

CORE JAVA PROJECT

PROJECT TITLE:

BANK MANAGEMENT SYSTEM

UNDER THE GUIDANCE OF :

INDRAKKA MALI

S.No.	Description
1.1.	INTRODUCTION
1.2.	Motivation
1.3.	Problem definition
1.4.	Objective of Project
1.5.	Limitations of project
2.	LITERATURE SURVEY
2.1.	Introduction
2.2.	Existing Banking System
2.3.	Disadvantages of Existing System
2.4.	Proposed System
2.5.	Conclusion
3.1	Introduction
3.2	Software requirement Specification
3.2.1.	Software requirement
3.2.2.	Hardware requirement
3.3.	Content diagram of Project
3.4	IMPLEMENTATION & RESULTS

INTRODUCTION

1.MOTIVATION

During the past several decades personnel function has been transformed from a relatively obscure record keeping staff to central and top level management function. There are many factors that have influenced this transformation like technological advances, professionalism, and general recognition of human beings as most important resources.

A computer based management system is designed to handle all the primary information required to calculate monthly statements of customer account which include monthly statement of any month. Separate database is maintained to handle all the details required for the correct statement calculation and generation.

This project intends to introduce more user friendliness in the various activities such as record updation, maintenance, and searching. The searching of record has been made quite simple as all the details of the customer can be obtained by simply keying in the identification or account number of that customer. Similarly, record maintenance and updation can also be accomplished by using the account number with all the details being automatically generated. These details are also being promptly automatically updated in the master file thus keeping the record absolutely up-to-date.

The entire information has maintained in the database or Files and whoever wants to retrieve can't retrieve, only authorization user can retrieve the necessary information which can be easily be accessible from the file. This system provides fast, efficient, reliable and User friendly interfaces in banking and has no chance of losing data while processing of user data i.e. customer account transactions. This software provides a good user interface such that a user of basic computer knowledge can operate the application. It also reduces effort done by the accountant and also reduces the load of real time computation. This software enables faster transaction like new account creation, withdrawal of cash from the account, deposit of cash to the account, checking account balance of the account holder even if there are large amount of data in the system database. A Bank is a financial institution which accepts deposits, pays interest on pre-defined rates, clears checks, makes loans, and often acts as an intermediary in financial transactions. It also provides other financial services to its customers.

2. OBJECTIVE OF THE PROJECT

A computer based management system is designed to handle all the primary information required to calculate monthly statements of customer account which include monthly statement of any month. Separate database is maintained to handle all the details required for the correct statement calculation and generation.

This project intends to introduce more user friendliness in the various activities such as record updation, maintenance, and searching. The searching of record has been made quite simple as all the details of the customer can be obtained by simply keying in the identification or account number of that customer. Similarly, record maintenance and updation can also be accomplished by using the account number with all the details being automatically generated. These details are also being promptly automatically updated in the master file thus keeping the record absolutely up-to-date.

The main objective of our project is providing the different typed of customers facility, the main objective of this system is to find out the actual customer service. Etc.

- It should fulfill almost all the process requirements of any Bank.
- It should increase the productivity of bank by utilizing the working hours more and more, with minimum manpower.

This project includes the entire upgraded feature required for the computerization banking system. This system is very easy to use, so that any user can use without getting pre-knowledge about this. Its very much user friendly and meet almost all daily working process requirements. This system is completely GUI based and can be use by mouse and as well as keyboard. This system is melded in such a way that has got all features to upgrade without making much change in existing components.

LITERATURE SURVEY

2.1 Introduction

SYSTEM ANALYSIS

System analysis is a process of gathering and interpreting facts, diagnosing problems and the information to recommend improvements on the system. It is a problem solving activity that requires intensive communication between the system users and system developers. System analysis or study is an important phase of any system development process. The system is studied to the minutest detail and analyzed. The system analyst plays the role of the interrogator and dwells deep into the working of the present system. The system is viewed as a whole and the input to the system are identified. The outputs from the organizations are traced to the various processes. System analysis is concerned with becoming aware of the problem, identifying the relevant and decisional variables, analyzing and synthesizing the various factors and determining an optimal or at least a satisfactory solution or program of action.

