



**A Report on: -
Banking Management System**

Under Subject of Object-Oriented Programming.

B-tech. – 5th(sem.)

Nirma University
Sarkhej-Gandhinagar Highway, Ahmedabad- 382 481



Sr. no.	Name	Roll no.
1.	Pratik Sharma	21bee089

Academic Year (2023-2024)

Nirma University
Sarkhej-Gandhinagar Highway, Ahmedabad- 382 481

Table Of Content

- Introduction
 - About Java (OOP).
 - About Bank Management System.
 - About GUI.
- Abstract
- Proto-Type Model
- Objective/ AIM of the Project
- Working of System
- Software/Solution /Skill requirements
- Source Code
- Conclusion

Introduction

About JAVA:

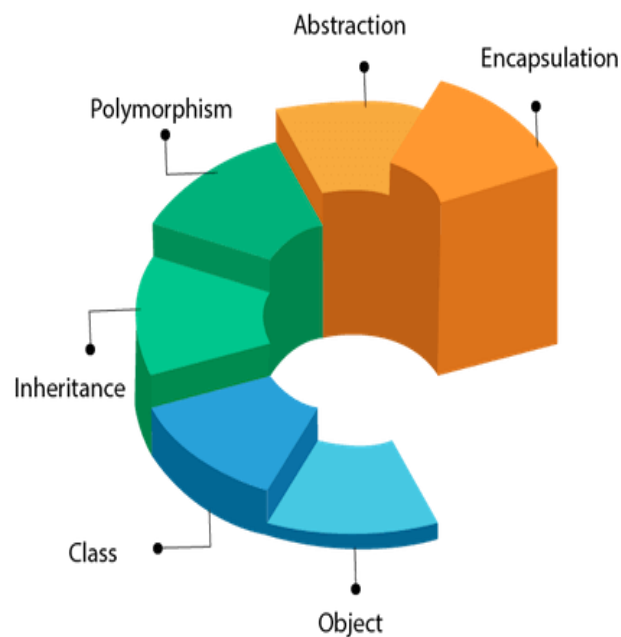
OOP stands for Object-Oriented Programming.

Procedural programming is about writing procedures or methods that perform operations on the data, while object-oriented programming is about creating objects that contain both data and methods.

Object-oriented programming has several advantages over procedural programming:

- OOP is faster and easier to execute
- OOP provides a clear structure for the programs
- OOP helps to keep the Java code DRY "Don't Repeat Yourself", and makes the code easier to maintain, modify and debug
- OOP makes it possible to create full reusable applications with less code and shorter development time

OOPs (Object-Oriented Programming System)



About Bank-Management System:

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" project is a model Desktop Banking Site. This desktop version 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 desktop version for viewing their Account details and perform the transactions on account as per their requirements.

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.

About GUI:

GUI stands for Graphical User Interface. It refers to an interface that allows one to interact with electronic devices like computers and tablets through graphic elements.



Basics Components of GUI:

- Pointer
- Pointing device
- Icons
- Desktop

GUI Benefits:

- It allows you to place more information within a program.
- The graphics allow users to use complex programs with greater ease.
- It saves time as you do not need to edit configurations manually.
- You can easily memorize the tasks (point-and-click).
- Helps create user-friendly software with a point-and-click.

Abstract

The online banking management system is a software application that enables customers of a bank to perform various financial transactions through the bank's website.

The system provides customers with a secure and convenient way to access their accounts, view account balances, transfer funds, pay bills, and more.

The system is designed to be user-friendly and easy to navigate, with features such as online help and customer support available 24/7.

The online banking management system is also beneficial for banks as it helps them reduce costs associated with traditional banking methods.

By providing customers with online access to their accounts, banks can reduce the need for physical branches and staff, which can result in significant cost savings.

Additionally, the system provides banks with a way to offer additional services to their customers, such as investment advice and financial planning.

Modules of Online Banking System:

Account Management Module: Used for managing the Account Details.

Bank Module: Create an Account, Saving Account, Student Account, Current Account.

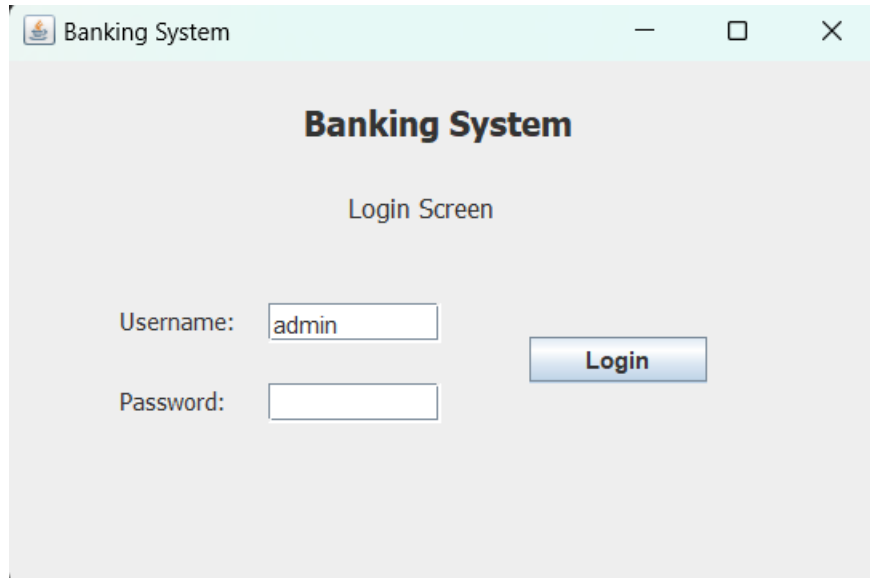
User Module: used for managing the users of the system.

