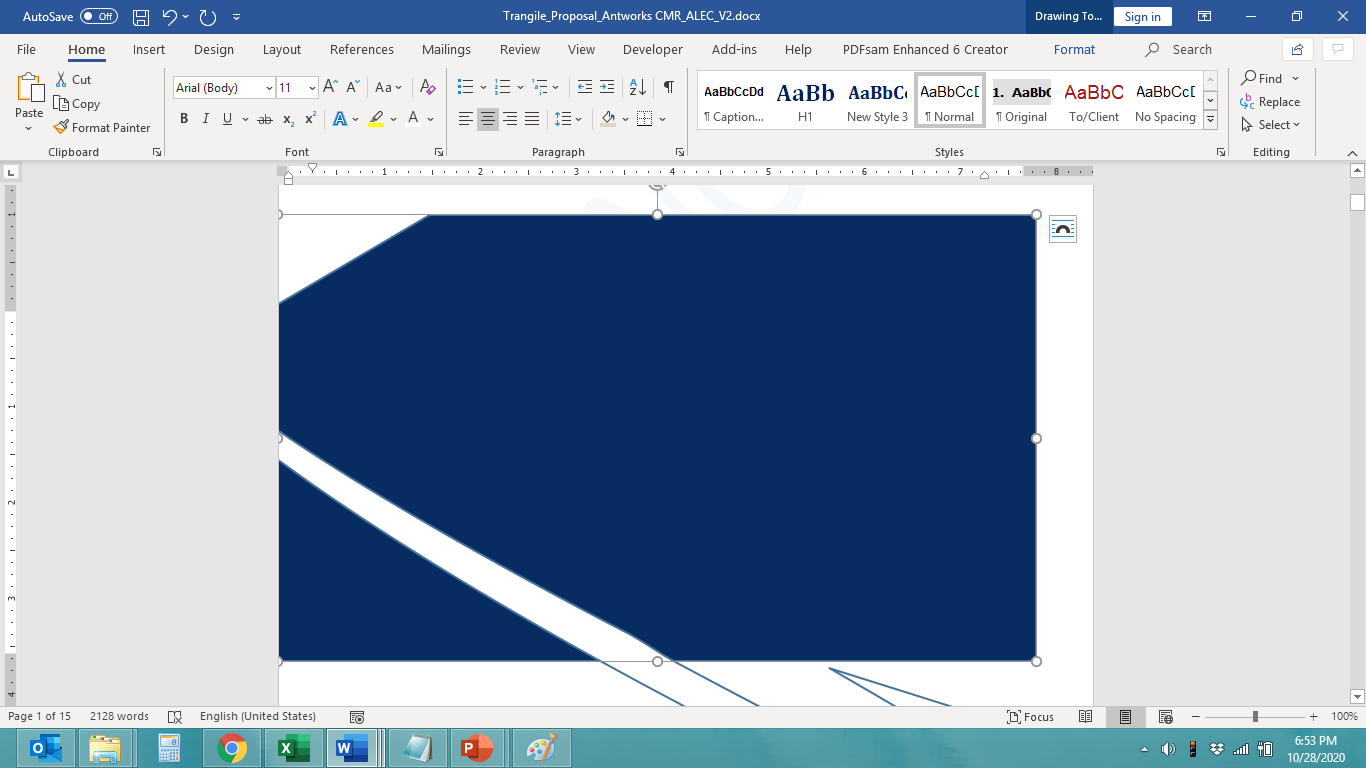
|  |  |  |
| --- | --- | --- |
|  |  |  |



**TO BE Functional Specifications**

**for**

**Bank Management System**

Submitted by-

Sindhuja Billakanti

Vamshi Artham

Jaji Madhuri Vallurupalli

Ganji Akhil Chandra

Hindhu Preethi Gunduka

The data, information or material provided herein is confidential and proprietary to Technical Ltd. and its partner organization's and shall not be disclosed, duplicated or used in whole or in part for any purpose other than as approved by an authorized official of Technical Ltd. in writing. The copyrights logo’s & brand name mentioned are of respective organizations. The recipient agrees to maintain complete confidentiality of the information; data received and shall take all reasonable precautions / steps in maintaining confidentiality of the same, however in any event not less than the precautions / steps taken for its own confidential material. If you are not the intended recipient of this information, you are not authorized to read, forward, print, retain, copy or disseminate this document or any part of it

**Document Control**

**Table of CONTENTS**

[1. Document Acceptance and Revision History 3](#_Toc200358324)

[Acceptance Signoff 3](#_Toc200358325)

[Document Revision History 3](#_Toc200358326)

[2. Requirement Summary 4](#_Toc200358327)

[3. Design Overview 5](#_Toc200358328)

[4. Assumptions 7](#_Toc200358329)

[5. High Level Process 8](#_Toc200358330)

[6. Screen Specifications and Functional Design 11](#_Toc200358331)

[6.1 Launcher 11](#_Toc200358332)

[6.2 Login Screen 12](#_Toc200358333)

[6.3 Customer Transaction Menu Screen 13](#_Toc200358334)

[6.4 Deposit Screen 14](#_Toc200358335)

[6.5 Cash Withdrawal Screen 15](#_Toc200358336)

[6.6 Balance Inquiry Screen 16](#_Toc200358337)

[6.7 Mini Statement Screen 17](#_Toc200358338)

[6.8 Fast Cash 19](#_Toc200358339)

[6.9 PIN Change Screen 20](#_Toc200358340)

[6.10 Employee Login Screen 21](#_Toc200358341)

[7. Database Design 26](#_Toc200358342)

[8. UML and ER diagrams 29](#_Toc200358343)

[9. Integration Requirements 31](#_Toc200358344)

[10. Reporting Requirements 33](#_Toc200358345)

[11. Process Dependencies/Frequency 34](#_Toc200358346)

[12. Security Requirements 35](#_Toc200358347)

[13. Availability and Processing Integrity Requirements 36](#_Toc200358348)

[12.1. Availability Requirements 36](#_Toc200358349)

[12.2. Processing Integrity Requirements 36](#_Toc200358350)

[14. Abbreviations 37](#_Toc200358351)

[15. Related Documents 38](#_Toc200358352)

[16. References 39](#_Toc200358353)

[17. Issues/Questions 40](#_Toc200358354)

# Document Acceptance and Revision History

## Acceptance Signoff

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Project Role** | **Signature** | **Date** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

## Document Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Revision Notes** |
| 1.0 | 15-May-2025 |  | Initial Draft |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Requirement Summary

The goal of this project is to simulate Bank Management System using a graphical user interface (GUI) in Java. The system allows users to perform common banking operations such as depositing money, withdrawing money, checking balance, downloading and email of mini-statements, and changing PIN by verifying the existing pin.

