Government Polytechnic, Pune-21

(An Autonomous Institute of Government of Maharashtra)

Project Report On

Gui-Based Banking System

SUBMITTED BY

Name	Enrollment No.	Division
1. Deshmukh Akash Abhay	1906021	G3
2. Deshmukh Sagar Abhay	1906022	G3
2. Ingale Ankush Dinesh	1906038	G3

Under the Guidance of

Mrs. Saraswati Panchakshari Mam

ACKNOWLEDGEMENT

The success of any project depends largely on the encouragement and guidelines of many others. This research project would not have been possible without their support. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project.

First and foremost, we wish to record our sincere gratitude to the Management of this college and our Respected Principal Mr.V.S. Bandal, for his constant support and encouragement in the preparation of this report and for the availability of library facilities needed to prepare this report.

Our sincere thanks to Dr.S.B. Nikam, Head, Department of Computer, Govt. Polytechnic, Pune was abundantly helpful and offered invaluable assistance, support and guidance.

We express our sincere gratitude to our Guide, **Saraswati Panchakshari mam** for guiding us in the investigations of this project and in carrying out experimental work. Our numerous discussions were extremely helpful. We are highly indebted to him for his guidance and constant supervision as well as for providing necessary information regarding the project & also for his support in completing the project. We hold him in esteem for the guidance, encouragement, and inspiration received from him.

We wish to thank our parents for financing our studies and helping us throughout our life for achieving perfection and excellence. Their help in making this report and project worth presentation is gratefully acknowledged. Last but not least we thank the Almighty for the continued strength we were bestowed for the completion of this project.

Content

Abstract

Java is a high-level, class-based, object-oriented programming language that is designed to have as few implementation dependencies as possible. It is a general-purpose programming language intended to let programmers write once, and run anywhere (WORA), meaning that compiled Java code can run on all platforms that support Java without the need for recompilation. Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of the underlying computer architecture. The syntax of Java is similar to C and C++ but has fewer low-level facilities than either of them. The Java runtime provides dynamic capabilities (such as reflection and runtime code modification) that are typically not available in traditional compiled languages. As of 2019, Java was one of the most popular programming languages in use according to GitHub, particularly for client-server web applications, with a reported 9 million developers.

Our project title is **Gui Based Banking System**. It is developed using all java concepts and mostly focused on **Swing components** and some advanced java concepts like collection, and JDBC. It is a GUI application that has user-friendly interface and stores data in the database. Along with banking operations, our application provides a facility to recharge a mobile phone. We have used oracle database and SQL as query language.

Our GUI application has customer sections that are to be used by all users:

- Customer Section

❖ **Customer Section**: The customer first needs to register to our banking application before doing any transaction. After registering he or she can sign in using the password registered at the time of registration. After Login, the customer can perform a transaction, and the customer can also recharge his or her phone by entering a number.

  - **Profile**: Using this functionality Customers can see their profile details added at the time of registration.
  - **Change Password**: The customer can change the login password by entering an old password for security purposes.

- **Recharge**: Using this functionality, the Customer can recharge his or her phone by entering mobile phone and service provider.
- **Deposit**: user can enter the amount that he wants to deposit
- **Withdrawl**: customer can withdraw an amount from his bank account.
- **Transaction History:** The customer can see previous transactions done by him.
- **Logout:** The customer can logout from his account

Platform and/Hardware Specifications: -

Hardware Specifications:

- Processor: AMD Ryzen 3 3250U
- Speed: 1.1GHz
- Ram: 8GB
- Key Board: Standard windows keyboard
- System type: 64-bit Operating system

Eclipse IDE:

Eclipse is an integrated development environment (IDE) used in computer programming. It contains a base workspace and an extensible plug-in system for customizing the environment. Eclipse is written mostly in Java and its primary use is for developing Java applications, but it may also be used to develop applications in other programming languages via plug-ins, including Ada, ABAP, C, C++, C#, Clojure, COBOL, D, Erlang, Fortran, Groovy, Haskell, JavaScript, Julia, Lasso, Lua, NATURAL, Perl, PHP, Prolog, Python, R, Ruby (including Ruby on Rails framework), Rust, Scala, and Scheme. It can also be used to develop documents with LaTeX (via a TeXlipse plug-in) and packages for the software Mathematica. Development environments include the Eclipse Java development tools (JDT) for Java and Scala, Eclipse CDT for C/C++, and Eclipse PDT for PHP, among others.

The initial codebase originated from IBM VisualAge. The Eclipse software development kit (SDK), which includes the Java development tools, is meant for Java developers. Users can extend their abilities by installing plug-ins written for the Eclipse Platform, such as development toolkits for other programming languages, and can write and contribute their plug-in modules. Since the introduction of the OSGi implementation (Equinox) in version 3 of Eclipse, plug-ins can be plugged-stopped dynamically and are termed (OSGI) bundles.

Eclipse software development kit (SDK) is free and open-source software, released under the terms of the Eclipse Public License, although it is incompatible with the GNU General Public License. It was one of the first IDEs to run under GNU Classpath and it runs without problems under IcedTea.