Login Module: used for managing the login details.

Transaction Module: Used for managing the Transaction details.

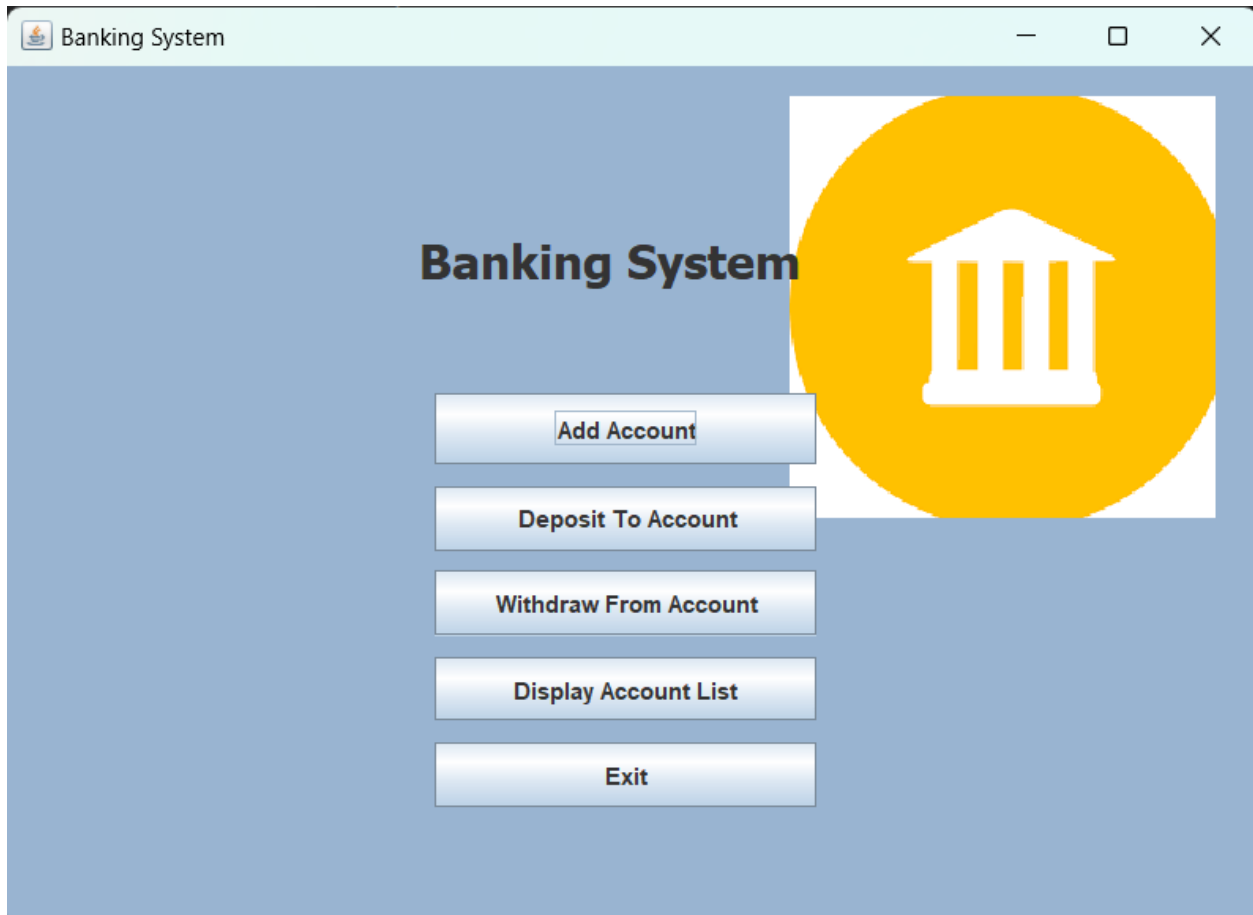
Proto-Type Model:

1. Login Screen:



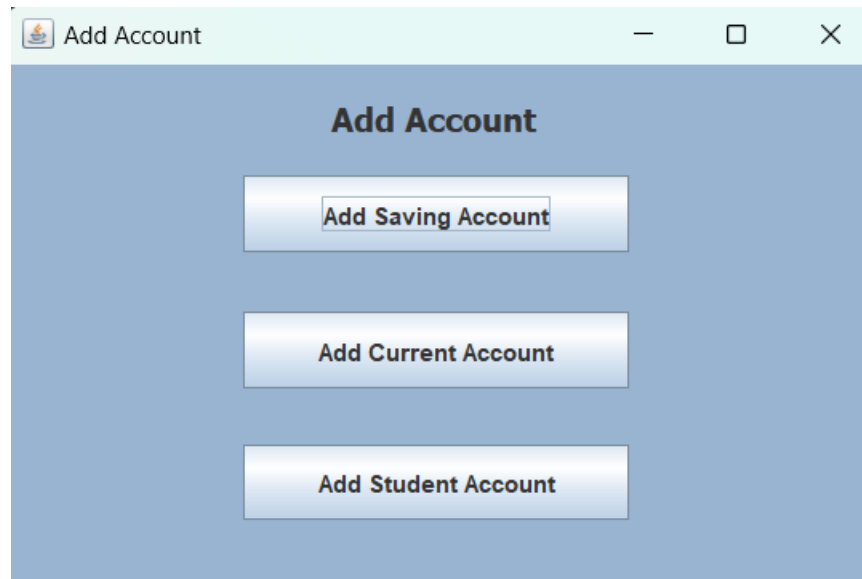
A screenshot of a web browser window titled "Banking System". The page has a light gray background. At the top center, the text "Banking System" is displayed in a bold, black font. Below it, the text "Login Screen" is centered. There are two input fields: "Username:" with the text "admin" entered, and "Password:" which is empty. To the right of the password field is a blue button with the text "Login".