In addition to facilitating secure transactions, the system incorporates a role-based access model where employees can approve new customer registrations. It provides a realistic banking experience by handling user authentication, transaction recording, and interaction flow through intuitive screens.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Application** | **Module** | **Business Process** | **Req. No.** | **Requirement** | **Solution Element (Fit/Gap**  **/SOP/WIP/CIS /CFG** | **Gap Desc** | **Solution Description** |
| |  | | --- | | ATM\_SIM |  |  | | --- | |  | | Customer | Customer Login | |  | | --- | |  |  |  | | --- | | R001 | | Existing customer signin and new customer signup and wait for bank employee Approval.   |  | | --- | |  |  |  | | --- | |  | | FIT |  |  |
| ATM\_SIM | |  | | --- | |  |  |  | | --- | |  |   Deposit | Deposit | R002 | Deposit money in the bank | FIT |  |  |
| ATM\_SIM | Cash  Withdrawal | Cash  Withdrawal | R003 | |  | | --- | |  |  |  | | --- | | Withdraw amount from the bank | | FIT |  |  |
| ATM\_SIM | Balance  Enquiry | Balance Check | R004 | |  | | --- | |  |  |  | | --- | | Display current balance | | FIT |  |  |
| ATM\_SIM | Mini  Statement | |  | | --- | |  |  |  | | --- | | Mini  Statement | | R005 | Show all last transactions, and able to download and email it. | FIT |  |  |
| ATM\_SIM | Pin Change | PIN Change | R006 | Allow secure PIN change by verifying the existing pin in the database. | FIT |  |  |

# Design Overview

The system simulates a real-world Bank Management System using a Java Swing-based GUI. Users must authenticate using a simulated card and PIN. The interface presents transactional options such as deposit, withdrawal, balance inquiry, PIN change, and mini statement. It processes user selections, updates the system state, and provides real-time feedback for each operation.





# Assumptions

|  |  |  |
| --- | --- | --- |
| **S. No.** | **Assumption** | |
| 1. | |  | | --- | |  |  |  | | --- | | All transactions are handled locally (no network connection). | | |
| 2. | |  | | --- | |  |  |  | | --- | | Each card is mapped to a user with an initial account balance. | | |
| 3. | |  | | --- | |  |  |  | | --- | | PIN authentication is mandatory for all operations. | | |
| 4. | Transaction data is stored in memory or flat files. | |
| 5. | No physical hardware interaction (e.g., cash dispenser). | |
| 6. | Application should behave based on privilege levels, where employees and customers have separate access rights | |
|  | |

# High Level Process

To provide a structured, secure, and intuitive banking experience through a Java Swing-based GUI that simulates real-world banking operations.

**Main Menu Options:**

* Login as Customer
* Login as Employee

**Customer Flow:**

1. **Login:**
   * Existing customers log in using:
     + Card Number
     + PIN
2. **Sign-Up (for new users):**
   * Three-step registration process:
     + Step 1: Personal Details
     + Step 2: Demographic Details
     + Step 3: Account Details
3. **Post Sign-Up:**
   * Directed to a **"Waiting for Approval"** screen.
   * The screen **auto-refreshes periodically** to check approval status.

**Employee Flow:**

1. **Login:**
   * Accessed through the Menu Screen.
   * Employees log in using:
     + Employee Card Number
     + PIN
2. **Customer Approval Dashboard:**
   * Displays **pending customer sign-up requests**.
   * Validates:
     + Aadhar Number
     + PAN
     + Email
     + Date of Birth (Age)
3. **Approval Actions:**
   * **If valid:**
     + Approves request.
     + System generates and displays Card Number & PIN on the customer’s screen.
   * **If invalid:**
     + Rejects request.
     + Provides a **custom rejection reason**, shown to the customer.

**Customer Transaction Services:**

Once logged in, the customer accesses a **Transaction Menu** offering:

* **Deposit**



* **Cash Withdrawal**



* **Balance Inquiry**



* **Mini Statement**



* **PIN Change**



* **Exit**

# Screen Specifications and Functional Design

The **Bank Management System** offers a set of GUI-based screens developed using Java Swing to simulate essential banking operations. Each screen is functionally designed to reflect real-world banking workflows, guiding users through secure and interactive customer transactions.

## 6.1 Launcher

To act as the **entry point** for the Bank Management System, clearly separating user roles and providing secure navigation to respective workflows.

**Main Options:**

* Login as Customer
* Login as Employee

**Customer Flow:**

* Existing customers:
  + Enter **Card Number** and **PIN** to access the **Transactions Menu**.
* New customers:
  + Click **Sign Up**.
  + Complete a **3-step registration form**:
    1. Personal Information
    2. Demographic Details
    3. Account Preferences
  + Redirected to a **"Waiting for Approval"** screen.
  + Screen **auto-refreshes** to check approval status in real-time.

**Employee Flow:**

* Employees:
  + Log in using **Employee Card Number** and **PIN**.
  + Access the **Customer Approval Dashboard**.
  + Verify pending customer details:
    - Aadhar Number
    - PAN
    - Email Address
    - Date of Birth (Age Check)

**Possible Outcomes:**

* ✅ **If Approved:**
  + System generates and displays the **Card Number** and **PIN** to the customer.
* ❌ **If Rejected:**
  + A **custom rejection message** is shown to the customer.
  + Message includes guidance or reasons for rejection.