Oracle SQL:

Structured Query Language (SQL) is the set of statements with which all programs and users access data in an Oracle database. Application programs and Oracle tools often allow users access to the database without using SQL directly, but these applications in turn must use SQL when executing the user's request. This chapter provides background information on SQL as used by most database systems.

Oracle SQL Developer is a free, integrated development environment that simplifies the development and management of Oracle Database in both traditional and Cloud deployments. SQL Developer offers complete end-to-end development of your PL/SQL applications, a worksheet for running queries and scripts, a DBA console for managing the database, a reports interface, a complete data modeling solution, and a migration platform for moving your 3rd party databases to Oracle.

Concepts Used

1. **Swing:** Swing is a GUI widget toolkit for Java. It is part of Oracle's Java Foundation Classes – an API for providing a graphical user interface for Java programs
2. **Exception Handling:**

   Exception Handling in Java is one of the powerful mechanisms to handle the runtime errors so that the normal flow of the application can be maintained.
3. **Loops:**

   Looping in programming languages is a feature that facilitates the execution of a set of instructions/functions repeatedly while some condition evaluates to true.
4. **Conditional Statements:**
   - Use if to specify a block of code to be executed, if a specified condition is true.
   - Use else to specify a block of code to be executed if the same condition is false.
   - Use else if to specify a new condition to test, if the first condition is false.
5. **Getters and Setters:**

   Getters and Setters play an important role in retrieving and updating the value of a variable outside the encapsulating class. A setter updates the value of a variable, while a getter reads the value of a variable.
6. **Scanner class:**

   A scanner is a class in java. util package used for obtaining the input of the primitive types like int, double, etc., and strings. ... To create an object of Scanner class, we usually pass the predefined object System. in, which represents the standard input stream.

7. Collection:

The Collection in Java is a framework that provides an architecture to store and manipulate a group of objects.

- **LinkedList:**

  LinkedList implements the Collection interface. It uses a doubly linked list internally to store the elements. It can store duplicate elements. It maintains the insertion order and is not synchronized. In LinkedList, the manipulation is fast because no shifting is required.

8. Java Database Connectivity (JDBC):

Java Database Connectivity is an application programming interface for the programming language Java, which defines how a client may access a database. It is a Java-based data access technology used for Java database connectivity. It is part of the Java Standard Edition platform, from Oracle Corporation.

Coding of the project:

❖ Methods file

```
package com.dao;




import java.beans.Statement;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Timestamp;

import java.util.Date;

import java.util.LinkedList;

import java.util.List;

import java.util.Random;


import javax.swing.JLabel;

import javax.swing.JOptionPane;





public class RegisterDao {



	Connection con=null;

	int i=0;

	public int createUser(int s10,String s1, String s2, String s8, String s9, 
String s5, String s6, String s7,int s11) {

		try {
			con=dbConnection.getConnection();

			PreparedStatement ps = con.prepareStatement("insert into reg values(?,?,?,?,?,?,?,?)");

			ps.setInt(1, s10);

			ps.setString(2, s1);

			ps.setString(3, s2);

			ps.setString(4, s8);

			ps.setString(5, s5);

			ps.setString(6, s6);

			ps.setString(7, s7);

			ps.setInt(8, s11);

			i =ps.executeUpdate();

		} catch (SQLException e) {

			// TODO Auto-generated catch block

			e.printStackTrace();

		}

		return i;


	}

	public ResultSet validateUser(String str1,String str2) {

		ResultSet rs=null;

		con=dbConnection.getConnection();

		try {

			PreparedStatement ps = con.prepareStatement("select * from reg where email=? and password=?");

			ps.setString(1, str1);

			ps.setString(2, str2);

			rs = ps.executeQuery();

		} catch (SQLException e) {

			// TODO Auto-generated catch block

			e.printStackTrace();
			}
			return rs;
		}
		public ResultSet profile(int regno)
		{
			ResultSet rs=null;
			con=dbConnection.getConnection();
			try {
				PreparedStatement ps=con.prepareStatement("select *from reg 
where regid=?");
				ps.setInt(1, regno);
				rs=ps.executeQuery();
			} catch (SQLException e) {
				// TODO Auto-generated catch block
				e.printStackTrace();
			}
			return rs;
		}
		public int updatePassword(String s,int regno)
		{
			con=dbConnection.getConnection();
			try {
				PreparedStatement ps=con.prepareStatement("update reg set 
password=? where regid=?");
				ps.setString(1, s);
				ps.setInt(2, regno);
				i=ps.executeUpdate();
			} catch (SQLException e) {
				// TODO Auto-generated catch block
				e.printStackTrace();
			}
		return i;
	}
	public int updateBal(int regno,int amt)
	{
		PreparedStatement ps=null;
		ResultSet rs=null;
		con=dbConnection.getConnection();
		try {
			rs=profile(regno);
			if(rs.next())
			{
				ps=con.prepareStatement("update reg set balance=? where
regid=?");
				ps.setInt(1, rs.getInt(8)-amt);
				ps.setInt(2, regno);
				i=ps.executeUpdate();
			}
		} catch (SQLException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		}
		return i;
	}
	public int depositAmt(int regno,int amt)
	{
		PreparedStatement ps=null;
		ResultSet rs=null;
		con=dbConnection.getConnection();
		try {
			rs=profile(regno);
```

```
                if(rs.next())
                {
                        ps=con.prepareStatement("update reg set balance=? where
regid=?");
                        ps.setInt(1, rs.getInt(8)+amt);
                        ps.setInt(2, regno);
                        i=ps.executeUpdate();
                }
        } catch (SQLException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
        }
        return i;
}
public ResultSet validateID()
{
        ResultSet rs=null;
        java.sql.Statement st=null;
        con=dbConnection.getConnection();
        String str="select *from reg";
        try {
                st= con.createStatement();
                rs=st.executeQuery(str);
        } catch (SQLException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
        }
        return rs;
}
public int transactionHistory(int regno,int amt,String type)
```

```
	{
		i=0;
		con=dbConnection.getConnection();
		Timestamp date=new Timestamp(new Date().getTime());
		Random rd=new Random();
		//String type="Recharge";
		int tr_id=rd.nextInt(200000000);
		try {
			PreparedStatement ps=con.prepareStatement("insert into 
transaction values(?,?,?,?,?)");
			ps.setInt(1, tr_id);
			ps.setInt(2, amt);
			ps.setString(3, type);
			ps.setTimestamp(4,date);
			ps.setInt(5, regno);
			i=ps.executeUpdate();
		} catch (SQLException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		}
		return i;
	}

	public ResultSet retrive_transaction(int regno)
	{
		ResultSet rs=null;
		con=dbConnection.getConnection();
		try {
			PreparedStatement ps=con.prepareStatement("select *from 
transaction where regid=?");
			ps.setInt(1, regno);
```

```
                    rs=ps.executeQuery();

              } catch (SQLException e) {

                    // TODO Auto-generated catch block

                    e.printStackTrace();

              }

              return rs;

        }


}
```

