GUJARAT TECHNOLOGICAL UNIVERSITY

Government Engineering College, Bhavnagar

Subject: **Advanced Java Programming**

B.E. C.E. Semester-6th

(Computer Branch)

Submitted By:

Name: Vankani Arjun BakulBhai

Enrollment: 180210107060

Prof. Vishal Andodariya
(Faculty Guide)

Prof. KARSHAN KANDORIYA
(Head of the Department)

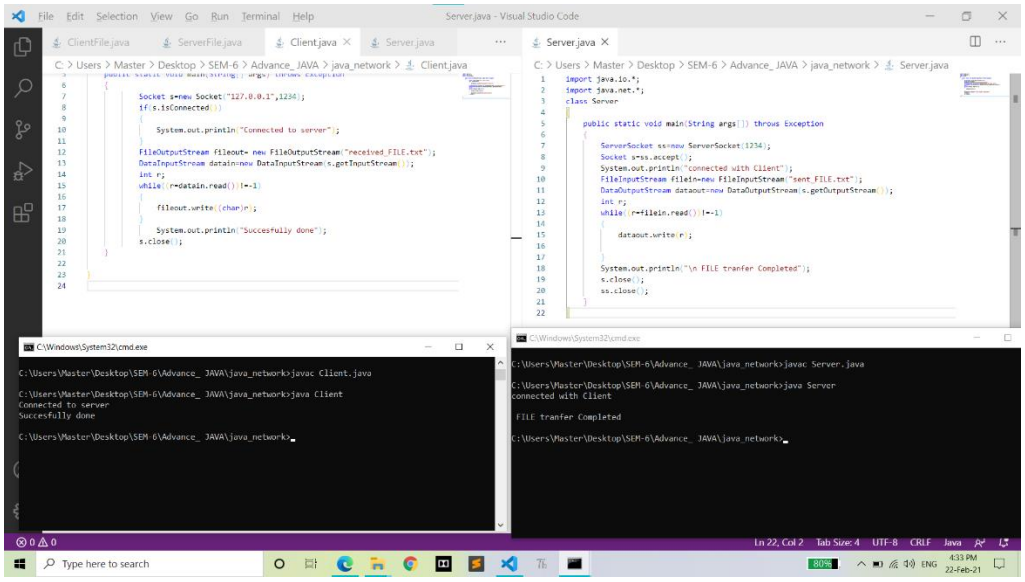
Academic Year (2020-21)

Index

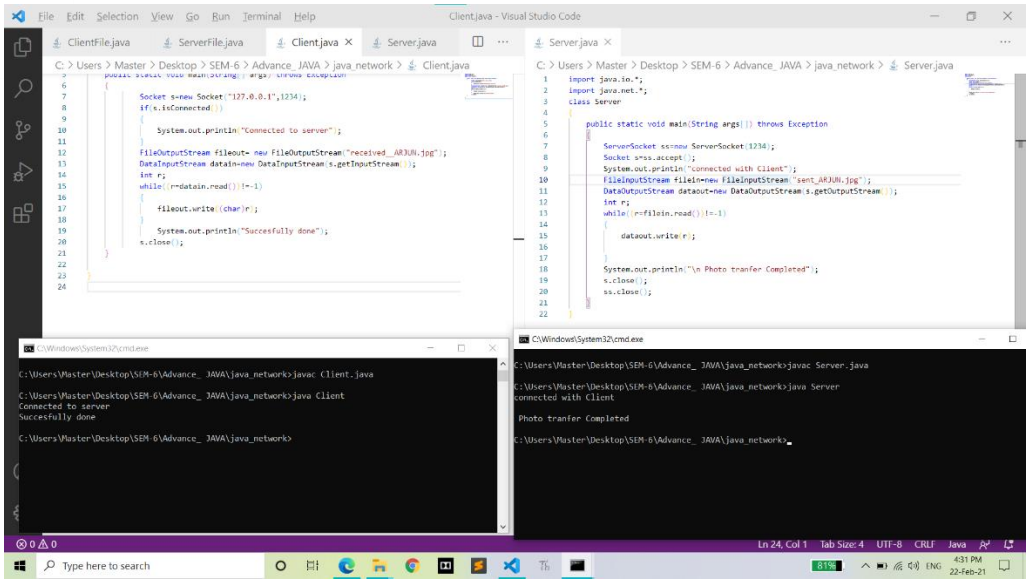
Advanced Java Programming
(Lab Session-1)

