Abstract:

The **Banking System** is a simple web-application in which an admin user can create customers, create accounts, update details of the account, like adding funds, removing funds. This application uses **J2EE** to work in backend and for frontend we used **JSP (Java Server Pages)** and uses **MySQL** for Database and **JDBC** To Connect with the Database

Introduction:

Building Banking Software is a complex job to undertake, so, by using technologies like J2EE we can efficiently build a Web Application for Banking System. So, this project demonstrates the basic system of banking software. Using technologies like JDBC, JSP, Servlets, MySQL and other Underlying Technologies. We have used HTML, CSS and Some JavaScript to make this project possible and we also used some frameworks like Bootstrap and others to build User Interface on Web Platform

Technologies Used:

J2EE:

which stands for Java 2 Platform, Enterprise Edition, is a set of specifications and technologies developed by Sun Microsystems for building and deploying enterprise-level applications using the Java programming language. J2EE provides a robust, scalable, and platform-independent framework for developing distributed, multitiered, and highly secure applications

- **Architecture:** J2EE is based on a multitiered architecture, where the application is typically divided into three main tiers: the client tier, the middle

tier, and the data tier. This architecture allows for scalability, maintainability, and flexibility.

- **Servlets:** Servlets are Java classes that run on the server and respond to client requests, typically for generating dynamic web content. They are a fundamental part of the J2EE platform for building web applications.

- **JSP:** Java Server Pages (JSP) allow developers to embed Java code within HTML templates. They are often used to create dynamic web pages and can interact with Servlets and other components

- **Platform Independence:** One of the key advantages of J2EE is platform independence. Java applications can run on different operating systems and hardware platforms if there is a J2EE-compliant application server available.

JDBC:

JDBC Stands for Java Database Connectivity, it is used for connecting Java Applications to Database such as MySQL, Oracle, PostgreSQL, Microsoft SQL Server, and more.

- **Database Connectivity:** JDBC provides a means to establish a connection between a Java application and a relational database. It allows the application to send SQL (Structured Query Language) statements to the database for various operations, such as retrieving data, updating records, and executing stored procedures.

MySQL:

It is a Relational database management System. Used to store data in table format. It is an open-source database management system (RDBMS) that is widely used for

managing and storing structured data. Developed by Oracle Corporation, MySQL is known for its reliability, performance, and ease of use.

Operations:

1. Creating Customer, and his Associated Account by Generating Random Number for Account Number
2. Searching Customers Account Using Account Number, Updating Balance like Debiting, Crediting the amount into database
3. CRUD (Create, Read, Update, Delete) Operation using Banking System

Code:

Login:

```
protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

// TODO Auto-generated method stub

String name = request.getParameter("uname");

String psw = request.getParameter("psw");

HttpSession session = request.getSession();

PrintWriter out = response.getWriter();

RequestDispatcher dispatcher = null;

try {

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

Connectionconn = DriverManager.getConnection("jdbc:mysql://localhost:3306/bank", "root", "Shivu@1102");

if(conn != null)

{

out.print("Connection Succesfull");

}

else
```

```
{
out.print("Connection Failed");
}


// Create the SQL query with a prepared statement
String sql = "select * from admin where name = ? and psw = ?;";
PreparedStatement preparedStatement = conn.prepareStatement(sql);
preparedStatement.setString(1, name);
preparedStatement.setString(2, psw);
// Execute the query
ResultSet re = preparedStatement.executeQuery();
dispatcher = request.getRequestDispatcher("cus_create.jsp");
        if(re.next())
        {
session.setAttribute("name", name);
dispatcher = request.getRequestDispatcher("admindash.jsp");
        }
        else
        {
request.setAttribute("status", "failed");
dispatcher = request.getRequestDispatcher("login.jsp");
        }
        conn.close();
        dispatcher.forward(request, response);
}
catch (Exception e)
{
```

```
e.printStackTrace();
}
```

Code To Create Customer and Account:

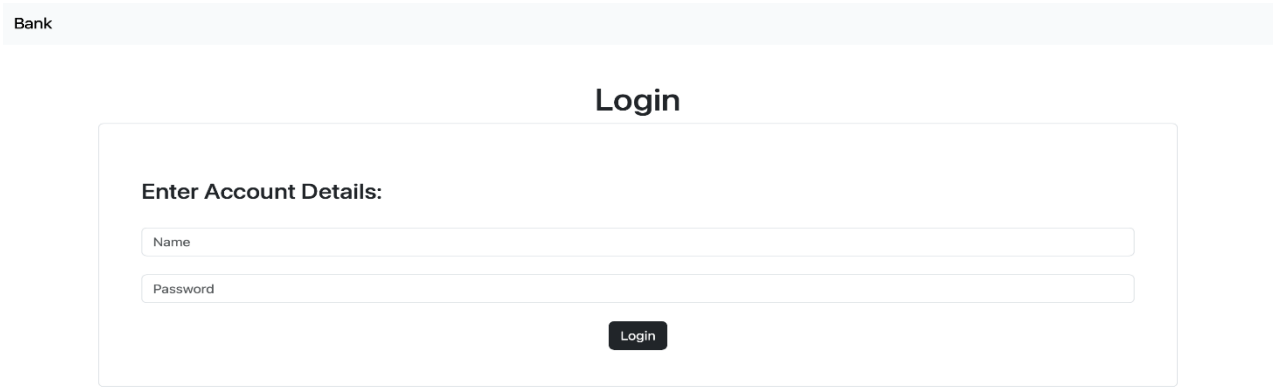
```
try {
Class.forName("com.mysql.cj.jdbc.Driver");
Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/bank", "root", "Shivu@1102");
if(conn != null)
{
out.print("Connection Succesfull");
}
else
{
out.print("Connection Failed");
}

// Create the SQL query with a prepared statement
String sql = "INSERT INTO customer (name, phone, address, DOB, created) VALUES (?, ?, ?, ?, ?);";
String stmt = "select id from customer where phone=?;";

PreparedStatement preparedStatement = conn.prepareStatement(sql);
preparedStatement.setString(1, name);
```

```
preparedStatement.setString(2, phone);
preparedStatement.setString(3, address);
preparedStatement.setDate(4, java.sql.Date.valueOf(dob));
preparedStatement.setDate(5, java.sql.Date.valueOf(doc));


// Execute the query
int rowsAffected = preparedStatement.executeUpdate();
PreparedStatement stmt1 = conn.prepareStatement(stmt);
stmt1.setString(1, phone);
ResultSet re = stmt1.executeQuery();
if(re.next())
{
int id = re.getInt("id");

String sql1 = "insert into account(cusid, account_number, type, balance) values(?, ?, ?, ?);";

PreparedStatement stm = conn.prepareStatement(sql1);

int n = getRandomNumber();
stm.setInt(1, id);
stm.setInt(2, n);
stm.setString(3, type);
stm.setFloat(4, (float) 0.0);

int act = stm.executeUpdate();
```

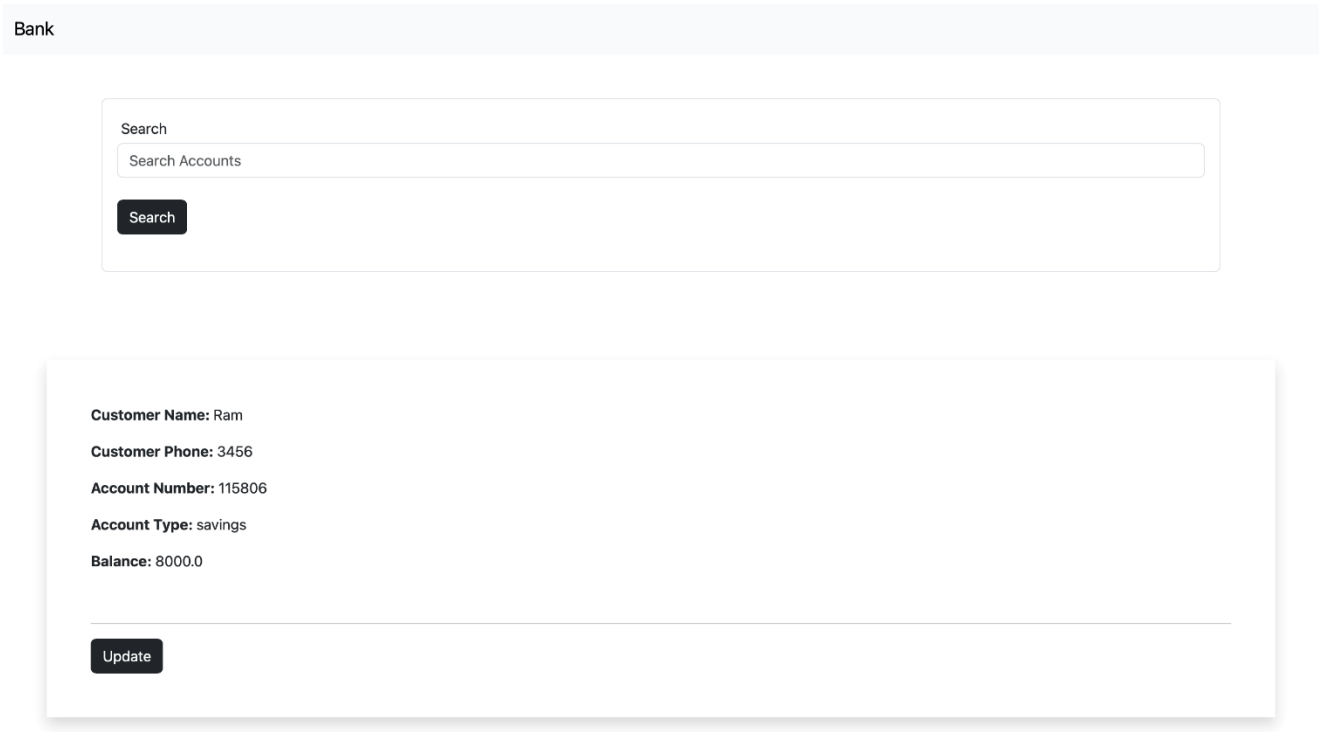
request.setAttribute("account", n);