❖ Dashboard file

```
package com.gui;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.image.BufferedImage;
import java.io.File;
import java.io.IOException;
import java.sql.ResultSet;
import java.sql.SQLException;

import javax.imageio.ImageIO;
import javax.swing.*;
import javax.swing.border.Border;

import com.dao.RegisterDao;
public class DashBoard extends JFrame implements ActionListener{

    JPanel pnl1,pnl2,pnl3,pnl4,pnl5;
    JButton btn1,btn2,btn3,btn4,btn5,btn6,btn7,btn8 ,btn9,btn10,btn11,btn12;
    JLabel lb1,lb2,lb3;
    int regno;
    public DashBoard(int regno) {
            super("Dashboard..");
            setSize(970,800);
            setVisible(true);
            this.regno=regno;

            BorderLayout bl=new BorderLayout();
            setLayout(bl);

            GridLayout gl = new GridLayout(7,0,25,25);
```

```
Font f=new Font("Monospace", Font.BOLD, 24);


pnl1=new JPanel();
pnl2=new JPanel();
pnl3=new JPanel();
pnl4=new JPanel();
pnl5=new JPanel();

pnl1.setBackground(Color.cyan);
pnl2.setBackground(Color.gray);
pnl3.setBackground(Color.cyan);
pnl4.setBackground(Color.ORANGE);
pnl5.setBackground(Color.yellow);

//pnl2.setLayout(gl);
lb1=new JLabel("Reserve Bank Of India");
lb1.setFont(new Font("Monospace", Font.BOLD, 40));
add(pnl5,bl.CENTER);
RegisterDao rdao=new RegisterDao();
ResultSet rs=rdao.profile(regno);
try {
        if(rs.next())
        {
                lb3=new JLabel("Account Holder Name :"+rs.getString(2));
                lb2=new JLabel("Account Balance :"+rs.getInt(8));
        }
} catch (SQLException e1) {
        // TODO Auto-generated catch block
        e1.printStackTrace();
}
pnl5.add(lb3);
lb3.setFont(f);
pnl5.add(lb2);
lb2.setFont(f);

add(pnl2,bl.NORTH);
BufferedImage image1=null;
/*try {
        image = ImageIO.read(new File("./bank3.png"));
} catch (IOException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
}*/
ImageIcon icon = new ImageIcon("./bank3.png");
JLabel label1 = new JLabel(icon,JLabel.LEFT);
label1.setBounds(10, 10, 10, 10);
pnl2.add(label1);
pnl2.add(lb1);


//pnl1.setFont(f);
```

