Project Report on
"BANK MANAGEMENT SYSTEM"

DIPLOMA 3rd Year

(Branch – CSE)

Submitted To:

Manoj Sir

Submitted By:

Nitesh Kumar Mishra

SID-2247072

ACKNOWLEDGEMENT

I would like to express my special thanks of gratitude to my teacher Mr. **Manoj Sir**, who gave me the golden opportunity to do this wonderful project on **“BANK MANEGEMENT SYSTEM”**, Who also helped me in completing my project. I came to know about so many new things I am really thankful to them. Secondly I would also like to thank my parents and friends who helped me a lot in finalizing this project within the limited time frame.

Student’s Name

Nitesh Kumar Mishra

SID - 2247072

CONTENTS

INTRODUCTION

Overview

The “Bank Account Management System” project is a model Internet Banking Site. This site enables the customers to perform the basic banking transactions by sitting at their office or at homes through PC or laptop. The system provides the access to the customer to create an account, deposit/withdraw the cash from his account, also to view reports of all accounts present. The customers can access the banks website for viewing their Account details and perform the transactions on account as per their requirements. With Internet Banking, the brick and mortar structure of the traditional banking gets converted into a click and portal model, thereby giving a concept of virtual banking a real shape. Thus, today's banking is no longer confined to branches. E-banking facilitates banking transactions by customers round the clock globally.

The primary aim of this “Bank Account Management System” is to provide an improved design methodology, which envisages the future expansion, and modification, which is necessary for a core sector like banking. This necessitates the design to be expandable and modifiable and so a modular approach is used in developing the application software.
Anybody who is an Account holder in this bank can become a member of Bank Account Management System. He has to fill a form with his personal details and Account Number.
Bank is the place where customers feel the sense of safety for their property. In the bank, customers deposit and withdraw their money. Transaction of money also is a part where customer takes shelter of the bank. Now to keep the belief and trust of customers, there is the positive need for management of the bank, which can handle all this with comfort and ease.
Smooth and efficient management affects the satisfaction of the customers and staff members, indirectly. And of course, it encourages management committee in taking some needed decision for future enhancement of the bank.

Now a day’s, managing a bank is tedious job up to certain limit. So software that reduces the work is essential. Also, today’s world is a genuine computer world and is getting faster and faster day-by-day. Thus, considering above necessities, the software for bank management has become necessary which would be useful in managing the bank more efficiently.

Purpose and Objectives

The Traditional way of maintaining details of a user in a bank was to enter the details and record them. Every time the user needs to perform some transactions he has to go to bank and perform the necessary actions, which may not be so feasible all the time. It may be a hard-hitting task for the users and the bankers too. The project gives real life understanding of Online Banking System and activities performed by various roles in the supply chain. Here, we provide automation for banking system through Internet. Online Banking System project captures activities performed by different roles in real life banking which provides enhanced techniques for maintaining the required information up-to-date, which results in efficiency. The project gives real life understanding of Online Banking System and activities performed by various roles in the supply chain.

Main Goal

1. Motto- Our motto is to develop a software program for managing the entire bank process related to Administration accounts customer accounts and to keep each every track about their property and their various transaction processes efficiently.
Hereby, our main objective is the customer's satisfaction considering today's faster in the world.

2. Customer Satisfaction: Client can do his operations comfortably without any risk or losing of his privacy. Our software will perform and fulfill all the tasks that any customer would desire.

3. Saving Customer Time: Client doesn't need to go to the bank to do small operation.

4. Protecting the Customer: It helps the customer to be satisfied and comfortable in his choices, this protection contains customer's account, money and his privacy.

5. Transferring Money: Help client transferring money to/or another bank or country.

Methods

• We need to be able to generate an account number
• Account types: Savings or Current Account
• Maintain/update Balance
• Open/Close Account
• Withdraw/Deposit

Administrative Modules

Here in my project there are two types of modules. This module is the main module which performs all the main operations in the system. The major operations in the system are:

Admin Module

Admin can access this project there is an authorization process. If you login as an Admin then you will be redirected to the Admin Home Page and if you are a simple user you will be redirected to your Account Home Page. This performs the following functions: Create
Individual Accounts, manage existing accounts, View all transactions, Balance enquiry,
Delete/close account etc.

1- Admin login
2- Add/delete/update account
3- Withdrawal/deposit/statements transaction
4- Account Information
5- User details list
6- Active/Inactive account
7- View transaction histories

User Module

A simple user can access their account and can deposit/withdraw money from their account.
User can also transfer money from their account to any other bank account. User can see their transaction report and balance enquiry too.

1- User login, use PIN system
2- Creating/open new account registration
3- Funds transfer (local/international/domestic)
4- View statements transaction
5- User account details
6- Change Password and Pin

Scope of the Project - Bank Management System

The scope of the "Bank Management System" project encompasses the design, development, and implementation of a comprehensive software solution aimed at improving the operational efficiency and customer service of a banking institution. The system will primarily focus on managing various banking processes and operations while ensuring data security and compliance with regulatory standards. The following key elements define the scope of this project:

Inclusions:

Customer Account Management: The system will provide features for creating, updating, and managing customer accounts, including savings, current, and fixed deposit accounts.

1. *Transaction Processing:* The system will facilitate various banking transactions, including deposits, withdrawals, fund transfers, and account statements.

2. *Loan Management:* The project will include features to manage loan applications, approvals, disbursements, and repayments.

3. *Employee Management:* The system will include user roles and permissions for bank employees, allowing them to perform their specific job functions efficiently.

4. *Security Measures:* The project will prioritize data security and implement authentication, authorization, and encryption to protect sensitive customer information.
5. *Reporting and Analytics:* The system will provide tools for generating financial reports, customer

Exclusions:

1. *Third-Party Integrations:* The project will not encompass integration with external third-party services, such as payment gateways or credit bureaus.

2. *Physical Hardware Setup:* The project will not involve the procurement or setup of physical hardware components like servers or networking equipment.

3. *Training and Documentation:* While the system will include user documentation, the project will not provide training to bank employees.

4. *Legal and Compliance:* While the system will be designed to adhere to legal and regulatory requirements, the project will not address changes in legal and compliance standards beyond its initial implementation.

5. *Branch Expansion:* The project will not include plans for opening new branches or expanding the bank's physical presence.

Abstract

The Bank Account Management System is an application for maintaining a person's account in a bank. In this project I tried to show the working of a banking account system and cover the basic functionality of a Bank Account Management System. To develop a project for solving financial applications of a customer in banking environment in order to nurture the needs of an end banking user by providing various ways to perform banking tasks. Also, to enable the user's work space to have additional functionalities which are not provided under a conventional banking project.

The Bank Account Management System undertaken as a project is based on relevant technologies. The main aim of this project is to develop software for Bank Account Management System. This project has been developed to carry out the processes easily and quickly, which is not possible with the manuals systems, which are overcome by this software. This project is developed using Java language. Creating and managing requirements is a challenge of IT, systems and product development projects or indeed for any activity where you have to manage a contractual relationship. Organization need to effectively define and manage requirements to ensure they are meeting needs of the customer, while proving compliance and staying on the schedule and within budget.

The impact of a poorly expressed requirement can bring a business out of compliance or even cause injury or death. Requirements definition and management is an activity that can deliver a high, fast return on investment. The project analyzes the system requirements and then comes up with the requirements specifications. It studies other related systems and then come up with system specifications. The system is then designed in accordance with specifications to satisfy the requirements. The system design is then implemented with Java. The system is designed as an interactive and content management system. The content management system deals with data entry, validation confirm and updating whiles the interactive system deals with system interaction with the administration and users. Thus, above features of this project will save transaction time and therefore increase the efficiency of the system

AIM of this project

The main aim of designing and developing this Internet banking System Java primarily based
Engineering project is to provide secure and efficient net banking facilities to the banking customers over the internet. Apache Server Pages, MYSQL database used to develop this bank application where all banking customers can login through the secured web page by their account login id and password. Users will have all options and features in that application like get money from western union, money transfer to others, and send cash or money to inter banking as well as other banking customers by simply adding them as payees.