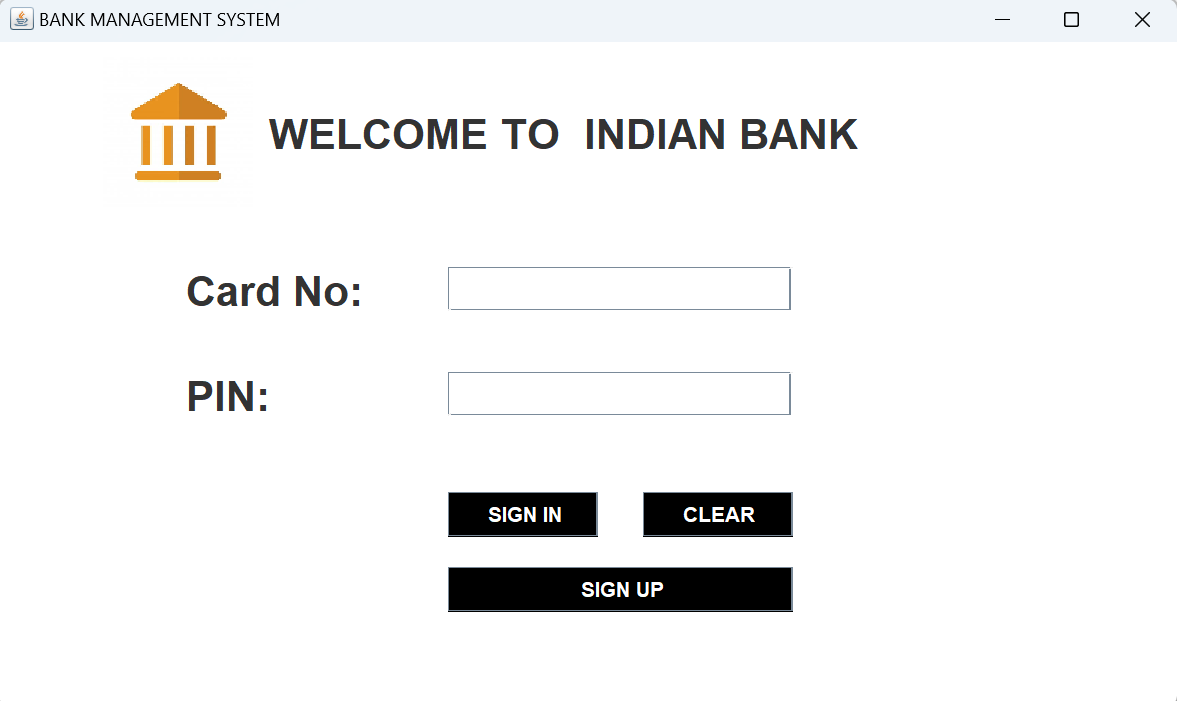
A screenshot of a computer screen

AI-generated content may be incorrect.

## 6.2 Login Screen

When the application starts, users are first shown the **Menu Screen** with two options:  
**Login as Customer** or **Login as Employee**.

* **Login as Customer:**  
  Existing users can log in with their card number and PIN.  
  New users must complete a 3-step signup process. After submission, they are shown a **Waiting for Approval** screen.
* **Login as Employee:**  
  Employees log in using their credentials. Once logged in, they can access the **Approval Dashboard** to approve or reject pending customer requests.
* **If approved**, the system generates a **Card Number and PIN**, shown in the waiting screen.
* **If rejected**, a custom message is displayed, guiding the user to contact the bank.



## 6.3 Customer Transaction Menu Screen

Once authenticated, the system displays a **customer transaction menu** that serves as the central navigation point. This screen includes the following transaction options:

* Withdraw
* Deposit
* Balance Inquiry
* Mini Statement
* PIN Change
* Exit

Each option navigates the user to a dedicated screen for that transaction. The menu remains consistent across all operations, allowing users to return to it after completing a transaction.



## 6.4 Deposit Screen

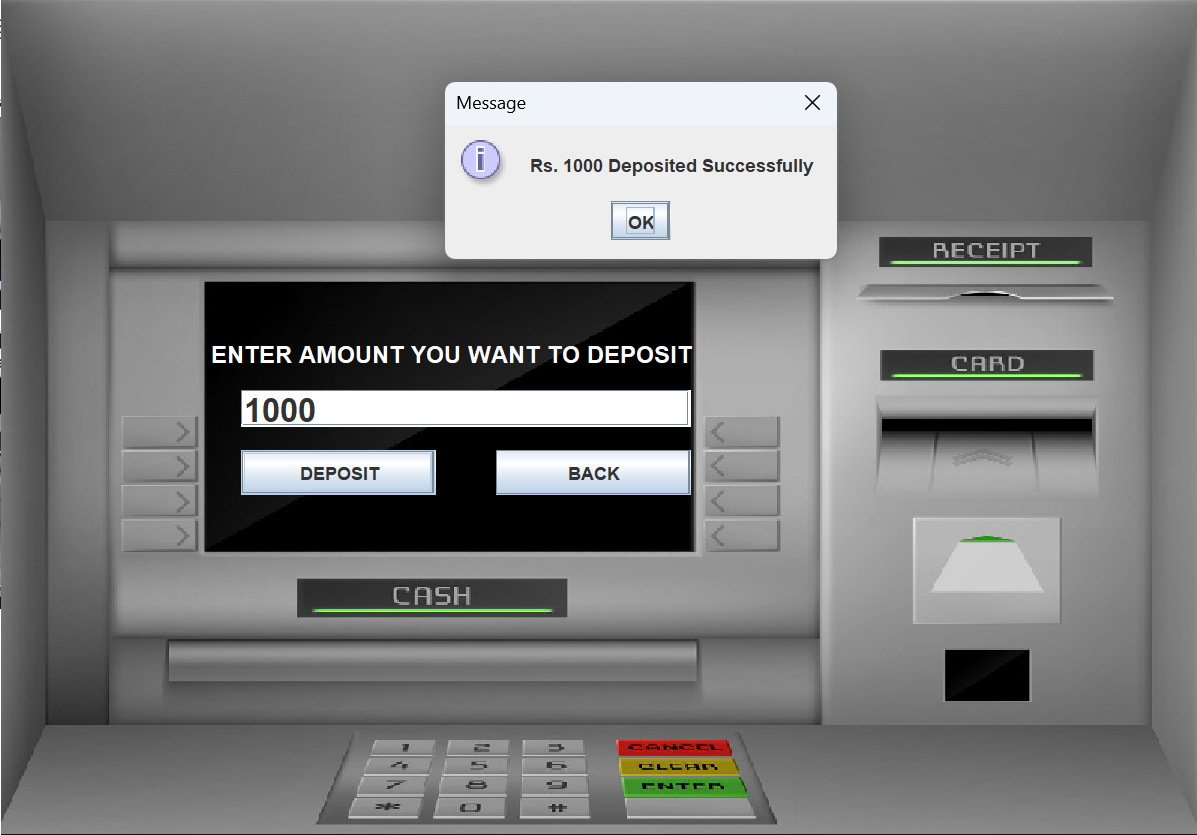
To allow users to deposit funds into their account through a simple, validated process.