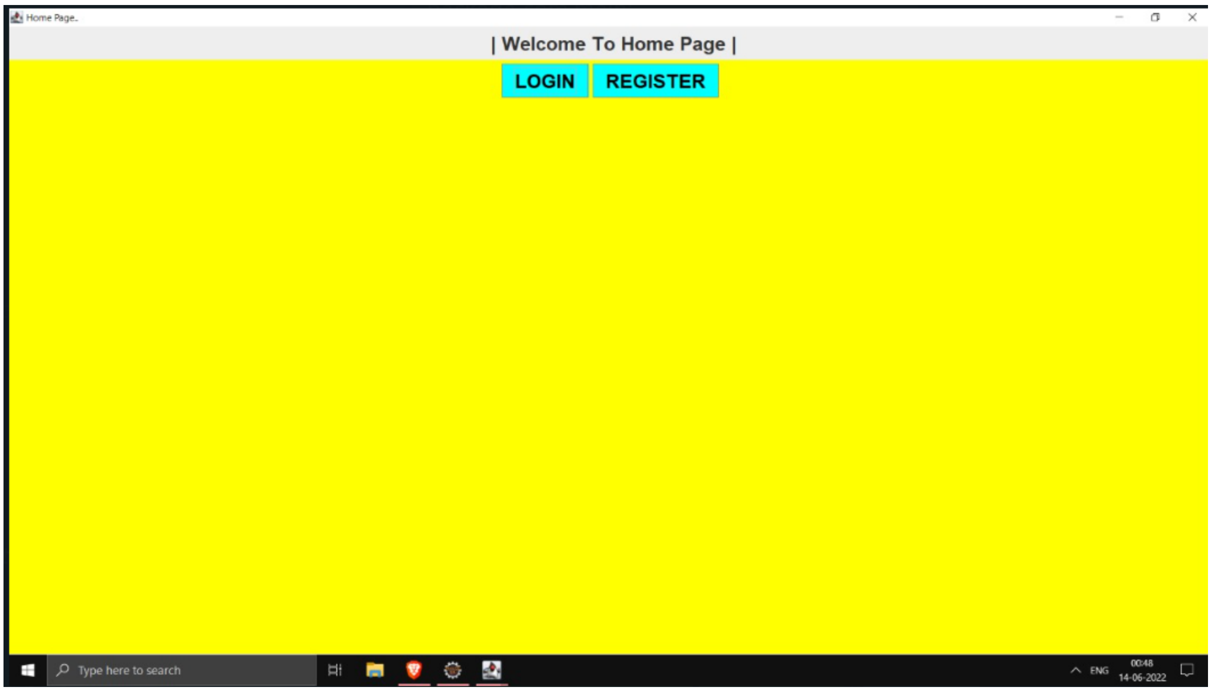
```
btn2=new JButton("Profile");
btn2.setPreferredSize(new Dimension(250, 250));
btn2.setFont(f);

btn3=new JButton("Change Password");
btn3.setPreferredSize(new Dimension(250, 250));
btn3.setFont(f);

btn4=new JButton("Recharge");
btn4.setPreferredSize(new Dimension(250, 250));
btn4.setFont(f);

btn5=new JButton("Deposit");
btn5.setPreferredSize(new Dimension(250, 250));
btn5.setFont(f);

btn6= new JButton("Withdrawl");
btn6.setPreferredSize(new Dimension(250, 250));
btn6.setFont(f);

btn9= new JButton("Transaction History");
btn9.setPreferredSize(new Dimension(250, 250));
btn9.setFont(f);

btn1=new JButton("Logout");
btn1.setPreferredSize(new Dimension(250, 250));
btn1.setFont(f);

GridLayout gl1=new GridLayout(2,0,20,20);
add(pnl1,bl.WEST);
pnl1.setLayout(gl);
pnl1.add(btn2);//profile
pnl1.add(btn3);//update password
pnl1.add(btn4);//recharge
pnl1.add(btn5);//deposit
pnl1.add(btn6);//withdrawl
pnl1.add(btn9);//history
pnl1.add(btn1);//log out

add(pnl3,bl.EAST);
pnl3.setLayout(gl1);
BufferedImage image2=null;
BufferedImage image3=null;
try {
        image1 = ImageIO.read(new File("./bank1.png"));
        image2=ImageIO.read(new File("./bank2.jpg"));
} catch (IOException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
}
JLabel label2 = new JLabel(new ImageIcon(image1));
JLabel label3 = new JLabel(new ImageIcon(image2));
```

