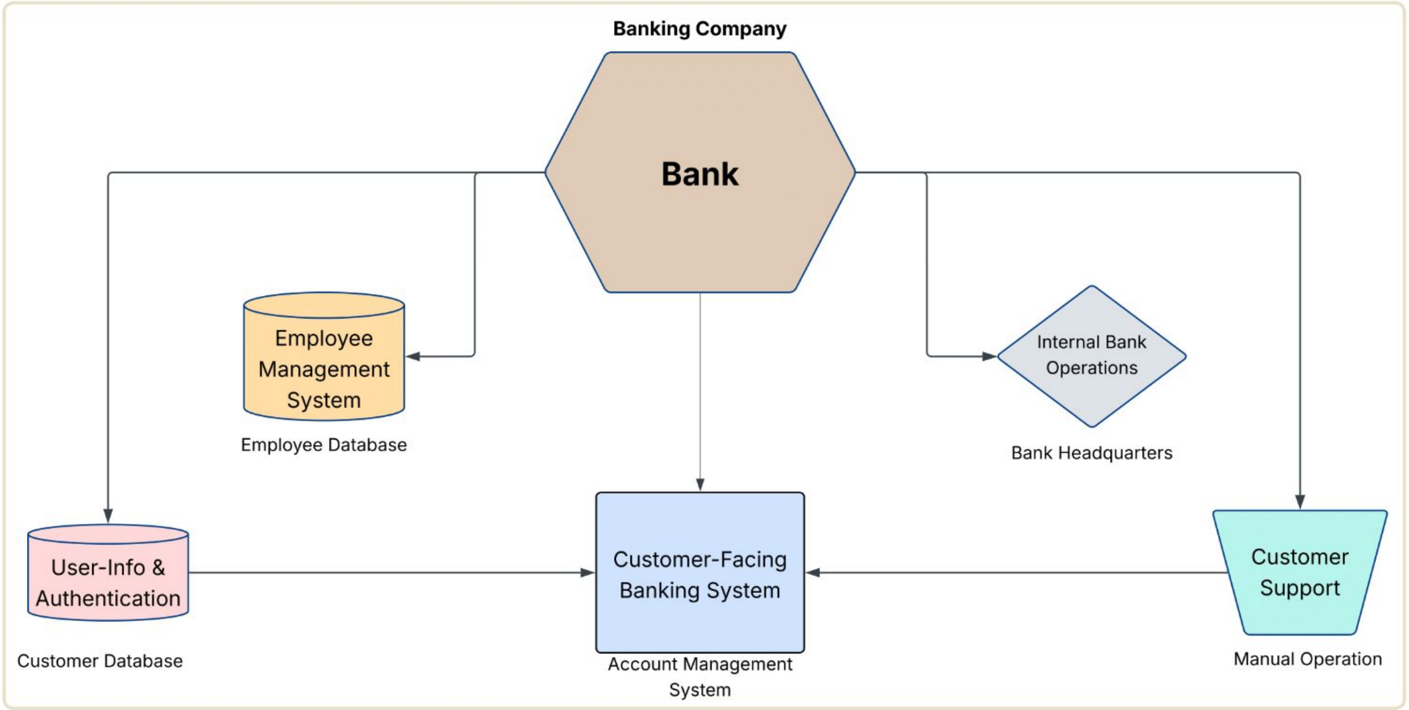
Customer-facing Banking System

Team Tiger

By: Bilal, Anthony, Sokina, Fre, Momen

High Level Architecture

Customer-Facing Banking System High-Level Architecture



Customer-Facing Banking System High-Level Architecture

By ***Momen Suliman***

Diagram information

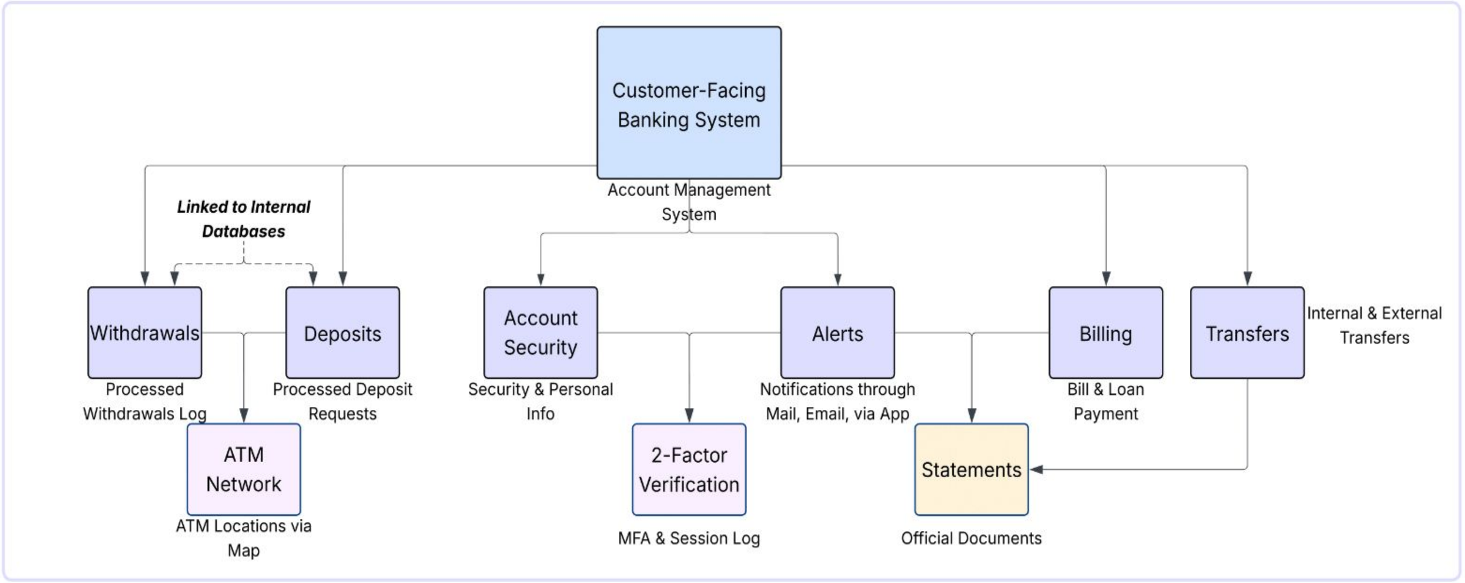
This diagram illustrates the major structure and components of a customer-facing banking system, highlighting the interactions between various internal systems and databases. Key components include the Employee Database, Bank Headquarters, Banking Company, and several systems like Employee Management, Internal Operations, and Customer Support. The chart shows how these elements connect to facilitate customer interactions and manage operations within the bank.

High-Level Architecture Modular Flow

1. The Banking company is the major module as its the owner and funding body.
2. The employee management system is what enables the bank to function.
3. The customer authentication system ensures customer data and bank accounts are secure and encrypted.
4. The customer facing banking-system is where the user handles and manages their account.
    4.1. Its the core experience that the bank offers to its customers.
    4.2. It relies on customer support to handle and fix major issues.
    4.3. Its directly linked to the customer database as that's where the customers' data is stored.
5. The headquarters are the official Bank's base of operation and handles the business planning.
6. Customer Support is the primary operation that handles non-automated customer issues with 24/7 availability and moment's notice.

Module Design

Customer-Facing Banking System Module Design



Customer-Facing Banking System
Account Management System
Linked to Internal Databases
Withdrawals
Processed Withdrawals Log
Deposits
Processed Deposit Requests
Account Security
Security & Personal Info
Alerts
Notifications through Mail, Email, via App
Billing
Bill & Loan Payment
Transfers
Internal & External Transfers
ATM Network
ATM Locations via Map
2-Factor Verification
MFA & Session Log
Statements
Official Documents