Banks terms:

1. All requests received from customers are logged for backend fulfillment and are effective from the time they are recorded at the branch.
2. Rules and regulations applicable to normal banking transactions in India will be applicable mutatis mutandis for the transactions executed through this site.
3. The BAMS Bank service cannot be claimed as a right. The bank may also convert this into a discretionary service anytime.
4. Dispute between the customer and the Bank in this service is subject to the jurisdiction of the courts in the Republic of India and governed by the laws prevailing in India.
5. The Bank reserves the right to modify the services offered or the Terms of service of
BAMS Bank. The changes will be notified to the customers through a notification on the Site.

Customer’s obligations

1. The customer has an obligation to maintain secrecy in regard to Username & Password registered with the Bank. The bank presupposes that login using valid Username and Password is a valid session initiated by none other than the customer.

2. Transaction executed through a valid session will be construed by RR to have emanated from the registered customer and will be binding on him/her.
3. The customer will not attempt or permit others to attempt accessing the BAMS Bank through any unlawful means.

Benefits of online banking

Many of us lead busy lives. Some of us are up before the crack of dawn, getting ourselves prepared so we can in turn get our families ready for the day. We rush to work, rush to get the kids to school, and at the end of the day we rush home only to brace ourselves for the next day. After a hectic day, the last thing you want to do is spend time waiting in line at the bank, or even the post office. That's where Online Banking comes in. Many of the benefits of doing our banking online are obvious:

1- You don't have to wait in line.
2- You don't have to plan your day around the bank's hours.
3- You can look at your balance whenever you want, not just when you get a statement.

There are some hidden benefits too. As a young bank customer, you're just learning how to manage your money and observe your spending patterns.
Online banking allows you to watch your money on a daily basis if you want to.
By keeping close tabs on your funds, you'll always be aware of what's happening in your bank account.
For those experienced spenders, this option is far more appealing than the sudden discovery that you're broke!

It's also helpful to watch how much interest you're gathering on investments and savings or what service charges you have incurred.

Most available benefits

1. Online banking with key bank is fast, secure, convenient and free.
2. Quick, simple, authenticated access to accounts via the web application.
3. Simply scalable to grow with changing system requirement.
4. Global enterprise wide access to information.
5. Improved data security, restricting unauthorized access.
6. Minimize Storage Space.

System Requirements

Hardware Requirements:

Server:

A powerful server to host the database and the application.

Multi-core CPU (e.g., Intel Xeon or AMD Ryzen)

Sufficient RAM (e.g., 8GB or more)

Adequate storage (e.g., 128 SSD for faster database access)

Client Machines:

Computers for bank employees and customers to access the system.

Moderate CPU and RAM (4GB or more)

Network connectivity to access the system over a local network or the internet.

Networking:

A reliable network infrastructure, especially if the system is accessed remotely.

Security Measures:

Firewalls, intrusion detection systems, and other security measures to protect sensitive financial data.

Software Requirements:

Operating System:

Server: Linux (e.g., CentOS, Ubuntu) or Windows Server

Client: Windows, macOS, or Linux

Database Management System (DBMS):Java-based projects often use databases like MySQL, PostgreSQL, or Oracle. The choice depends on your specific requirements and scalability needs.

Required: import the jcalendar and jdbc connector jar file for connecting database.

Java Development Kit (JDK):

Install the latest version of the Java SE Development Kit (JDK) to develop and run your Java application.

Integrated Development Environment (IDE):

You can use popular Java IDEs such as Eclipse, IntelliJ IDEA,Vs Code , or NetBeans for development.

Web Server (if applicable):

If your Bank Management System has a web-based interface, you might need a web server such as Apache Tomcat.

Version Control System:

Use version control tools like git for source code management.

Programming Frameworks and Libraries:

Depending on the design and functionality of your system, you may need various Java libraries and frameworks.

Security Software:

Implement security measures like antivirus and firewall software on all machines to protect against threats.

Backup and Recovery Tools:

Set up regular data backup and recovery procedures to ensure data integrity.

Documentation Tools:

Use documentation tools such as Microsoft Word, or Google Docs to create user manuals and technical documentation.

Collaboration and Communication Tools:

Tools for project management, team communication, and collaboration (e.g., JIRA, Slack, Microsoft Teams).

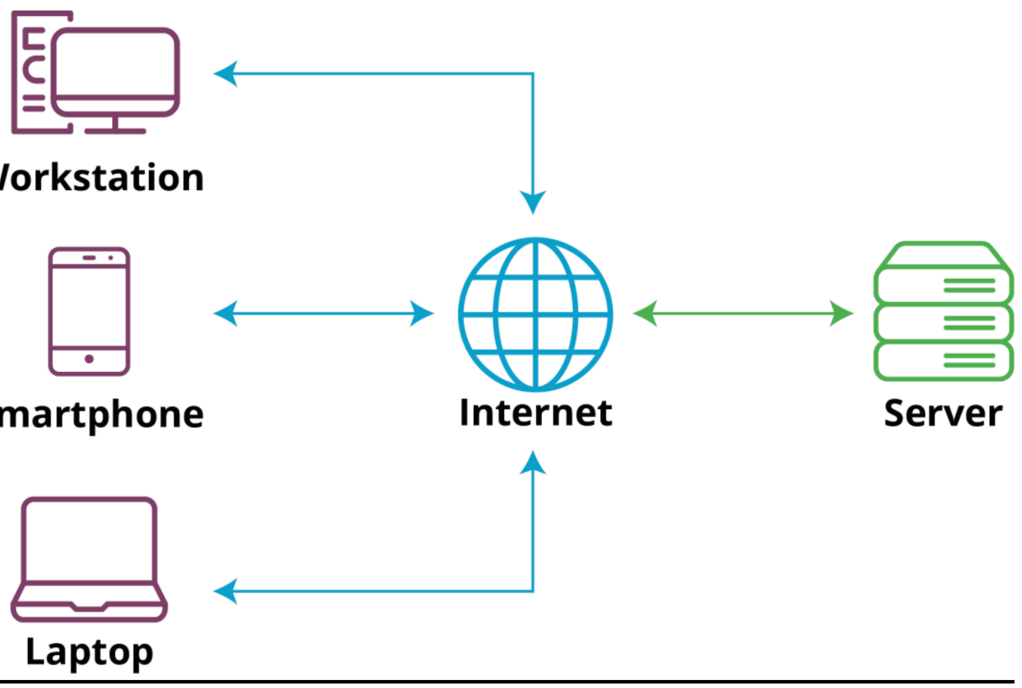
System Design

- Designing a Bank Management System involves creating a blueprint for how the system will be structured and how its components will interact. Here's a high-level overview of the system design for a Bank Management System:

System Architecture:

Client-Server Architecture:

The system follows a client-server model where client machines (bank employees and customers) interact with a central server.