2. Banking System Screen.

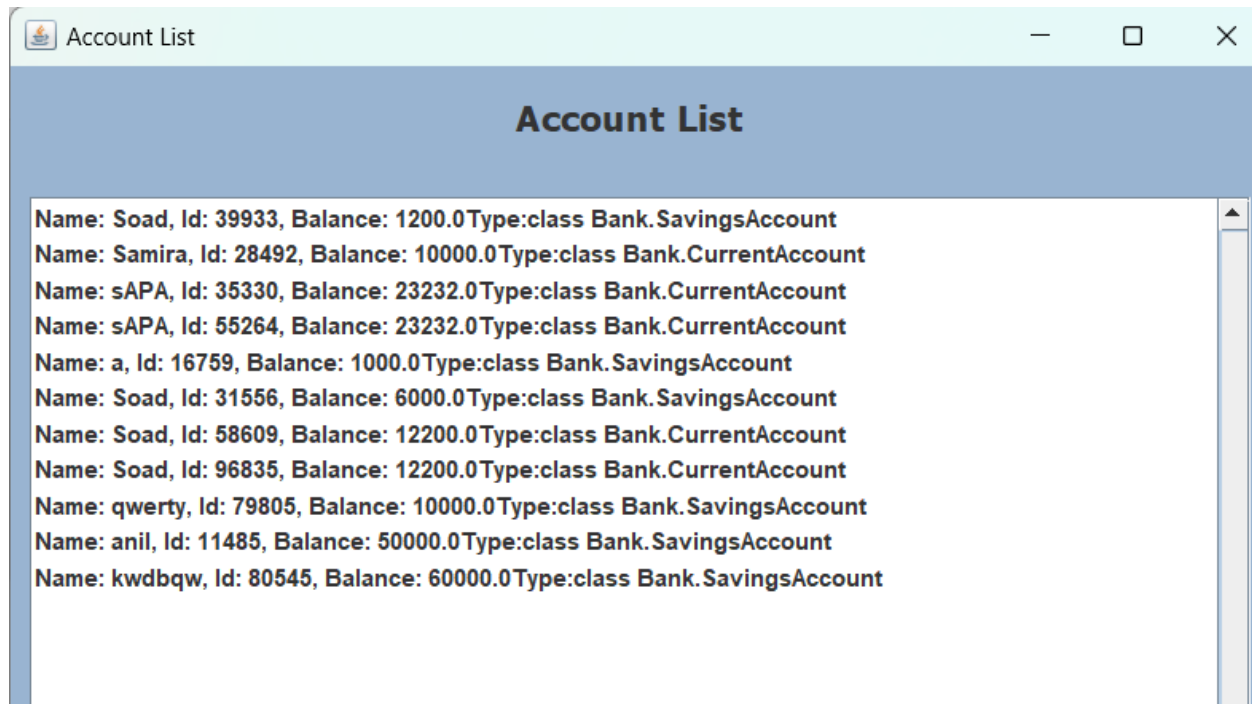


A screenshot of a web browser window titled "Banking System". The page has a blue background. On the right side, there is a large yellow circle containing a white icon of a classical building with columns. On the left side, the text "Banking System" is displayed in a bold, black font. Below the text, there are five blue buttons stacked vertically: "Add Account", "Deposit To Account", "Withdraw From Account", "Display Account List", and "Exit".