A detailed study of the process must be made by various techniques like interviews, questionnaires etc. The data collected by these sources must be scrutinized to arrive to a conclusion. The conclusion is an understanding of how the system functions. This system is called the existing system. Now the existing system is subjected to close study and problem areas are identified. The designer now functions as a problem solver and tries to sort out the difficulties that the enterprise faces. The solutions are given as proposals. The proposal is then weighed with the existing system analytically and the best one is selected. The proposal is presented to the user for an endorsement by the user. The

proposal is reviewed on user request and suitable changes are made. This is loop that ends as soon as the user is satisfied with proposal. Preliminary study is the process of gathering and interpreting facts, using the information for further studies on the system. Preliminary study is problem solving activity that requires intensive communication between the system users and system developers. It does various feasibility studies. In these studies a rough figure of the system activities can be obtained, from which the decision about the strategies to be

EXISTING SYSTEM

In the existing system the transactions are done only manually but in proposed system we have to computerize all the banking transaction using the software system.

DISADVANTAGES OF EXISTING SYSTEM

- Lack of security of data.
- More man power.
- Time consuming.
- Consumes large volume of pare work.
- Needs manual calculations.
- No direct role for the higher officials.
- Damage of machines due to lack of attention.

To avoid all these limitations and make the working more accurately the system needs be computerized.

PROPOSED SYSTEM

The aim of proposed system is to develop a system of improved facilities. The proposed system can overcome all the limitations of the existing system . The system proper security and reduces the manual work.

ADVANTAGES OF THE PROPOSED SYSTEM

The system is very simple in design and to implement . The system requires very low resources and the system will work in almost all configurations . It has got
Security of data

- Proper control of the higher officials.
- Reduce the damages of the machines.
- Minimize manual data entry.
- Minimum time needed for the various processing.
- Greater efficiency
- .•Better service.
- User friendliness and interactive

CONCLUSION

In many ways, consumer banking is like other types of consumer activity. But banking customers expect more than an excellent mix of products: they are looking for superior customer experiences that fulfill basic expectations while providing added value.

In our survey, customers selected “the way I am treated” as the second most important reason for trusting their banking provider, trailing only the predictable “financial stability” of their bank. Customer experience is also the most common reason for opening and closing accounts, more so than fees, rates, locations and convenience.

This idea of trust is what transforms customers from static sources of revenue into advocates for their banks. In an era where social and digital media enable consumers to immediately share their experiences, customers who trust their bank will drive the most referrals and be more willing to consolidate their banking needs with a single financial services provider. This makes them the growth engines of any bank.

Creating customized banking

Each of the [eight customer segments](#) shares common behaviors and expectations when it comes to their banking experience. By focusing on the type of customer rather than the number of customers, banks can build a reputation for excellent customer service.

Although crafting common strategies can be more efficient, banks that approach each customer the same way risk offering individual customers the wrong products, services and advice across less-effective channels. To optimize investments in customer experience, banks should deploy segment-based strategies that take advantage of these inherent similarities – but also their differences.

Competition from all sides

The good news is that consumer confidence in the banking industry is on the rise, with 93% survey respondents reporting moderate or complete trust in their banks. Likewise, 77% of customers are satisfied enough with their banking relationship to recommend their primary provider. The global economic recovery appears to be taking hold, and banks are among the beneficiaries.

The challenge, however, is that an increasing number of financial service providers are competing for the same customers. Emerging technology and the increasing use of mobile.

3.2 SOFTWARE REQUIREMENT SPECIFICATION

SYSTEM SPECIFICATION

HARDWARE REQUIREMENTS

Processor : X86 Compatible processor with 1.7 GHz Clock speed

RAM : 512 MB or more

Hard disk : 20 GB or more

Monitor : VGA/SVGA

Keyboard : 104 Keys

Mouse : 2 buttons/ 3 buttons

SOFTWARE REQUIREMENTS

Operating System : Windows 2000/XP

Front end : Java Eclipse IDE

Back end : SQL Server 2005

SOURCE CODE:

Bank Connection Class:

This class is used for connecting MySQL with Eclipse IDE

```
package com.prog;

import java.sql.Connection;
import java.sql.DriverManager;

public class BankConnection {

    private static Connection conn=null;

    private static String driver="com.mysql.cj.jdbc.Driver";

    private static String
url="jdbc:mysql://localhost:3306/banksystem";

    private static String un="root";

    private static String up="root";

    public static Connection getConnection() {
        try {
            Class.forName(driver);

            conn=DriverManager.getConnection(url,un,up);

            if(conn==null) {

                System.out.println("Connection Error!");
            }
        }
    }
}
```

```
        }  
    }catch(Exception e) {  
        e.printStackTrace();  
    }  
  
    return conn;  
}
```

Bank Employee Class :