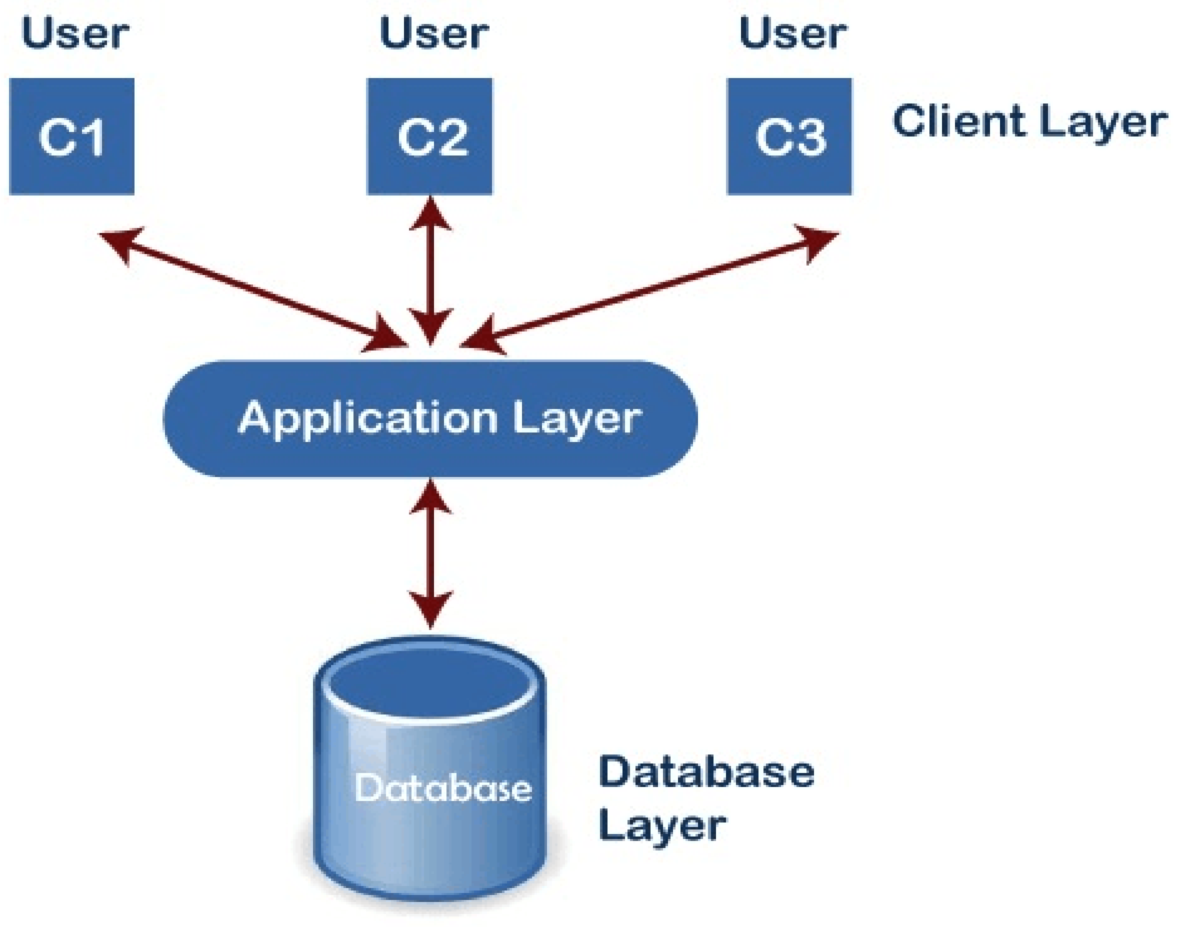
Three-Tier Architecture:

It's common to implement a three-tier architecture:

Presentation Tier (Client): This is where the user interacts with the system through a graphical user interface (GUI). Java Swing or JavaFX can be used to build desktop applications, or web technologies like JSP/Servlets or Spring Boot for web-based applications.

Application Tier (Server): This layer contains the application logic. It handles user requests, communicates with the database, and manages the core functionalities. Java EE or Spring can be used for building server-side components.

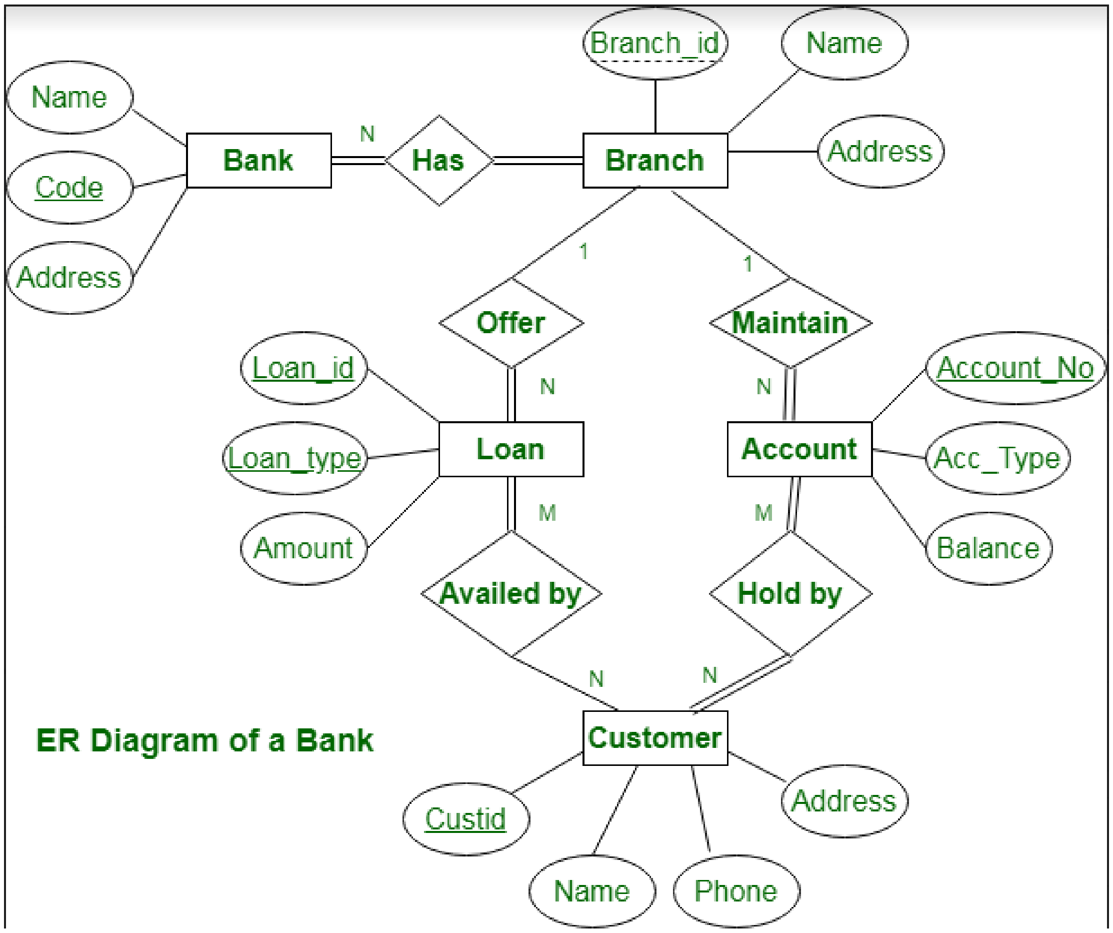
Data Tier (Database): This is where customer data, account details, transaction records, and other information are stored. You can use a relational database management system (RDBMS) like MySQL, PostgreSQL, or Oracle.



Database Design:

Entity-Relationship Diagram (ERD): Create an ERD to model the data entities such as customers, accounts, transactions, loans, and employees. Define the relationships between these entities.