3. Add Account



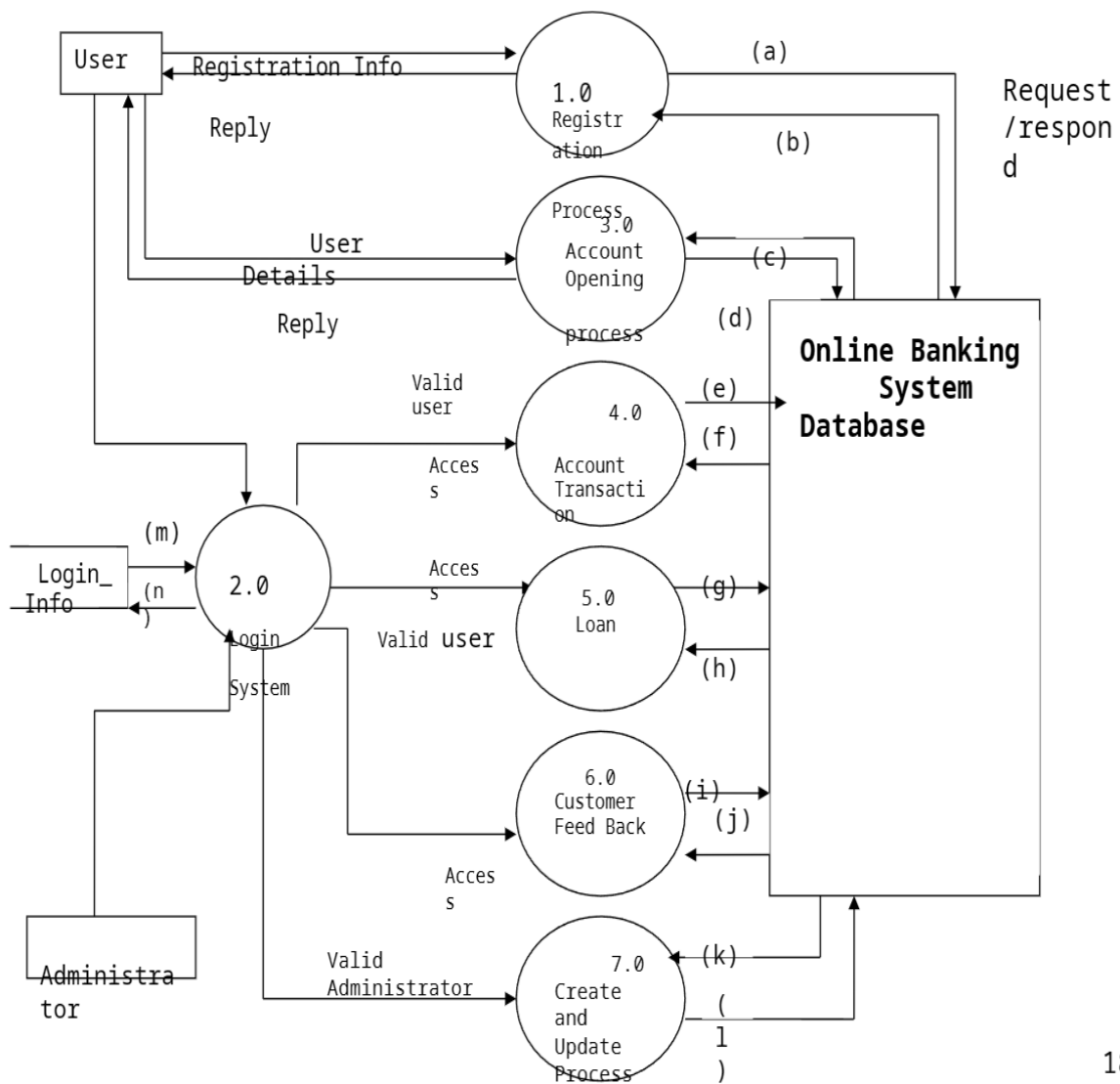
4. Account List.



Objective/AIM of this project

- User-friendly interface
- Fast access to the database
- Less error
- More Storage Capacity
- Search facility
- Look and Feel Environment
- Quick transaction
- Methods
 - We need to be able to generate an account number.
 - Account Type: Saving or Current Account
 - Update Balance
 - Open/Close Account
 - Withdraw/Deposit

Working System



Software/Solution /Skill requirements

1. JAVA
2. OOP Concept
3. VS Code
4. IntelliJ idea
5. GUI
6. Chrome
7. My SQL Workbench
8. HTML/CSS

Source Code.

Bank Module codes.

```
package Bank;
import java.io.Serializable;
import javax.swing.DefaultListModel;
import Exceptions.AccNotFound;
import Exceptions.InvalidAmount;
import Exceptions.MaxBalance;
import Exceptions.MaxWithdraw;
public class Bank implements Serializable {
    private static final long serialVersionUID = 1L;
    private BankAccount[] accounts= new BankAccount[100];
    public int addAccount(BankAccount acc)
    {
        int i=0;
        for(i=0;i<100;i++)
        {
            if(getAccounts()[i]==null)
            {
                break;
            }
        }
        getAccounts()[i]=acc;
        return i;
    }

    public int addAccount(String name, double balance, double maxWithLimit
)
    {
        SavingsAccount acc=new SavingsAccount(name, balance,
maxWithLimit);
        return this.addAccount(acc);
    }

    public int addAccount(String name, double balance, String
tradeLicense) throws Exception
    {
        CurrentAccount acc = new CurrentAccount(name,
balance,tradeLicense);
        return this.addAccount(acc);
    }
}
```

```

    }

    public int addAccount(String name, String institutionName, double
balance, double min_balance)
    {
        StudentAccount acc= new
StudentAccount(name,balance,institutionName);
        return this.addAccount(acc);
    }

    public BankAccount findAccount(String aaccountNum)
    {
        int i;
        for(i=0;i<100;i++)
        {
            if(getAccounts()[i]==null)
            {
                break;
            }
            if(getAccounts()[i].acc_num.equals(aaccountNum))
            {
                return getAccounts()[i];
            }
        }
        return null;
    }

    public void deposit(String aaccountNum, double amt) throws
InvalidAmount,AccNotFound

    {
        if(amt<0)
        {
            throw new InvalidAmount("Invalid Deposit amount");
        }
        BankAccount temp=findAccount(aaccountNum);
        if(temp==null)
        {
            throw new AccNotFound("Account Not Found");
        }
        if(temp!=null)
        {