This is used for adding functionality for the admin

```
package com.prog;  
  
import java.sql.Connection;  
  
import java.sql.PreparedStatement;  
  
import java.sql.ResultSet;  
  
import java.sql.SQLException;  
  
import java.util.Scanner;  
  
public class BankEmployee {  
  
    private static Connection conn;  
  
    private static ResultSet rs;  
  
    private static PreparedStatement pst;
```

```
public static void addAccountNumber() throws SQLException {

    conn=BankConnection.getConnection();

    Scanner sc = new Scanner(System.in);

    String acc = null;

    String cname,mname,lname,hno,village,pin,pno;

    float inamt;

    System.out.println("Enter customer name");

    cname=sc.next();

    System.out.println("Enter customer house number");

    hno=sc.next();

    System.out.println("Enter customer city name");

    village=sc.next();

    System.out.println("Enter pin code");

    pin=sc.next();

    System.out.println("Enter the phone number");

    pno=sc.next();

    System.out.println("Enter the initial amount you want to
deposit");

    inamt=sc.nextFloat();

    System.out.println("Assign a new account number for the
user.");
```

```
acc=sc.next();
```

```
String s="insert into account values(?,?,?,?,?,?,?)";
```

```
pst=conn.prepareStatement(s);
```

```
pst.setString(1, acc);
```

```
pst.setString(2, cname);
```

```
pst.setString(3, pno);
```

```
pst.setString(4, hno);
```

```
pst.setString(5, village);
```

```
pst.setString(6, pin);
```

```
pst.setFloat(7, inamt);
```

```
int rv=pst.executeUpdate();
```

```
if(rv>0) {
```

```
    System.out.println("Record inserted Successfully.");
```

```
}
```

```
else {
```

```
    System.out.println("Unable to insert record");
```

```
}
```

```

    }

    public static void deleteRecord() throws SQLException {

        conn=BankConnection.getConnection();

        String acc;

        Scanner sc = new Scanner(System.in);

        System.out.println("Enter the account number to delete
record");

        acc=sc.next();

        String sel="Select * from account where acc_no=?";

        pst=conn.prepareStatement(sel);

        pst.setString(1, acc);

        rs=pst.executeQuery();

        if(rs.next()) {

            String del="delete from account where acc_no=?";

            pst=conn.prepareStatement(del);

            pst.setString(1, acc);

            int rv=pst.executeUpdate();

            if(rv>0) {

                System.out.println("Record is deleted");

            }else {

```

```

        System.out.println("ERROR!!!!");
    }
    }else {
        System.out.println("Student id "+acc+" not
exists");
    }

}

public static void updateRrecord() throws SQLException {

    conn=BankConnection.getConnection();

    String cname,acc,phone;

    Scanner sc= new Scanner(System.in);

    System.out.println("Enter customer account number to update
record");

    acc=sc.next();

    String s ="select * from account where acc_no=?";

    pst=conn.prepareStatement(s);

    pst.setString(1,acc);

    rs=pst.executeQuery();

    if(rs.next()) {

```

```
System.out.println("Enter the operation you want to perform");

System.out.println("1. To update name of the customer");

System.out.println("2. to update Phone number");

        int input;

        input = sc.nextInt();

        switch(input) {

            case 1:

                System.out.println("Enter the name to change");

                cname=sc.next();

String upd="update account set cname=? where acc_no=?";

                pst=conn.prepareStatement(upd);

                pst.setString(1,cname);

                pst.setString(2, acc);

                int rv=pst.executeUpdate();

                if(rv>0) {

System.out.println("Name is changed succesfully");

                    }else {

                        System.out.println("Error!!");

                    }

                break;
```

```
}
```

```
case 2:
```

```
System.out.println("Enter the phone number you want to update");
```

```
phone=sc.next();
```

```
String upd1="update account set phone=? where acc_no=?";
```

```
pst=conn.prepareStatement(upd1);
```

```
pst.setString(1, phone);
```

```
pst.setString(2, acc);
```

```
int rv1=pst.executeUpdate();
```

```
if(rv1>0) {
```

```
System.out.println("Phone number is changed succesfully");
```

```
}else {
```

```
System.out.println("Error!!");
```

```
}
```

```
break;
```

```
}
```

```
}
```

```
}
```

```
public static void retrieveRecords() throws SQLException {
```



```

        conn = BankConnection.getConnection();

        String s="select * from account";

        pst=conn.prepareStatement(s);

        rs=pst.executeQuery();


        System.out.println("acc_no\t\tcustomer_name\tphone_no\thouse_no\t
city\t\tzipcode\t\tamount");

        while(rs.next()) {

            String ac=rs.getString("acc_no");

            String cname=rs.getString("cname");

            String phone=rs.getString("phone");

            String hno=rs.getString("house_no");

            String dist = rs.getString("city");

            String pin=rs.getString("zipcode");

            int amt=rs.getInt("amount");

            System.out.println(ac+"\t\t"+cname+"\t\t"+phone+"\t"+hno+"\t\t"+dist+"
\t\t"+pin+"\t\t"+amt);

        }

        System.out.println("Error!!!");

    }

