Replace logo Holy-wood Academy, Kolhapur.

# {Institute Name}

*Institute Address*

# DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

**A PROJECT REPORT**

# ON

**“ *Bank Management System* ”**

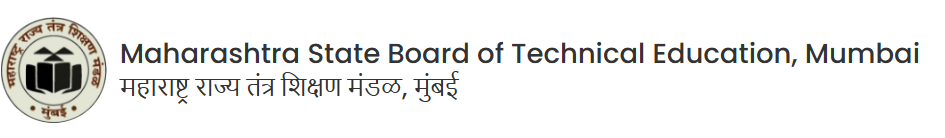
**Submitted in the partial fulfillment of the requirements for the award of degree of**

# Diploma(Polytechnic)

**IN**

# COMPUTER SCIENCE AND ENGINEERING

**Of**



# SUBMITTED BY

Project Group ID: 7

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr. No** | **Student Name** | **Roll No.** | **PRN** |
| 1 | Avinash Kamble | 46 | 23213570000 |
| 2 | Avdhut Taral | 14 | 23213570000 |
| 3 | Kartik Raut | 12 | 2321357000 |
| 4 | Aniruddh Hujare | 19 | 2321351200 |
| 5 | Gaurav Patil | 18 | 2321357000 |

# UNDER THE GUIDANCE OF

1

)

Ms. S. S. Patil

*Department of Computer Science & Engineering Sanjeevan Group Of Institution, Panhala*

# Academic Year: 2024-2025

Holy-wood Academy, Kolhapur.

# SANJEEVAN GROUP OF INSTITUTION

*Sanjeevan Knowledge City, Somwar Peth, Panhala, Kolhapur - 416201*

# DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that Project entitled **“Bank Management System”** is a Bonafide Project work carried out by:

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr. No** | **Student Name** | **Roll No.** | **PRN** |
| 1 | Avinash Kamble | 46 | 20000000121 |
| 2 | Avdhut Taral | 14 | 20300000006 |
| 3 | Kartik Raut | 12 | 23225035120 |
| 4 | Aniruddh Hujare | 19 | 23455110011 |
| 5 | Gaurav Patil | 18 | 23564550130 |

of Fourth Semester of Diploma in Computer Science And Engineering of Institute Sanjeevan Group Of Institution (Institute Code: 1739) has completed the Micro Project satisfactorily in Subject Java Programming (314317) for the academic year 2024-2025 as prescribed in the curriculum.

|  |  |  |
| --- | --- | --- |
| **Guide** | **HOD** | **Principal** |
| **Ms.** | **Mr.** | **Mr.** |

## Name of the Examiners: External Viva Signature with Date

**1**

**2.**

# DECLARATION

We hereby declare that this project work entitled “**Bank Management System**” has been prepared by us during the year 2024-25 under the guidance of **Ms. S. S. Patil,** Department of Computer Science, Sanjeevan Group Of Institution, Panhala in the partial fulfilment of the Micro Project prescribed by the college.

# ACKNOWLEDGEMENT

Motivation and guidance is the key towards the success. We would like to thank all the source of motivation and who guided us with due respect and gratitude.

We express our deep sense of gratitude to our project guide **Ms.{name}**, Lectural, Department of Computer Science and Engineering for his timely help and advice. We extend our heart full thanks to our beloved HOD **Mr.{name}**, Computer Science and Engineering who has been the constant driving behind the completion of this project.

We are extremely grateful to our beloved Principal **Mr. {name}** who encouraged us.

Finally we would like to extend our thanks to all teaching and non-teaching staff-members of Computer Science and Engineering for their co-operation in completing this project.

|  |  |  |
| --- | --- | --- |
| **SR.NO.** | **NAME** | **ROLL.NO.** |
| 1 | Avinash Kamble | 46 |
| 2 | Avdhut Taral | 14 |
| 3 | Kartik Raut | 12 |
| 4 | Aniruddh Hujare | 19 |
| 5 | Gaurav Patil | 18 |

# ABSTRACT

The **Bank Management System** is a desktop-based application developed using **Java AWT (Abstract Window Toolkit)** and **Swing** to provide an interactive and user-friendly interface for managing basic banking operations. The project aims to replicate essential banking functionalities such as **account creation**, **deposit**, **withdrawal**, **balance inquiry**, and **transaction history**, providing a digital alternative to manual banking processes.

This system is designed for **single-branch simulation** where an administrator or bank staff can manage customer accounts securely and efficiently. The GUI components from AWT and Swing allow for the creation of responsive forms, buttons, menus, and dialogs that enhance user interaction and usability. Data is handled through backend logic and can optionally be connected to a file system or database for persistence.

In real-world banking, such systems play a vital role in reducing human error, saving time, and ensuring the security of sensitive information. This project serves as a foundational model for larger enterprise banking applications, showcasing how a basic banking system can be prototyped and scaled using object-oriented principles and Java GUI components.

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 analyses 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.

**Keywords:-** MySQL Database, GUI Components.

# INDEX

|  |  |  |
| --- | --- | --- |
| **CHAPTER**  **NO** | **CONTENT** | **PAGE**  **NO** |
| **1** | **Introduction** | 1-1 |
| **2** | **Problem Statement**   * 1. Need of Work   2. Problem Definition   2.3 Objective | 2-4 |
| **3** | **System Design and Requirement Specification Analysis**   * 1. System Architecture   2. Block Diagrams   3. System Requirements | 5-7 |
| **4** | **Implementation**   * 1. Implementation Modules Description   2. Software Platforms, Languages & Tools Used | 8-10 |
| **5** | **Testing** | 11 |
| **6** | **Source Code & Snapshots** | 12-37 |
| **7** | **Conclusion & Future Work** | 37-38 |
| **8** | **References** | 38 |

1. **INTRODUCTION**

The **Bank Management System** is a mini-project designed to demonstrate how core banking functions can be automated and digitized using programming concepts and graphical user interfaces. This project is developed using **Java's AWT (Abstract Window Toolkit)** and **Swing**, which are powerful tools for creating platform-independent desktop applications with rich user interfaces.

The system simulates a basic banking environment where users can perform common tasks such as **creating a new account**, **depositing and withdrawing money**, **checking balances**, and **viewing transaction history**. The primary objective is to eliminate the need for paper-based processes and manual record-keeping, which are time-consuming and prone to human error.

Using Java AWT and Swing allows the application to offer a **graphical user interface (GUI)** that is not only user-friendly but also responsive and intuitive. The backend logic, written in Java, ensures data is processed efficiently while maintaining the security and integrity of user information.

This project is ideal for educational purposes and serves as a foundational prototype that can be scaled up to support more complex banking features such as multi-user logins, account types, loan management, and database integration.

By implementing this system, learners and developers gain hands-on experience with:

* GUI design using Java Swing components (JFrame, JPanel, JButton, JLabel, etc.)
* Event handling and user input validation
* Object-oriented programming principles
* Basic data persistence (using files or optional database)

Ultimately, this project bridges the gap between theoretical learning and practical application, showcasing how software can effectively solve real-world problems in the banking sector.

In today’s environment, a mobile has become one of the

In today’s environment, a mobile has become one of th

In today’s environment, a mobile has become one of theimportant necessities. If a person wants to search his family member or a child or if a person who loses his mobile wants to track it, or a Person wants to know who is disturbing him by giving missed calls or irrelevant SMS, a person becomes handicapped if he is unable to get in touch with the contacts. With the help of GSM technology, it is possible to trace and find the Location of person with the help of 2G and 3G mobile phones which the other person is using. There are many applications on the internet through which one can detect the location of user. These applications include survillence, detection of fraud, help in business marketing etc. There are two types of addresses that are assigned to user i.e. public IP addresses and private IP addresses for data distribution and performance tuning of a System. One can also find the Location of user through IP address for immediate or any specific service. A few data services such as Maxmind are provide that maintains the tables including IP addresses with the locations that matches. The important requirement of general user is to get the Location of his family member. This can be done through sending and receiving SMS. The Paper describes the general architecture of GSM and an IP cellular Network to determine the location of user. In further Sections, a paper will determine various methods of Location Tracking and compare their resu

1. **PROBLEM STATEMENT**

In the traditional banking process, many operations such as account creation, deposits, withdrawals, and balance inquiries are often handled manually or through outdated systems. These methods are **time-consuming**, **error-prone**, and **inefficient**, leading to delays in service, poor customer experience, and challenges in data management.

Smaller financial institutions or educational simulations often lack access to fully developed banking software due to **cost, complexity, or limited technical resources**. As a result, there is a critical need for a **simplified, user-friendly, and cost-effective desktop application** that can efficiently manage basic banking operations without requiring a complex backend infrastructure.

The lack of an **automated banking solution** causes the following issues:

* Difficulty in maintaining and retrieving customer records.
* High chances of human errors during financial transactions.
* Lack of real-time updates and secure access.
* Poor user experience due to command-line or paper-based operations.
* Inability to scale or adapt to modern banking needs.