**Functionality:**

* The user is prompted to **enter the deposit amount**.
* The system performs a **validation check**:
  + Ensures the amount entered is a **positive number**.

**Outcomes:**

* ✅ **If the amount is valid:**
  + The **account balance is updated** accordingly.
  + A **success message** is displayed to the user.
* ❌ **If the amount is invalid (e.g., negative or zero):**
  + The system **rejects the input** and prompts the user to enter a valid amount.



## 6.5 Cash Withdrawal Screen

To allow users to withdraw money from their account, with necessary validations to ensure secure and valid transactions.

**Functionality:**

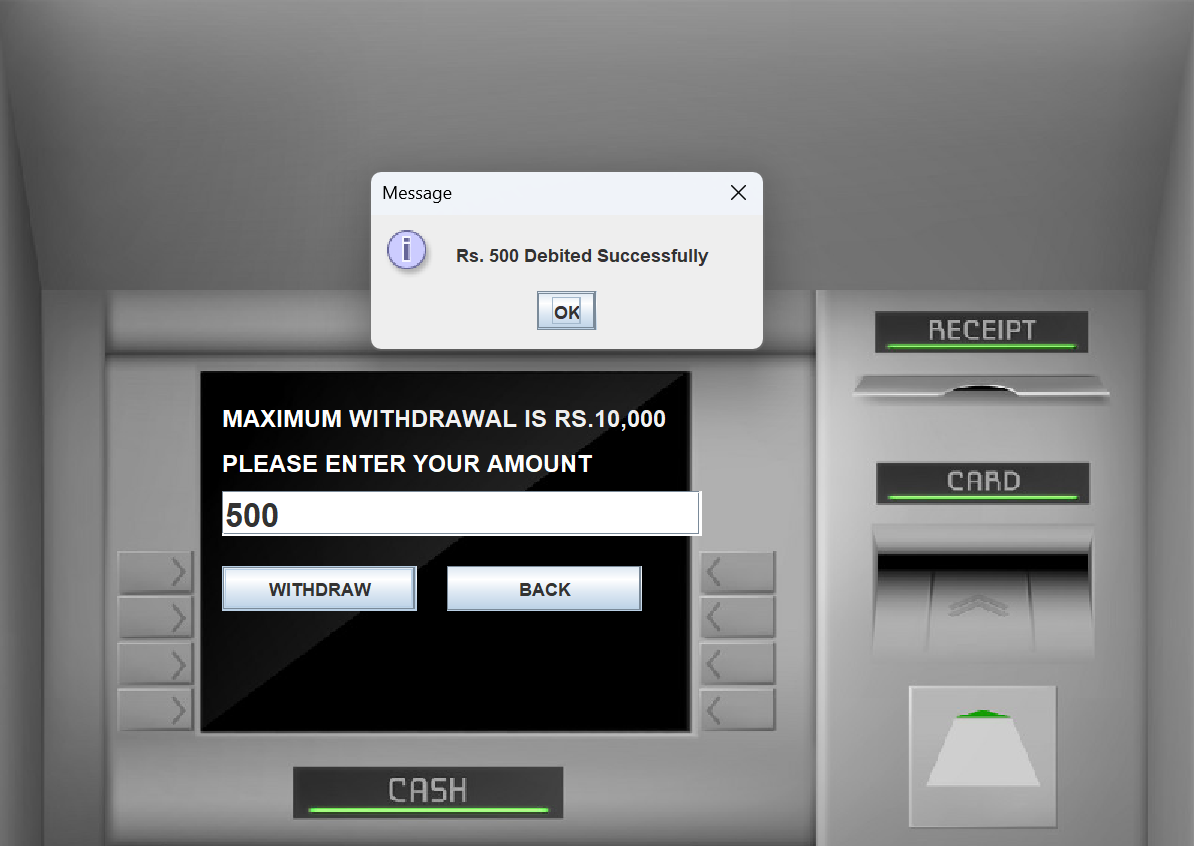
* The user is prompted to enter the withdrawal amount.

**Validation Checks:**

* ✅ The amount must be:
  + A positive number
  + Typically a multiple of 100
* ✅ The user must have sufficient account balance.

**Outcomes:**

* ✅ If all validations pass:
  + The specified amount is deducted from the account.
  + The account balance is updated.
  + A withdrawal confirmation message is displayed.
* ❌ If any validation fails:
  + An error message is shown indicating:
    - Invalid amount
    - Insufficient balance



## 6.6 Balance Inquiry Screen

To allow users to view their current account balance instantly without any input.

**Functionality:**

* No user input is required.
* The system automatically:
  + Retrieves the account balance from the database.
  + Displays the balance either:
    - Directly on the screen, or
    - Via a popup message.

A screen shot of a machine

AI-generated content may be incorrect.

## 6.7 Mini Statement Screen

To provide users with a summary of recent transactions, simulating a printed receipt or activity report.