Customer-Facing Banking System Module Design

By ***Momen Suliman***

Diagram information

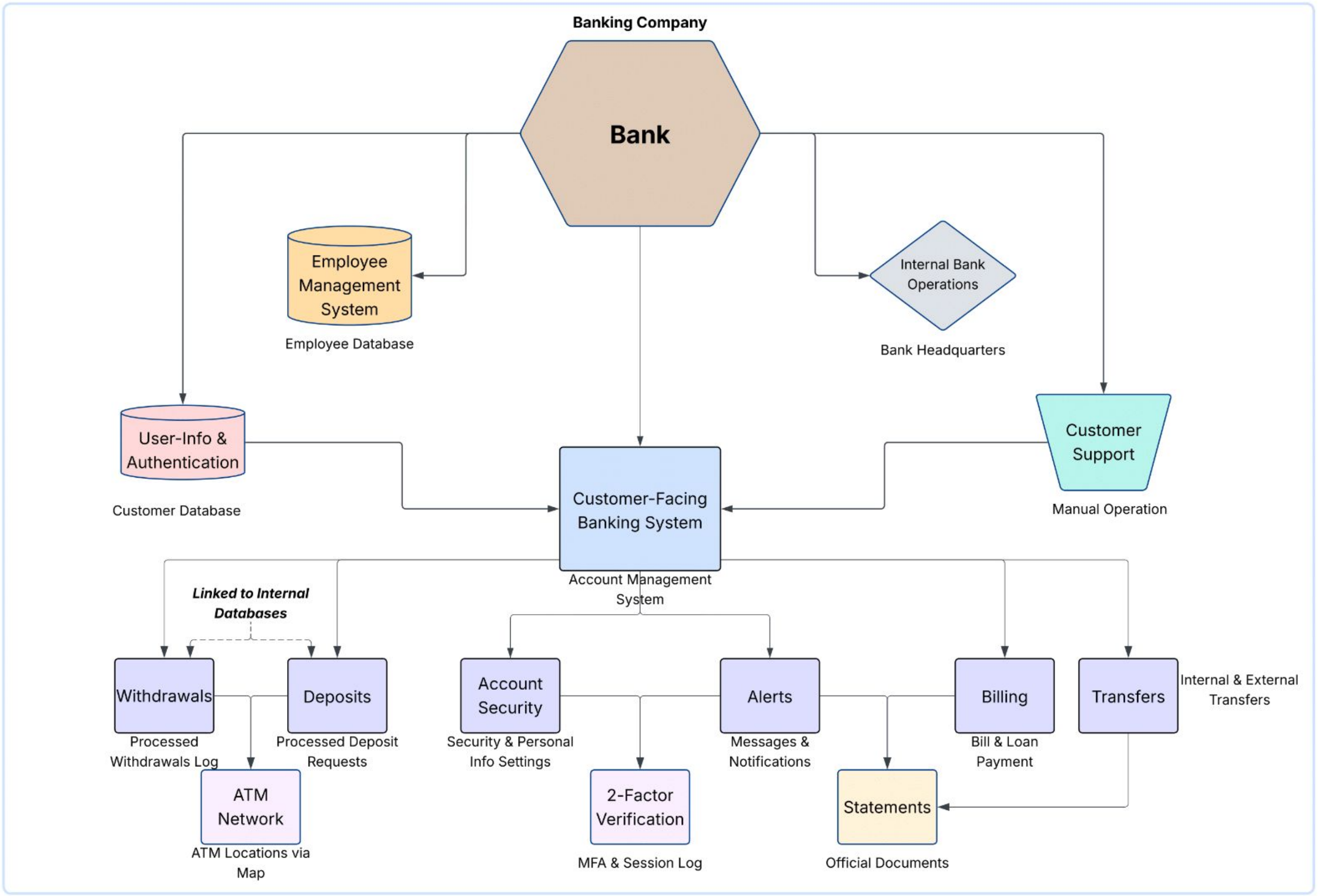
This diagram shows the interconnected modules of a customer facing banking system, focusing on account security, transaction processes, and account management. Key components include account security, withdrawals, deposits, account management, and notifications. The diagram shows how these modules interact with internal databases, ATM networks, and other services like billing and transfers.