To address these challenges, this project proposes the development of a **Bank Management System using Java AWT and Swing** that will provide an interactive GUI-based application. The system aims to automate common banking functions such as:

* Creating and managing customer accounts,
* Handling deposits and withdrawals,
* Displaying account balances,
* Tracking transaction history.

By leveraging Java's AWT and Swing libraries, the system will offer a **platform-independent**, **graphical user interface** that is intuitive, responsive, and suitable for small banking environments or as a learning model for understanding core financial application development.

* 1. **Need of work.**

The increasing demand for **automation and digitalization** in the banking sector has made it essential to replace outdated manual systems with reliable, fast, and secure software solutions. Traditional banking processes that involve maintaining physical records, managing transactions manually, and handling account-related tasks through paperwork are not only inefficient but also highly prone to human errors and data inconsistency.

Despite the availability of advanced banking software in large institutions, **small-scale banks, co-operative societies, educational institutions**, and **individual learners** often lack access to such systems due to cost and complexity. Therefore, there is a strong need for a **simplified and cost-effective banking application** that offers core functionality without the need for expensive infrastructure.

* **Reasons for the Need of Work:**

1. **Manual Operations are Inefficient:** Manual record-keeping and financial transactions consume a significant amount of time and effort, leading to slow service and dissatisfied customers.
2. **Error-Prone Processes:** Calculations and data entry done by hand can result in errors that may lead to financial discrepancies and loss of trust.
3. **Lack of Real-Time Access:** Manual or command-line systems do not offer real-time access to account information or transaction history, which is expected in today’s banking environment.
4. **No User-Friendly Interface:** Many existing systems in smaller setups do not have a proper graphical user interface, making them hard to use for staff without technical training.
   1. **Problem Definition :**

In the current era of digital transformation, banking operations have evolved significantly to improve customer experience, operational efficiency, and data accuracy. However, in many small-scale banks, cooperative societies, and educational environments, banking processes are still handled through **manual systems** or **basic command-line applications**, which pose serious limitations in terms of usability, reliability, and scalability.

The primary problem is the **absence of an interactive, easy-to-use, and secure system** to manage day-to-day banking operations such as **creating accounts, depositing money, withdrawing funds, checking balances, and tracking transaction history**. These tasks, when handled manually, are not only time-consuming but also vulnerable to errors and data mismanagement.

* 1. **Objectives :**

The primary objective of this project is to design and develop a **desktop-based Bank Management System** using **Java AWT (Abstract Window Toolkit)** and **Swing** that can simulate essential banking operations through a graphical user interface. This system aims to automate and simplify the banking process while offering a hands-on understanding of GUI-based application development in Java.

**Main Objectives:**

1. **To Develop a User-Friendly GUI:**
   * Create an intuitive and interactive graphical user interface using AWT and Swing components for seamless navigation and operation.
2. **To Simulate Core Banking Operations:**
   * Implement key functionalities such as:
     + Creating a new customer account
     + Deposit and withdrawal of funds
     + Checking account balance
     + Viewing transaction history
3. **To Ensure Data Accuracy and Security:**
   * Apply validation checks to avoid invalid inputs and ensure secure processing of transactions.
4. **To Replace Manual Banking Tasks:**
   * Automate record-keeping and account handling to reduce human errors and increase operational efficiency.

# SYSTEM DESIGN AND REQUIREMENT SPECIFICATION ANALYSIS

# System Architecture :

# The System Architecture of the Bank Management System outlines the structure, key components, and the flow of data within the application. This system follows a modular and layered architecture to ensure a clean separation of concerns, easy maintenance, and potential scalability.

# Developed using Java AWT (Abstract Window Toolkit) and Swing, the system is a standalone desktop application that allows users to perform fundamental banking operations through a graphical user interface (GUI).

# Concept Diagram :

# How to Create a Bank ATM Use Case Diagram | Banking System | UML use case diagram - Banking system | Use Case Diagram For Banking System In Uml

Fig 1: Conceptual Diagram of Bank Management System

* 1. **Block Diagram :**

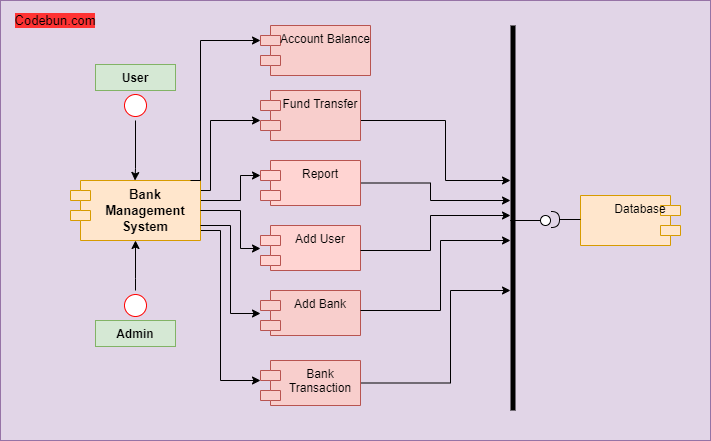


Fig 2: Block Diagram of Bank Management System

* 1. **System Requirements :**
* **Hardware Requirements**:

| **Component** |  | **Minimum Requirement** | **Recommended Requirement** |
| --- | --- | --- | --- |
| **Processor** |  | Intel Pentium IV or equivalent | Intel Core i5 or higher |
| **RAM** |  | 2 GB | 4 GB or higher |
| **Hard Disk** |  | 200 MB of free space | 500 MB or more |
| **Display** |  | 1024 x 768 resolution | 1366 x 768 or higher |
| **Input Devices** |  | Keyboard, Mouse | Keyboard, Mouse |

* **Software Requirements**:

| **Software Component** | **Specification** |
| --- | --- |
| **Operating System** | Windows 7/8/10/11, Linux, or macOS |
| **JDK (Java Development Kit)** | JDK 8 or above (preferably JDK 11 or newer) |
| **IDE (Optional)** | VS code,Eclipse, IntelliJ IDEA, or NetBeans |
| **Compiler** | Java Compiler (javac) |
| **File Storage (Optional)** | Text files (.txt, .dat) or lightweight DB like Mysql |
| **Java Runtime** | Java Runtime Environment (JRE) 8 or higher |

1. **IMPLEMENTATION**
   1. **Implementation Module Description :**

The **Bank Management System** is divided into several functional modules to enhance usability, maintainability, and scalability. Each module is responsible for handling specific tasks in the system, from user interaction to data processing and storage. The implementation is done using **Java AWT and Swing** for GUI components, supported by Java's object-oriented programming principles for business logic.

**🔹 1. Login Module**

**Purpose:**  
To authenticate users before granting access to the banking system.

**Implementation Details:**

* GUI built using JFrame, JTextField, JPasswordField, and JButton.
* Validates username and password against stored credentials (in file or hardcoded for simplicity).
* If valid, grants access to the main dashboard.
* Includes error messages for invalid login attempts.

**🔹 2. Account Creation Module**

**Purpose:**  
To create and register a new customer account in the system.

**Implementation Details:**

* GUI includes fields for name, age, address, mobile number, account type, and initial deposit.
* Uses JTextField, JComboBox, JButton, etc.
* Generates a **unique account number** for each user.
* Stores account details in a file (or array/list) for persistence.
* Validates that all required fields are filled and data is correctly formatted.

**🔹 3. Deposit Module**

**Purpose:**  
To allow users to deposit money into an existing account.

**Implementation Details:**

* Prompts for account number and deposit amount.
* Validates account existence and ensures the deposit amount is valid (positive number).
* Updates the account balance and logs the transaction.
* Displays updated balance to the user after deposit.

**🔹 4. Withdrawal Module**

**Purpose:**  
To allow users to withdraw money from their account.

**Implementation Details:**

* Requires account number and withdrawal amount.
* Checks whether the account exists and has sufficient balance.
* Deducts the amount and updates the new balance.
* Displays error message if funds are insufficient.

**🔹 5. Balance Inquiry Module**

**Purpose:**  
To show the current balance of a specific account.

**Implementation Details:**

* Takes account number as input.
* Searches stored data for the account.
* Displays the current balance and basic customer information.
* Optional: Can include account type and recent transaction date.

**🔹 6. Transaction History Module**

**Purpose:**  
To maintain and display a log of transactions performed by a customer.

**Implementation Details:**

* Stores transaction logs (date, time, type, amount, resulting balance).
* Displays transactions in a readable format using JTextArea or JTable.
* Useful for tracking account activity and generating basic reports.

**🔹 7. Exit / Logout Module**

**Purpose:**  
To safely exit the application or log out of the current session.

**Implementation Details:**

* Implemented using simple JButton events.
* Optionally asks for confirmation before closing the app.
* Clears session variables if implemented.
  1. **Software Platforms, Languages & Tools Used :**
* Software Platforms : VS Code (macOS, Linus and Windows)
* Languages : Java
* Documentation Tools : MS Word, Draw.io (for diagrams)
* GUI Libraries : AWT, Swing

1. **TESTING**

**Testing** is a crucial phase in the software development lifecycle. It ensures that the application functions as expected and meets the requirements defined during the design phase. For the **Bank Management System developed using Java AWT and Swing**, testing was conducted to verify the **correctness, usability, stability, and performance** of the system.