```

Bank Main Class:

This class is responsible for running all the classes

```
package com.prog;

import java.sql.SQLException;

import java.util.Scanner;

public class BankMain {

    public static void main(String[] args) throws SQLException {

        Scanner sc = new Scanner(System.in);

        int choice,c2,c3;

        char ch;

        String upass,uname;

        while(true) {

            System.out.println("_____WELCOME_____");

            System.out.println("Enter your choice from below");

            System.out.println("1.For Customer");

            System.out.println("2.For Bank Staff");

            choice=sc.nextInt();

            switch(choice) {

                case 1:

                    while(true) {

                        System.out.println("*****__WELCOME__*****");

                        System.out.println("1. Display the account details");
```

```
System.out.println("2. Witihdraw Amount");

System.out.println("3. Deposit Amount");

c2=sc.nextInt();

switch(c2) {

case 1:

System.out.println("Enter your account number to view account
details:");

Customer.showDetails();

        break;

        case 2:

                System.out.println("Withdrawl Section");

                Customer.withdrawAmount();

                break;

case 3:

                System.out.println("Deposit Section");

                Customer.depositAmount();


                default: System.out.println("Invalid input");

                        break;

}

System.out.println("Do you want to continue? Y/N");

        char ch1=sc.next().charAt(0);

        if(ch1=='n' || ch1=='N') {

                break;
```

```

    }

    }

    break;

case 2:

System.out.println("Welcome to the login window");

System.out.println("Enter username:");

uname=sc.next();

System.out.println("Enter your password:");

upass=sc.next();

    if(uname.equalsIgnoreCase("Varsha123")&&upass.equals("varsha@100"
)) {

        while(true) {

System.out.println("_***WELCOME***_");

System.out.println("Choose the options you want to perform:");

System.out.println("1. Open new account.");

System.out.println("2. Delete existing details.");

System.out.println("3. Update the Record as per request.");

System.out.println("4. Retrieve all records.");

c3=sc.nextInt();

switch(c3) {

case 1:

System.out.println("Fill the following details correctly to open an

BankEmployee.addAccountNumber();

        break;

```

```
case 2:

System.out.println("Enter account number to delete record.");

BankEmployee.deleteRecord();

break;

case 3:

System.out.println("Update existing records.");

BankEmployee.updateRrecord();

break;

case 4:

System.out.println("To view all records.");

BankEmployee.retriveRecords();

break;

default:

System.out.println("Invalid Input");

break;

    }

System.out.println("Do you want to continue?");

char inp=sc.next().charAt(0);

if(inp=='n' || inp=='N') {

break;

}

System.out.println("You have been logged out.");
```

```

        }
    }else {
        System.out.println("Incorrect username or password! ");
    }

    break;

default:

    System.out.println("Invalid Input");

    break;

}

System.out.println("Do you want to continue y/n?");

char ch2 = sc.next().charAt(0);

if(ch2=='n' || ch2=='N') {

    break;

}

}

}

}

}

```

Customer Class:

This class is responsible for the customer functionality.

```

package com.prog

import java.sql.Connection;

import java.sql.PreparedStatement;

```

```

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.Scanner;


public class Customer {

    public static Connection conn;

    public static ResultSet rs;

    public static PreparedStatement pst;

    public static void showDetails() throws SQLException {

        conn = BankConnection.getConnection();

        Scanner sc = new Scanner(System.in);

        System.out.println("Enter your account number to view your account
        details ");

        String acc=sc.next();

        String s="select * from account where acc_no=?";

        pst=conn.prepareStatement(s);

        pst.setString(1, acc);

        rs=pst.executeQuery();

        if(rs.next()) {

            System.out.println("acc_no\t\tcustomer_name\tphone_no\thouse_no\t
            city_or_village\tpincode\tamount");

            String ac=rs.getString("acc_no");

            String cname=rs.getString("cname");

            String phone=rs.getString("phone");

```

```

        String hno=rs.getString("house_no");

        String dist = rs.getString("city");

        String pin=rs.getString("pincode");

        float amt=rs.getFloat("amount");

        System.out.println(ac+"\t\t"+cname+"\t"+phone+"\t"+hno+"\t"+dist+"\t"+
        pin+"\t"+amt);

    }

    System.out.println("Account not found");

}

public static void withdrawAmount() throws SQLException {

    conn = BankConnection.getConnection();

    float wamount;

    float a=0;

    Scanner sc = new Scanner(System.in);

    System.out.println("Enter the account number");

    String acc=sc.next();

    System.out.println("Enter the amount you want to withdraw");

    wamount=sc.nextFloat();

    String s = "select * from account where acc_no=?";

    pst=conn.prepareStatement(s);

    pst.setString(1,acc);

    rs=pst.executeQuery();

    if(rs.next()) {