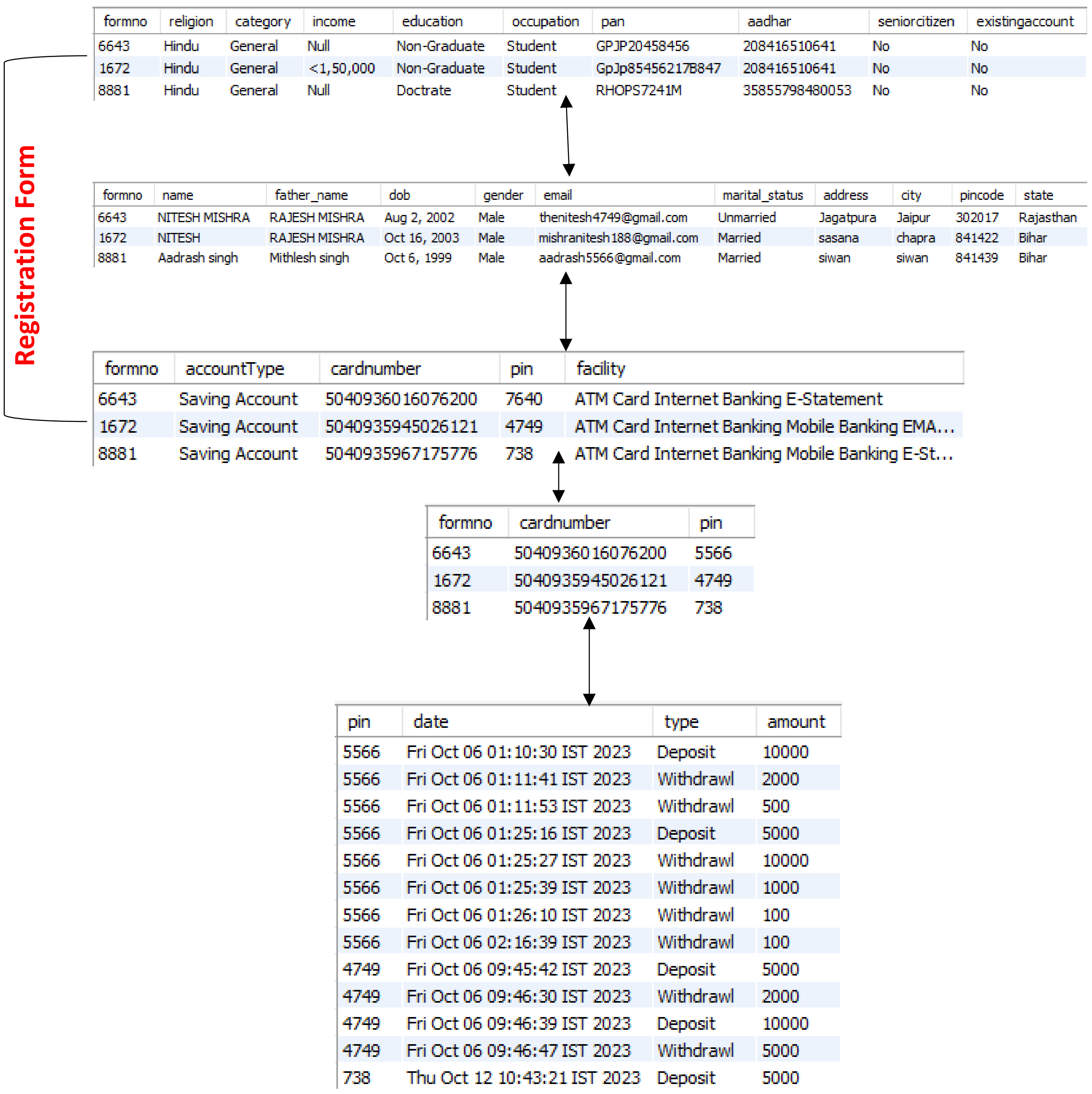
<u>ER Diagram</u>



Normalization: Apply the principles of database normalization to minimize redundancy and improve data integrity.

Table Structure: Design the database tables, specifying fields, data types, primary keys, foreign keys, and constraints.

Database (MYSQL) Table Structure(Bank Management System)



formno	religion	category	income	education	occupation	pan	aadhar	seniorcitizen	existingaccount
6643	Hindu	General	Null	Non-Graduate	Student	GPJP20458456	208416510641	No	No
1672	Hindu	General	<1,50,000	Non-Graduate	Student	GpJp85456217B847	208416510641	No	No
8881	Hindu	General	Null	Doctrate	Student	RHOPS7241M	35855798480053	No	No

formno	name	father_name	dob	gender	email	marital_status	address	city	pincode	state
6643	NITESH MISHRA	RAJESH MISHRA	Aug 2, 2002	Male	thenitesh4749@gmail.com	Unmarried	Jagatpura	Jaipur	302017	Rajasthan
1672	NITESH	RAJESH MISHRA	Oct 16, 2003	Male	mishranitesh188@gmail.com	Married	sasana	chapra	841422	Bihar
8881	Aadrash singh	Mithlesh singh	Oct 6, 1999	Male	aadrash5566@gmail.com	Married	siwan	siwan	841439	Bihar

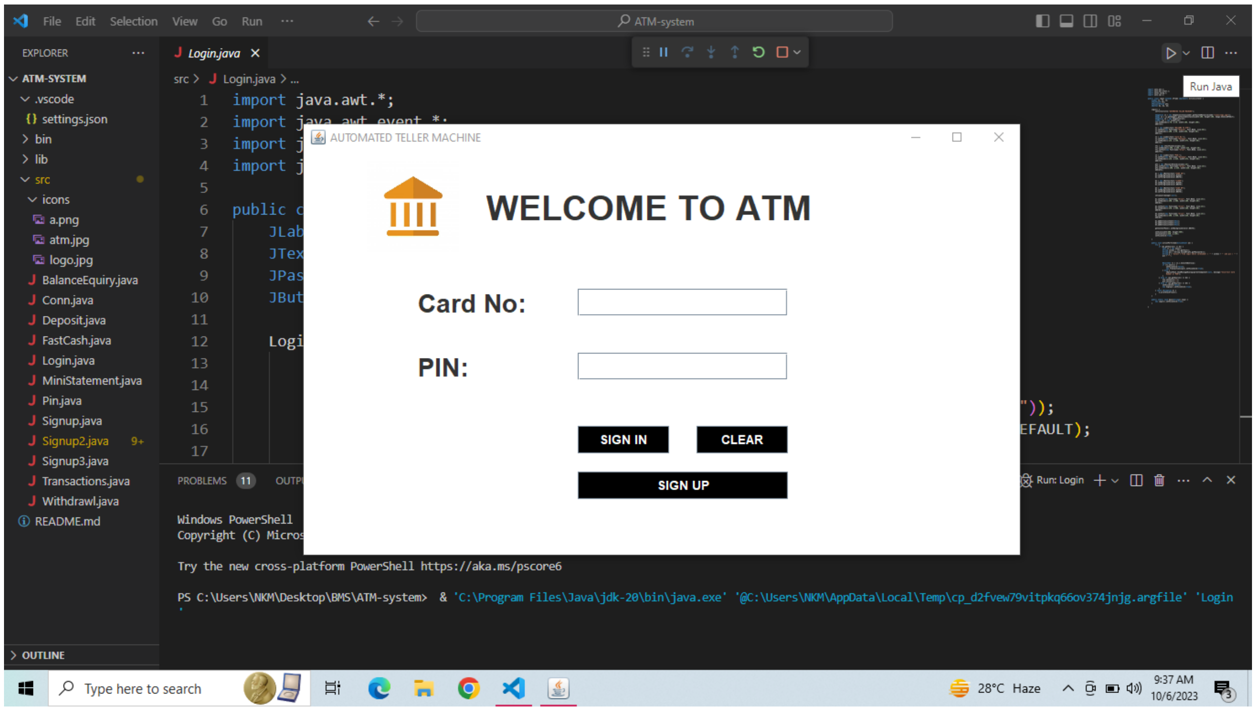
formno	accountType	cardnumber	pin	facility
6643	Saving Account	5040936016076200	7640	ATM Card Internet Banking E-Statement
1672	Saving Account	5040935945026121	4749	ATM Card Internet Banking Mobile Banking EMA...
8881	Saving Account	5040935967175776	738	ATM Card Internet Banking Mobile Banking E-St...

formno	cardnumber	pin
6643	5040936016076200	5566
1672	5040935945026121	4749
8881	5040935967175776	738

pin	date	type	amount
5566	Fri Oct 06 01:10:30 IST 2023	Deposit	10000
5566	Fri Oct 06 01:11:41 IST 2023	Withdrawl	2000
5566	Fri Oct 06 01:11:53 IST 2023	Withdrawl	500
5566	Fri Oct 06 01:25:16 IST 2023	Deposit	5000
5566	Fri Oct 06 01:25:27 IST 2023	Withdrawl	10000
5566	Fri Oct 06 01:25:39 IST 2023	Withdrawl	1000
5566	Fri Oct 06 01:26:10 IST 2023	Withdrawl	100
5566	Fri Oct 06 02:16:39 IST 2023	Withdrawl	100
4749	Fri Oct 06 09:45:42 IST 2023	Deposit	5000
4749	Fri Oct 06 09:46:30 IST 2023	Withdrawl	2000
4749	Fri Oct 06 09:46:39 IST 2023	Deposit	10000
4749	Fri Oct 06 09:46:47 IST 2023	Withdrawl	5000
738	Thu Oct 12 10:43:21 IST 2023	Deposit	5000

User Interface Design

Login and Authentication: Implement a secure login system for both employees and customers. Passwords should be hashed and stored securely.