The testing process involved both **manual testing techniques** and structured **test cases** to validate different modules and use-case scenarios.

**Testing Objectives**

* To ensure that all core functionalities (like account creation, deposit, withdrawal, and balance inquiry) work as intended.
* To verify that user interactions through the GUI are intuitive and free of errors.
* To ensure the system handles invalid inputs gracefully.
* To validate file-based data storage and retrieval functions.
* To identify and fix bugs, inconsistencies, or crashes.

# 6.Source Code

**Login.java**

package bank.management.system;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

import java.sql.\*;

public class Login extends JFrame implements ActionListener {

    JTextField t1;

    JPasswordField t2;

    JButton login,signup,clear;

String pinnum;

    public Login(String pinnum){

         this.pinnum=pinnum;

        setTitle("Bank Management System ");

        ImageIcon i1=new ImageIcon(ClassLoader.getSystemResource("icon/bank.png"));

        Image i2= i1.getImage().getScaledInstance(100,100, Image.SCALE\_DEFAULT);

        ImageIcon i3= new ImageIcon(i2);

        JLabel image1= new JLabel(i3);

        image1.setBounds(490,40,100,100);

        add(image1);

        ImageIcon ii1=new ImageIcon(ClassLoader.getSystemResource("icon/card.png"));

        Image ii2=ii1.getImage().getScaledInstance(100,100,Image.SCALE\_DEFAULT);

        ImageIcon ii3= new ImageIcon(ii2);

        JLabel image2=new JLabel(ii3);

        image2.setBounds(850,425,100,100);

        add(image2);

        JLabel label1=new JLabel("Welcome To SGI Bank");

        label1.setForeground(Color.BLACK);

        label1.setFont(new Font("Arial",Font.BOLD,20));

        label1.setBounds(430,90,500,150);

        add(label1);

        JLabel label2=new JLabel("Enter Login ID:");

        label2.setBounds(350,150,150,150);

        label2.setFont(new Font("Raleway ",Font.BOLD,15));

        label2.setForeground(Color.BLACK);

        add(label2);

        t1=new JTextField(12);

        t1.setBounds(490,215,160,18);

        t1.setFont(new Font("Arial",Font.ROMAN\_BASELINE,14));

        add(t1);

        JLabel label3=new JLabel("Enter Password:");

        label3.setBounds(350,205,150,150);

        label3.setFont(new Font("Raleway",Font.BOLD,15));

        label3.setForeground(Color.black);

        add(label3);

        t2= new JPasswordField(8);

        t2.setBounds(490,271,160,18);

        t2.setEchoChar('\*');

        t2.setFont (new Font("Arial",Font.ROMAN\_BASELINE,20));

        add(t2);

        ImageIcon iii1= new ImageIcon(ClassLoader.getSystemResource(""));//image url//

        Image iii2= iii1.getImage().getScaledInstance(1050,850,Image.SCALE\_DEFAULT);

        ImageIcon iii3=new ImageIcon(iii2);

        JLabel image3= new JLabel(iii3);

        image3.setBounds(0,0,1050,850);

        add(image3);

         signup=new JButton("Sign Up");

         signup.setBounds(410, 360, 90, 27);

         signup.setBackground(Color.black);

         signup.setForeground(Color.WHITE);

         signup.setFont(new Font("Arial",Font.BOLD,14));

         signup.addActionListener(this);

         add(signup);

         clear=new JButton("Clear");

         clear.setBounds(580,360,90,27);

         clear.setBackground(Color.BLACK);

         clear.setForeground(Color.WHITE);

         clear.setFont(new Font("Arial",Font.BOLD,14));

         clear.setVisible(true);

         clear.addActionListener(this);

         add(clear);

         login=new JButton("Sign In");

         login.setBounds(496,430,90,27);

         login.setBackground(Color.BLACK);

         login.setForeground(Color.WHITE);

         login.setFont(new Font("Arial",Font.BOLD,14));

         login.addActionListener(this);

         add(login);

        setLayout(null);

        setSize(1050,550);

        setVisible(true);

        setLocation(450,200);

        setBackground(Color.LIGHT\_GRAY);

    }

    public void actionPerformed(ActionEvent e){

        String id=t1.getText();

        String pass=t2.getText();

        try{

            if (e.getSource()==login){

                conn c = new conn();

                String cardnum = t1.getText();

                String pin = t2.getText();

                String q = "select \* from login where card\_number = '"+cardnum+"' and  pin = '"+pin+"'";

                ResultSet resultSet = c.statement.executeQuery(q);

                if (resultSet.next()){

                    setVisible(false);

                    new mainfile(pin);

                }

                else if (id.equals("")) {

                        JOptionPane.showMessageDialog(null,"Please Enter Your Card Number.");

                    }

                    else if (pass.equals("")) {

                        JOptionPane.showMessageDialog(null,"Please Enter Your PIN.");

                    }

                  else{

                    JOptionPane.showMessageDialog(null,"Incorrect Card Number or PIN");

                }

            }else if (e.getSource() == clear){

                t1.setText("");

                t2.setText("");

            }

            else if (e.getSource() == signup){

                new signup();

                setVisible(false);

            }

        }

    catch(Exception ea){

        ea.printStackTrace();

    }

}

    public static void main(String[] args) {

        new Login("");

    }

}

**Mainfile.java**

package bank.management.system;

import java.awt.event.\*;

import java.awt.\*;

import javax.swing.ImageIcon;

import javax.swing.JButton;

import javax.swing.JFrame;

import javax.swing.JLabel;

public class mainfile extends JFrame implements ActionListener {

    JButton b1,b2,b3,b4,b5,b6,b7,b8,b9;

    String pinnum;

    public mainfile(String pinnum){

        this.pinnum=pinnum;

        ImageIcon ii1=new ImageIcon(ClassLoader.getSystemResource("icon/atm2.jpg"));

        Image i1=ii1.getImage().getScaledInstance(1550,830,Image.SCALE\_DEFAULT);

        ImageIcon i2=new ImageIcon(i1);

        JLabel l3=new JLabel(i2);

        l3.setBounds(0,0,1550,830);

        add(l3);

        JLabel label = new JLabel("Please Select Your Transaction");

        label.setBounds(600,12,700,35);

        label.setForeground(Color.BLACK);

        label.setFont(new Font("Arial",Font.BOLD,28));

        l3.add(label);

        b1 = new JButton("DEPOSIT");

        b1.setForeground(Color.WHITE);

        b1.setBackground(Color.magenta);

        b1.setBounds(330,85,150,35);

        b1.addActionListener(this);

        l3.add(b1);

        b2 = new JButton("CASH WITHDRAWL");

        b2.setForeground(Color.white);

        b2.setBackground(Color.black);

        b2.setBounds(600,85,150,35);

        b2.addActionListener(this);

        l3.add(b2);

        b3 = new JButton("FAST CASH");

        b3.setForeground(Color.white);

        b3.setBackground(Color.magenta);

        b3.setBounds(464,140,150,35);

        b3.addActionListener(this);

        l3.add(b3);

        b4 = new JButton("MINI STATEMENT");

        b4.setForeground(Color.WHITE);

        b4.setBackground(Color.BLACK);

        b4.setBounds(1000,85,150,35);

        b4.addActionListener(this);

        l3.add(b4);

        b5 = new JButton("PIN CHANGE");

        b5.setForeground(Color.WHITE);

        b5.setBackground(Color.magenta);

        b5.setBounds(1129,140,150,35);

        b5.addActionListener(this);

        l3.add(b5);

        b6 = new JButton("BALANCE ENQUIRY");

        b6.setForeground(Color.WHITE);

        b6.setBackground(Color.magenta);

        b6.setBounds(1260,85,150,35);

        b6.addActionListener(this);

        l3.add(b6);

        b7 = new JButton("EXIT");

        b7.setForeground(Color.WHITE);

        b7.setBackground(Color.RED);

        b7.setBounds(804,140,150,35);

        b7.addActionListener(this);

        l3.add(b7);

        b8 = new JButton("BACK");

        b8.setForeground(Color.WHITE);

        b8.setBackground(Color.BLUE);

        b8.setBounds(804,85,150,35);

        b8.addActionListener(this);

        l3.add(b8);

        b9 = new JButton("SIGN IN");

        b9.setFont(new Font("Arial",Font.BOLD,25));

        b9.setForeground(Color.WHITE);

        b9.setBackground(Color.BLUE);

        b9.setBounds(154,508,215,55);

        b9.addActionListener(this);

        l3.add(b9);

setLayout(null);

setVisible(true);

setBounds(0,0,1750,900);

    }

    @Override

    public void actionPerformed(ActionEvent e) {

        if (e.getSource()==b1){

            new Deposite(pinnum);

             setVisible(false);

         }else if (e.getSource()==b7){

             System.exit(0);

         }else if (e.getSource()==b2) {

             new Withdrawl(pinnum);

             setVisible(false);

         } else if (e.getSource()==b6) {

             new Bal(pinnum);

             setVisible(false);