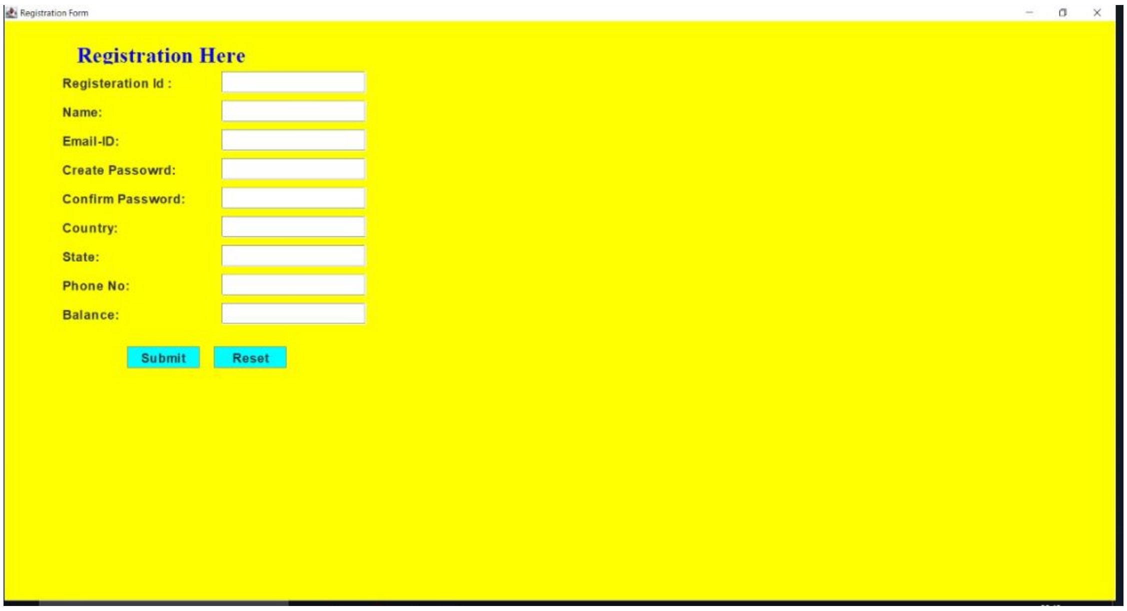
```
                pnl3.add(label2);
                pnl3.add(label3);

                btn2.addActionListener(this);
                btn3.addActionListener(this);
                btn4.addActionListener(this);
                btn5.addActionListener(this);
                btn6.addActionListener(this);
                btn9.addActionListener(this);
                btn1.addActionListener(this);
        }
        @Override
        public void actionPerformed(ActionEvent e) {
                if(e.getSource()==btn2)
                {
                        Profile p=new Profile(regno);
                        this.dispose();

                }
                if(e.getSource()==btn3)
                {
                        changePassword c=new changePassword(regno);
                        this.dispose();
                }
                if(e.getSource()==btn4)
                {
                        recharge r=new recharge(regno);
                        this.dispose();
                }
                if(e.getSource()==btn5)
                {
                        deposit dp=new deposit(regno);
                        this.dispose();
                }
                if(e.getSource()==btn6)
                {
                        withdrawl w=new withdrawl(regno);
                        this.dispose();
                }
                if(e.getSource()==btn9)
                {
                        history h=new history(regno);
                        this.dispose();
                }
                if(e.getSource()==btn1)
                {
                      final JLabel label = new JLabel();
                      int result = JOptionPane.showConfirmDialog(pnl5,"Sure? You want to Log out?", "Log
Out", JOptionPane.YES_NO_OPTION, JOptionPane.QUESTION_MESSAGE);
                          if(result == JOptionPane.YES_OPTION)
                          {
                            Home h=new Home();
                            this.dispose();
                          }
                          else if (result == JOptionPane.NO_OPTION)
                          {
                            DashBoard d=new DashBoard(regno);
                            this.dispose();
                          }
                }

        }


}
```

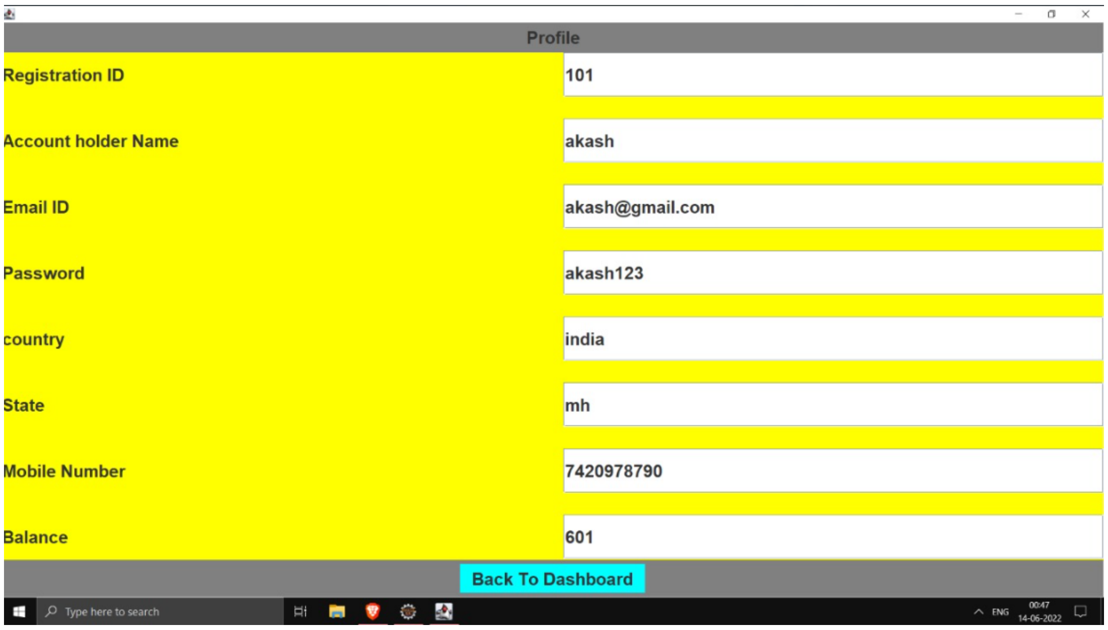
❖ Main File

```
package com.main;

import com.gui.DashBoard;
import com.gui.Home;
import com.gui.Login;
import com.gui.Register;

public class Main {

	public static void main(String[] args) {

		//DashBoard dh=new DashBoard();
		Home home=new Home();
	}

}
```

Outputs :

