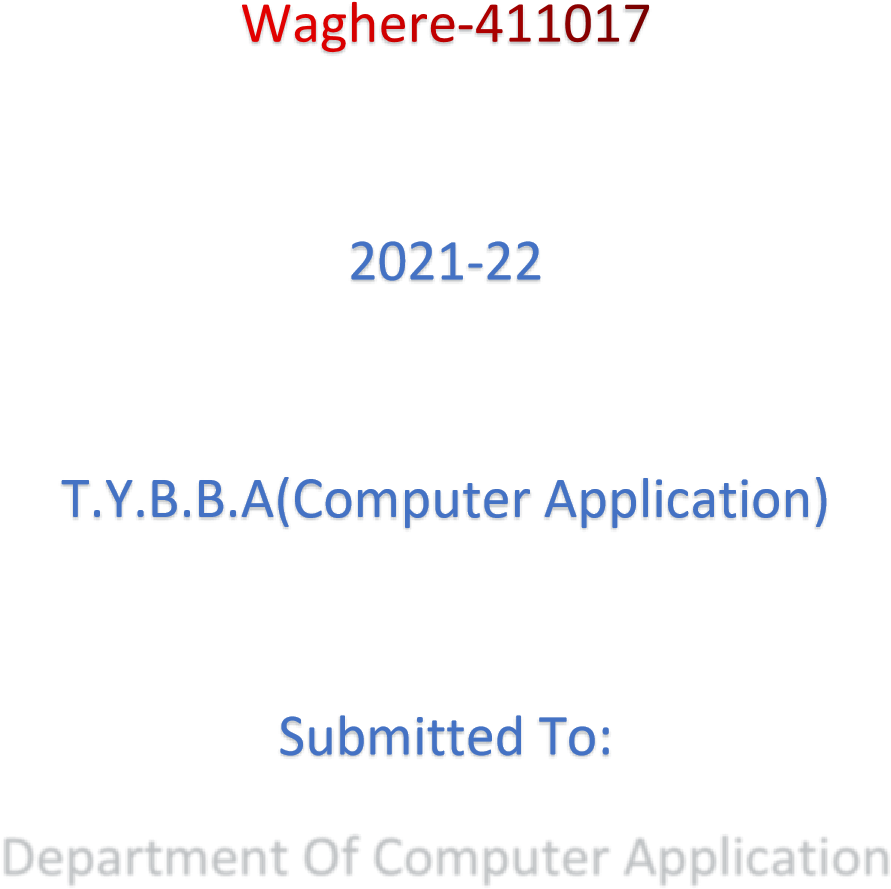
*SAVITRIBAI PHULE UNIVERSITY* **Project Report**   **On** “ BANKING SYSTEM” Submitted by  **VISHAL S. SALPE & 3726 DIPAK S. GUPTA**

# Rayat Shikshan Sanstha’s



**MAHATMA PHULE MAHAVIDYALAYA**

**PIMPRI**

# CERTIFICATE

This is to certify that the work on the project report entitled ”BANKING SYSTEM” has been satisfactorily completed by VISHAL SALPE & DIPAK GUPTA TYBCA (Sem- VI) has been carried out during the academic year 2021-2022. To the best of my knowledge and belief, the matter is presented in the project report is useful for study and practical knowledge

Prof. Nitanjali Mane Prof. Priti Nevase

Project Guide Head of Department

Prof. Priti Nevase **Prof. Mayur Mali** Internal ExaminerExternal Examiner

**CHAPTER 1: INTRODUCTION**

**1.1 Existing System**

The existing bank system is slow as every task is being performed by a human being and comparing the computer task speed with a computer is not fair. The complexity of this system is increased when an increase in the number of customers and with that there will be several transactions will be performed now everything needs to log in to a file for reference in the future which is simply not the kind of scenario we need at this time.

* 1. **Need for System**
* The main purpose of this software is to simplify the tedious task of banking by providing this service in a user-friendly environment.
* It also aims at increasing the efficiency and reducing the drawback of the existing manual banking process thus making it more convenient for the customers to do bank when they require it.
* This project is supported by a well-designed DBMS in which customers’ account information is integrated.
* A friendly is also provided so that the required made by the gives the give correct result by accessing the information stored in the database
  1. **Operating Environment Hardware and Software**

**At Server Side**

Hardware Configuration

|  |  |
| --- | --- |
| **Name** | **Details** |
| Processor | Intel Pentium IV or more |
| RAM | At least 128 MB RAM is required (formally 4 GB Ram) |
| Hard Drive | 1 TB |

Software Environment

|  |  |
| --- | --- |
| **Name** | **Details** |
| Operating System | Windows 7 |
| Database Server | MySQL Server (phpMyAdmin) |
| Web Server | Wamp Server |
| Browser | Chrome |
| Framework | Java Programming. |
| IDE | Eclipse Technologies |

**At Client Side**

Hardware Configuration

|  |  |
| --- | --- |
| **Name** | **Details** |
| Processor | Intel(R) Core(TM) i3-6100 CPU @ 3.70GHz 3.70 GH |
| RAM | 4.00 GB (3.87 GB usable) |
| Hard Drive | 1 TB |

Software Environment

|  |  |
| --- | --- |
| **Name** | **Details** |
| Operating System | Windows 7 |
| Database Server | MySQL Server (phpMyAdmin) |
| Web Server | Wamp Server |
| Framework | Java Programming. |

**CHAPTER 2: Proposed System**

**2.1 Proposed System (Introduction of the system)**

The bank management system is an application for maintaining a person's account in a bank the system provides the access to the customer to create an account deposit/withdraw the case from his account and also to view a report of all accounts pr ent the following presentation provides the specification for the system.

* 1. **Module specifications (Scope)**
* **Customer Module: -**This module allows users to **create** an account. **View** all the existing members. These are the main source of business for the bank.
* **Account Module: -** This module allows every customer will become a customer when they open an account in the bank and start depositing the money or take some other service. The account enables the customer to take advantage of the facilities provided by the bank. Every customer has a unique account number and the bank will identify you by only that account number.
* **Transaction Module: -** This module allows every time an account holder performs some activity on the account it will be updated through transactions this is like logs but only shows the required details. Any time a customer makes any changes in an account like pay or deposit it will be through transactions. This helps in keeping the track of cash flow in the bank. Also, help in managing the correct information if there is some data loss to the bankside there is any query at one customer side.
* **Functional requirements**
* User basic graphical tools such as shapes objects, brushes, colortoolsleraser, ser, etc.
* Should allow freehand drawing, object shapes such as circles ellipses, rectangles, and polygons.
* Should allow the usage of different colors in the form of brushes, shapes, and curves to manage the picture with tools such as pencil, airbrush, and clear all.
* **Non-Functional requirements**
  + Must provide the program in vivid color and format.
  + Should have adaptable to allow a single module at a time.
  + Must enable faster processing of operations when a module is selected.
* **Scope of the system**

The scope of the Bank Management System extends to all the users who wish for easy banking facilities. This software product will be used for storing users’ account information and the transactions made by them.