         }else if (e.getSource()==b3) {

             new fastcash(pinnum);

             setVisible(false);

         } else if (e.getSource()==b4) {

             new mini(pinnum);

         } else if (e.getSource()==b5) {

            new Pin(pinnum);

            setVisible(false);

        }else{

            new Login(pinnum);

            setVisible(false);

        }

     }

    public static void main(String[] args) {

        new mainfile("");

    }

}

**Conn.java**

package bank.management.system;

import java.sql.\*;

public class conn {

        Connection conn;

        Statement statement;

        public conn(){

        String url = "jdbc:mysql://localhost:3366/bankmanagmentsystem";

        String user = "root";

        String password = "Avixxxx@xxxx";

        try{

            conn = DriverManager.getConnection(url,user,password);

           statement=conn.createStatement();

        } catch (Exception e) {

            e.printStackTrace();

        }

    }

}

**Signup.java**

package bank.management.system;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

import com.toedter.calendar.JDateChooser;

public class signup extends JFrame implements ActionListener{

    String marriedSta;

    JTextField first,mid,last,mob,emai,add,city1,state1;

    JButton sumbit,clear;

    JRadioButton  male,female,sin,dou;

    JDateChooser choosedob;

    JCheckBox box;

 signup(){

     ImageIcon img1 =new ImageIcon(ClassLoader.getSystemResource("icon/sgibank.png"));

      Image img2 =img1.getImage().getScaledInstance(100, 100,Image.SCALE\_DEFAULT);

      ImageIcon img3=new ImageIcon(img2);

      JLabel imgbank=new JLabel(img3);

      imgbank.setBounds(10,-10,100,100);

      add(imgbank);

      JLabel signupform=new JLabel("SignUP Application Form No 1");

      signupform.setBounds(240, 30, 500, 50);

      signupform.setForeground(Color.BLACK);

      signupform.setFont(new Font("",Font.BOLD,20));

      add(signupform);

     JLabel personalDeatails= new JLabel("Personal Details");

     personalDeatails.setBounds(290,100,250,50);

     personalDeatails.setFont(new Font("Arial",Font.TYPE1\_FONT,15));

     personalDeatails.setForeground(Color.black);

     add(personalDeatails);

     JLabel name= new JLabel("NAME ");

     name.setBounds(50, 150, 200,50);

     name.setFont(new Font("Arial",Font.BOLD,18));

     add(name);

    JLabel fname=new JLabel("First");

    fname.setBounds(50,175,50,50);

    fname.setFont(new Font("Italic",Font.ITALIC,12));

    add(fname);

     first=new JTextField(12);

     first.setBounds(40,210,120,20);

     first.setFont(new Font("Arial",Font.PLAIN,14));

     add(first);

     JLabel mname=new JLabel("Middle ");

     mname.setBounds(260,175,50,50);

     mname.setFont(new Font("Italic",Font.ITALIC,12));

     add(mname);

     mid=new JTextField();

     mid.setBounds(250,210,120,20);

     mid.setFont(new Font("Arial",Font.PLAIN,14));

     add(mid);

     JLabel lname=new JLabel("Last ");

     lname.setBounds(460,175,50,50);

     lname.setFont(new Font("Italic",Font.ITALIC,12));

     add(lname);

     last=new JTextField();

     last.setBounds(450,210,120,20);

     last.setFont(new Font("Arial",Font.PLAIN,14));

     add(last);

//Line//

    JLabel line= new JLabel("------------------------------------------------------------------------------------------");

     line.setBounds(0,221,1500,50);

     line.setFont(new Font("Italic",Font.ROMAN\_BASELINE,22));

     add(line);

     JLabel mobile=new JLabel("MOBILE NO");

     mobile.setBounds(50,245,200,50);

     mobile.setFont(new Font("Arial",Font.BOLD,17));

     add(mobile);

     mob=new JTextField();

     mob.setBounds(40,300,180,22);

     mob.setFont(new Font("Arial",Font.PLAIN,14));

     add(mob);

     JLabel email=new JLabel("EMAIL");

     email.setBounds(410,245,200,50);

     email.setFont(new Font("Arial",Font.BOLD,17));

     add(email);

     emai=new JTextField();

     emai.setBounds(390,300,180,22);

     emai.setFont(new Font("Arial",Font.PLAIN,14));

     add(emai);

     JLabel line1= new JLabel("------------------------------------------------------------------------------------------");

     line1.setBounds(0,321,1500,50);

     line1.setFont(new Font("Italic",Font.ROMAN\_BASELINE,22));

     add(line1);

     JLabel dob=new JLabel("DOB");

     dob.setBounds(50,345,200,50);

     dob.setFont(new Font("Arial",Font.BOLD,17));

     add(dob);

     choosedob=new JDateChooser();

     choosedob.setBounds(40,390,180,22);

     choosedob.setForeground(Color.red);

     add(choosedob);

     JLabel married=new JLabel("GENDER");

     married.setBounds(410,345,200,50);

     married.setFont(new Font("Arial",Font.BOLD,17));

     add(married);

     male= new JRadioButton("Male");

     male.setBounds(350,380,80,50);

     add(male);

     female= new JRadioButton("Female");

     female.setBounds(480,380,100,50);

     add(female);

     JLabel married1=new JLabel("MARRIED STATUS");

     married1.setBounds(228,418,200,50);

     married1.setFont(new Font("Arial",Font.BOLD,17));

     add(married1);

     sin= new JRadioButton("Married");

     sin.setBounds(190,455,80,50);

     add(sin);

     dou= new JRadioButton("Unmarried");

     dou.setBounds(305,455,100,50);

     add(dou);

     JLabel line2= new JLabel("------------------------------------------------------------------------------------------");

     line2.setBounds(0,479,1500,50);

     line2.setFont(new Font("Italic",Font.ROMAN\_BASELINE,22));

     add(line2);

     JLabel address=new JLabel("ADDRESS");

     address.setBounds(50,512,200,50);

     address.setFont(new Font("Arial",Font.BOLD,17));

     add(address);

     add= new JTextField();

     add.setBounds(40,555,580,31);

     add.setFont(new Font("Arial",Font.PLAIN,15));

     add(add);

     JLabel state=new JLabel("STATE");

     state.setBounds(50,590,200,50);

     state.setFont(new Font("Arial",Font.BOLD,17));

     add(state);

     state1=new JTextField();

     state1.setBounds(40,630,180,22);

     state1.setFont(new Font("Arial",Font.PLAIN,14));

     add(state1);

     JLabel city=new JLabel("CITY");

     city.setBounds(410,590,200,50);

     city.setFont(new Font("Arial",Font.BOLD,17));

     add(city);

     city1=new JTextField();

     city1.setBounds(390,630,180,22);

     city1.setFont(new Font("Arial",Font.PLAIN,14));

     add(city1);

     JLabel line3= new JLabel("------------------------------------------------------------------------------------------");

     line3.setBounds(0,655,1500,50);

     line3.setFont(new Font("Italic",Font.ROMAN\_BASELINE,22));

     add(line3);

    JLabel mandatory=new JLabel("\*");

    mandatory.setBounds(218,418,200,50);

    mandatory.setForeground(Color.RED);

    mandatory.setFont(new Font("Arial",Font.BOLD,15));

    add(mandatory);

    JLabel mandatory1=new JLabel("\*");

    mandatory1.setBounds(39,148,200,50);

    mandatory1.setForeground(Color.RED);

    mandatory1.setFont(new Font("Arial",Font.BOLD,15));

    add(mandatory1);

    JLabel mandatory2=new JLabel("\*");

    mandatory2.setBounds(39,505,200,50);

    mandatory2.setForeground(Color.RED);

    mandatory2.setFont(new Font("Arial",Font.BOLD,15));

    add(mandatory2);

    JLabel mandatory3=new JLabel("\*");

    mandatory3.setBounds(401,245,200,50);

    mandatory3.setForeground(Color.RED);

    mandatory3.setFont(new Font("Arial",Font.BOLD,15));

    add(mandatory3);

    JLabel mandatory4=new JLabel("\*");

    mandatory4.setBounds(39,345,200,50);

    mandatory4.setForeground(Color.RED);

    mandatory4.setFont(new Font("Arial",Font.BOLD,15));

    add(mandatory4);

    JLabel mandatory5=new JLabel("\*");

    mandatory5.setBounds(401,345,200,50);

    mandatory5.setForeground(Color.RED);

    mandatory5.setFont(new Font("Arial",Font.BOLD,15));

    add(mandatory5);

    JLabel mandatory6=new JLabel("\*");

    mandatory6.setBounds(41,590,200,50);

    mandatory6.setForeground(Color.RED);

    mandatory6.setFont(new Font("Arial",Font.BOLD,15));

    add(mandatory6);

    JLabel mandatory7=new JLabel("\*");

    mandatory7.setBounds(400,590,200,50);

    mandatory7.setForeground(Color.RED);

    mandatory7.setFont(new Font("Arial",Font.BOLD,15));

    add(mandatory7);