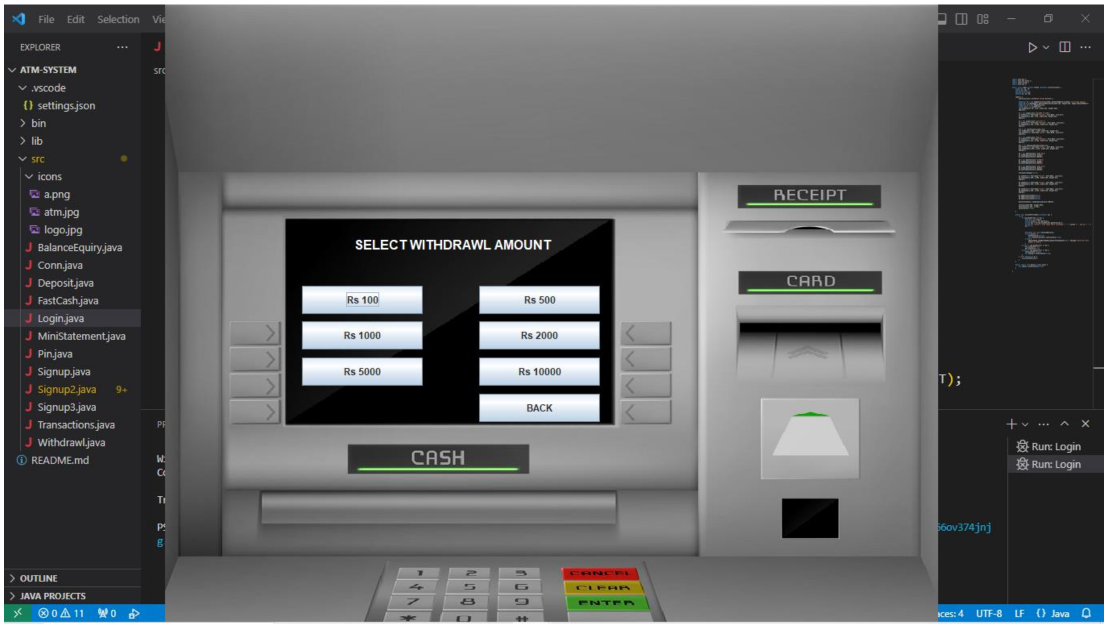


```
import java.awt.*;
import java.awt.event.*;
import j
import j

public c
    JLab
    JTex
    JPas
    JBut

    Logi
```

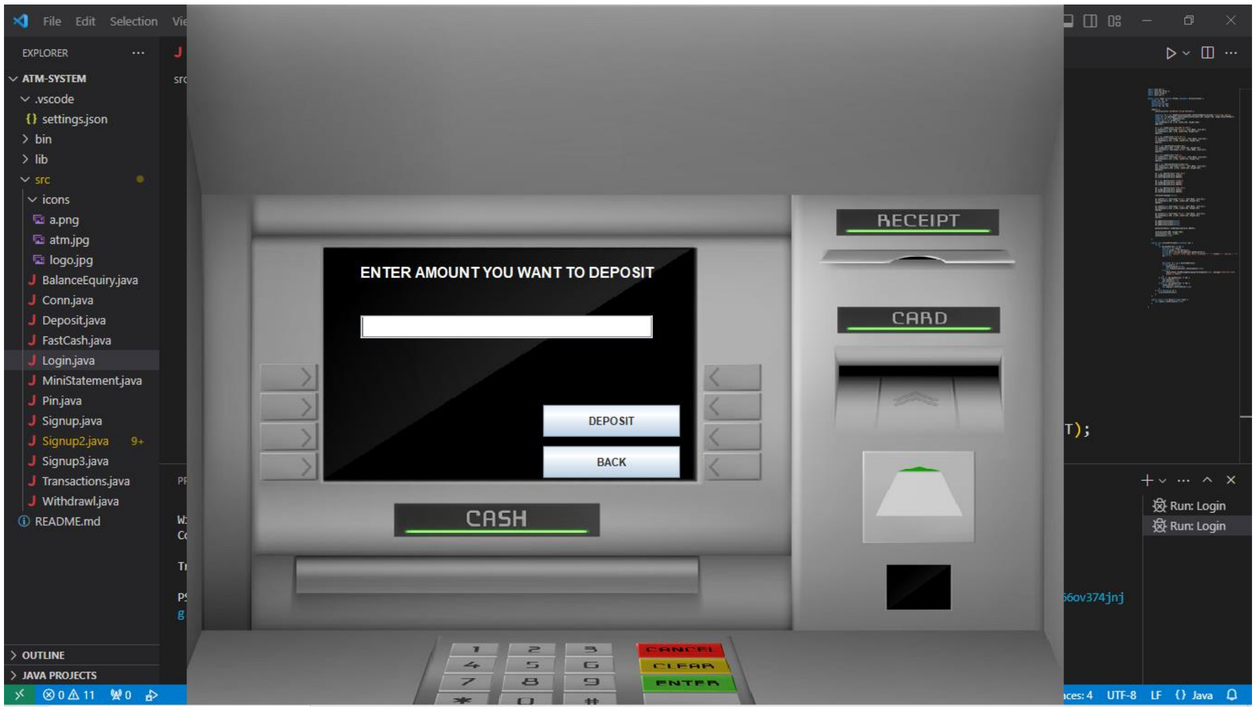
```
"));
EFAULT);
```

```
Windows PowerShell
Copyright (C) Micros

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\NKM\Desktop\BMS\ATM-system>  & 'C:\Program Files\Java\jdk-20\bin\java.exe' '@C:\Users\NKM\AppData\Local\Temp\cp_d2fvew79vitpkq66ov374jnjg.argfile' 'Login'
```