```



```
        a=rs.getInt("amount");

        a=a-wamount;

        String upd="update account set amount=? where acc_no=?";

        pst=conn.prepareStatement(upd);

        pst.setFloat(1, a);

        pst.setString(2, acc);

        int rv=pst.executeUpdate();

        if(rv>0) {

System.out.println("Transaction successful");

System.out.println("Remaining account balance is :"+a);

        }else {

            System.out.println("Insufficient Balance");

        }

        }else {

            System.out.println("Account number not found!!");

        }

    }

}
```

```
public static void depositAmount() throws SQLException {

    conn = BankConnection.getConnection();

    float damount,a = 0;

    Scanner sc = new Scanner(System.in);

    System.out.println("Enter the account number");

    String acc=sc.next();

    System.out.println("Enter the amount you want to Deposit");

    damount=sc.nextFloat();

    String s = "select * from account where acc_no=?";

    pst=conn.prepareStatement(s);

    pst.setString(1, acc);

    rs=pst.executeQuery();

    if(rs.next()) {

        a=rs.getFloat("amount");

    }

    a=a+damount;

    damount=a;

    String upd="update account set amount=? where acc_no=?";

    pst=conn.prepareStatement(upd);

    pst.setFloat(1, a);

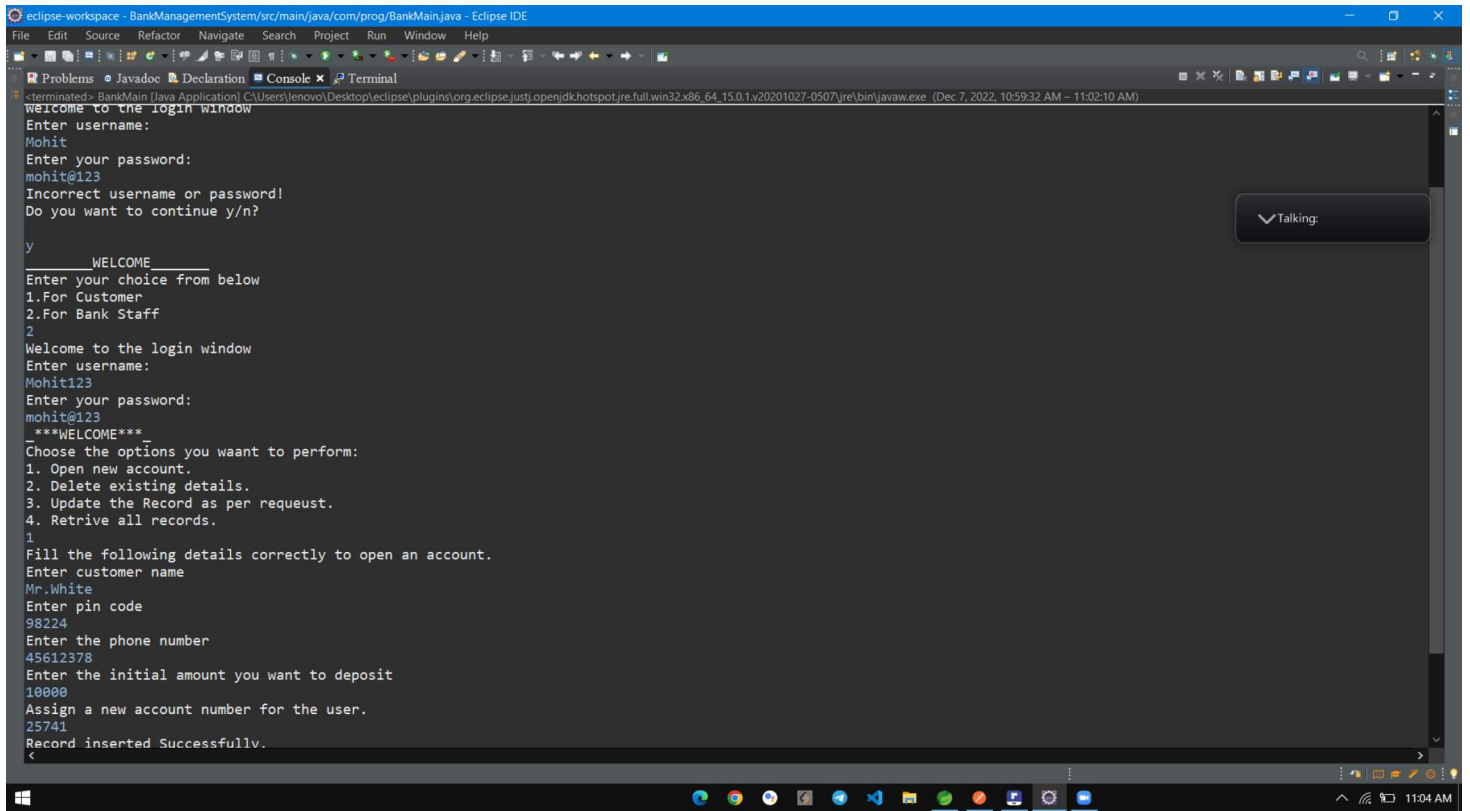
    pst.setString(2, acc);

    int rv=pst.executeUpdate();

}
```

```
        if(rv>0) {  
System.out.println("Transaction successful");  
System.out.println("Remaining account balance is :"+a);  
        }  
        }else {  
            System.out.println("Account holder not found!!");  
        }  
    }  
}
```