    JLabel mandatory8=new JLabel("\*");

    mandatory8.setBounds(41,250,200,50);

    mandatory8.setForeground(Color.RED);

    mandatory8.setFont(new Font("Arial",Font.BOLD,15));

    add(mandatory8);

    sumbit=new JButton("Next");

    sumbit.setBounds(100,750,80,30);

    sumbit.setBackground(Color.BLACK);

    sumbit.setForeground(Color.WHITE);

    sumbit.setFont(new Font("Arial",Font.BOLD,10));

    sumbit.addActionListener(this);

    add(sumbit);

    clear=new JButton("Clear");

    clear.setBounds(460,750,80,30);

    clear.setBackground(Color.BLACK);

    clear.setForeground(Color.WHITE);

    clear.setFont(new Font("Arial",Font.BOLD,10));

    clear.addActionListener(this);

    add(clear);

     setLayout(null);

      setBounds(850,0,650,850);

      setVisible(true);

      setTitle("Signup Page");

 }

    public void actionPerformed(ActionEvent ea){

        if (ea.getSource() == clear) {

            first.setText("");

            mid.setText("");

            last.setText("");

            add.setText("");

            emai.setText("");

            city1.setText("");

            state1.setText("");

            mob.setText("");

        }

        String fname=first.getText();

        String mname=mid.getText();

        String lname=last.getText();

        String mobile=mob.getText();

        String email=emai.getText();

        String dob=((JTextField) choosedob.getDateEditor().getUiComponent()).getText();

        String gender=null;

        if (male.isSelected()) {

            gender="Male";

        } else {

            gender="Female";

        }

        marriedSta=null;

        if (sin.isSelected()) {

            marriedSta="Married";

        }

        else{

         marriedSta="Unmarried";

        }

        String address=add.getText();

        String state=state1.getText();

        String city=city1.getText();

        try {

            if (fname.equals("")) {

                JOptionPane.showMessageDialog(null,"Please Enter First Name.");

            }  else if (mname.equals("")) {

                JOptionPane.showMessageDialog(null,"Please Enter Middle Name.");

            }

            else if (lname.equals("")) {

                JOptionPane.showMessageDialog(null,"Please Enter Last Name.");

            }

            else if (mobile.equals("")) {

                JOptionPane.showMessageDialog(null,"Please Enter Mobile N0.");

            }

            else if (email.equals("")) {

                JOptionPane.showMessageDialog(null,"Please Enter Your Email.");

            }

            else if (dob.equals("")) {

                JOptionPane.showMessageDialog(null,"Please Enter Your DOB.");

            }

            else if (gender.equals("")) {

                JOptionPane.showMessageDialog(null,"Please Select Your Gender.");

            }

            else if (address.equals("")) {

                JOptionPane.showMessageDialog(null,"Please Enter Your Address.");

            }

            else if (city.equals("")) {

                JOptionPane.showMessageDialog(null,"Please Enter Your City.");

            }

            else if (state.equals("")) {

                JOptionPane.showMessageDialog(null,"Please Enter Your State.");

            }

            else if (marriedSta.equals("")) {

                JOptionPane.showMessageDialog(null,"Please Select Your Status.");

            }

            else{

                conn con=new conn();

                String qeury= "INSERT INTO signup values('"+fname+"', '"+mname+"', '"+lname+"','"+mobile+"', '"+email+"', '"+dob+"','"+gender+"','"+address+"','"+state+"','"+city+"','"+marriedSta+"')";

                con.statement.executeUpdate(qeury);

               new signup1();

               setVisible(false);

            }

        } catch(Exception E){

            E.printStackTrace();

        }

        }

  public static void main(String []args){

      new signup();

}

}

**Diposite.java**

package bank.management.system;

import java.awt.\*;

import java.awt.event.\*;

import java.util.\*;

import javax.swing.\*;

public class Deposite extends JFrame implements ActionListener{

    static String pinnum;

    TextField t1;

    JButton b1,b2;

    Deposite(String pinnum){

        Deposite.pinnum=pinnum;

      ImageIcon i1=new ImageIcon(ClassLoader.getSystemResource("icon/bg.jpg"));

      Image i2=i1.getImage().getScaledInstance(1500, 835,Image.SCALE\_DEFAULT);

      ImageIcon i3=new ImageIcon(i2);

      JLabel l3=new JLabel(i3);

      l3.setBounds(0,0,1500,835);

      add(l3);

      JLabel l1=new JLabel("ENTER AMOUNT YOU WANT TO DIPOSITE.");

      l1.setBounds(560,180,700,35);

      l1.setFont(new Font("Arial",Font.BOLD,23));

      l1.setForeground(Color.BLACK);

      l3.add(l1);

      t1 = new TextField();

      t1.setBackground(Color.WHITE);

      t1.setForeground(Color.BLACK);

      t1.setBounds(642,290,360,25);

      t1.setFont(new Font("Raleway", Font.BOLD,22));

      l3.add(t1);

      b1 = new JButton("DEPOSIT");

      b1.setBounds(900,450,150,35);

      b1.setBackground(Color.BLUE);

      b1.setForeground(Color.WHITE);

      b1.addActionListener(this);

      l3.add(b1);

      b2 = new JButton("BACK");

      b2.setBounds(600,450,150,35);

      b2.setBackground(Color.RED);

      b2.setForeground(Color.WHITE);

      b2.addActionListener(this);

      l3.add(b2);

       setLayout(null);

       setBounds(0,0,1750,900);

       setVisible(true);

       setTitle("Deposite Form");

    }

    public void actionPerformed(ActionEvent e) {

        try {

            String amount = t1.getText();

            Date date = new Date();

            if (e.getSource()==b1){

                if (t1.getText().equals("")){

                    JOptionPane.showMessageDialog(null,"Please enter the Amount you want to Deposit");

                }else {

                    conn c1 = new conn();

                    String qu="INSERT INTO bankd values('"+pinnum+"', '"+date+"','Deposit', '"+amount+"')";

                    c1.statement.executeUpdate(qu);

                    JOptionPane.showMessageDialog(null,"Rs. "+amount+" Deposited Successfully");

                    setVisible(false);

                   new mainfile(pinnum);

                }

            }else if (e.getSource()==b2){

                setVisible(false);

                new mainfile(pinnum);

            }

        }catch (Exception E){

            E.printStackTrace();

        }

    }

    public static void main(String[] args) {

        new Deposite(pinnum);