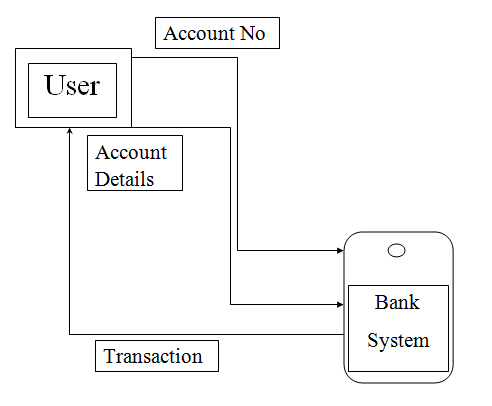
**2.3 Objectives of the system**

The proposed sythe stem aims to develop a system of improved facilities. The Propoproposedem can overcome all the limitations of the existing system. The

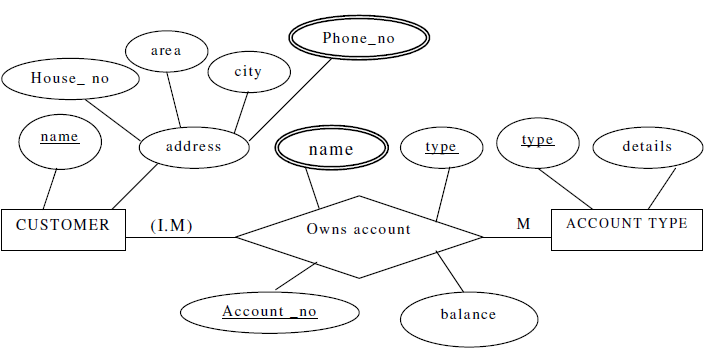
The system provides proper security and reduces manual work.

**CHAPTER 3: ANALYSIS & DESIGN**

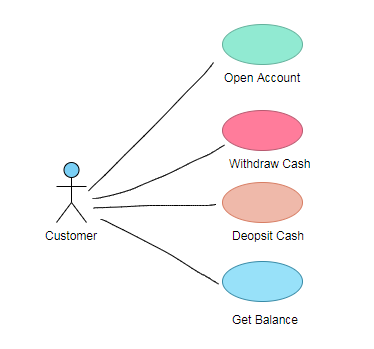
**3.1 0th Level DFD**



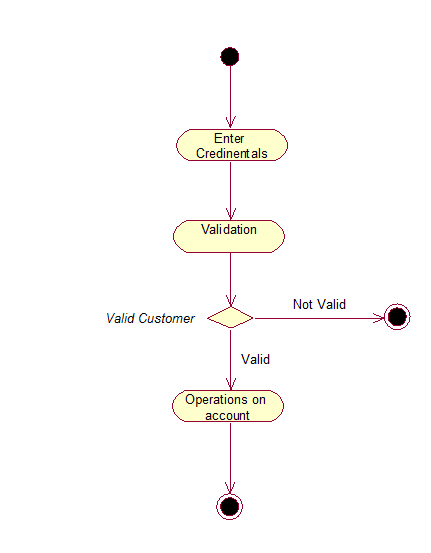
**3.2ERD (Entity Relationship Diagram)**



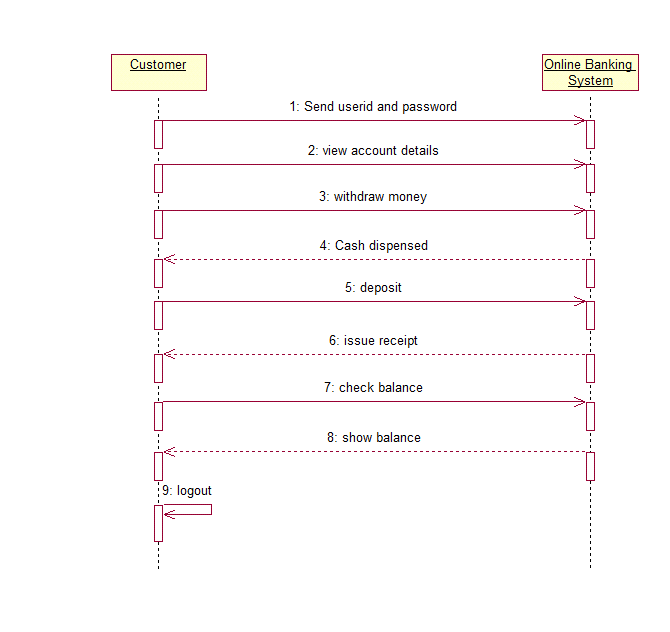
3.3 Use Case Diagrams

****

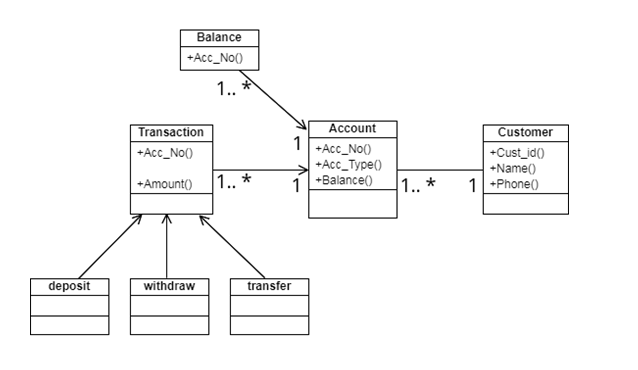
3.4 Activity Diagram



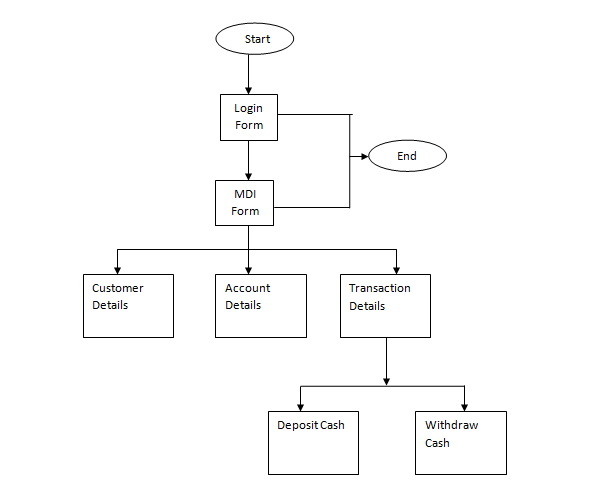
**3.5 Sequence Diagram**



3.6 Class Diagram

****

Module Hierarchy Diagram

****

3.7 Table specifications (Database design)

Login Table/New User Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr.no | Column name | Data Type | Width | Constraint |
| 1. | Id | int | 10 | Primary Key |
| 2. | First\_Name | varchar | 255 | Not null |
| 3. | Last\_Name | varchar | 255 | Not null |
| 4. | Email | varchar | 255 | Not null |
| 5. | Username | varchar | 255 | Not null |
| 6. | Password | varchar | 255 | Not null |
| 7. | Conform\_Password | varchar | 255 | Not null |