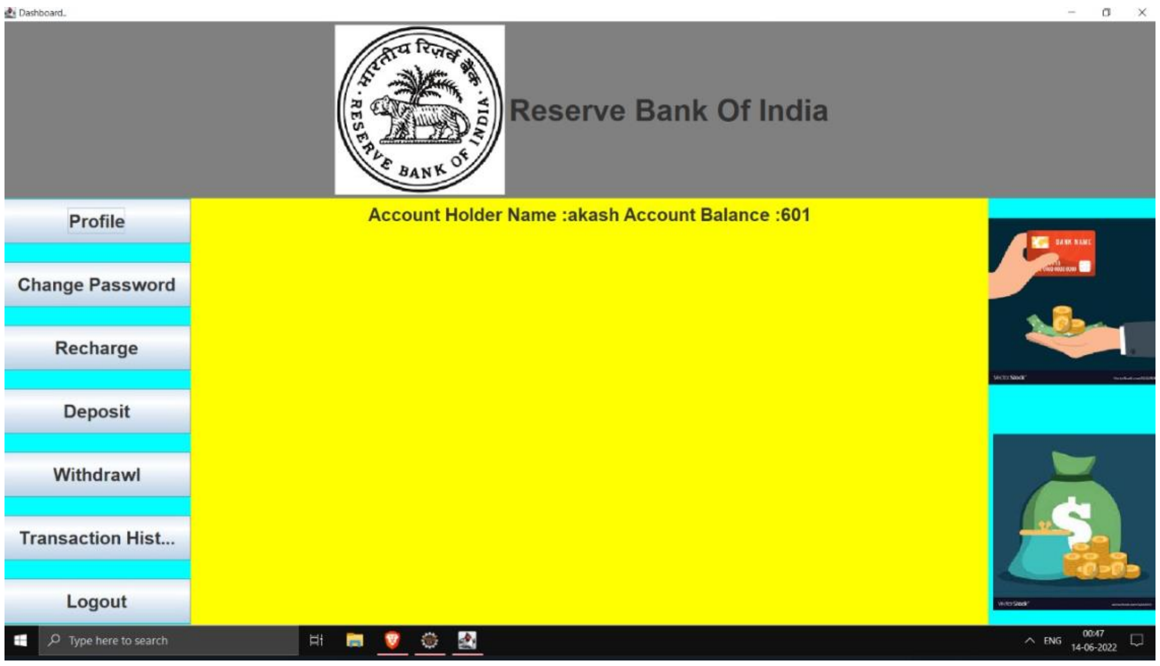

Home Page..
| Welcome To Home Page |
LOGIN
REGISTER
Type here to search
ENG
00:48
14-06-2022


Registration Form
Registration Here
Registeration Id :
Name:
Email-ID:
Create Passowrd:
Confirm Password:
Country:
State:
Phone No:
Balance:
Submit
Reset