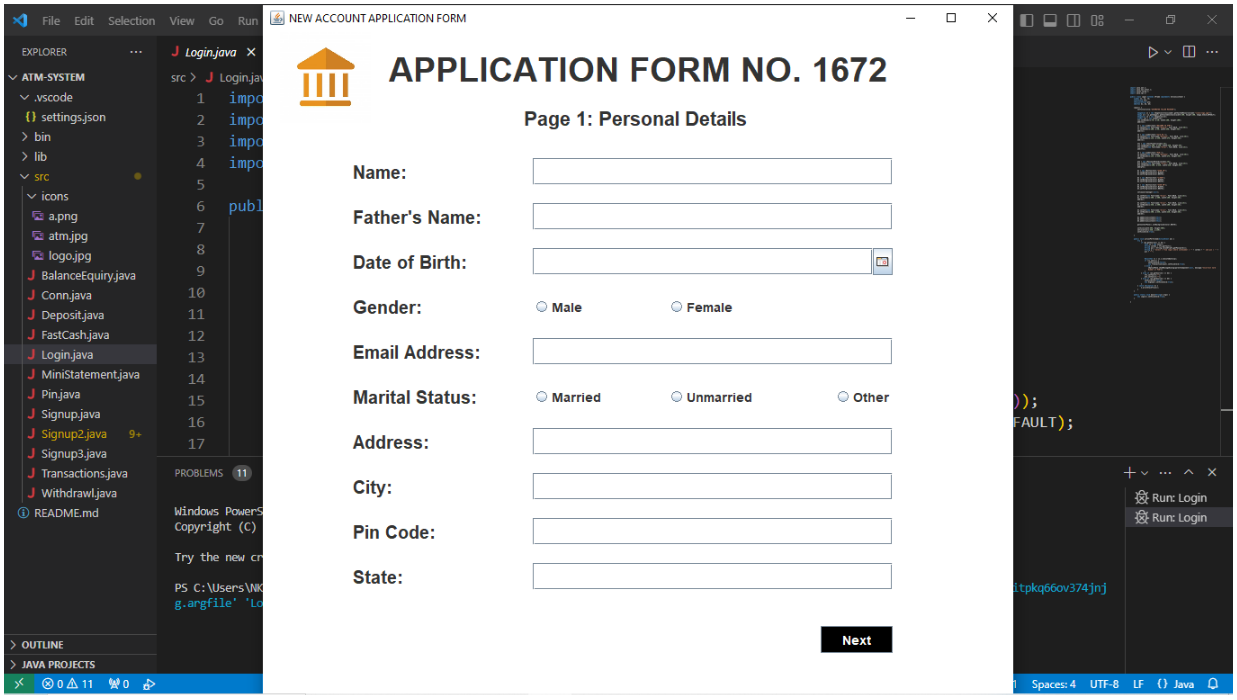
Dashboard: Create a user-friendly dashboard for bank employees, showing an overview of customer accounts, transactions, and other key information.



Customer Portal: Develop a user interface for customers to access their accounts, view balances, make transactions, and request services.



Forms: Design forms for account creation, transaction processing, loan applications, and more. Implement validation to ensure data accuracy.



NEW ACCOUNT APPLICATION FORM

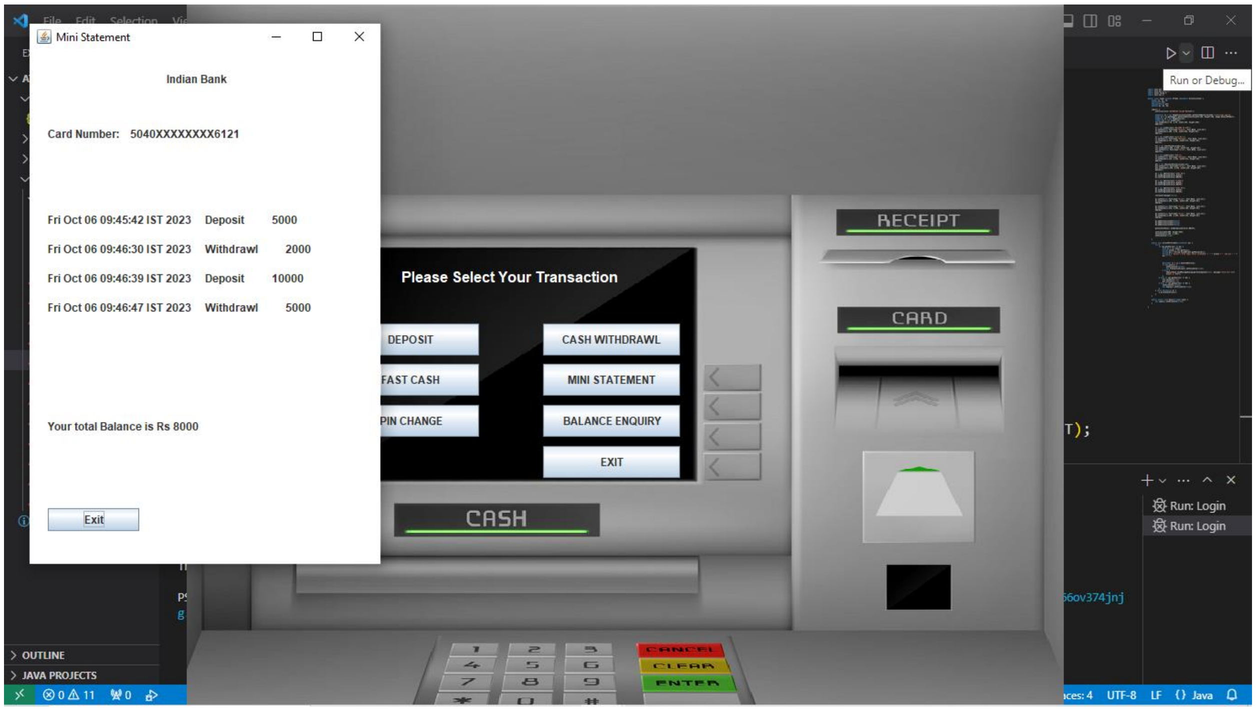
APPLICATION FORM NO. 1672

Page 1: Personal Details

Name:	
Father's Name:	
Date of Birth:	
Gender:	Male Female
Email Address:	
Marital Status:	Married Unmarried Other
Address:	
City:	
Pin Code:	
State:	

Next

***Reports**: Include a reporting module for generating financial reports, transaction history, and other analytics.*



Mini Statement

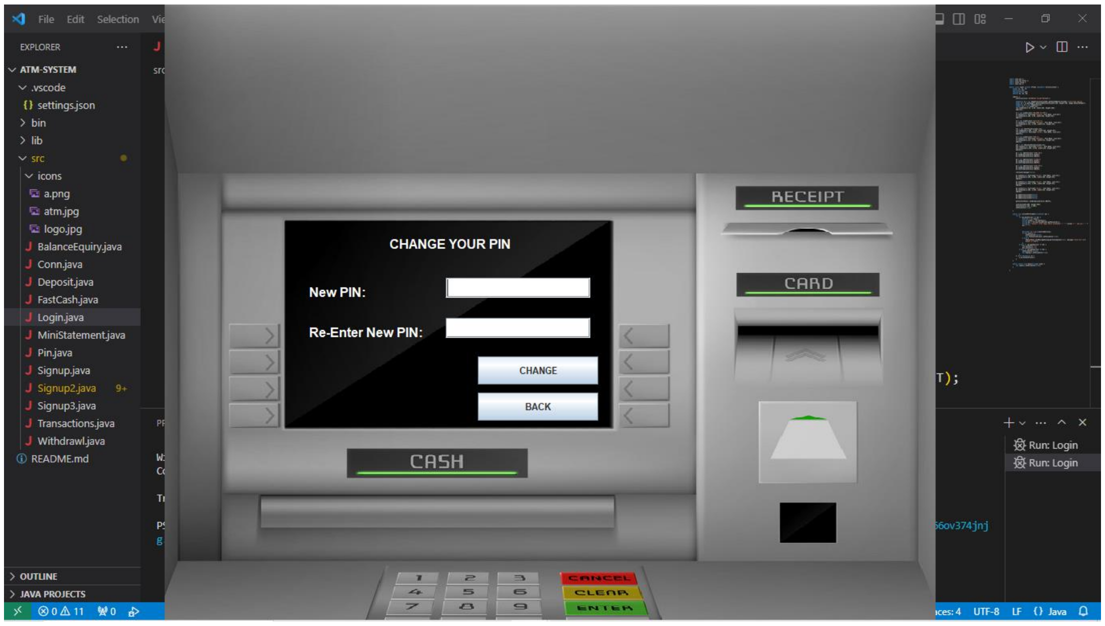
Indian Bank

Card Number: 5040XXXXXXXX6121

```
Fri Oct 06 09:45:42 IST 2023   Deposit     5000
Fri Oct 06 09:46:30 IST 2023   Withdrawl    2000
Fri Oct 06 09:46:39 IST 2023   Deposit     10000
Fri Oct 06 09:46:47 IST 2023   Withdrawl    5000
```

Your total Balance is Rs 8000

***Authentication and Authorization:** Implement secure login and access control mechanisms to ensure that only authorized users can perform specific actions.*



System Implementation

- Implementing a Bank Management System in Java and MySQL involves creating a software application that allows you to manage customer accounts, perform transactions, and maintain a database to store customer information and account details. Below, I'll provide a simplified outline of how you can design and implement such a system.