OUTPUT

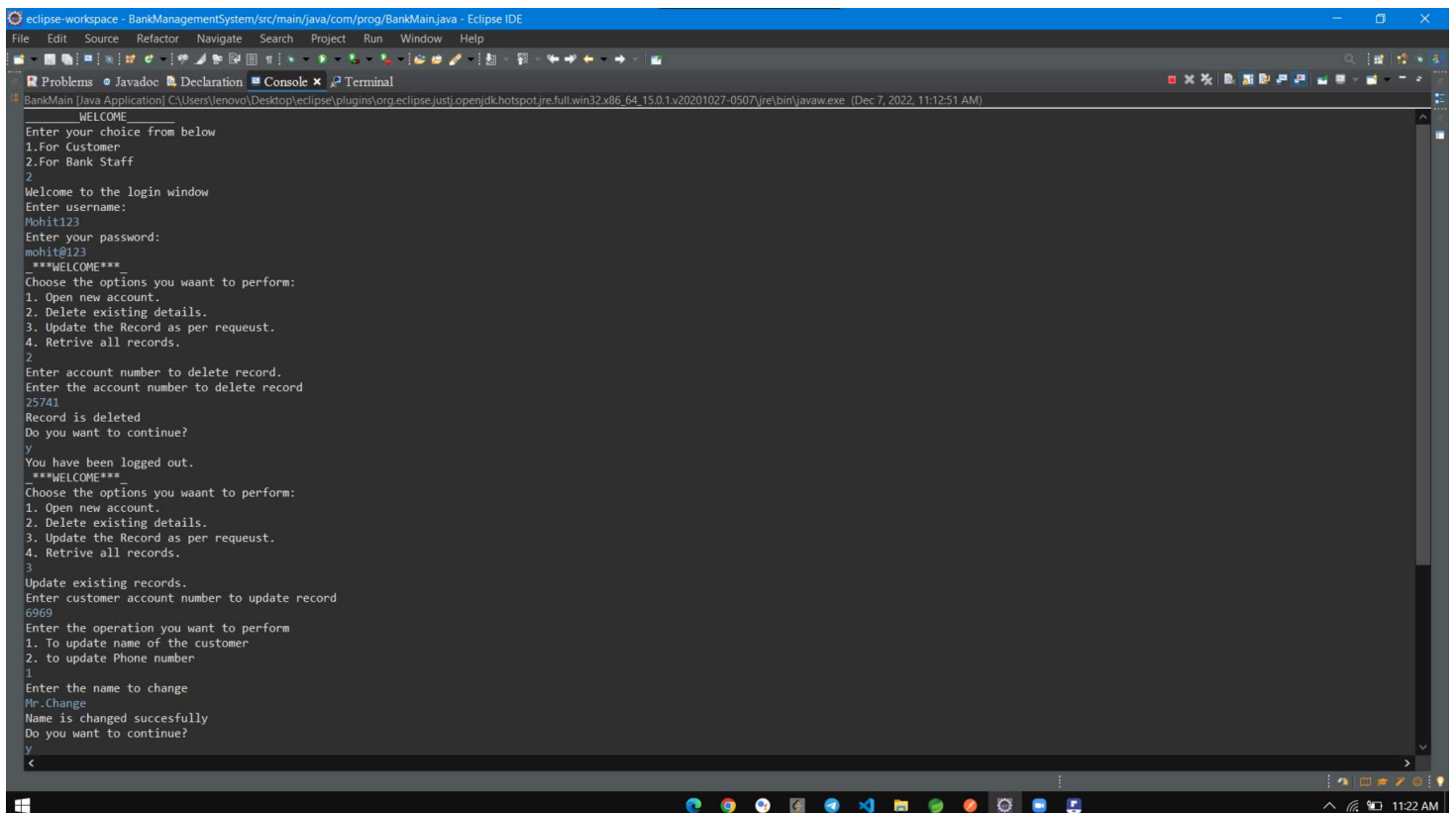


```
eclipse-workspace - BankManagementSystem/src/main/java/com/prog/BankMain.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help

terminated: BankMain [Java Application] C:\Users\lenovo\Desktop\eclipse\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_15.0.1.v20201027-0507\jre\bin\javaw.exe (Dec 7, 2022, 10:59:32 AM - 11:02:10 AM)

Welcome to the login window
Enter username:
Mohit
Enter your password:
mohit@123
Incorrect username or password!
Do you want to continue y/n?
y
WELCOME
Enter your choice from below
1.For Customer
2.For Bank Staff
2
Welcome to the login window
Enter username:
Mohit123
Enter your password:
mohit@123
***WELCOME***
Choose the options you want to perform:
1. Open new account.
2. Delete existing details.
3. Update the Record as per request.
4. Retrive all records.
1
Fill the following details correctly to open an account.
Enter customer name
Mr.White
Enter pin code
98224
Enter the phone number
45612378
Enter the initial amount you want to deposit
10000
Assign a new account number for the user.
25741
Record inserted Successfully.
<
```

