the transaction history associated with the financial account that the teller selected

(3) the central module sends the transaction history of the account to the Teller Module

(4) the transaction history associated with the financial account selected is shown to the teller

Alternate Flows:

[none]

Exceptions:

(1) the central server does not receive the request to access financial account’s transaction history

(2) the transaction history is not properly sent back to the Teller Module

Related Use Case(s): 2.2.5

2.2.16. Viewing Users on a Customer’s Financial Account

Relevant Requirement(s) from SRS: [none]

Precondition(s): the Teller Module is online and connected to the central server, and the teller has verified the customer’s identity and is accessing financial account

Postcondition(s):

Basic Flow:

(1) the teller selects to view the list of users authorized to access that financial account

(2) the Teller Module sends a request to the central server to find all users that can access the selected financial account

(3) the central server sends a list of authorized users back to the Teller Module

(4) the list of users authorized to view the financial account is shown to the teller

Alternate Flows:

[none]

Exceptions:

(1) the central server does not properly receive the request to find all users that can access the financial account

(2) the central server cannot find all users associated with the selected financial account

(3) the list of authorized users is not properly sent back to the Teller Module

Related Use Case(s): 2.2.5, 2.2.11, 2.2.12

2.2.17. Teller Logging-Out of Teller Module

Relevant Requirement(s) from SRS: [none]

Precondition(s): the Teller Module is online and connected to the central server, and the teller is logged-in to the Teller Module

Postcondition(s): the teller is logged-out of the Teller Module

Basic Flow:

(1) the teller initiates a log-out request

(2) the Teller Module prompts the teller to confirm that they want to log out of the module

(3) if the teller confirms the log-out request, then the Teller Module sends the request to the central server to log the teller out

(4) if the teller is logged in to any customer’s user account, then the user account will automatically be logged out of before the teller is logged out of the Teller Module

(5) the central server sets the teller’s status to logged out

(6) the central server sends a confirmation back to the Teller Module to notify the teller that they are no longer logged in

(7) the Teller Module returns to the log-in page, ready for the next teller to log in

Alternate Flows:

(1) if the teller does not confirm that they want to log out, then the log-out request is canceled and nothing is sent to the central server

Exceptions:

(1) the central server does not receive the log-out request

(2) the confirmation message is not properly sent back to the Teller Module after the teller’s status is set to logged out

Related Use Case(s): 2.2.1

* 1. **Central Server Module Use Cases**

2.3.1. Record Transaction History: Deposits

Relevant Requirement(s) from SRS: 3.1.1.7

Precondition(s): a deposit is made by a customer, either directly through an ATM or indirectly through a teller

Postcondition(s): the deposit is recorded in the user account’s transaction history

Basic Flow:

(1) a deposit is made into a customer’s financial account

(2) the deposit request is sent either by the ATM Module or the Teller Module to the central server

(3) the central server records the date and time that the deposit was made, the amount deposited, whether the deposit was cash or check, and whether the deposit was performed through an ATM or by a teller, into a comma-separated text file associated with the financial account

Alternate Flows:

[none]

Exceptions:

(1) the data is not properly recorded in the comma-separated text file

Related Use Case(s): 2.1.4, 2.2.9

2.3.2. Record Transaction History: Withdrawals

Relevant Requirement(s) from SRS: 3.1.1.7

Precondition(s): a withdrawal is made by a customer, either directly through an ATM or indirectly through a teller

Postcondition(s): the deposit is recorded in the user account’s transaction history

Basic Flow:

(1) a withdrawal is made from a customer’s financial account

(2) the withdrawal request is sent either by the ATM Module or the Teller Module to the central server

(3) the central server records the date and time that the withdrawal was made, the amount withdrawn, and whether the withdrawal was performed through an ATM or by a teller, into a comma-separated text file associated with the financial account

Alternate Flows:

[none]

Exceptions:

(1) the data is not properly recorded in the comma-separated text file

Related Use Case(s): 2.1.5, 2.2.10

2.3.3. Record History: User Account Creation

Relevant Requirement(s) from SRS: 3.1.1.7

Precondition(s): a new user account is created by a teller

Postcondition(s): the user account’s creation is recorded in that account’s history