Open Account

- Customer Details Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr.no | Column name | Data Type | Width | Constraint |
| 1. | Id | int | 10 | Primary Key |
| 2. | Cust\_id | varchar | 255 | Not null |
| 3. | First\_Name | varchar | 255 | Not null |
| 4. | Last\_Name | varchar | 255 | Not null |
| 5. | Phone | int | 10 | Not null |
| 6. | Email | varchar | 255 | Not null |
| 7. | Gender | varchar | 255 | Not null |
| 8. | DOB | date | 55 | Not null |
| 9. | City | varchar | 255 | Not null |
| 10. | State | varchar | 255 | Not null |
| 11. | Address | varchar | 255 | Not null |
| 12. | Photo | varchar | 255 | Not null |

- Account Details Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr.no | Column name | Data Type | Width | Constraint |
| 1. | Id | int | 10 | Primary Key |
| 2. | Acc\_id | varchar | 255 | Not null |
| 3. | Cust\_id | varchar | 255 | Not null |
| 4. | First\_Name | varchar | 255 | Not null |
| 5. | Last\_Name | varchar | 255 | Not null |
| 6. | Phone | int | 10 | Not null |
| 7. | Email | varchar | 255 | Not null |
| 8. | Acc\_Type | varchar | 255 | Not null |
| 9. | Balance | varchar | 255 | Not null |

Deposit Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr.no | Column name | Data Type | Width | Constraint |
| 1. | Id | int | 10 | Primary Key |
| 2. | Acc\_id | varchar | 255 | Not null |
| 3. | Cust\_id | varchar | 255 | Not null |
| 4. | First\_Name | varchar | 255 | Not null |
| 5. | Last\_Name | varchar | 255 | Not null |
| 6. | Date | varchar | 255 | Not null |
| 7. | Balance | int | 10 | Not null |
| 8. | Deposite\_Amount | int | 10 | Not null |

Withdraw Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr.no | Column name | Data Type | Width | Constraint |
| 1. | Id | int | 10 | Primary Key |
| 2. | Acc\_id | varchar | 255 | Not null |
| 3. | Cust\_id | varchar | 255 | Not null |
| 4. | First\_Name | varchar | 255 | Not null |
| 5. | Last\_Name | varchar | 255 | Not null |
| 6. | Date | varchar | 255 | Not null |
| 7. | Balance | int | 10 | Not null |
| 8. | Withdraw\_Amount | int | 10 | Not null |

Transfer Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr.no | Column name | Data Type | Width | Constraint |
| 1. | Id | int | 10 | Primary Key |
| 2. | F\_Account | varchar | 255 | Not null |
| 3. | Balance | int | 10 | Not null |
| 4. | To\_Account | varchar | 255 | Not null |
| 5. | Transfer\_Amount | int | 10 | Not null |
| 6. | Date | varchar | 255 | Not null |

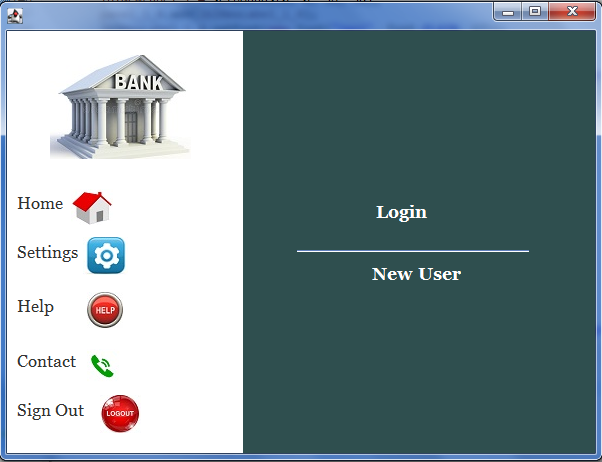
3.5 Data dictionary

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Sr.no | Column name | Data Type | Width | Constraint | Description | Table Name |
| 1. | Id | int | 10 | Primary Key | 1 | Create\_Account |
| 2. | First\_Name | varchar | 255 | Not null | 2 | Create\_Account |
| 3. | Last\_Name | varchar | 255 | Not null | 3 | Create\_Account |
| 4. | Email | varchar | 255 | Not null | 4 | Create\_Account |
| 5. | Username | varchar | 255 | Not null | 5 | Create\_Account |
| 6. | Password | varchar | 255 | Not null | 6 | Create\_Account |
| 7. | Conform\_Password | varchar | 255 | Not null | 7 | Create\_Account |
| 1. | Id | int | 10 | Primary Key | 1 | customer |
| 2. | Cust\_id | varchar | 255 | Not null | 2 | customer |
| 3. | First\_Name | varchar | 255 | Not null | 3 | customer |
| 4. | Last\_Name | varchar | 255 | Not null | 4 | customer |
| 5. | Phone | int | 10 | Not null | 5 | customer |
| 6. | Email | varchar | 255 | Not null | 6 | customer |
| 7. | Gender | varchar | 255 | Not null | 7 | customer |
| 8. | DOB | date | 55 | Not null | 8 | customer |
| 9. | City | varchar | 255 | Not null | 9 | customer |
| 10. | State | varchar | 255 | Not null | 10 | customer |
| 11. | Address | varchar | 255 | Not null | 11 | customer |
| 12. | Photo | varchar | 255 | Not null | 12 | customer |
| 1. | Id | int | 10 | Primary Key | 1 | account |
| 2. | Acc\_id | varchar | 255 | Not null | 2 | account |
| 3. | Cust\_id | varchar | 255 | Not null | 3 | account |
| 4. | First\_Name | varchar | 255 | Not null | 4 | account |
| 5. | Last\_Name | varchar | 255 | Not null | 5 | account |
| 6. | Phone | int | 10 | Not null | 6 | account |
| 7. | Email | varchar | 255 | Not null | 7 | account |
| 8. | Acc\_Type | varchar | 255 | Not null | 8 | account |
| 9. | Balance | varchar | 255 | Not null | 9 | account |
| 1. | Id | int | 10 | Primary Key | 1 | deposit |
| 2. | Acc\_id | varchar | 255 | Not null | 2 | deposit |
| 3. | Cust\_id | varchar | 255 | Not null | 3 | deposit |
| 4. | First\_Name | varchar | 255 | Not null | 4 | deposit |
| 5. | Last\_Name | varchar | 255 | Not null | 5 | deposit |
| 6. | Date | varchar | 255 | Not null | 6 | deposit |
| 7. | Balance | int | 10 | Not null | 7 | deposit |
| 8. | Deposite\_Amount | int | 10 | Not null | 8 | deposit |
| 1. | Id | int | 10 | Primary Key | 1 | withdraw |
| 2. | Acc\_id | varchar | 255 | Not null | 2 | withdraw |
| 3. | Cust\_id | varchar | 255 | Not null | 3 | withdraw |
| 4. | First\_Name | varchar | 255 | Not null | 4 | withdraw |
| 5. | Last\_Name | varchar | 255 | Not null | 5 | withdraw |
| 6. | Date | varchar | 255 | Not null | 6 | withdraw |
| 7. | Balance | int | 10 | Not null | 7 | withdraw |
| 8. | Withdraw\_Amount | int | 10 | Not null | 8 | withdraw |
| 1. | Id | int | 10 | Primary Key | 1 | transfer |
| 2. | F\_Account | varchar | 255 | Not null | 2 | transfer |
| 3. | Balance | int | 10 | Not null | 3 | transfer |
| 4. | To\_Account | varchar | 255 | Not null | 4 | transfer |
| 5. | Transfer\_Amount | int | 10 | Not null | 5 | transfer |
| 6. | Date | varchar | 255 | Not null | 6 | transfer |