```

```

        temp.deposit(amt);

    }

}

    public void withdraw(String aaccountNum, double amt) throws
MaxBalance, AccNotFound, MaxWithdraw, InvalidAmount
    {
        BankAccount temp=findAccount(aaccountNum);

        if(temp==null)
        {
            throw new AccNotFound("Account Not Found");
        }

        if(amt<=0)
        {
            throw new InvalidAmount("Invalid Amount");
        }

        if(amt>temp.getbalance())
        {
            throw new MaxBalance("Insufficient Balance");
        }
        if(temp!=null)
        {
            temp.withdraw(amt);
        }
    }

    public DefaultListModel<String> display()
    {
        DefaultListModel<String> list=new DefaultListModel<String>();
        int i;

        for(i=0;i<100;i++)
        {
            if(getAccounts()[i]==null)
            {
                break;
            }
        }
    }
}

```

```

    }

    list.addElement(getAccounts()[i].toString());

}

return list;
}

public BankAccount[] getAccounts() {
    return accounts;
}

public void setAccounts(BankAccount[] accounts) {
    this.accounts = accounts;
}
}

```

```

package Bank;
import java.io.Serializable;

import Exceptions.MaxBalance;
import Exceptions.MaxWithdraw;

public class BankAccount implements Serializable {

    private static final long serialVersionUID = 1L;
    String name;
    private double balance;
    double min_balance;
    String acc_num;

    public BankAccount(String name, double balance, double min_balance) {
        this.name = name;
        this.balance = balance;
    }
}

```



```

        this.min_balance = min_balance;
        acc_num = 10000 + (int)(Math.random()*89999) + "";
    }

    public void deposit(double amount)
    {
        balance+=amount;
    }

    public void withdraw(double amount) throws MaxWithdraw, MaxBalance
    {
        if((balance-amount)>=min_balance && amount<balance)
        {
            balance-=amount;

        }

        else
        {
            throw new MaxBalance("Insufficient Balance");
        }
    }

    public double getbalance()
    {
        return balance;
    }

    @Override
    public String toString() {
        return "Name: " + name + ", Id: " + acc_num + ", Balance: " +
balance +"Type:"+this.getClass();
    }
}

```

```
package Bank;

public class CurrentAccount extends BankAccount {

    private static final long serialVersionUID = 1L;
    String tradeLicenseNumber;

    public CurrentAccount(String name, double balance,String
tradeLicenseNumber) throws Exception {
        super(name, balance, 5000);
        this.tradeLicenseNumber= tradeLicenseNumber;
        if(balance<5000) throw new Exception("Insufficient Balance");
    }

}
```

```
package Bank;
import Exceptions.MaxBalance;
import Exceptions.MaxWithdraw;

public class SavingsAccount extends BankAccount {

    private static final long serialVersionUID = 1L;
    float rate= .05f;
    double maxWithLimit;

    public SavingsAccount(String name, double balance,double maxWithLimit)
{
        super(name, balance, 2000);
        this.maxWithLimit= maxWithLimit;
    }
}
```

```

    }

    public double getNetBalance()
    {
        double NetBalance= getbalance()+(getbalance()*rate);
        return NetBalance;
    }

    public void withdraw(double amount) throws MaxWithdraw, MaxBalance
    {
        if(amount<maxWithLimit)
        {
            super.withdraw(amount);
        }
        else
        {
            throw new MaxWithdraw("Maximum Withdraw Limit Exceed");
        }
    }
}

```

```

package Bank;

public class StudentAccount extends SavingsAccount {

    private static final long serialVersionUID = 1L;
    String institutionName;

    public StudentAccount(String name, double balance
    ,String institutionName) {
        super(name, balance, 20000);
        min_balance=100;
        this.institutionName=institutionName;
    }
}

```

Data Module

```
package Data;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
import Bank.*;
public class FileIO {
    public static Bank bank=null;
    public static void Read()
    {FileInputStream fis =null;
        ObjectInputStream oin=null;
        try {
            fis =new FileInputStream("data");
            oin=new ObjectInputStream(fis);
            FileIO.bank=(Bank)oin.readObject();
        }

        catch (Exception en) {
            FileIO.bank=new Bank();
        }
    finally{
        try{
            if(oin!=null) oin.close();
            if(fis!=null) fis.close();
        }
        catch (IOException en) {
        }}
    }

    public static void Write()
    {
        try {
            FileOutputStream fout=new FileOutputStream("data");
            ObjectOutputStream out=new ObjectOutputStream(fout);
            out.writeObject(FileIO.bank);
            out.flush();
            fout.close();
        }
        catch(Exception en)
```

```
        {  
            }  
        }  
    }  
}
```

Exception Module

Account not found.

```
package Exceptions;  
public class AccNotFound extends Exception {  
    private static final long serialVersionUID = 1L;  
    public AccNotFound(String s)  
    {  
        super(s);  
    }  
}
```

Invalid amount

```
package Exceptions;  
public class InvalidAmount extends Exception {  
    private static final long serialVersionUID = 1L;  
    public InvalidAmount(String s)  
    {  
        super(s);  
    }  
}
```

Max Balance

```
package Exceptions;  
public class MaxBalance extends Exception {  
    private static final long serialVersionUID = 1L;  
  
    public MaxBalance(String s)  
    {  
        super(s);  
    }  
}
```

Max withdraws

```
package Exceptions;
public class MaxWithdraw extends Exception {
private static final long serialVersionUID = 1L;
public MaxWithdraw(String s)
{
    super(s);
}
}
```

GUI codes.