Module design Diagram Flow

1. Ensure Account Security is integrated with 2-Factor Verification.
2. Manage Deposits & Withdrawals through an ATM Network.
3. Deposits & Withdrawals are logged to Internal Databases.
   3.1. Verify database linkage.
   3.2. Process Deposits & Withdrawals requests.
4. Conduct Account Management:
   4.1. Oversee Deposits & Withdrawals.
   4.2. Handle Deposits requests.
   4.3. Set up Alerts.
   4.4. Maintain Account Security via MFA.
   4.5. Manage Billing.
   4.6. Facilitate Transfers.
5. Finally the customer would be able to request a statement in-person or through the mobile app.

Overall System Architecture

Customer-Facing Banking System Overall Architecture



Customer-Facing Banking System Overall Architecture

By ***Momen Suliman***

Diagram information

This diagram represents the comprehensive flow of a Customer-Facing Banking System, illustrating how various modules and processes are interconnected. It includes key modules such as account management, billing, transfers, security settings, and customer support. The flowchart demonstrates the relationship between different banking functions, starting from registration and authentication, to handling deposits and withdrawals, and managing account security and notifications, along with official documents.

Customer-Facing Banking System Modular Flow

1. Begin with the Registration Database where user info is stored.
2. Authenticate the user through the Authentication process
3. Once logged into the Account; navigate the 'View Page'.
4. Manage the account by accessing the Account Management tab where the user can...
   4.1. Handle Billing through the Billing function.
   4.2. Execute Transfers using the Transfers function.
   4.3. Monitor Alerts for account notifications.
      4.3.1. Direct Alerts to Statements.
      4.3.2. Implement 2-Factor Verification for security.
5. Access Account Security Settings to update Security & Personal Information.
6. Make Withdrawals through the ATM Network.
7. Make Deposits via the Deposits function.
8. Utilize Customer Support for additional assistance.
9. View all logged processed transactions and pending transactions via the Messages & Notifications.
10. Review Official Documents and Statements for compliance and record-keeping.
11. Locate ATM Locations via the Map for physical transactions.

MoSCoW

Must Have

1. The systems **M**ust allow users to log in with a username and a password and to login securely.

2. The ATM **M**ust process transactions such as deposits and withdrawals in real time.

3. The system **M**ust accept debit and credit cards at the atm and the user must authenticate the account by entering their PIN.

Should Have

1.The system **S**hould provide users with the ability to recover their passwords through a secure email involving verification such as login process must complete within 1 minute .

2.The system **S**hould allow users to set up and manage recurring payments for regular expenses.

3. The atm **S**hould have assistance for handicapped people/visually impaired

Must Have

1. We **Must** have a system in place to properly reject checks that cannot be read / accepted.

2. The ATM and web services **Must** be connected and updated in real time, changes in one should reflect to the other within 5 seconds.

Should Have

1.The ATM deposit system **Should** be able to accept most checks, a goal of 90% accepted

2. The ATM **S**hould allow users to withdraw money safely using a debit or credit card with a secure PIN. Users should be able to select how much they want to withdraw.

Could Have

1. The ATM and Web applications **Could** have a support chatbot in case service lines are busy.

2. The Atm **C**ould have an app that connects to your banking system so you wouldn’t need to enter your card at the atm it would be all an online transaction

Won’t Have

1. ATM **W**on't do currency exchanges since it is too complex to do on a basic ATM.

2. The customer **W**on’t be able to deposit or withdraw non-USD currencies,

User Stories

ATM Error Messages

As an ATM user, I want clear, helpful error messages when something goes wrong so that I can resolve issues without confusion.

Acceptance Criteria:

1. **Handle Common Errors:**
   a. When there's a problem like not enough money or a “swallowed” card, the ATM should explain it clearly.
   b. For example: “Please try a smaller amount” or “Your card was taken for safety. Please contact the bank.”
2. **Guide the User:**
   a. The ATM should help the user know what to do next.
   b. It should show simple buttons like “Try Again” or “Cancel.”
   c. If something is fixed, the ATM should guide the user step by step.
3. **Accessibility:**
   a. Text on the screen should be big and easy to read.
   b. There should also be sound options for people with vision problems.
   c. The ATM should support screen readers and keyboard use.