    }

}

**SNAPSHOTS**

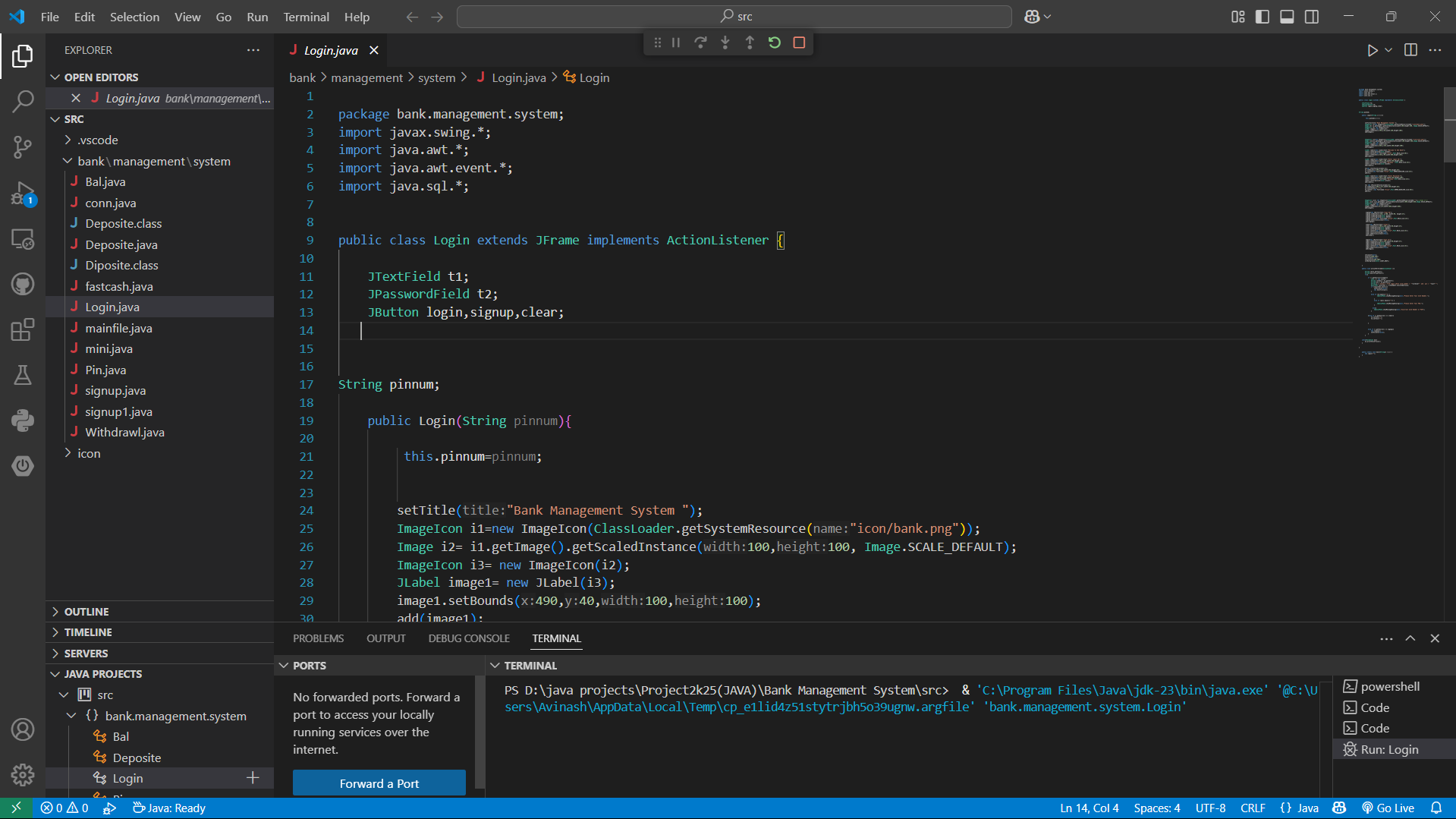


Fig 1 : LOGIN Code Page

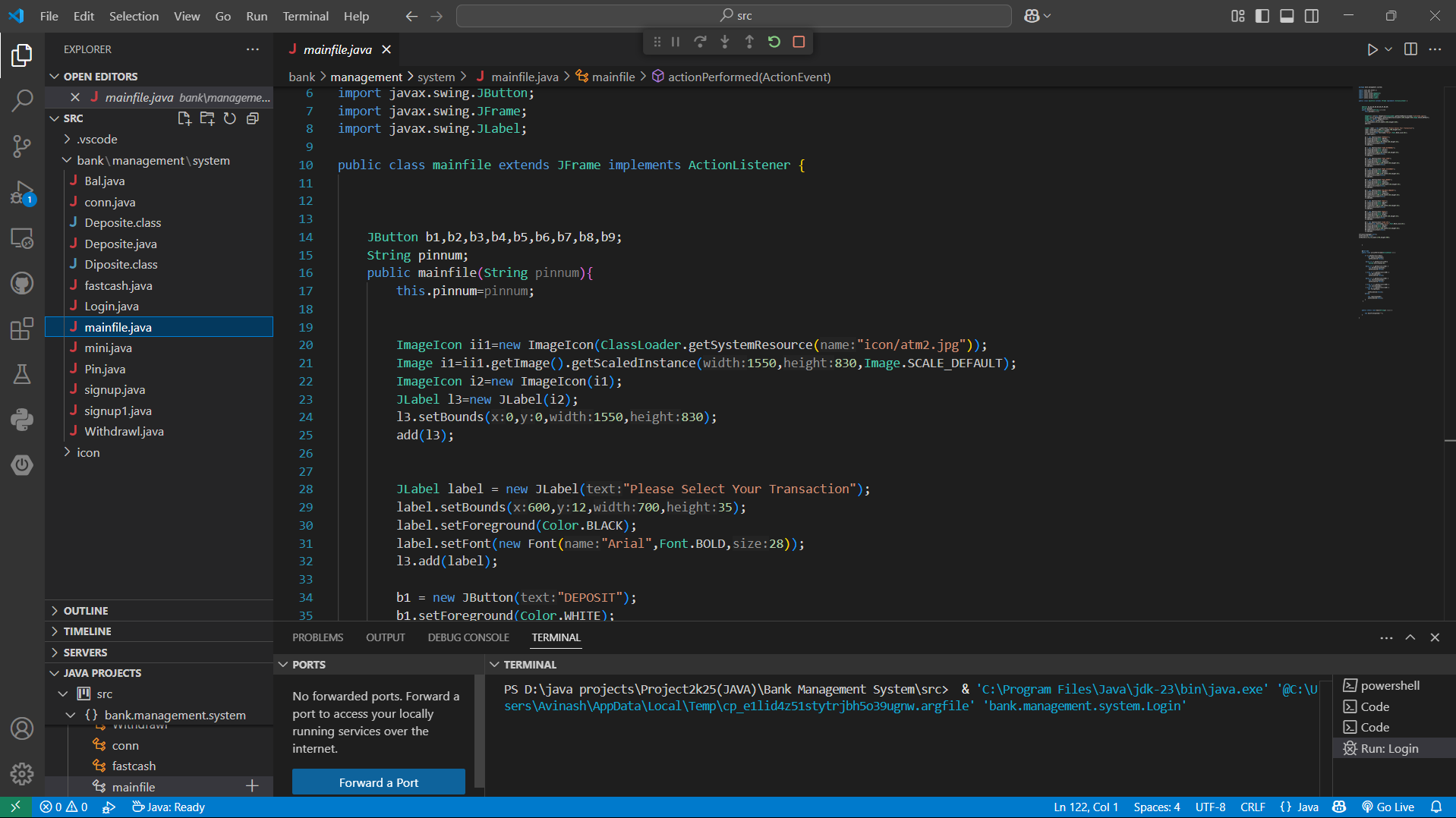


Fig 2 : Main Window. Cash Diposit,Withdrawal, Statement Page etc

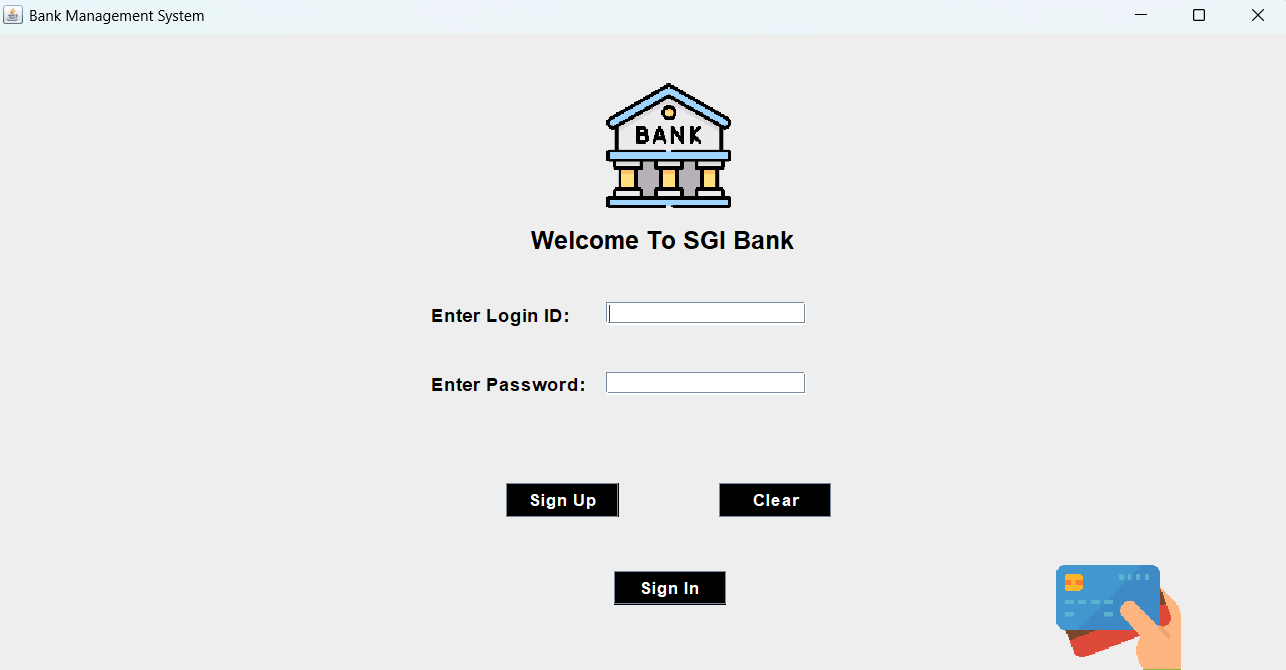


Fig 3 Login Page Interface :