Add account

```
package GUI;
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.border.EmptyBorder;
import javax.swing.JButton;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.SwingConstants;
import java.awt.Font;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import java.awt.SystemColor;
public class AddAccount extends JFrame {
    private static final long serialVersionUID = 1L;
    private JPanel contentPane;
    public AddAccount() {
        setTitle("Add Account");
        setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
        setBounds(100, 100, 450, 300);
        contentPane = new JPanel();
        contentPane.setBackground(SystemColor.activeCaption);
        contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
        setContentPane(contentPane);
        contentPane.setLayout(null);
        JButton btnAddCurrentAccount = new JButton("Add Saving Account");
        btnAddCurrentAccount.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                if(!GUIForm.addsavingsaccount.isVisible())
                {
                    GUIForm.addsavingsaccount.setVisible(true);
                }
                else
                {
                    JOptionPane.showMessageDialog(getComponent(0),
"Already Opened", "Warning", 0);
                }dispose();
            }
        });
        btnAddCurrentAccount.setBounds(118, 56, 193, 38);
    }
}
```

```

contentPane.add(btnAddCurrentAccount);

JButton button = new JButton("Add Current Account");
button.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        if(!GUIForm.addcurrentacc.isVisible())
        {
            GUIForm.addcurrentacc.setVisible(true);
            dispose();
        }
        else
        {
            JOptionPane.showMessageDialog(getComponent(0),
"Already Opened", "Warning", 0);
        }

    }
});
button.setBounds(118, 124, 193, 38);
contentPane.add(button);

JButton btnAddStudentAccount = new JButton("Add Student Account");
btnAddStudentAccount.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        if(!GUIForm.addstudentaccount.isVisible())
        {
            GUIForm.addstudentaccount.setVisible(true);
            dispose();
        }

        else
        {
            JOptionPane.showMessageDialog(getComponent(0),
"Already Opened", "Warning", 0);
        }

    }
});
btnAddStudentAccount.setBounds(118, 190, 193, 38);
contentPane.add(btnAddStudentAccount);
JLabel lblAddAccount = new JLabel("Add Account");

```



```

        lblAddAccount.setFont(new Font("Tahoma", Font.BOLD, 16));
        lblAddAccount.setHorizontalAlignment(SwingConstants.CENTER);
        lblAddAccount.setBounds(108, 11, 210, 34);
        contentPane.add(lblAddAccount);
    }
}

```

Add current Account

```

package GUI;
import javax.swing.JFrame;
import Data.FileIO;
import javax.swing.JPanel;
import javax.swing.border.EmptyBorder;

import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.SwingConstants;
import java.awt.Font;
import javax.swing.JTextField;
import javax.swing.JButton;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import java.awt.SystemColor;

public class AddCurrentAccount extends JFrame {

    private static final long serialVersionUID = 1L;
    private JPanel contentPane;
    private JTextField textField;
    private JTextField textField_1;
    private JTextField textField_2;

    /**
     * Create the frame.
     */
    public AddCurrentAccount() {
        setTitle("Add Current Account");
        setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
        setBounds(100, 100, 450, 300);
        contentPane = new JPanel();
        contentPane.setBackground(SystemColor.activeCaption);
    }
}

```

```

contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
setContentPane(contentPane);
contentPane.setLayout(null);

JLabel lblAddCurrentAccount = new JLabel("Add Current Account ");
lblAddCurrentAccount.setFont(new Font("Tahoma", Font.BOLD, 16));
lblAddCurrentAccount.setHorizontalAlignment(SwingConstants.CENTER)
;

lblAddCurrentAccount.setBounds(10, 11, 414, 34);
contentPane.add(lblAddCurrentAccount);

JLabel lblName = new JLabel("Name:");
lblName.setFont(new Font("Tahoma", Font.PLAIN, 11));
lblName.setBounds(10, 72, 124, 14);
contentPane.add(lblName);

textField = new JTextField();
textField.setBounds(144, 69, 254, 20);
contentPane.add(textField);
textField.setColumns(10);

JLabel lblBalance = new JLabel("Balance:");
lblBalance.setFont(new Font("Tahoma", Font.PLAIN, 11));
lblBalance.setBounds(10, 118, 124, 14);
contentPane.add(lblBalance);

textField_1 = new JTextField();
textField_1.setColumns(10);
textField_1.setBounds(144, 115, 254, 20);
contentPane.add(textField_1);

JLabel lblMaximumWithdrawLimit = new JLabel("Trade Licence
Number:");
lblMaximumWithdrawLimit.setFont(new Font("Tahoma", Font.PLAIN,
11));

lblMaximumWithdrawLimit.setBounds(10, 163, 135, 14);
contentPane.add(lblMaximumWithdrawLimit);

textField_2 = new JTextField();
textField_2.setColumns(10);
textField_2.setBounds(144, 160, 254, 20);
contentPane.add(textField_2);