**Functionality:**

* Displays the last five transactions (or recent transactions).
* Each transaction includes:
  + 📅 Date and Time
  + 🔄 Transaction Type (e.g., Withdrawal, Deposit)
  + 💰 Transaction Amount
  + 💼 Remaining Balance
* Presented in a formatted view resembling:
  + A printed mini statement, or
  + An on-screen activity report

**Available User Actions:**

* 📄 Download:
  + Exports the mini statement as a .txt file for offline use.
* 📧 Email:
  + Sends the mini statement to a user-specified email address.
  + Uses the customer’s pre-configured email credentials.
* ❌ Exit:
  + Closes the mini statement screen.
  + Returns the user to the Customer Transaction Menu.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

## 6.8 Fast Cash

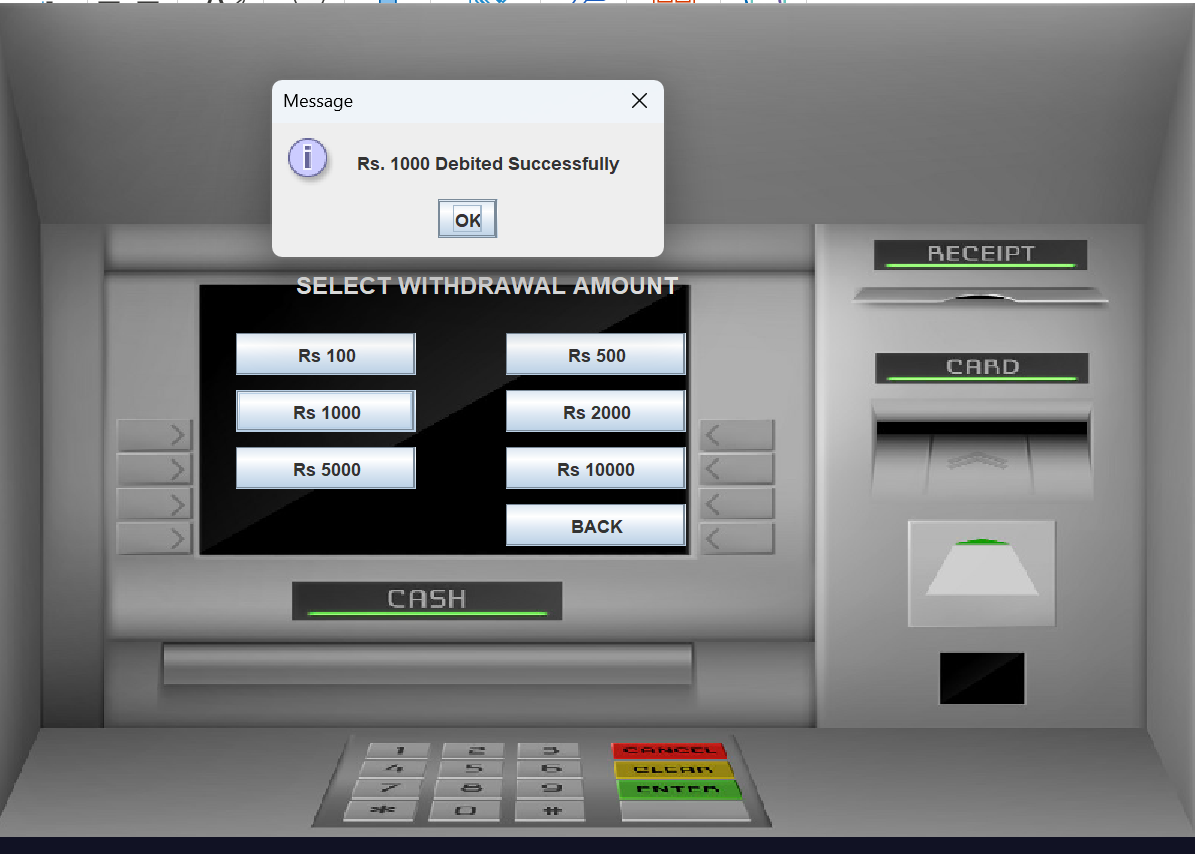
To enable customers to quickly withdraw predefined cash amounts without manually entering the withdrawal value, enhancing speed and convenience.

**Available Denominations:**

* Rs 100
* Rs 500
* Rs 1000
* Rs 2000
* Rs 5000
* Rs 10,000

**Features and Behavior:**

* When a user clicks a denomination button, the system:
  + Validates if the customer has sufficient balance.
  + Deducts the selected amount from the customer’s account.
  + Updates the transaction record in the database.
  + Displays a success message if the withdrawal is successful.
  + Shows an error message if the balance is insufficient or any other issue occurs.
* **BACK button:**
  + Allows the user to **return to the Customer Transaction Menu** without performing a withdrawal.



## 6.9 PIN Change Screen

This screen allows users to securely change their PIN. It prompts users to:

* Enter the current PIN
* Enter a new PIN
* Confirm the new PIN

The system validates the old PIN and ensures both new PIN entries match. If valid, the PIN is updated securely.

A screen shot of a machine

AI-generated content may be incorrect.

## 6.10 Employee Login Screen

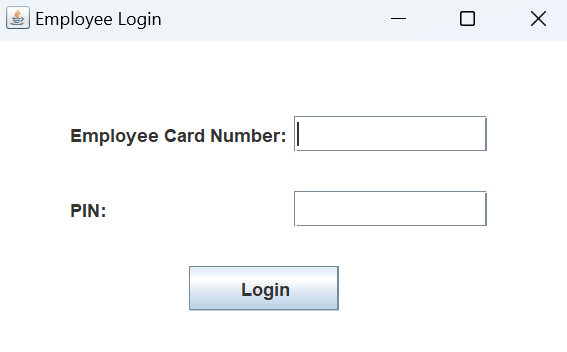
To allow authorized bank employees to securely log in and access employee-specific functionalities.