Practical No: 1) Implement TCP Server for transferring files using Socket and Server Socket

File Transfer with Socket IO



- **Photo Transfer**



File Transferred Successfully

Code:

1)Server side:

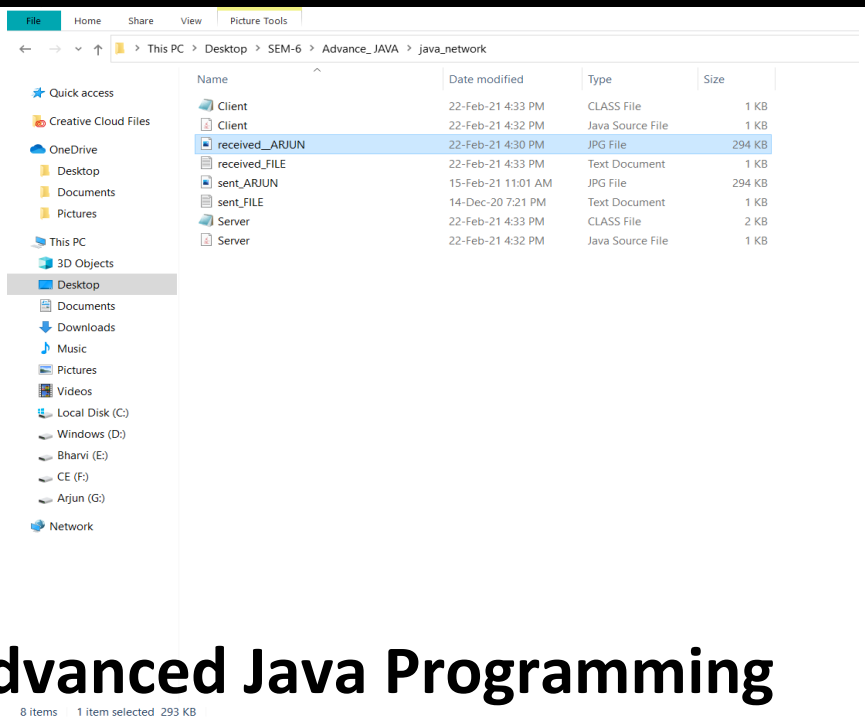
```
import java.io.*;
import java.net.*;
public class Client {

   public static void main(String[] args) throws Exception
   {
         Socket s=new Socket("127.0.0.1",1234);
         if(s.isConnected())
         {
            System.out.println("Connected to server");
         }
         FileOutputStream fileout= new FileOutputStream("received_FI
LE.txt");
         DataInputStream datain=new DataInputStream(s.getInputStre
am());
         int r;
```

```
        while((r=datain.read())!=-1)
        {
           fileout.write((char)r);
        }
           System.out.println("Succesfully done");
        s.close();
   }

}
```

2) **Client Side:**

```
import java.io.*;
import java.net.*;
class Server
{
   public static void main(String args[]) throws Exception
   {
      ServerSocket ss=new ServerSocket(1234);
      Socket s=ss.accept();
      System.out.println("connected with Client");
      FileInputStream filein=new FileInputStream("sent_FILE.txt");
      DataOutputStream dataout=new DataOutputStream(s.getOutpu
tStream());
      int r;
      while((r=filein.read())!=-1)
      {
         dataout.write(r);

      }
      System.out.println("\n FILE tranfer Completed");
      s.close();
      ss.close();
   }
}
```