Note: This is a high-level overview, and you would need to write a substantial amount of code and create a MySQL database with the appropriate schema. I'll provide code snippets and explanations for key parts of the system.

Prerequisites:

- Install Java Development Kit (JDK)
- Set up MySQL and a MySQL Java connector (like JDBC)

Database Setup:

- Create a MySQL database to store information about customers, accounts, and transactions. Here's a basic schema:

Registration Table:

- ✓ Form Number (Primary Key)
- ✓ Full Name
- ✓ Father’s Name
- ✓ Gender
- ✓ Address
- ✓ Phone
- ✓ Email
- ✓ Pan No.
- ✓ Aadhar No.
- ✓ Account-Type

Accounts Table:

- ✓ Card Number (Primary Key)
- ✓ ATM PIN(Foreign Key)
- ✓ Balance
- ✓ AccountType (e.g., Savings, Checking)

Transactions Table:

- ✓ Transaction ID (Primary Key)
- ✓ Card Number (Foreign Key)
- ✓ Amount
- ✓ Transaction Type (Deposit, Withdrawal, Transfer)
- ✓ Transaction Date

Java Implementation:

- Registration Class: Create a Java class to represent a customer, with fields like full Name, Father's Name, etc.
- Account Class: Create a class to represent an account, with fields like Card-number, customer, balance, and methods for deposit, withdrawal, etc.
- Transaction Class: Implement a class to represent transactions, with fields like Transaction ID, account, amount, and Transaction Type, Mini-statement.
- Database Connection: Establish a connection to your MySQL database using JDBC.

Customer Management:

Create, read, update, and delete customer records in the database.

Account Management:

Create, read, update, and delete accounts.

Implement methods to deposit and withdraw money from accounts.

Coding

CLASS Login :

```java
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.sql.*;

public class Login extends JFrame implements ActionListener {
    JLabel l1, l2, l3;
    JTextField tf1;
    JPasswordField pf2;
    JButton b1, b2, b3;

    Login() {
        setTitle("AUTOMATED TELLER MACHINE");

        ImageIcon i1 = new
ImageIcon(ClassLoader.getSystemResource("icons/logo.jpg"));
        Image i2 = i1.getImage().getScaledInstance(100, 100,
Image.SCALE_DEFAULT);
        ImageIcon i3 = new ImageIcon(i2);
        JLabel l11 = new JLabel(i3);
        l11.setBounds(70, 10, 100, 100);
        add(l11);

        l1 = new JLabel("WELCOME TO ATM");
        l1.setFont(new Font("Osward", Font.BOLD, 38));
        l1.setBounds(200, 40, 450, 40);
        add(l1);

        l2 = new JLabel("Card No:");
        l2.setFont(new Font("Raleway", Font.BOLD, 28));
        l2.setBounds(125, 150, 375, 30);
        add(l2);
        tf1 = new JTextField(15);
        tf1.setBounds(300, 150, 230, 30);
        tf1.setFont(new Font("Arial", Font.BOLD, 14));
        add(tf1);

        l3 = new JLabel("PIN:");
        l3.setFont(new Font("Raleway", Font.BOLD, 28));
        l3.setBounds(125, 220, 375, 30);
        add(l3);

        pf2 = new JPasswordField(15);
        pf2.setFont(new Font("Arial", Font.BOLD, 14));
        pf2.setBounds(300, 220, 230, 30);
        add(pf2);

        b1 = new JButton("SIGN IN");
        b1.setBackground(Color.BLACK);
        b1.setForeground(Color.WHITE);

        b2 = new JButton("CLEAR");
        b2.setBackground(Color.BLACK);
        b2.setForeground(Color.WHITE);

        b3 = new JButton("SIGN UP");
        b3.setBackground(Color.BLACK);
        b3.setForeground(Color.WHITE);

        setLayout(null);

        b1.setFont(new Font("Arial", Font.BOLD, 14));
        b1.setBounds(300, 300, 100, 30);
        add(b1);

        b2.setFont(new Font("Arial", Font.BOLD, 14));
        b2.setBounds(430, 300, 100, 30);
        add(b2);

        b3.setFont(new Font("Arial", Font.BOLD, 14));
        b3.setBounds(300, 350, 230, 30);
        add(b3);

        b1.addActionListener(this);
        b2.addActionListener(this);
        b3.addActionListener(this);

        getContentPane().setBackground(Color.WHITE);

        setSize(800, 480);
        setLocation(320, 130);
        setVisible(true);

    }

    public void actionPerformed(ActionEvent ae) {
        try {
            if (ae.getSource() == b1) {
                Conn c1 = new Conn();
                String cardno = tf1.getText();
                String pin = String.valueOf(pf2.getPassword());
                String q = "select * from login where
cardnumber = '" + cardno + "' and pin = '" + pin + "'";



                ResultSet rs = c1.s.executeQuery(q);
                if (rs.next()) {
                    setVisible(false);
                    new Transactions(pin).setVisible(true);
                } else {
                    JOptionPane.showMessageDialog(null,
"Incorrect Card Number or PIN");
```

```
                }
            } else if (ae.getSource() == b2) {
                tf1.setText("");
                pf2.setText("");
            } else if (ae.getSource() == b3) {
                setVisible(false);
                new Signup().setVisible(true);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    public static void main(String[] args) {
        new Login().setVisible(true);
    }

}
```

CLASS Withdrawl :

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.util.Date;
import java.sql.*;

public class Withdrawl extends JFrame implements
ActionListener{
    JTextField t1,t2;
    JButton b1,b2,b3;
    JLabel l1,l2,l3,l4;
    String pin;
    Withdrawl(String pin){
        this.pin = pin;
        ImageIcon i1 = new
ImageIcon(ClassLoader.getSystemResource("icons/atm.jpg"));
        Image i2 = i1.getImage().getScaledInstance(1000, 850,
Image.SCALE_DEFAULT);
        ImageIcon i3 = new ImageIcon(i2);
        JLabel l3 = new JLabel(i3);
        l3.setBounds(0, 0, 960, 850);
        add(l3);

        l1 = new JLabel("MAXIMUM WITHDRAWAL IS RS.10,000");
        l1.setForeground(Color.WHITE);
        l1.setFont(new Font("System", Font.BOLD, 16));

        l2 = new JLabel("PLEASE ENTER YOUR AMOUNT");
        l2.setForeground(Color.WHITE);
        l2.setFont(new Font("System", Font.BOLD, 14));

        t1 = new JTextField();
        t1.setFont(new Font("Raleway", Font.BOLD, 20));

        b1 = new JButton("WITHDRAW");
        b2 = new JButton("BACK");

        setLayout(null);

        l1.setBounds(190,290,400,15);
        l3.add(l1);

        l2.setBounds(190,340,400,15);
        l3.add(l2);
        t1.setBounds(190,380,230,25);
        l3.add(t1);

        b1.setBounds(390,438,150,30);
        l3.add(b1);

        b2.setBounds(390,483,150,30);
        l3.add(b2);

        b1.addActionListener(this);
        b2.addActionListener(this);

        setSize(960,850);
        setLocation(200,0);
        setUndecorated(true);
        setVisible(true);
    }


    public void actionPerformed(ActionEvent ae){
        try{
            String amount = t1.getText();
            Date date = new Date();
            if(ae.getSource()==b1){
                if(t1.getText().equals("")){
                    JOptionPane.showMessageDialog(null, "Please
enter the Amount to you want to Withdraw");
                }else{
                    Conn c1 = new Conn();