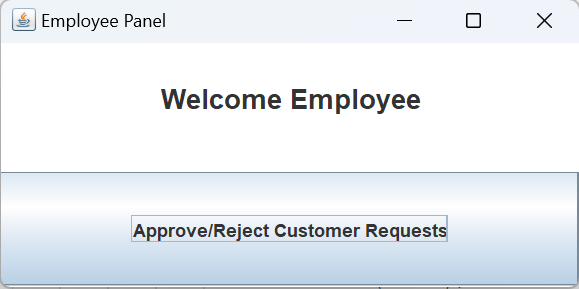
**Functionality:**

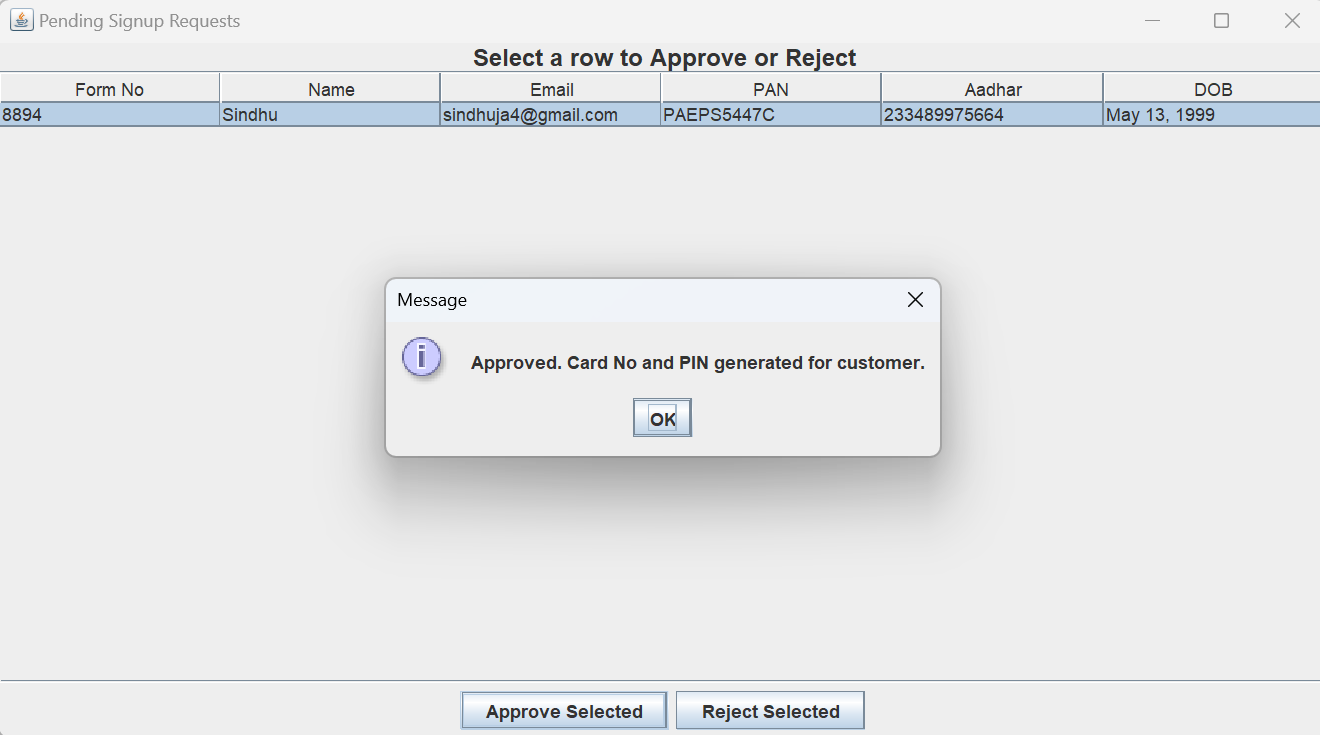
* Employees enter:
  + Employee Card Number
  + PIN
* The system authenticates the credentials.

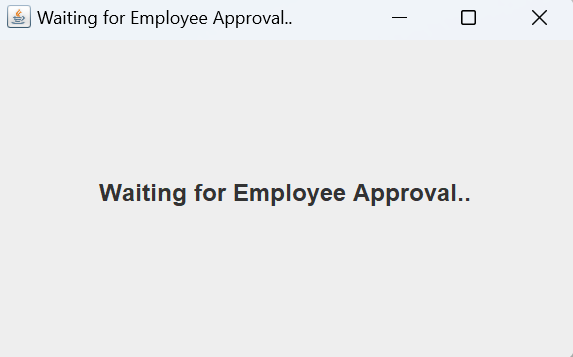
**Post-Login Behavior:**

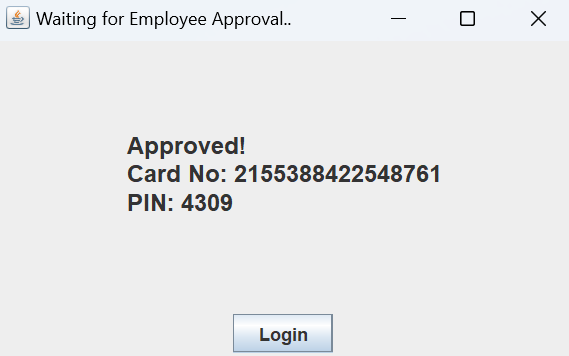
* Upon successful authentication:
  + Employees are redirected to the Employee Options screen.
  + From there, they can access the Customer Approval Dashboard to:
    - Review pending customer registration requests.
    - Approve or reject customer applications.