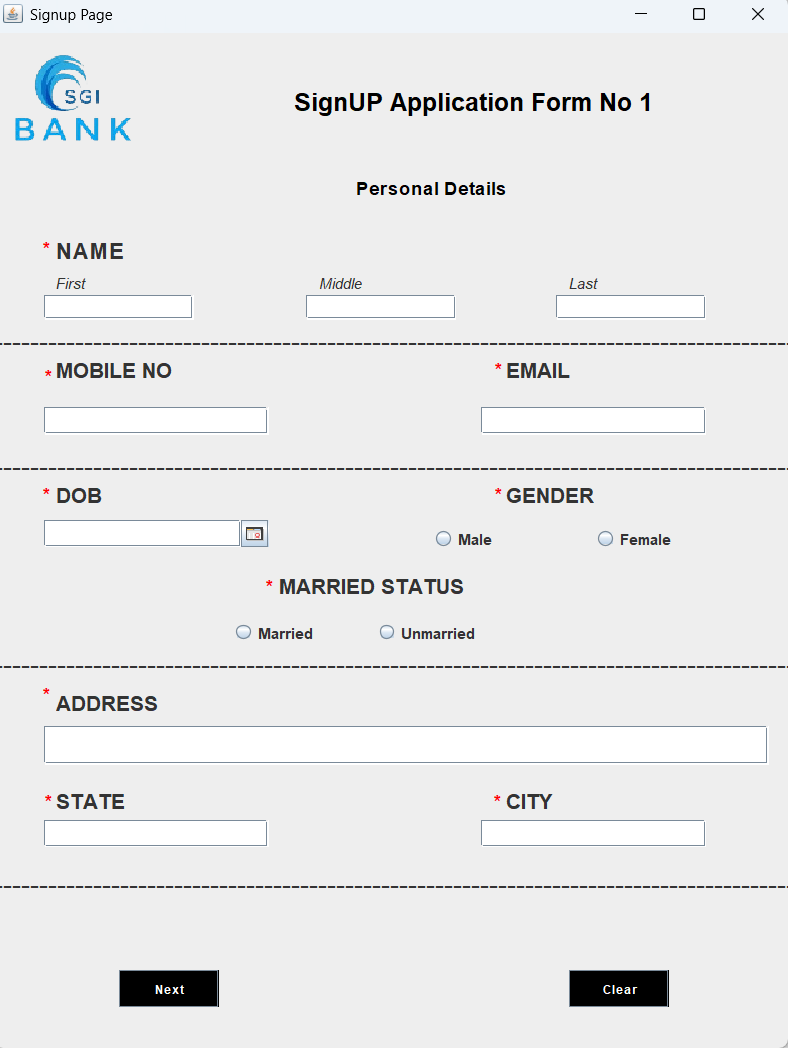


Fig 4 : Signup Form Page

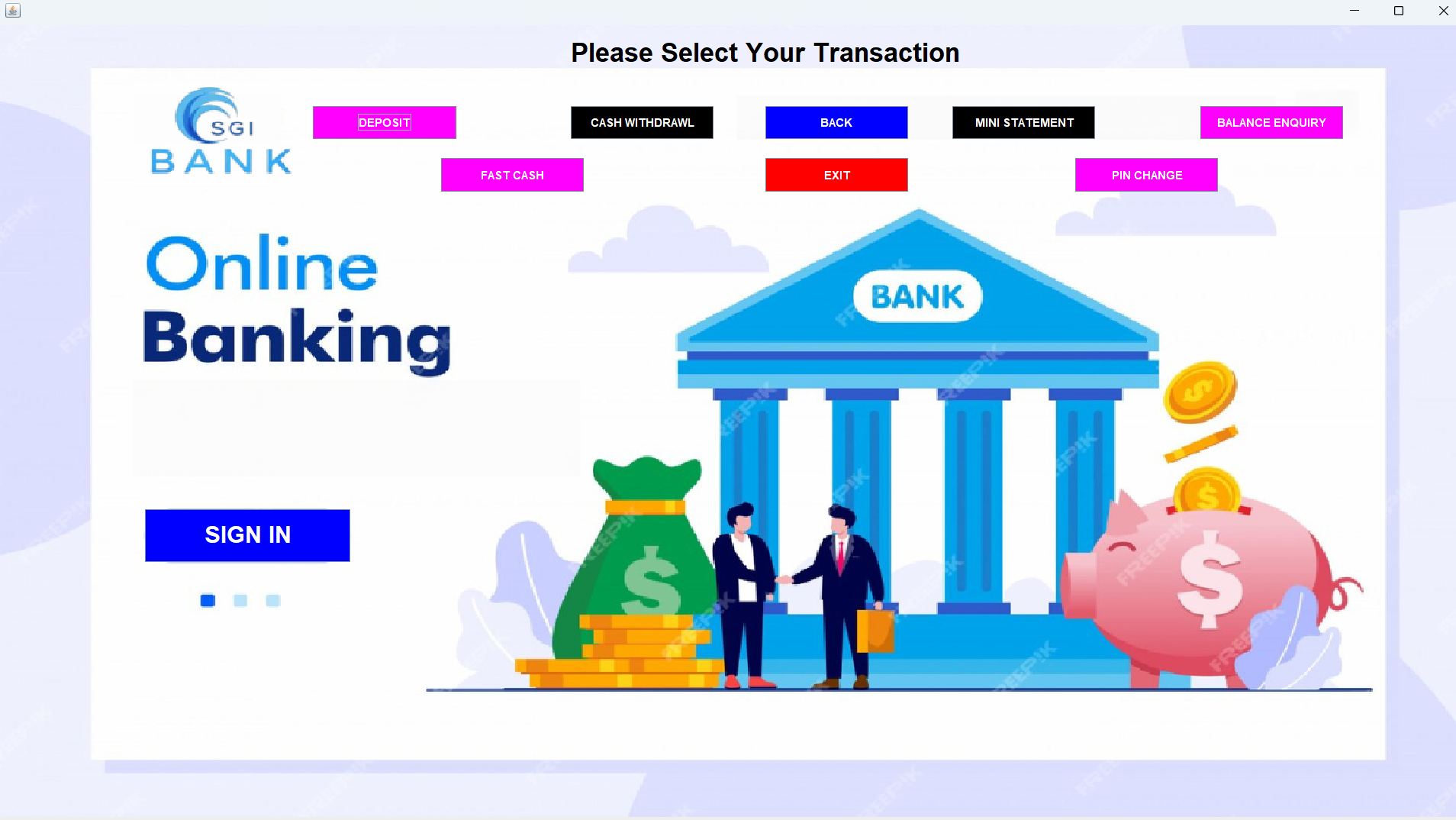


Fig 5 : Main Page(Sign In)