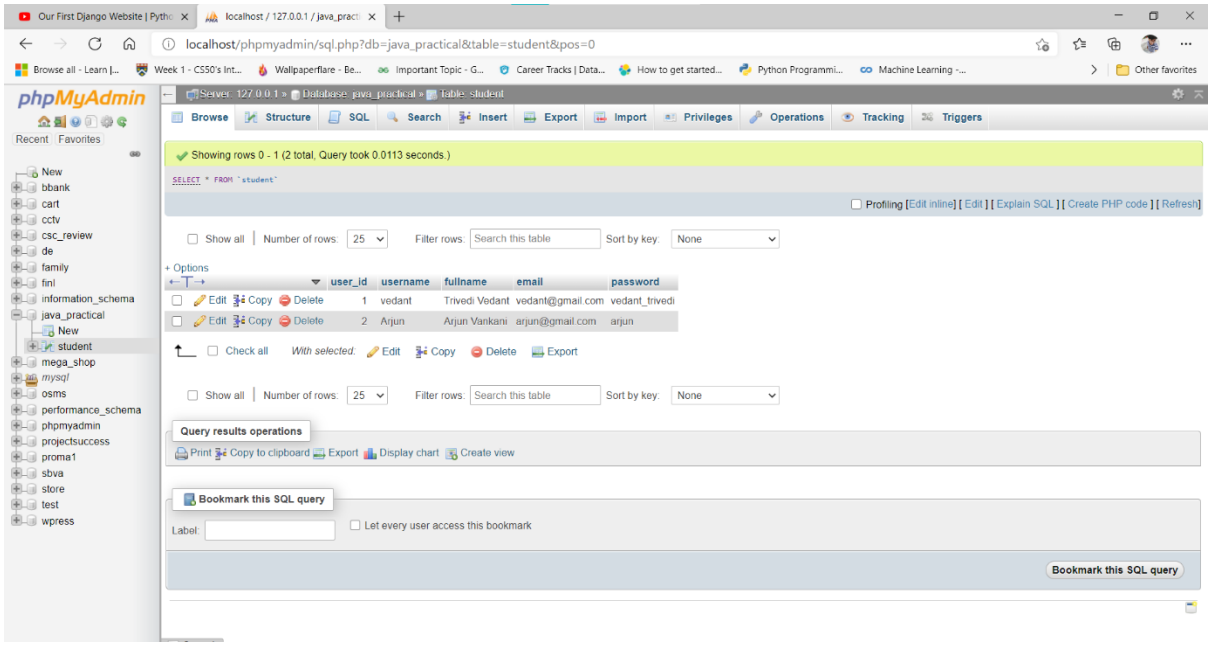
[redacted]



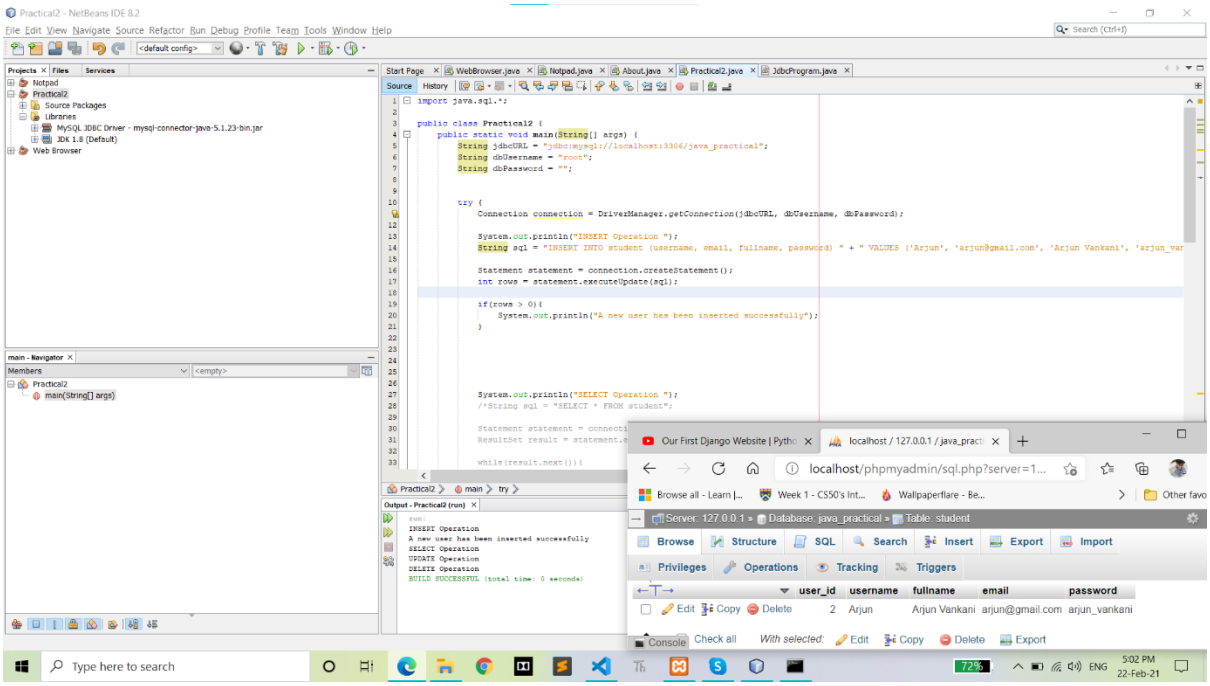
Advanced Java Programming (Lab Session-2)