Screenshots:

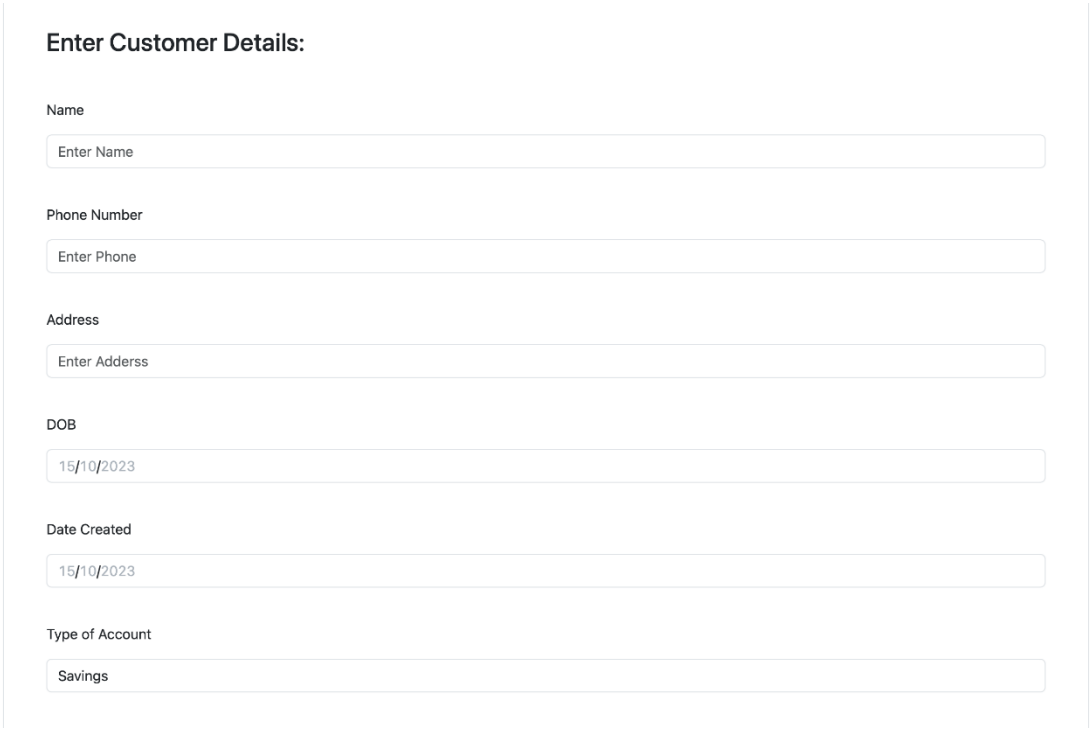
Login Page:



Search Page:



Customer Creation and Account Creation Page:



Account Transaction Update (Credit/Debit) Page:

Bank

Account Number: 63961

Account Type: savings

Balance: 7800.0

Update

Transaction:

63961

7800.0

Enter Amount

Select