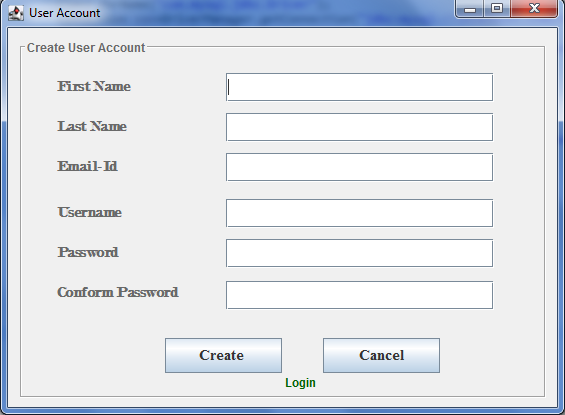
**CHAPTER 4: USER MANUAL**

4.1 User Interface Screens (Input)

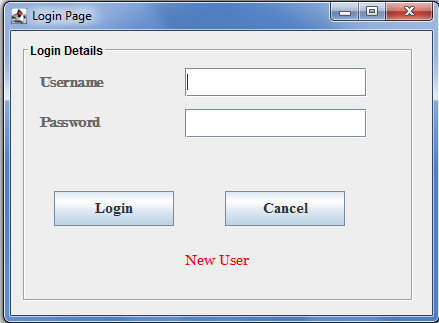
* Login/New User Page



* New User



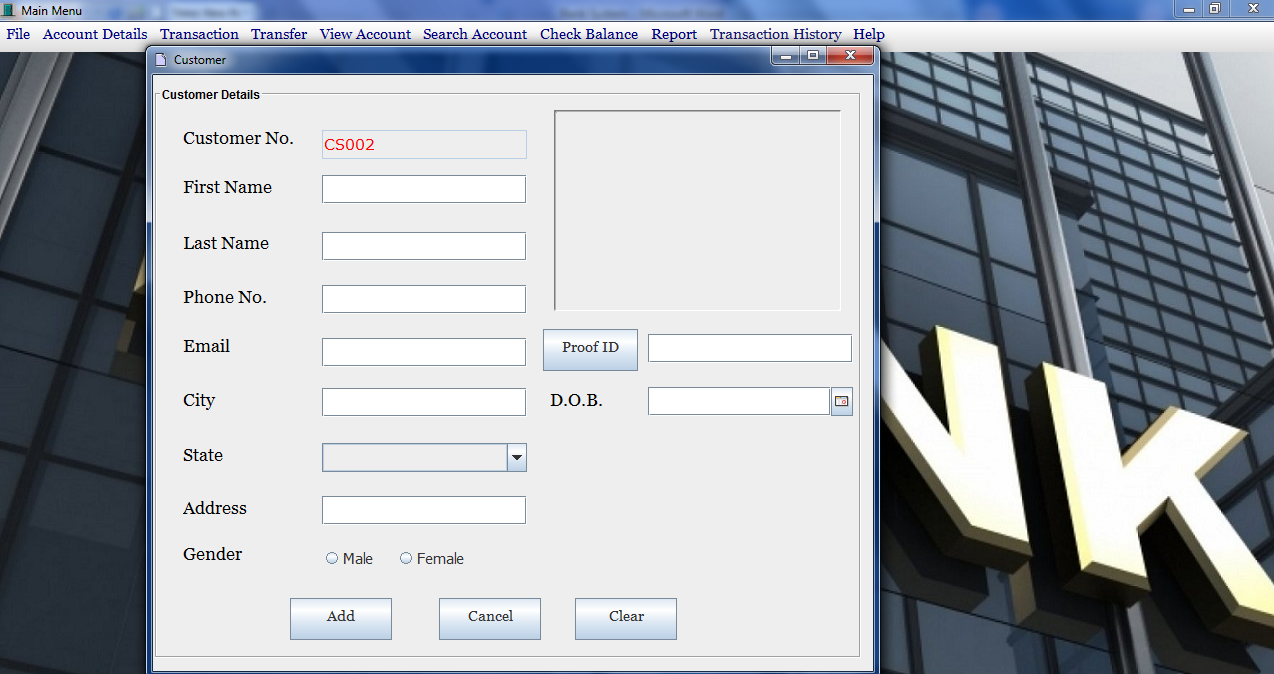
* Login Page



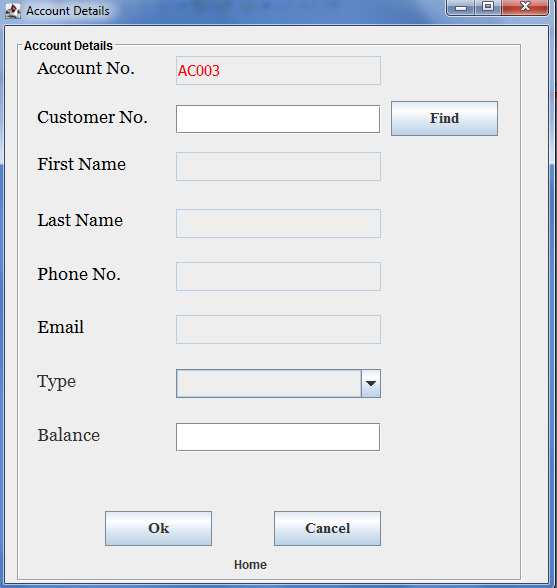
* Menu Page



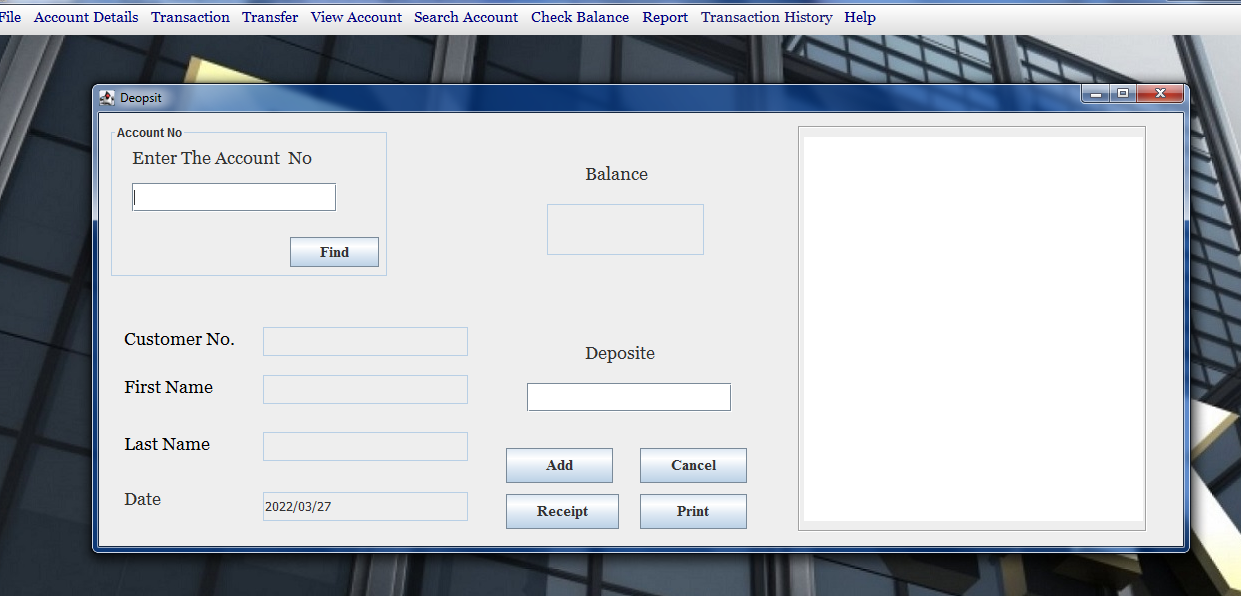
* Open Account
* Customer Details



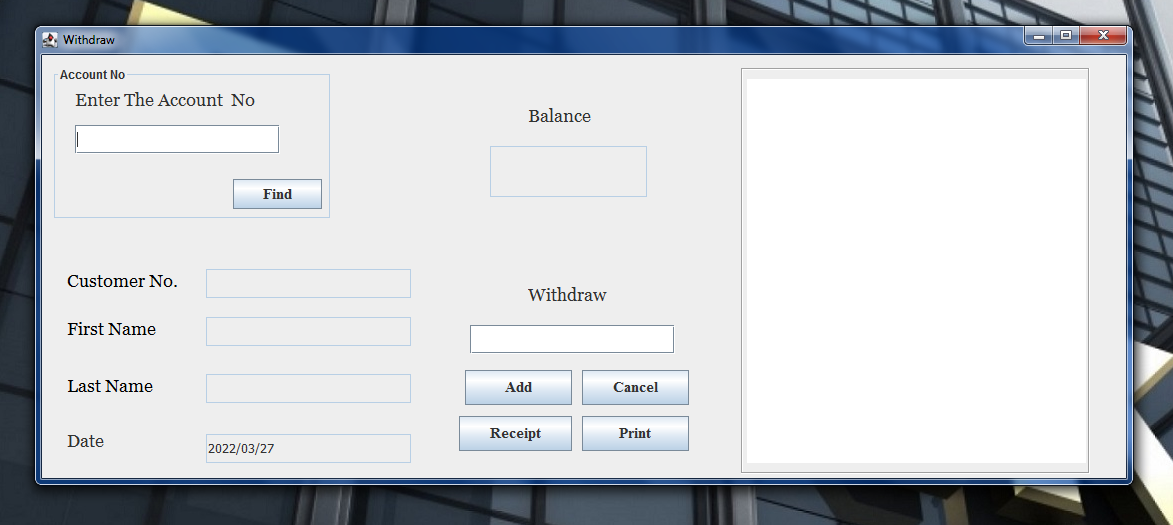
* Account Details



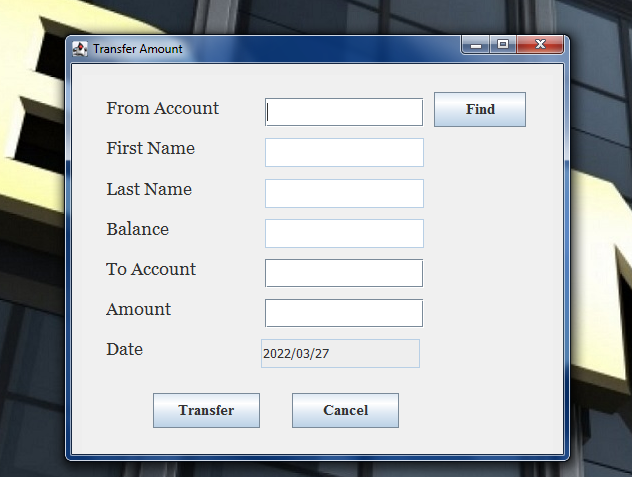
* Deposit Cash



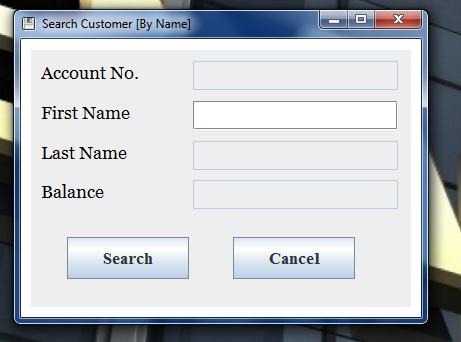
* Withdraw Cash



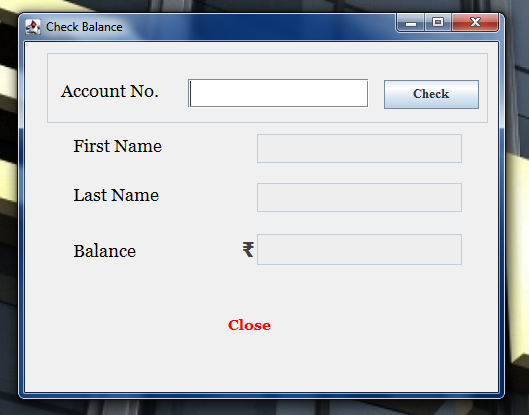