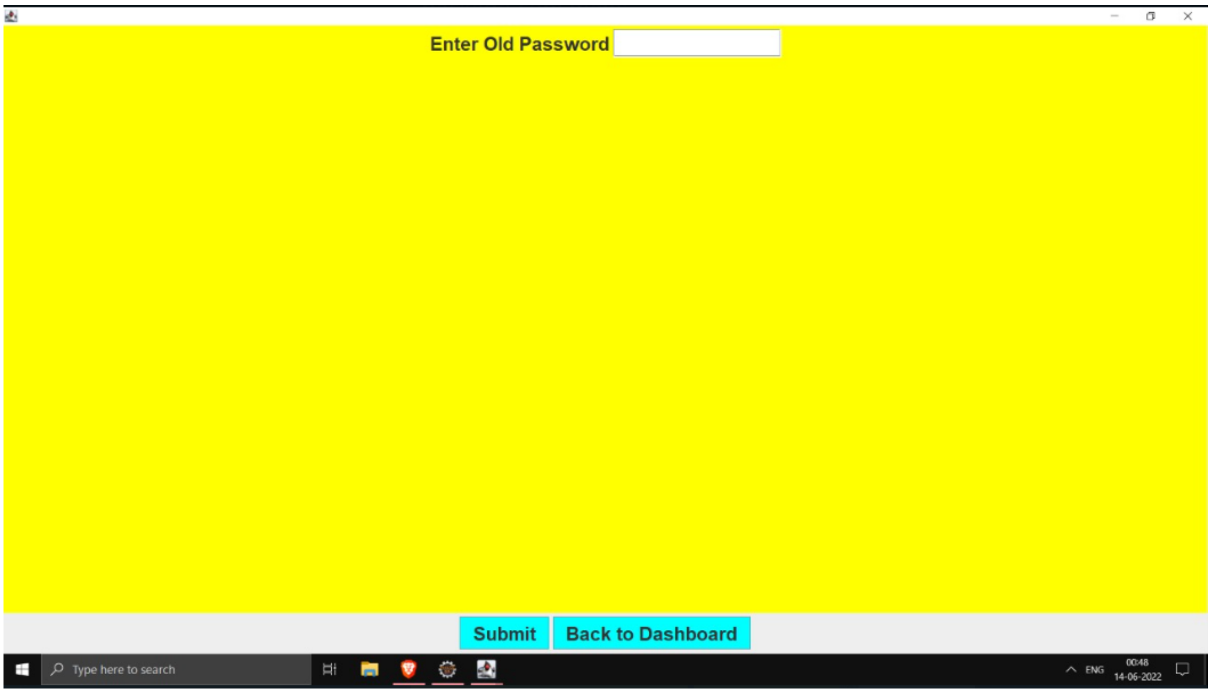


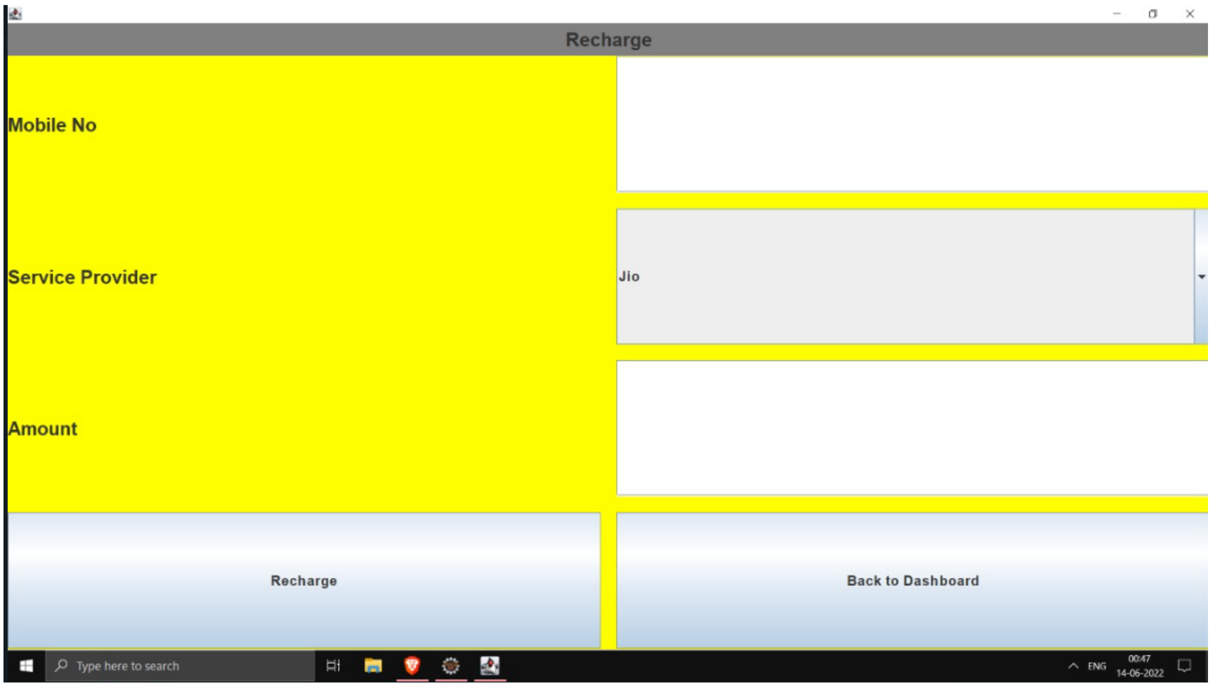
Profile

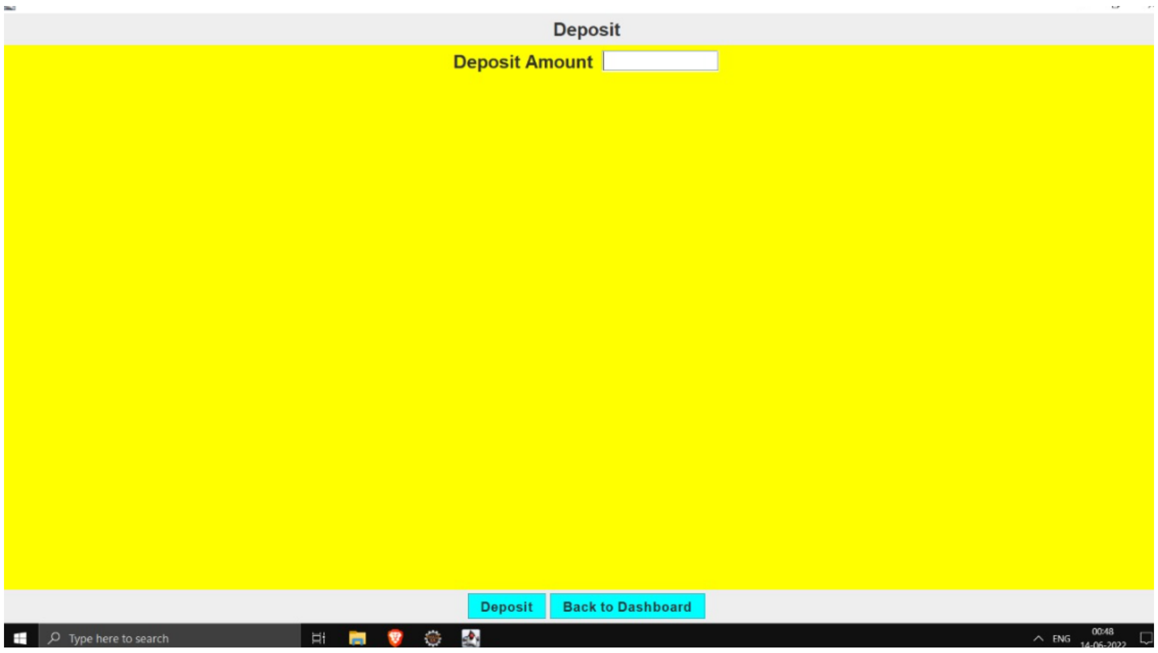
Registration ID	101
Account holder Name	akash
Email ID	akash@gmail.com
Password	akash123
country	india
State	mh
Mobile Number	7420978790
Balance	601