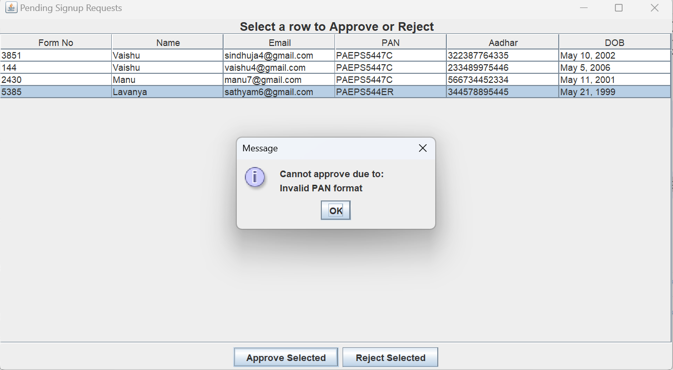


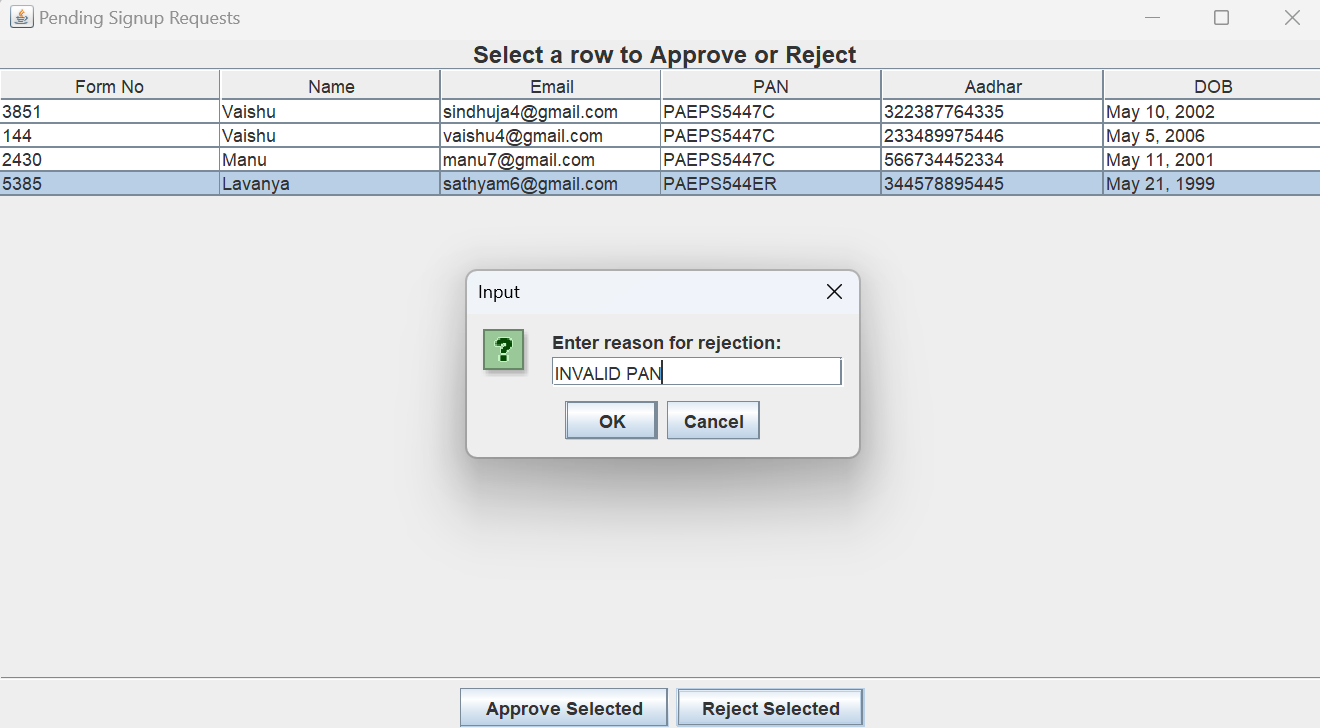












A screenshot of a computer screen

AI-generated content may be incorrect.

# Database Design

The database for the Bank Management System is structured to reflect real-world banking workflows and supports features such as user registration, employee approval, authentication, transactions, and access control. The design is based on normalized relational tables created in MySQL and follows entity-relationship principles.

1. **Login Table**

Stores login credentials for all users (Customers and Employees).

| Field | Data Type | Description |
| --- | --- | --- |
| formno | VARCHAR(20) | Reference to signup form (Foreign Key) |
| card\_number | VARCHAR(20) | Unique card number used for login (Primary Key) |
| pin | VARCHAR(10) | Personal Identification Number (Authentication) |
| usertype | ENUM | Role of the user: 'Customer' or 'Employee' |

1. **Customerdetails1**

Captures the personal and contact information of customers during account creation.

| Field | Data Type | Description |
| --- | --- | --- |
| formno | VARCHAR(20) | Unique form number (Primary Key) |
| name | VARCHAR(50) | Customer's full name |
| fname | VARCHAR(50) | Father's name |
| dob | VARCHAR(20) | Date of Birth |
| gender | VARCHAR(10) | Gender |
| email | VARCHAR(50) | Email Address |
| marital | VARCHAR(30) | Marital Status |
| address | VARCHAR(100) | Residential Address |
| city | VARCHAR(30) | City |
| pincode | VARCHAR(10) | Postal Code |
| state | VARCHAR(30) | State of Residence |

1. **Customerdetails2**

Stores extended demographic and identification data.

| Field | Data Type | Description |
| --- | --- | --- |
| formno | VARCHAR(20) | Foreign Key → customerdetails.formno |
| religion | VARCHAR(20) | Religion |
| category | VARCHAR(20) | Category (e.g., General, SC, ST) |
| income | VARCHAR(30) | Income Bracket |
| education | VARCHAR(30) | Educational Qualification |
| occupation | VARCHAR(30) | Occupation |
| pan | VARCHAR(20) | PAN Number |
| aadhar | VARCHAR(20) | Aadhaar Number |
| scitizen | VARCHAR(5) | Senior Citizen (Yes/No) |
| eaccount | VARCHAR(5) | Existing Account (Yes/No) |

1. **Account**

Holds account-specific details including type and associated services.

| Field | Data Type | Description |
| --- | --- | --- |
| formno | VARCHAR(20) | Foreign Key → customerdetails.formno |
| account\_type | VARCHAR(50) | Type of Account (e.g., Savings, Current) |

1. **Bank**

Logs all financial transactions such as deposits and withdrawals.