* Transfer Amount



* Search Customer

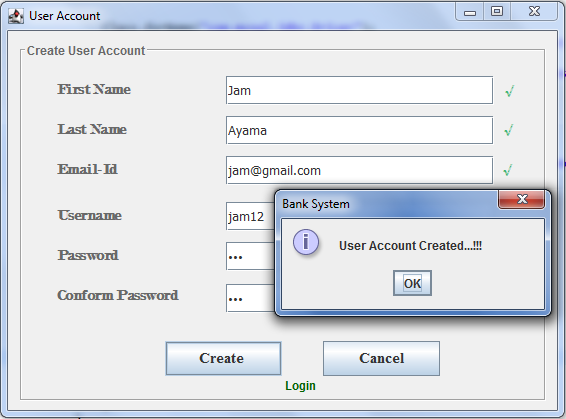


* Check Balance

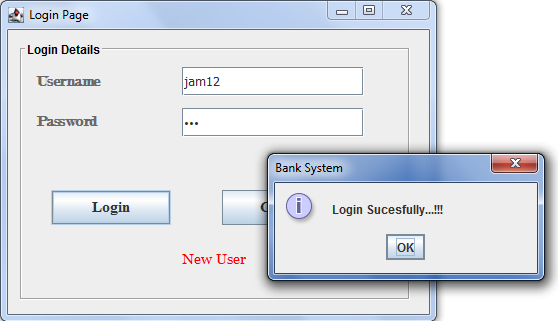


**4.2 Output Screens with data**

* New User



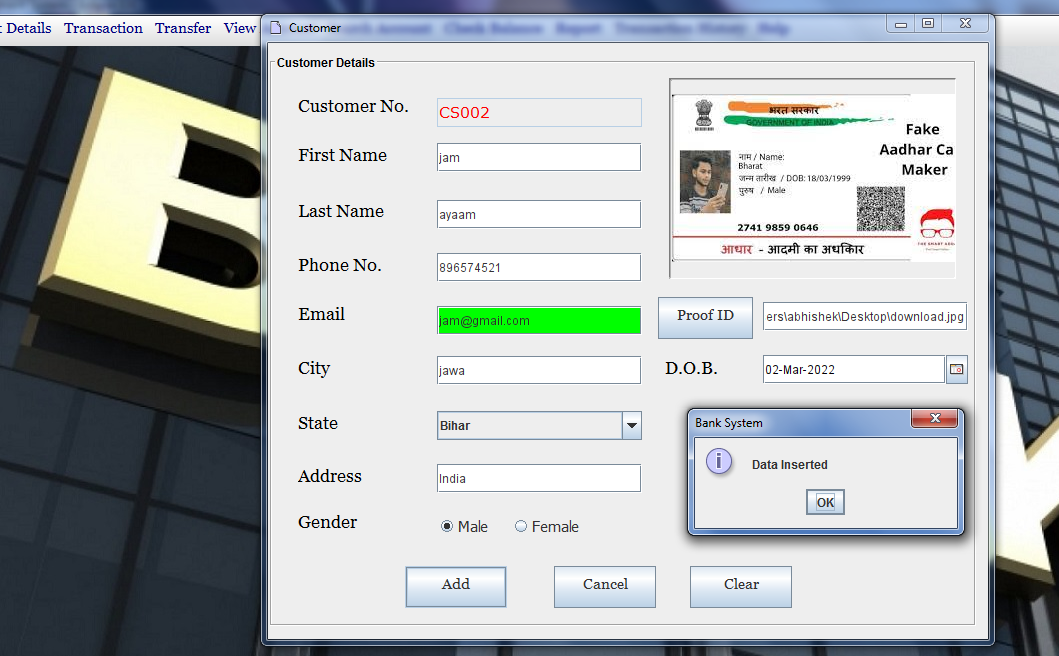
* Login Page



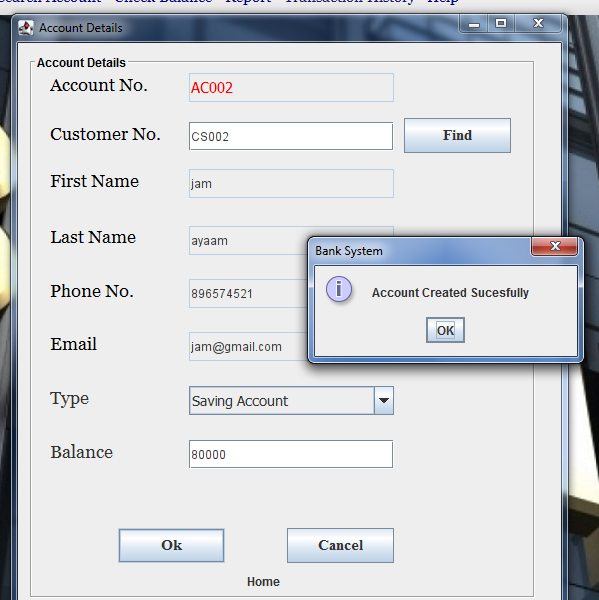
* Menu Page



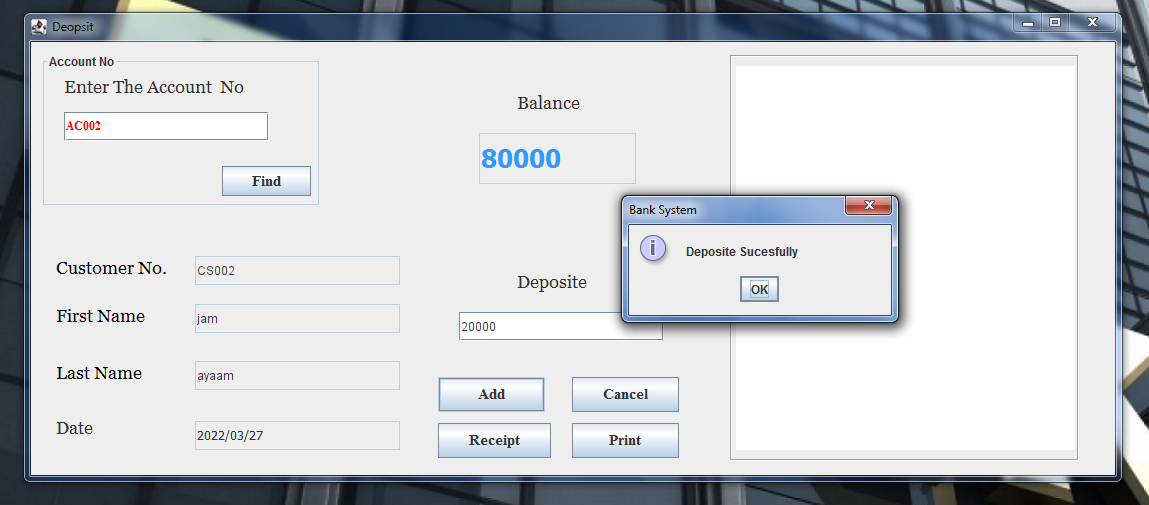
* Open Account
* Customer Details



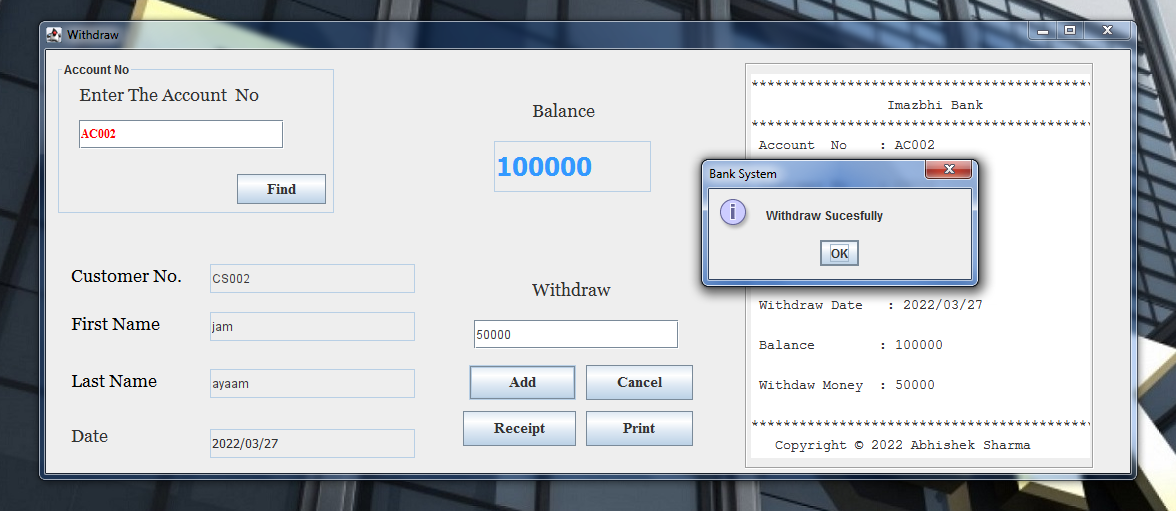
* Account Details



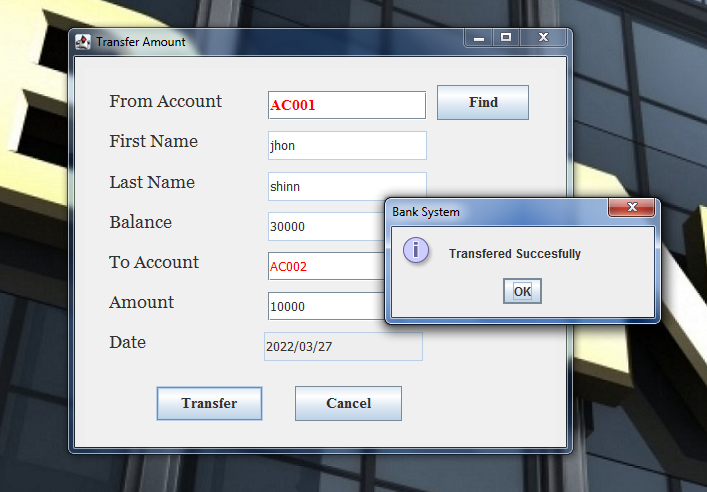
* Deposit Cash



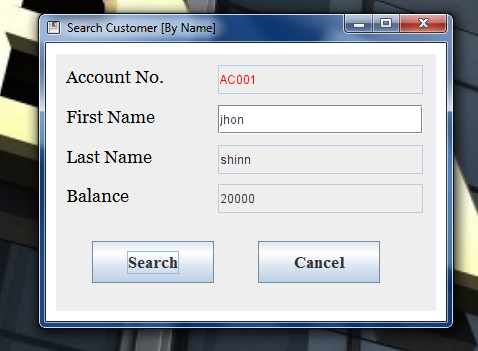
* Withdraw Cash