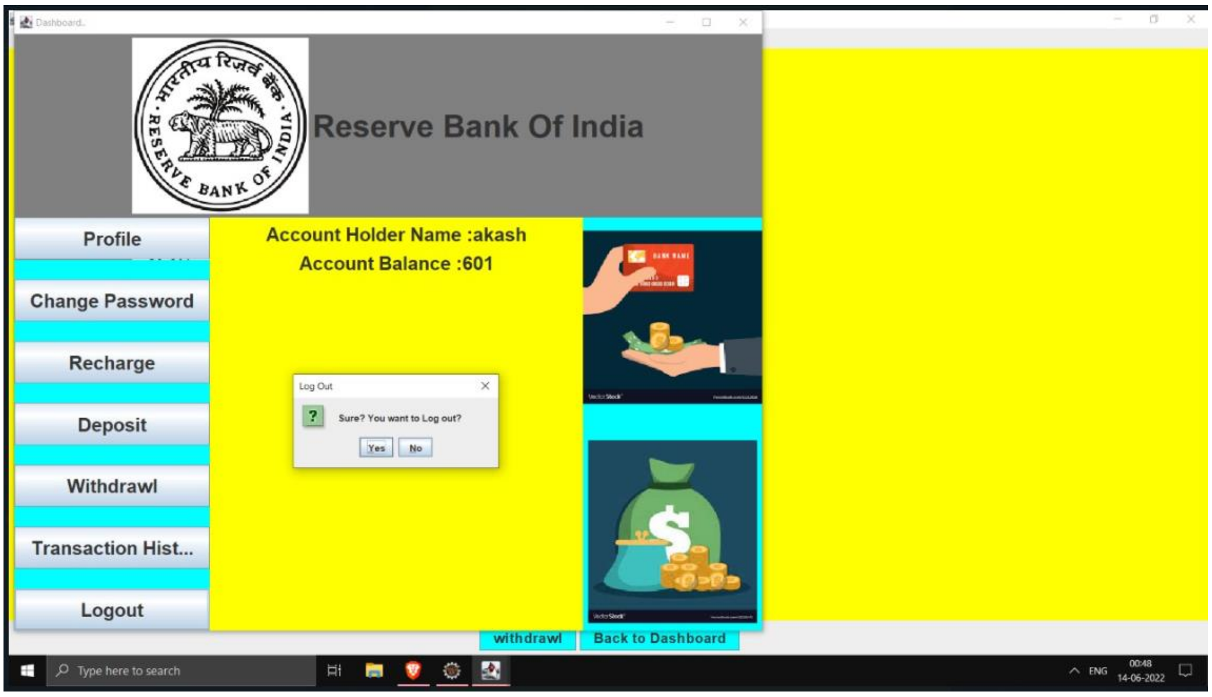


Account Holder Name :akash Account Balance :601


Enter Old Password
Submit
Back to Dashboard
Type here to search
ENG
00:48
14-06-2022


Recharge
Mobile No
Service Provider
Jio
Amount
Recharge
Back to Dashboard
Type here to search
ENG
00:47
14-06-2022


Deposit
Deposit Amount
Deposit
Back to Dashboard
Type here to search
ENG
00:48
14-06-2022


Dashboard
Reserve Bank Of India
Profile
Change Password
Recharge
Deposit
Withdrawl
Transaction Hist...
Logout
Account Holder Name :akash
Account Balance :601
Log Out
Sure? You want to Log out?
Yes
No
withdrawl
Back to Dashboard
Type here to search
ENG
00:48
14-06-2022

Outcome Achieved

a) Working together in a group/teamwork.

b) Working towards achieving the desired outcome.

c) Finding out ways to achieve the outcome in the simplest and fastest way possible.

d) Applying All concepts in Java programming.

e) Achieving the desired outcome using the shortest way.

f) Logic Building according to Different concepts.

g) How Real-Time java Applications are developed.

❖ **Future Scope :**

An increasing demand for a digital banking experience from millennials and Gen Zers is transforming how the entire banking industry operates. Consumers' growing desire to access financial services from digital channels has led to a surge in new banking technologies that are reconceptualizing the banking industry.

References / Websites

1. https://www.w3schools.com/java/java_intro.asp
2. https://www.javatpoint.com/java-tutorial
3. https://www.programiz.com/java-programming
4. https://www.javatpoint.com/collections-in-java
5. https://www.geeksforgeeks.org/exceptions-in-java/
6. https://www.tutorialspoint.com/jdbc/index.htm