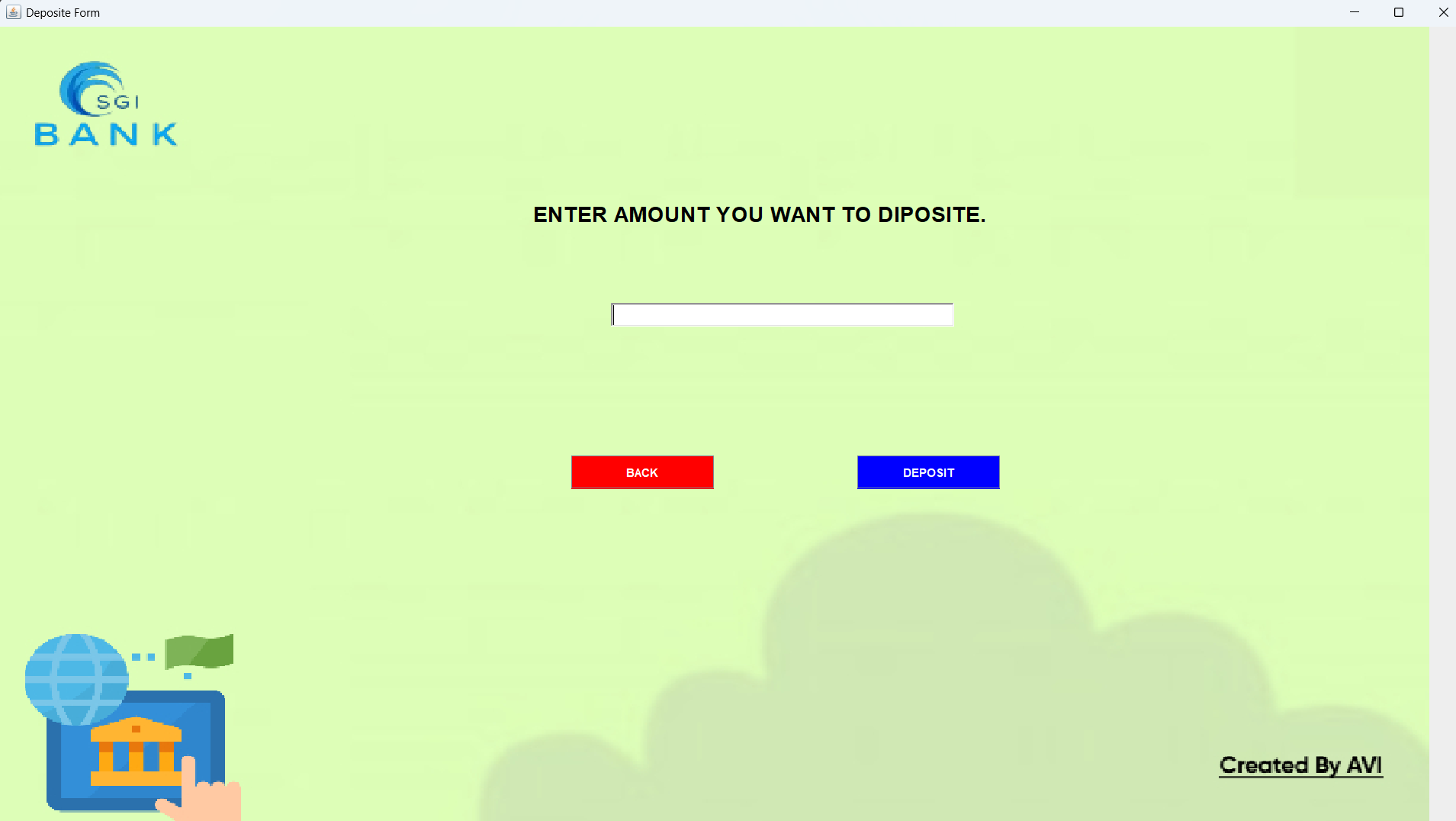


Fig 6 : Diposite Page



Fig 7 : Statement page

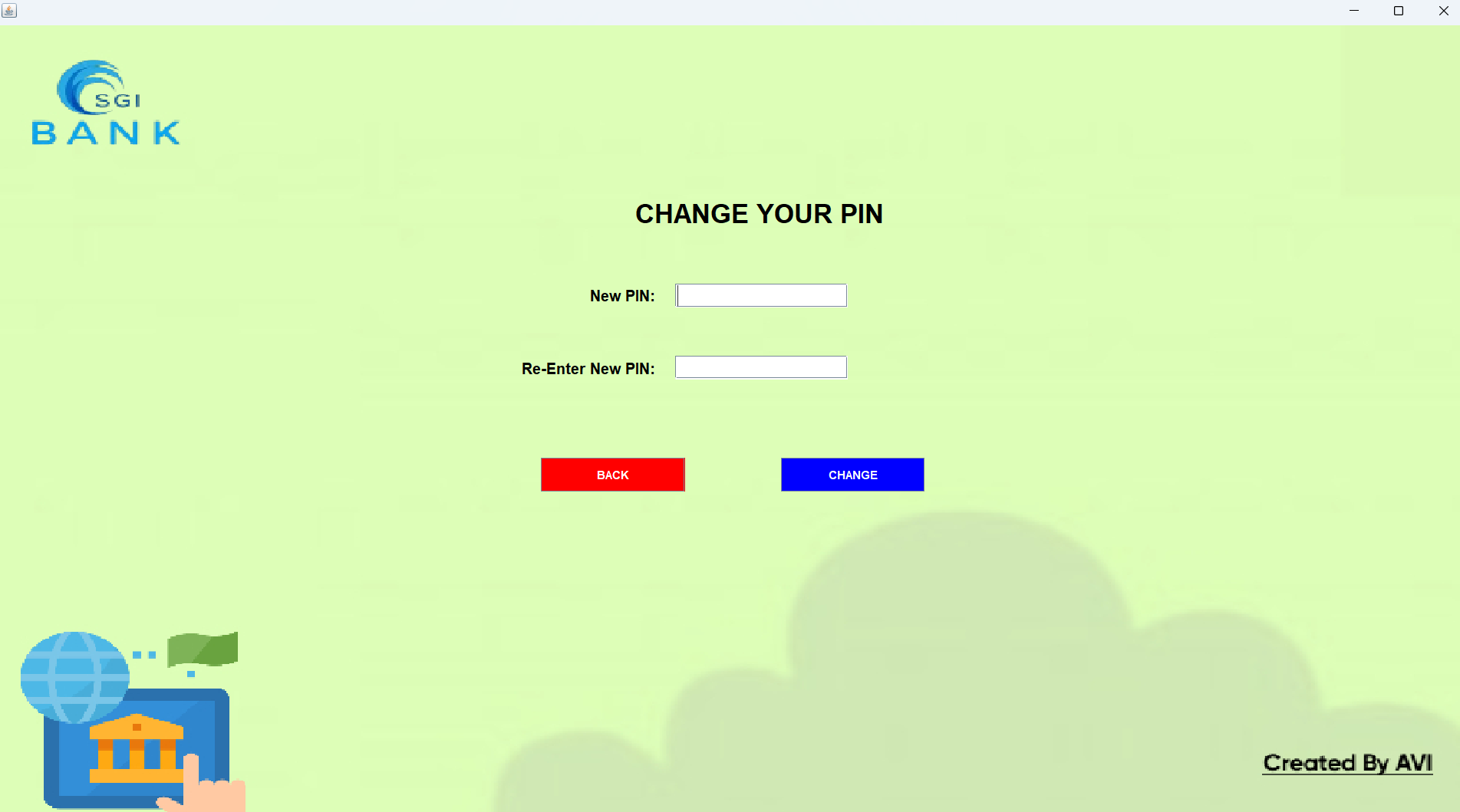
****

Fig 7 : Pin Change Page

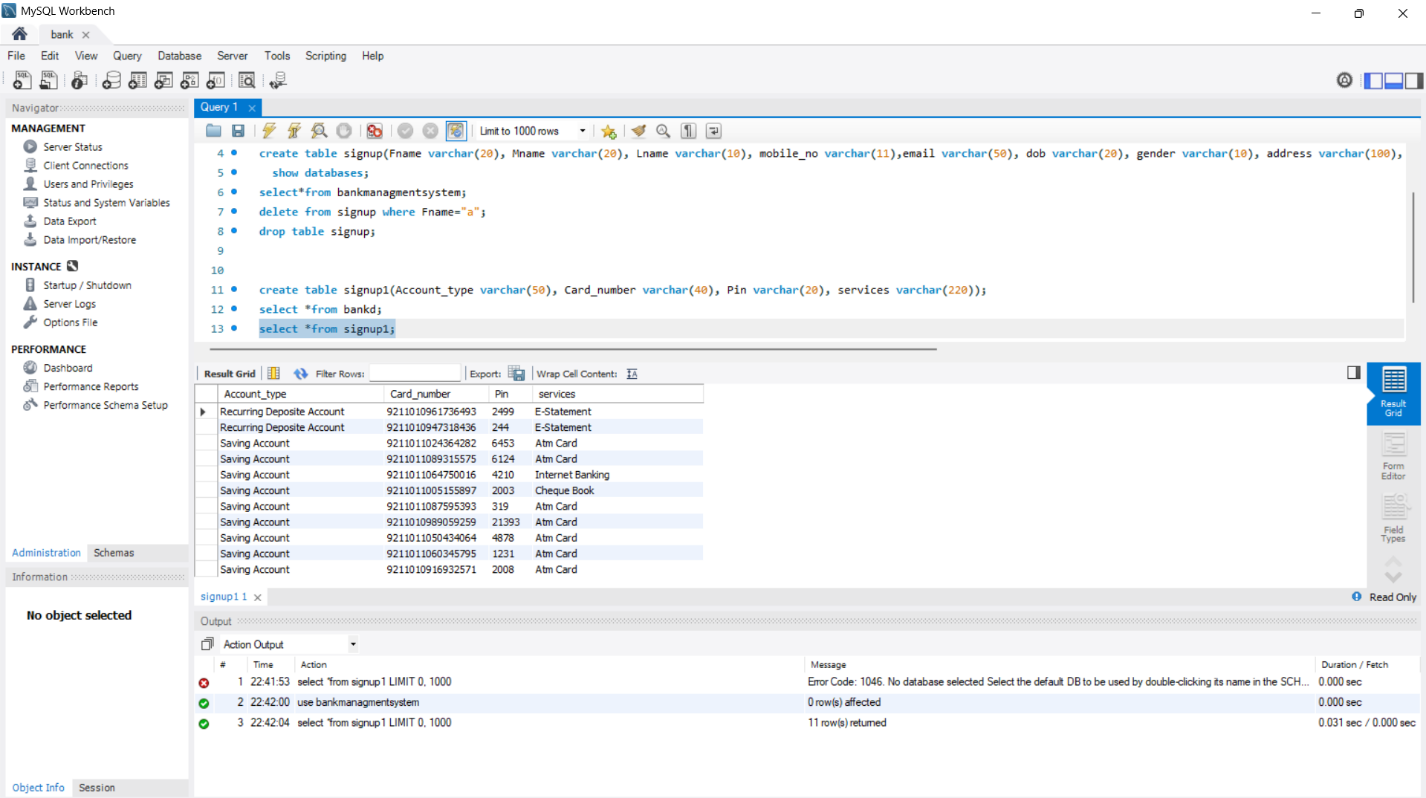


Fig 8 : Database

# CONCLUSION AND FUTURE WORK

## 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 Work :**

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. Code for ------ YouTube Channel

2. Online Bank Account Management System

Website: <https://www.slideshare.net> (Collect some info for report documents)

3. Learning MYSQL

Website: <https://www.w3schools.com/mysql/default.asp>

# REFERENCES

1. <https://www.researchgate.net/publication/328346114_Mobile_Tracking_System_using_Web_Application_and_Android_Apps>
2. <https://www.researchgate.net/publication/299998138_A_Research_on_Mobile_Applications_for_Location_Tracking_through_Web_Server_and_Short_Messages_Services_SMS>
3. <http://www.ijergs.org/files/documents/MOBILE-30.pdf>
4. <https://www.researchgate.net/publication/304129283_GPS_Based_Vehicle_Tracking_System_and_Using_Analytics_to_Improve_The_Performance>
5. <https://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=979B62ED6E07E451764781C030E918F3?doi=10.1.1.258.7662&rep=rep1&type=pdf>