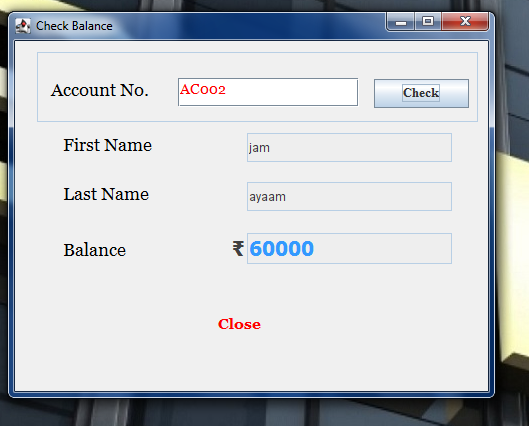
* Transfer Amount



* Search Customer



* Check Balance

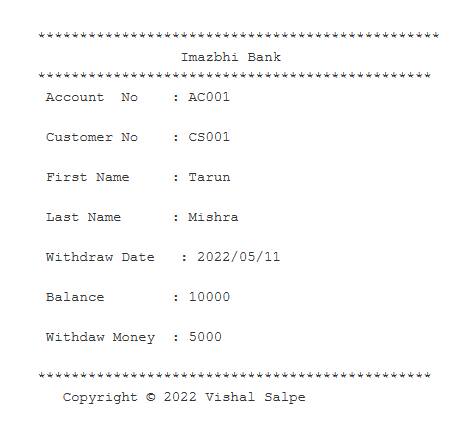


**4.3 Data Reports**

* Deposit Cash



* Withdraw Cash



**4.4 Sample program code**

package banksystem;

import java.awt.BorderLayout;

import java.awt.EventQueue;

import javax.swing.JFrame;

import javax.swing.JPanel;

import javax.swing.border.EmptyBorder;

import java.awt.Toolkit;

import java.awt.SystemColor;

import java.awt.Color;

import javax.swing.JLabel;

import javax.swing.JOptionPane;

import java.awt.Font;

import javax.swing.JTextField;

import javax.swing.JButton;

import java.awt.event.ActionListener;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.awt.event.ActionEvent;

import java.awt.event.KeyAdapter;

import java.awt.event.KeyEvent;

import java.awt.event.KeyListener;

public class SearchByName extends JFrame {

private JPanel contentPane;

private JTextField tAno;

private JTextField t1;

private JTextField bal;

public static void main(String[] args) {

EventQueue.invokeLater(new Runnable() {

public void run() {

try {

SearchByName frame = new SearchByName();

frame.setVisible(true);