Story Points: 5
Status: To Do
Assignee: Firehiwot

Secure Account Login

As a user, I want to log in to my account securely so that I can access my financial information.

Acceptance Criteria:

1. Login Form:
   a. The login page includes fields for username/email and password.
   b. Forgot Password? link is available.
2. Security:
   a. Passwords are hashed and never stored in plain text.
   b. MFA is required after password entry.
   c. After 5 failed attempts, account is temporarily locked with a message shown to the user.
3. Successful Login:
   a. On correct credentials and successful MFA, the user is taken to the dashboard.
   b. Session is securely stored and timed out after 15 minutes of inactivity.

Story Points: 5
Status: To Do
Assignee: Bilal

Account Balance Dashboard

As a user, I want to view all my account balances in one place so that I can manage my finances better.

Acceptance Criteria:

1. Dashboard Visibility:
   a. Upon successful login, the user is automatically directed to the dashboard displaying all linked accounts.
   b. Each account is represented with a clear label Checking, Savings, Credit Card.
2. Real-Time Data & Refresh:
   a. The dashboard must display real-time or near real-time balances.
   b. A Refresh button is available for the user to manually update the account balances.
3. Detailed Account View:
   a. Clicking on an individual account directs the user to a detailed view with transaction history and account-specific insights.
4. Security and Privacy:
   a. All account information is displayed only after secure authentication.

Story Points: 8
Status: To Do
Assignee:Bilal

Cash Withdrawal via ATm

As a bank customer, I want to withdraw money from an ATM so that I can access cash when needed.

Acceptance Criteria:

1. Login:
   a. Tap or insert bank card
   b. Enter PIN
      i. Forgot it ? can reset new one at the bank or email via
2. Withdrawal:
   a. Tap “Get Cash” on the screen
   b. Choose account type
      i. Checking
      ii. Saving
   c. Can choose amount of money
3. Receipt:
   a. Can choose receipt:
      i. Print
      ii. Text
      iii. Email
      iv. None
4. Finish Up:
   a. ATM gives you the money
   b. Show:
      i. How much you took
      ii. Which account
      iii. New balance
      iv. Transaction saved in your account history
5. Errors:
   a. Such as not enough money or wrong PIN
      i. It will shows a message

Story Point: 2
Status: TO DO
Assignee: Sokina

Classes

Class Design: Banking System Components

Setting up 4 classes for the Customer-facing banking system:

- Bank:
    - Manages all accounts
    - Updates account balance

      Functions:
        - addAccount() - add a new account
        - updateBalance()- change balance
- ATM:
    - Connect to the bank
    - Shows options like withdraws/Deposit

      Functions:
        - withdrawCash() - take out money
        - showMenu() - display user options
- UserAccount
    - Stores person's info (Name, ID)
    - Connects to their bank account

      Functions:
        - getUserInfo() - show name and ID
        - linkToAccount() - connect to their money
- BankAccount :
    - Track balance and transactions
    - Belongs to a specific user

      Functions:
        - getBalance() - show current balance
        - deposit() - add money

These 4 classes define the system. Each class gives instructions for what an object should be and do.
These classes work together to create a system that customers (users) will use to manage their bank accounts.

Objects

Object Instantiation

Setting up different objects in the customer-facing banking system:

- user1 = new UserAccount(“user’s name”, “ID”)
  - Stores name and ID
  - Identifies the customer user in the system
- account1 = new BankAccount(user1, 500.00)
  - User’s bank account
  - Starts with $500
  - Tracks deposit/withdrawals
- bank = new Bank()
  - Stores all accounts
  - Updates Balances
  - Help process user actions
- atm = new ATM (bank)
  - Let the user use the system
  - Sends user requests to the bank
  - Allows withdrawals and balance checks

UserAccount → BankAccount → Bank → ATM

The four objects instantiates are based on these classes, setting up a complete system where a user can interact with their bank account through an ATM connected to the Bank.

Test Plan

Test Plan and Timeline

Test Plan