BANK EMPLOYEE INSERTING VALUES



```
eclipse-workspace - BankManagementSystem/src/main/java/com/prog/BankMain.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help

BankMain [Java Application] C:\Users\lenovo\Desktop\eclipse\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_15.0.1.v20201027-0507\jre\bin\javaw.exe (Dec 7, 2022, 11:12:51 AM)

WELCOME
Enter your choice from below
1.For Customer
2.For Bank Staff
2
Welcome to the login window
Enter username:
Mohit123
Enter your password:
mohit@123
***WELCOME***
Choose the options you want to perform:
1. Open new account.
2. Delete existing details.
3. Update the Record as per request.
4. Retrive all records.
2
Enter account number to delete record.
Enter the account number to delete record
25741
Record is deleted
Do you want to continue?
y
You have been logged out.
***WELCOME***
Choose the options you want to perform:
1. Open new account.
2. Delete existing details.
3. Update the Record as per request.
4. Retrive all records.
3
Update existing records.
Enter customer account number to update record
6969
Enter the operation you want to perform
1. To update name of the customer
2. to update Phone number
1
Enter the name to change
Mr.Change
Name is changed succesfully
Do you want to continue?
y
<
```

DELETING RECORD

```

***WELCOME***
Choose the options you want to perform:
1. Open new account.
2. Delete existing details.
3. Update the Record as per request.
4. Retrieve all records.
3
Update existing records.
Enter customer account number to update record
6969
Enter the operation you want to perform
1. To update name of the customer
2. To update Phone number
1
Enter the name to change
Mr.Change
Name is changed successfully
Do you want to continue?
y
You have been logged out.
***WELCOME***
Choose the options you want to perform:
1. Open new account.
2. Delete existing details.
3. Update the Record as per request.
4. Retrieve all records.
4
To view all records.

```

acc_no	customer_name	pincode	amount
2248	Hector	1001001	20202020
4560	Sandstorm	590006	10850
4620	Him	59887	1500
6969	Mr.Change	590004	8000
69420	Tyler Durden	69420	51

```

Do you want to continue?

```

UPDATING EXISTING RECORDS

```
eclipse-workspace - BankManagementSystem/src/main/java/com/prog/BankMain.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Problems Javadoc Declaration Console x Terminal
<terminated> BankMain [Java Application] C:\Users\lenovo\Desktop\eclipse\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_15.0.1.v20201027-0507\jre\bin\javaw.exe (Dec 7, 2022, 11:08:01 AM - 11:09:16 AM)

Enter your choice from below
1.For Customer
2.For Bank Staff
1
***** _WELCOME_ *****
1. Display the account details
2. Withdraw Amount
3. Deposit Amount
1
Enter your account number to view account details:
25741
acc_no      customer_name  phone_no  pincode    amount
25741      Mr.White         45612378  98224      10000.0
Do you want to continue? Y/N
y
***** _WELCOME_ *****
1. Display the account details
2. Withdraw Amount
3. Deposit Amount
2
Withdrawal Section
Enter the account number
25741
Enter the amount you want to withdraw
500
Transaction successful
Remaining account balance is :9500.0
Do you want to continue? Y/N
y
***** _WELCOME_ *****
1. Display the account details
2. Withdraw Amount
3. Deposit Amount
3
Deposit Section
Enter the account number
25741
Enter the amount you want to Deposit
5000
Transaction successful
Remaining account balance is :14500.0
Do you want to continue? Y/N
n
<
```