} catch (Exception e) {

e.printStackTrace();

}

}

});

}

public SearchByName()

{

intialize();

connect();

tAno.setEditable(false);

t2.setEditable(false);

bal.setEditable(false);

}

Connection con;

PreparedStatement pstmt;

ResultSet rs;

private JTextField t2;

public void intialize()

{

setIconImage(Toolkit.getDefaultToolkit().getImage("F:\\Online project\\Banking-System-in-Core-Java-master\\Images\\save.gif"));

setTitle("Search Customer [By Name]");

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setBounds(100, 100, 416, 316);

contentPane = new JPanel();

contentPane.setBackground(SystemColor.text);

contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));

setContentPane(contentPane);

contentPane.setLayout(null);

JPanel panel = new JPanel();

panel.setBounds(10, 11, 380, 257);

contentPane.add(panel);

panel.setLayout(null);

JLabel lblAmount = new JLabel("Balance");

lblAmount.setBounds(10, 130, 122, 33);

panel.add(lblAmount);

lblAmount.setForeground(Color.BLACK);

lblAmount.setFont(new Font("Vani", Font.PLAIN, 18));

bal = new JTextField();

bal.setBounds(162, 130, 205, 29);

panel.add(bal);

bal.setColumns(10);

JLabel lblLastName = new JLabel("Last Name ");

lblLastName.setBounds(10, 96, 122, 23);

lblLastName.setForeground(Color.BLACK);

lblLastName.setFont(new Font("Vani", Font.PLAIN, 18));

panel.add(lblLastName);

t2 = new JTextField();

t2.setBounds(162, 91, 205, 29);

t2.setColumns(10);

panel.add(t2);

JLabel lblNewLabel = new JLabel("Account No. ");

lblNewLabel.setBounds(10, 16, 102, 23);

panel.add(lblNewLabel);

lblNewLabel.setForeground(Color.BLACK);

lblNewLabel.setFont(new Font("Vani", Font.PLAIN, 18));

tAno = new JTextField();

tAno.setForeground(Color.RED);

tAno.setBounds(162, 11, 205, 29);

panel.add(tAno);

tAno.setColumns(10);

JLabel lblCustomerName\_1 = new JLabel("First Name ");

lblCustomerName\_1.setBounds(10, 50, 122, 35);

panel.add(lblCustomerName\_1);

lblCustomerName\_1.setForeground(Color.BLACK);

lblCustomerName\_1.setFont(new Font("Vani", Font.PLAIN, 18));

t1 = new JTextField();

t1.setBounds(162, 51, 205, 29);

panel.add(t1);

/\*

t1.addKeyListener(new KeyAdapter() {

@Override

public void keyPressed(KeyEvent e) {

t1.addKeyListener ((KeyListener) new KeyAdapter()

{

public void keyTyped (KeyEvent ke)

{

char c = ke.getKeyChar ();

if (!((Character.isAlphabetic (c) || (c == KeyEvent.VK\_BACK\_SPACE))))

{

getToolkit().beep ();

ke.consume ();

}

}

} );

}

});\*/

t1.setColumns(10);

JButton btnSearch = new JButton("Search");

btnSearch.setBounds(36, 187, 122, 42);

panel.add(btnSearch);

btnSearch.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

try

{

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

{

JOptionPane.showMessageDialog (null, "Please! Provide Acccount Id.","BankSystem - EmptyField", JOptionPane.PLAIN\_MESSAGE);

t1.requestFocus();

}

String fname=t1.getText();

PreparedStatement pstmt=con.prepareStatement("select \* from account where Fname=?");

pstmt.setString(1,fname);

rs=pstmt.executeQuery();

if(rs.next()==false)

{

JOptionPane.showMessageDialog(null, "Account Not Found.... Please try again","Bank System",JOptionPane.ERROR\_MESSAGE);

tAno.setText("");

t1.setText("");

bal.setText("");

}

else

{

String Ano=rs.getString(2);

tAno.setText(Ano.trim());

String lname=rs.getString(5);

t2.setText(lname.trim());

String Amount=rs.getString(9);

bal.setText(Amount.trim());

}

}

catch(Exception ex)

{

ex.printStackTrace();

}

}

});

btnSearch.setFont(new Font("Times New Roman", Font.BOLD, 17));

JButton btnCancel = new JButton("Cancel");

btnCancel.setBounds(202, 187, 122, 42);

panel.add(btnCancel);

btnCancel.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

System.exit(0);

}

});

btnCancel.setFont(new Font("Times New Roman", Font.BOLD, 17));

}

public void connect()

{

try

{

Class.forName("com.mysql.jdbc.Driver");

con=DriverManager.getConnection("jdbc:mysql://localhost:3306/banksystem","root","");

}

catch(Exception sq)

{ s

q.printStackTrace();

}

}

}

**4.5 Limitations and Bibliography**

**Some other limitations system:**

* Less security of customer and bank information.
* Require more physical work and manpower.
* All the manual entry and editing will take more time.
* No level of clearance for the different levels of employees.
* Safety of paper documents from the disaster.
* No backup of the information.

By looking at disadvantages these are pretty serious for any banking system as they are capable of bringing down the whole system. Digitalization in the banking system, will not only achieve its goals and also will give some benefits like less manual calculation will be required.

**CONCLUSION** Our project is only a humble venture to satisfy the needs to manage their project work. Several user-friendly coding has also been adopted. This package shall prove to be powerful in satisfying all the requirements of the school. The objective of software planning is to provide a framework that enables the manager to make reasonable estimates made within a limited time frame at the beginning of the software project and should be updated regularly as the project

**Bibliography**

Suresh, P•&Paul, J(2012).Management of Banking and Financial Services; New Delhi, India.Pearson. Gordon, E• & Natrajan, K (2015) Banking: Theory Law and Practice, Mumbai.Himalaya Publishing House. Nigam. L (1985) Banking Law and Practice, NewDelhi, Vani Educational Books.• Yadav,S.K. (2015) Elements of Research Writing,New Delhi,UDH Publishers• & Distributors,Ltd.

**Project Guide Name**

**Signature**