Basic Flow:

(1) a new user account is made by a teller for a customer

(2) the user account’s credentials are sent to the central server from the Teller Module

(3) the central server records the credentials, including the customer’s full name, phone number, and password, along with the date and time that the user account was created, into a log file

Alternate Flows:

[none]

Exceptions:

(1) the data is not properly recorded in the text file

Related Use Case(s): 2.2.2

2.3.4. Record History: Account Blocking

Relevant Requirement(s) from SRS: 3.1.1.7

Precondition(s): a user account is blocked by a teller, with the customer’s approval

Postcondition(s): the block is recorded in the user account’s history

Basic Flow:

(1) a customer’s user account is blocked by a teller

(2) the block request is sent to the central server from the Teller Module

(3) the central server sets a flag on the user account to mark it as blocked, meaning that no other users can access that user account from an ATM Module or a Teller Module until the user account is unblocked

(4) the central server records the date and time that the block was performed, along with the ID of the teller that performed the block, into a log file

Alternate Flows:

[none]

Exceptions:

(1) the data is not properly recorded in the text file

(2) the user account is not properly flagged as being blocked

Related Use Case(s): 2.2.13

2.3.5. Record History: Account Unblocking

Relevant Requirement(s) from SRS: 3.1.1.7

Precondition(s): a user account is unblocked by a teller, with the customer’s approval

Postcondition(s): the unblock is recorded in the user account’s history

Basic Flow:

(1) a customer’s user account is unblocked by a teller

(2) the unblock request is sent to the central server from the Teller Module

(3) the central server removes the flag on the user account that marks it as being blocked

(4) the central server records the date and time that the unblock was performed, along with the ID of the teller that performed the unblock, into a log file

Alternate Flows:

[none]

Exceptions:

(1) the data is not properly recorded in the text file

(2) the user account is not properly set to being unblocked

Related Use Case(s): 2.2.14

2.3.6. Savings Account: Monthly Interest Added

Relevant Requirement(s) from SRS: 3.1.1.1

Precondition(s): one month has elapsed since monthly interest was last added

Postcondition(s): the financial account’s balance gets increased by adding interest

Basic Flow:

1. a monthly timer, which runs on its own thread in the central server, reaches a trigger value, causing the appropriate Central Server Module method to be called
2. the central server iterates over the financial account files and checks the account type
3. for each file where the account type is “savings”, the central server updates the financial account’s balance by adding interest
4. after processing all files, the monthly timer is reset

Alternate Flows:

[none]

Exceptions:

1. an error occurs in the middle of updating the financial account files, and the central server cannot process all files
2. the timer does not properly reset once all files are processed
3. the timer does not properly trigger at the end of each month

Related Use Case(s): [none]

2.3.7. Daily Limits Are Reset

Relevant Requirement(s) from SRS: 3.1.1.8, 3.1.2.4, 3.1.2.5

Precondition(s): one day has elapsed since last reset

Postcondition(s): the ATM deposit and withdrawal limits for each financial account are reset

Basic Flow:

1. a daily timer, which runs on its own thread in the central server, reaches a trigger value, causing the appropriate Central Server Module method to be called
2. the central server traverses over the daily limits file and resets the limits for each account
3. the daily timer is reset

Alternate Flows:

[none]

Exceptions:

1. an error occurs in the middle of updating the daily limits file
2. the timer does not properly reset once all files are processed
3. the timer does not properly trigger at the end of each day

Related Use Case(s): 2.3.8

2.3.8. ATM Cash Reserves Are Reset

Relevant Requirement(s) from SRS: 3.1.1.8, 3.1.2.3

Precondition(s): one day has elapsed since the last reset

Postcondition(s): the ATM cash reserves get filled again

Basic Flow:

1. a daily timer in each ATM thread, reaches a trigger value, causing the appropriate helper method to be called
2. in each ATM client, receipt of this message triggers the appropriate function which refills reserves
3. the daily timer is reset

Alternate Flows:

[none]

Exceptions:

1. an error occurs in the middle of resetting the cash reserves
2. the timer does not properly reset
3. the timer does not properly trigger at the end of each day

Related Use Case(s): 2.3.7