CUSTOMER SCREEN FOR INTERACTING WITH BANK

Sample test cases

For Admin

- **Verify Admin login with valid and Invalid data**
- **verify admin login without data**
- **verify all admin home links**
- **verify admin change password with valid and invalid data**
- **verify admin change password without data**
- **Verify admin change password with existing data**
- **verify admin logout**

For new Branch

- **create a new branch with valid and invalid data**
- **create a new branch without data**
- **create a new branch with existing branch data**
- **verify reset and cancel option**
- **Update branch with valid and invalid data**
- **update branch without data**
- **update branch with existing branch data**
- **Verify cancel option**
- **verify branch deletion with and without dependencies**
- **Verify branch search option**

For New Role

- **create a new role with valid and invalid data**

- **create a new role without data**

- **verify new role with existing data**

- **verify role description and role types**

- **Verify cancel and reset option**

- **Verify role deletion with and without dependency**

- **verify links in role details page**

For customer & Visitors

- **Verify all visitor or customer links**

- **Verify customers login with valid and invalid data**

- **Verify customers login without data**

- **Verify bankers login without data**

- **Verify bankers login with valid or invalid data**

For New users

- **create a new user with valid and invalid data**

- **create a new user without data**

- **create a new user with existing branch data**

- **verify cancel and reset option**

- **Update user with valid and invalid data**

- **update user with existing data**

- **verIFy cancel option**

- **vefiry deletion of the user**

Challenges in testing Banking domain & their Mitigation

Challenges tester might face during testing banking domain are

Challenge	Mitigation
<ul style="list-style-type: none">Getting access to production data and replicating it as test data, for testing is challenging	<ul style="list-style-type: none">Ensure that test data meets regulatory compliances requirements and guidelinesMaintain the data confidentiality by following techniques like data masking, synthetic test data, testing system integration, etc.
<ul style="list-style-type: none">The biggest challenge in testing banking system is during the migration of the system from the old system to the new system like testing of all the routines, procedures and plans. Also how the data will be fetched, uploaded and transferred to the new system after migration	<ul style="list-style-type: none">Ensure Data Migration Testing is completeEnsure Regression Test cases are executed on old and new systems, and the results match.
<ul style="list-style-type: none">There may be the cases where requirements are not documented well and may lead to functional gaps in test planMany non-functional requirements are not fully documented, and testers do not know whether to test it or not	<ul style="list-style-type: none">The test should participate in the project right from Requirement Analysis phases and should actively review the Business Requirements
<ul style="list-style-type: none">The most important point is to check whether the said system follows the desired policies and procedures	<ul style="list-style-type: none">Compliance or Regulatory Policies testing must be done
<ul style="list-style-type: none">The scope and the timelines increases as banking application are integrated with other application like internet or mobile banking	<ul style="list-style-type: none">Ensure Time budget for Integration testing is accounted if your banking application has many external interfaces

Summary

Banking domain is the most vulnerable area for cyber-theft, and safeguarding the software requires precise testing. This tutorial gives a clear idea of what it takes for banking domain testing and how important it is. One must understand that -

- Majority of banking software are developed on **Mainframe** and **Unix**
- Testing helps to lessen possible glitches encounter during software development
- Proper testing and compliance to industry standards, save companies from penalties
- Good practices help develop good results, reputation and more business for companies
- Both manual and automated testing have respective merits and usability

OUTPUT TESTING OR USER ACCEPTANCE TESTING

The system considered is tested for user acceptance; here it should satisfy the firm's need. The software should keep in touch with perspective system; user at the time of 13 developing and making changes whenever required. This done with respect to the following points

Input Screen Designs,

Output Screen Designs,⌘

Online message to guide the user and the like.⌘

The above testing is done taking various kinds of test data. Preparation of test data plays a vital role in the system testing. After preparing the test data, the system under study is tested using that test data. While testing the system by which test data errors are again uncovered and corrected by using above testing steps and corrections are also noted for future use.

For testing banking applications, different stages of testing include

- **Requirement Analysis:** It is done by business analyst; requirements for a particular banking application are gathered and documented
- **Requirement Review:** Quality analysts, business analysts, and development leads are involved in this task. The requirement gathering document is reviewed at this stage, and cross-checked to ensure that it does not affect the workflow
- **Business Requirements Documentation:** Business requirements documents are prepared by quality analysts in which all reviewed business requirements are covered