```

```

JButton btnAdd = new JButton("Add");
btnAdd.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {

        String name=textField.getText();
        double bal=Double.parseDouble(textField_1.getText());
        String trlic=textField_2.getText();
        if(bal<5000)
        {
            JOptionPane.showMessageDialog(getComponent(0),
"Minimum Limit 5000", "Warning", 0);
            textField.setText(null);
            textField_1.setText(null);
            textField_2.setText(null);
        }
        else
        {
            if(name==null||bal<=0||trlic==null)
            {
                JOptionPane.showMessageDialog(getComponent(0),"Typing
Mismatch!! Try Again");
                textField.setText(null);
                textField_1.setText(null);
                textField_2.setText(null);
            }
            else
            {
                int ch=JOptionPane.showConfirmDialog(getComponent(0),
"Confirm?");
                if(ch==0)
                {
                    int index = 0;
                    try {
                        index = FileIO.bank.addAccount(name, bal, trlic);
                    } catch (Exception e1) {

                        e1.printStackTrace();
                    }
                    DisplayList.arr.addElement(FileIO.bank.getAccounts()[i
ndex].toString());

```

```

        JOptionPane.showMessageDialog(getComponent(0), "Success
");
        dispose();
    }
    else
    {
        JOptionPane.showMessageDialog(getComponent(0), "Failed"
);
        textField.setText(null);
        textField_1.setText(null);
        textField_2.setText(null);
    }
}

});
btnAdd.setBounds(86, 209, 89, 23);
contentPane.add(btnAdd);

JButton btnReset = new JButton("Reset");
btnReset.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        textField.setText(null);
        textField_1.setText(null);
        textField_2.setText(null);
    }
});
btnReset.setBounds(309, 209, 89, 23);
contentPane.add(btnReset);
}
}

```

Add saving Account

```
package GUI;
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.border.EmptyBorder;

import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.SwingConstants;
import java.awt.Font;
import javax.swing.JTextField;
import javax.swing.JButton;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import Data.FileIO;

public class AddSavingsAccount extends JFrame {

    private static final long serialVersionUID = 1L;
    private JPanel contentPane;
    private JTextField textField;
    private JTextField textField_1;
    private JTextField textField_2;

    public AddSavingsAccount() {
        setTitle("Add Savings Account ");
        setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
        setBounds(100, 100, 450, 300);
        contentPane = new JPanel();
        contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
        setContentPane(contentPane);
        contentPane.setLayout(null);

        JLabel lblAddCurrentAccount = new JLabel("Add Savings Account ");
        lblAddCurrentAccount.setFont(new Font("Tahoma", Font.BOLD, 16));
        lblAddCurrentAccount.setHorizontalAlignment(SwingConstants.CENTER);

        ;

        lblAddCurrentAccount.setBounds(10, 11, 414, 34);
        contentPane.add(lblAddCurrentAccount);

        JLabel lblName = new JLabel("Name:");
        lblName.setFont(new Font("Tahoma", Font.PLAIN, 11));
        lblName.setBounds(10, 72, 124, 14);
```

```

contentPane.add(lblName);

textField = new JTextField();
textField.setBounds(144, 69, 254, 20);
contentPane.add(textField);
textField.setColumns(10);

JLabel lblBalance = new JLabel("Balance:");
lblBalance.setFont(new Font("Tahoma", Font.PLAIN, 11));
lblBalance.setBounds(10, 118, 124, 14);
contentPane.add(lblBalance);

textField_1 = new JTextField();
textField_1.setColumns(10);
textField_1.setBounds(144, 115, 254, 20);
contentPane.add(textField_1);

JLabel lblMaximumWithdrawLimit = new JLabel("Maximum Withdraw
Limit:");
lblMaximumWithdrawLimit.setFont(new Font("Tahoma", Font.PLAIN,
11));
lblMaximumWithdrawLimit.setBounds(10, 163, 135, 14);
contentPane.add(lblMaximumWithdrawLimit);

textField_2 = new JTextField();
textField_2.setColumns(10);
textField_2.setBounds(144, 160, 254, 20);
contentPane.add(textField_2);

JButton btnAdd = new JButton("Add");
btnAdd.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        String name=textField.getText();
        double bal=Double.parseDouble(textField_1.getText());
        double maxw=Double.parseDouble(textField_2.getText());
        if(bal<2000)
        {
            JOptionPane.showMessageDialog(getComponent(0),
"Minimum Limit 5000", "Warning", 0);
            textField.setText(null);
            textField_1.setText(null);
            textField_2.setText(null);
        }
    }
});

```

```

        else
        {
            if(name==null||bal<=0||maxw<=0)
            {
                JOptionPane.showMessageDialog(getComponent(0),"Typing
Mismatch!! Try Again");
                textField.setText(null);
                textField_1.setText(null);
                textField_2.setText(null);
            }
            else
            {
                int ch=JOptionPane.showConfirmDialog(getComponent(0),
"Confirm?");
                if(ch==0)
                {
                    int index = FileIO.bank.addAccount(name, bal, maxw);
                    DisplayList.arr.addElement(FileIO.bank.getAccounts()[i
index].toString());
                    JOptionPane.showMessageDialog(getComponent(0),"Added
Successfully");
                    dispose();
                }
                else
                {
                    JOptionPane.showMessageDialog(getComponent(0),"Failed"
);
                    textField.setText(null);
                    textField_1.setText(null);
                    textField_2.setText(null);
                }
                textField.setText(null);
                textField_1.setText(null);
                textField_2.setText(null);
            }
        }
    });
    btnAdd.setBounds(86, 209, 89, 23);
    contentPane.add(btnAdd);

    JButton btnReset = new JButton("Reset");
    btnReset.addActionListener(new ActionListener() {