                    ResultSet rs = c1.s.executeQuery("select *
from bank where pin = '"+pin+"'");
                    int balance = 0;
                    while(rs.next()){
                       if(rs.getString("type").equals("Deposit"
)){
                           balance +=
Integer.parseInt(rs.getString("amount"));
                       }else{
                           balance -=
Integer.parseInt(rs.getString("amount"));
                       }
                    }
                    if(balance < Integer.parseInt(amount)){
                        JOptionPane.showMessageDialog(null,
"Insuffient Balance");
                        return;
                    }

                    c1.s.executeUpdate("insert into bank
values('"+pin+"', '"+date+"', 'Withdrawl', '"+amount+"')");
                    JOptionPane.showMessageDialog(null, "Rs.
"+amount+" Debited Successfully");

                    setVisible(false);
                    new Transactions(pin).setVisible(true);
                }
            }else if(ae.getSource()==b2){
                setVisible(false);
                new Transactions(pin).setVisible(true);
            }
        }catch(Exception e){
                e.printStackTrace();
                System.out.println("error: "+e);
        }

    }     public static void main(String[] args){
        new Withdrawl("").setVisible(true);
    }}
```

CLASS Conn:

```
import java.sql.*;

public class Conn{
    Connection c;
    Statement s;
    public Conn(){
        try{
            Class.forName("com.mysql.cj.jdbc.Driver");
            c
=DriverManager.getConnection("jdbc:mysql:///bankmanagementsyste
m","root","556624");
            s =c.createStatement();



        }catch(Exception e){
            System.out.println(e);
        }
    }
}
```

Features and Functionalities

➢ User Registration and Login:

***User Registration*:**

1. *User Information:* Collect and store essential user information, including name, address, contact details, date of birth, and identification details (e.g., social security number).
2. *Account Setup:* Allow users to select the type of account they want (e.g., savings, checking, etc.).
3. *Security Credentials:* Users should create and verify their security credentials, such as a username and password. Encourage strong password policies.
4. *Personal Identification:* Implement a verification process, such as email or SMS verification, to confirm the user's identity.
5. *Additional Documents:* Depending on local regulations, you might need to collect additional documents or identification proofs.
6. *Terms and Conditions:* Users should agree to the bank's terms and conditions and privacy policies.

User Login:

1. *Username and Password:* Allow registered users to log in using their previously set username and password.
2. *Multi-Factor Authentication (MFA):* Implement MFA for an extra layer of security, requiring users to provide a second form of authentication, like a one-time code sent to their mobile device.
3. *Forgot Password:* Provide a 'Forgot Password' option that allows users to reset their password securely.
4. *Remember Me:* Offer an option for users to stay logged in on trusted devices for convenience.

➢ Account Management

Account management is a fundamental component of a bank management system. It involves a range of features and functionalities to ensure efficient and secure handling of customer accounts. Here are some key features and functionalities in a bank management system for account management:

1. *Customer Registration:* The system should allow bank employees to register new customers by collecting their personal and contact information. This may include name, address, contact number, email, and identification documents.
2. *Account Creation:* The ability to create different types of accounts such as savings, checking, fixed deposit, and more. Each type of account may have its own set of rules and features.
3. *Account Modification:* Enable customers to update their account details, such as a change of address, phone number, or email address. Bank employees should also be able to make necessary changes.
4. *Account Closure:* Provide the option for customers to close their accounts. Closure processes should include settling outstanding balances and transferring funds to another account or issuing a check.
5. *Account Balances:* Display real-time account balances, including available balances and pending transactions. This allows customers to keep track of their finances.
6. *Transaction History:* Maintain a comprehensive transaction history for each account, including deposits, withdrawals, transfers, and other transactions. This history should be easily accessible to both customers and bank employees.
7. *Transfer Funds:* Enable customers to transfer funds between their own accounts or to other accounts within the bank or to external banks via various methods, such as wire transfer, ACH, or online banking.
8. *Account Statements:* Generate and provide account statements to customers regularly, summarizing their account activity and balances over a specific period.
9. *Overdraft Protection:* Offer overdraft protection services, where customers can link their accounts to a savings or credit account to cover insufficient fund situations.

10. *Account Lock and Unlock:* Allow customers to temporarily lock and unlock their accounts to prevent unauthorized access in case of lost or stolen cards or suspected fraud.
11. *Account Security:* Implement robust security measures, including multi-factor authentication, to protect customer accounts from unauthorized access and fraud.
12. *Credit Card Integration:* If applicable, integrate credit card accounts into the system, allowing customers to view and manage their credit card balances and transactions.
13. *Customer Support and Communication:* Provide channels for customers to communicate with the bank, report issues, and seek assistance with their accounts.
14. *Data Backup and Recovery:* Implement data backup and recovery mechanisms to safeguard customer account information and ensure continuity of service in case of system failures.

Conclusion

This project is developed to nurture the needs of a user in a banking sector by embedding all the tasks of transactions taking place in a bank. Future version of this project will still be much enhanced than the current version. Writing and depositing checks are perhaps the most fundamental ways to move money in and out of a checking account, but advancements in technology have added ATM and debit card transactions. All banks have rules about how long it takes to access your deposits, how many debit card transactions you're allowed in a day, and how much cash you can withdraw from an ATM. Access to the balance in your checking account can also be limited by businesses that place holds on your funds.

Banks are providing internet banking services also so that the customers can be attracted. By asking the bank employs we came to know that maximum numbers of internet bank account holders are youth and business man. Online banking is an innovative tool that is fast becoming a necessity. It is a successful strategic weapon for banks to remain profitable in a volatile and competitive marketplace of today.
If proper training should be given to customer by the bank employs to open an account will be beneficial secondly the website should be made friendlier from where the customers can directly make and access their accounts.
Thus, the Bank Management System it is developed and executed successfully.

Future Look

The “Banking Online System is a big and ambitious project. I am thankful for being provided this great opportunity to work on it. As already mentioned, this project has gone through extensive research work. On the basis of the research work, we have successfully designed and implemented banking online System. To know what the future of online banking looks like, it’s probably worth looking at the present – online banking isn’t new. When you think of online banking, you probably think about a computer (either a desktop or laptop), a three or four step security process and then an interface that lets you view the balance of your various bank accounts and credit cards, whilst permitting you to transfer money and pay bills. And you’re not wrong either. The most valuable future looks are following below:

1- More branches of the bank, maybe it will be international, that means more ATM machines outside.
2- Customer issues development based on their needs, so the help desk will be aware of their needs and easy to use.
3- Developing a mobile App for banking system that help users to do the obtained his operations without go to the bank only he needs to sign in using his A/C NO. And password and then use your own PIN. Finally the system will update automatically.

References

1. **Java Programming:**

- "Java: The Complete Reference" by Herbert Schildt: This book is an excellent resource for Java programming and covers various topics from basics to advanced features.
- Oracle's official Java documentation: The official Java documentation is a comprehensive resource for learning Java, including tutorials and API documentation.

2. **MySQL Database:**

- "Learning MySQL" by YouTube Tutorial (Wscube Tech): This Tutorial provides a good introduction to MySQL, covering both basics and more advanced topics.
- MySQL official documentation: The MySQL documentation provides detailed information about MySQL features, SQL queries, and administration.

3. **User Interface Development:**

- JavaFX documentation: If you plan to create a desktop application with a graphical user interface (GUI), JavaFX is a popular choice. You can find tutorials and documentation on the official Oracle website.
- Swing documentation: Swing is another Java GUI library. You can find resources on creating Java Swing applications on the Oracle website and other online tutorials.

4. **Database Connectivity:**

- Java Database Connectivity (JDBC) documentation: You'll need to connect your Java application to the MySQL database using JDBC. The official JDBC documentation explains how to do this.

5. **Project Management and Version Control:**

- Git and GitHub: These tools are essential for version control and collaborative development. You can find documentation and tutorials on the GitHub website.

6. **Online Learning Platforms:**

- Online platforms like javatpoint.com, w3schools.com ,geeksforgeeks.org, Stack Overflow website and YouTube offer various Java and database courses that can help you build your skills.

7. **Open Source Projects:**

- Analyze open-source banking or financial system projects on platforms like GitHub. Studying their source code can provide insights into how similar systems are implemented.