| Field | Data Type | Description |
| --- | --- | --- |
| pin | VARCHAR(10) | Foreign Key → login.pin |
| date | DATETIME | Timestamp of transaction |
| type | VARCHAR(20) | Transaction Type (Deposit/Withdrawal) |
| amount | VARCHAR(20) | Transaction Amount |

1. **user\_access**

Controls access privileges and tracks whether access is granted to customers or employees.

| Field | Data Type | Description |
| --- | --- | --- |
| id | INT (AUTO\_INCREMENT) | Unique Identifier |
| aadhar | VARCHAR(20) | Unique Aadhaar Number |
| pan | VARCHAR(20) | PAN Number |
| role | ENUM | Role of User ('Customer' / 'Employee') |
| access\_granted | BOOLEAN | Access Approval Status |

1. **signup\_requests**

Tracks customer signup requests and their approval status by employees.

| Field | Data Type | Description |
| --- | --- | --- |
| formno | VARCHAR(20) | Reference Form Number |
| aadhar | VARCHAR(20) | Aadhaar Number |
| pan | VARCHAR(20) | PAN Number |
| status | ENUM | Request Status ('PENDING', 'APPROVED', 'DENIED') |
| requested\_at | TIMESTAMP | Time of Request Submission |

# UML and ER diagrams

UML Diagram :



ER diagram :



# Integration Requirements

**1. Package Structure**

* ASimulatorSystem
  + Main package containing:
    - All GUI screens
    - Business logic classes
    - Database connectivity classes

**2. Java Class Integration**

* **Conn.java**
  + Central utility class for managing database connections using JDBC.
* **Login.java**
  + Main entry point for both customers and employees.
  + Handles credential verification and routes users to appropriate dashboards.
* **PersonalDetails.java, AdditionalDetails.java, AccountDetails.java**
  + Manage the 3-step customer signup process.
  + Insert user details into multiple database tables.
* **AccountApproval.java**
  + Displays status after signup submission.
  + Continuously polls for employee approval updates.
* **EmployeeLogin.java**
  + Handles employee authentication using Employee Card Number and PIN.
* **EmployeeOptions.java**
  + Shows employee options to:
    - Perform personal transactions
    - Approve or reject new customer requests
* **EmployeeApprovalDashboard.java**
  + Displays pending signup requests.
  + Enables employees to approve or reject customers.
  + Card Number and PIN generated only upon approval.
* **CustomerTransactionsMenu.java**
  + Customer transaction menu providing navigation to:
    - Deposit
    - Withdrawal
    - Mini Statement
    - PIN Change
    - Other transaction operations
* **BankDeposit.java, Withdrawal.java, FastCash.java, BalanceEnquiry.java, MiniStatement.java, Pin.java** 
  + Dedicated screens for each banking operation.
  + Handle UI and update bank tables accordingly.
* **EmailComposer.java**
  + Utility class for sending mini statements to customer emails (if enabled).
* **Practice.java**
  + Used for prototyping or testing individual components.

**3. Icons and Media**

* ASimulatorSystem.icons
  + Stores static assets like:
    - atm.jpg
    - logo.jpg
  + Used for UI backgrounds and branding.

**4. External Libraries**

* mysql-connector-j-9.3.0.jar
  + JDBC connector for MySQL database integration.
* jcalendar-1.4.jar
  + Provides date input fields (used in signup or forms).
* javax.mail-1.6.2.jar & javax.activation-1.2.0.jar
  + Enable email sending functionality, e.g., for mini statement sharing.

**5. Database Integration**

* **Database:** MySQL
* **Purpose:** Store user info, login credentials, transaction logs, approval statuses, etc.
* **Key Tables:**
  + signup, signup2, signup3
  + login
  + bank
  + signup\_requests
  + user\_access
  + account

# Reporting Requirements

1. The system must generate a mini statement showing the last 5 transactions with date, type, amount, and balance.
2. Only successful transactions should appear in the statement.
3. Users should be able to download the statement as a .txt file.
4. An option to email the mini statement to the registered email (optional).
5. Reports should be generated instantly during the user session from the transaction menu.

# Process Dependencies/Frequency

1. All operations are user-initiated and executed on demand.
2. The system performs real-time processing during active user sessions.
3. No external services or scheduled tasks are involved.
4. Data updates occur immediately per transaction (if persistence is used).
5. The system has no batch jobs or time-based dependencies.

# Security Requirements

* **PIN-Based Authentication**: Access is restricted to users with a valid card number and 4-digit numeric PIN.
* **PIN Format**: The PIN must be exactly **4 digits** (numeric only).
* **Credential Validation**: Both card number and PIN must match records in the database for successful login.
* **Access Control**: Role-based access is enforced (e.g., Customer vs. Employee privileges).
* **Attempt Limitation** *(optional)*: A login attempt limit can be added to lock accounts after repeated failures.
* **Data Handling**: All sensitive operations (PIN updates, transaction logs) are processed securely during each session.

# Availability and Processing Integrity Requirements

## 12.1. Availability Requirements

* Runs reliably on any system with a compatible Java Runtime Environment (JRE).
* GUI screens must load smoothly and stay responsive.
* Application should not crash or freeze during normal use.
* *(Optional)* Auto logout on session timeout for inactive users.

## 12.2. Processing Integrity Requirements

* All transactions must be validated before processing.
* Account balances update accurately after each operation.
* Inputs must follow correct formats (e.g., numeric amounts, 4-digit PIN).
* If persistence is used, data is stored and retrieved safely.
* User-friendly error messages must guide when inputs are invalid.

# Abbreviations

|  |  |
| --- | --- |
| Abbreviation | Full form |
| GUI | Graphical User Interface |
| JDBC | Java Database Connectivity |
| UI | User Interface |

# Related Documents

# References

| **Symbol** | | **Description** |
| --- | --- | --- |
|
|  |  | |
|  |  | |
|  |  | |
|  |  | |
|  |  | |
|  |  | |
|  |  | |

# Issues/Questions

|  |  |  |
| --- | --- | --- |
| **Status** | **Issue** | **Resolution** |
|  |  |  |
|  |  |  |

End of Document