```

```

        public void actionPerformed(ActionEvent e) {
            textField.setText(null);
            textField_1.setText(null);
            textField_2.setText(null);
        }
    });
    btnReset.setBounds(309, 209, 89, 23);
    contentPane.add(btnReset);
}
}

```

Add Student Account

```

package GUI;
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.border.EmptyBorder;

import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.SwingConstants;
import java.awt.Font;
import javax.swing.JTextField;
import javax.swing.JButton;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import Data.FileIO;

public class AddStudentAccount extends JFrame {

    private static final long serialVersionUID = 1L;
    private JPanel contentPane;
    private JTextField textField;
    private JTextField textField_1;
    private JTextField textField_2;

    public AddStudentAccount() {
        setTitle("Add Student Account");
        setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
        setBounds(100, 100, 450, 300);
        contentPane = new JPanel();
    }
}

```



```

contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
setContentPane(contentPane);
contentPane.setLayout(null);

JLabel lblAddCurrentAccount = new JLabel("Add Student Account ");
lblAddCurrentAccount.setFont(new Font("Tahoma", Font.BOLD, 16));
lblAddCurrentAccount.setHorizontalAlignment(SwingConstants.CENTER)
;

lblAddCurrentAccount.setBounds(10, 11, 414, 34);
contentPane.add(lblAddCurrentAccount);

JLabel lblName = new JLabel("Name:");
lblName.setFont(new Font("Tahoma", Font.PLAIN, 11));
lblName.setBounds(10, 72, 124, 14);
contentPane.add(lblName);

textField = new JTextField();
textField.setBounds(144, 69, 254, 20);
contentPane.add(textField);
textField.setColumns(10);

JLabel lblBalance = new JLabel("Balance:");
lblBalance.setFont(new Font("Tahoma", Font.PLAIN, 11));
lblBalance.setBounds(10, 118, 124, 14);
contentPane.add(lblBalance);

textField_1 = new JTextField();
textField_1.setColumns(10);
textField_1.setBounds(144, 115, 254, 20);
contentPane.add(textField_1);

JLabel lblMaximumWithdrawLimit = new JLabel("Institution Name:");
lblMaximumWithdrawLimit.setFont(new Font("Tahoma", Font.PLAIN,
11));
lblMaximumWithdrawLimit.setBounds(10, 163, 135, 14);
contentPane.add(lblMaximumWithdrawLimit);

textField_2 = new JTextField();
textField_2.setColumns(10);
textField_2.setBounds(144, 160, 254, 20);
contentPane.add(textField_2);

JButton btnAdd = new JButton("Add");

```

```

btnAdd.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {

        String name=textField.getText();
        double bal=Double.parseDouble(textField_1.getText());
        String insname=textField_2.getText();
        if(bal<100)
        {
            JOptionPane.showMessageDialog(getComponent(0),
"Minimum Limit 5000", "Warning", 0);
            textField.setText(null);
            textField_1.setText(null);
            textField_2.setText(null);
        }
        else
        {
            if(name==null||bal<=0||insname==null)
            {
                JOptionPane.showMessageDialog(getComponent(0),"Typing
Mismatch!! Try Again");
                textField.setText(null);
                textField_1.setText(null);
                textField_2.setText(null);
            }
            else
            {
                try {
                    FileIO.bank.addAccount(name, bal, insname);
                } catch (Exception e1) {

                }
                int ch=JOptionPane.showConfirmDialog(getComponent(0),
"Confirm?");
                if(ch==0)
                {
                    int index = 0;
                    try {
                        index = FileIO.bank.addAccount(name, bal,
insname);
                    } catch (Exception e1) {

                    }
                }
            }
        }
    }
});

```

```

        DisplayList.arr.addElement(FileIO.bank.getAccounts()[i
index].toString());

        JOptionPane.showMessageDialog(getComponent(0), "Added
Successfully");
        dispose();
    }
    else
    {
        JOptionPane.showMessageDialog(getComponent(0), "Failed"
);
        textField.setText(null);
        textField_1.setText(null);
        textField_2.setText(null);
    }

    }
}

});
btnAdd.setBounds(86, 209, 89, 23);
contentPane.add(btnAdd);

JButton btnReset = new JButton("Reset");
btnReset.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        textField.setText(null);
        textField_1.setText(null);
        textField_2.setText(null);

    }
});
btnReset.setBounds(309, 209, 89, 23);
contentPane.add(btnReset);
}
}

```

Final Login Application Program

```
import java.awt.EventQueue;
import GUI.GUIForm;
public class Application {
    public static void main(String[] args)
    {
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    GUIForm.login.frame.setVisible(true);
                } catch (Exception e) {
                }
            }
        });
    }
}
```

Conclusion:

This project will provide a practical experience in building a basic online banking system, giving customers the ability to interact with their accounts, make transactions, and manage their finances. It showcases a variety of Java concepts and their real-world application.

Analysis of the online banking system has revealed both strengths and areas for improvement. The system excels in providing a user-friendly experience, with an intuitive interface and a range of convenient features that cater to the needs of its diverse user base. The security measures in place, including robust encryption and multi-factor authentication, instill confidence in users regarding the safety of their financial transactions.

The online banking system plays a vital role in the financial industry, offering customers a convenient and secure means of managing their finances. It is a testament to the ongoing digital transformation of the sector.

Thank you for your attention to this report.