Practical No: 2) Implement java application that demonstrates CRUD operation

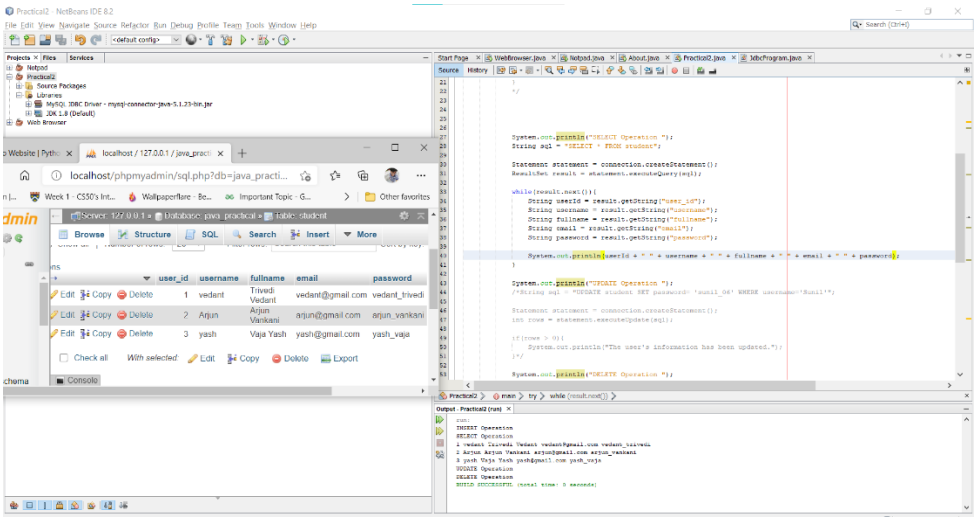
- Database



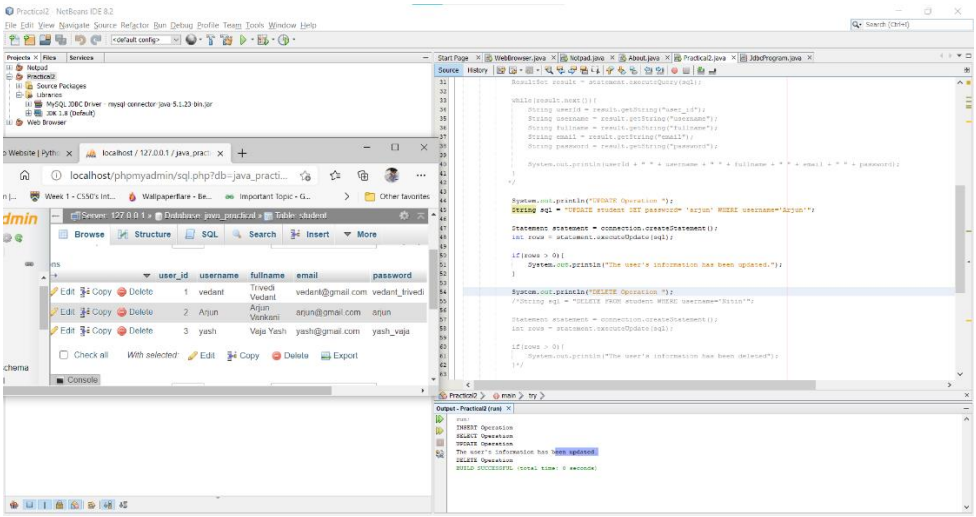
INSERT Query



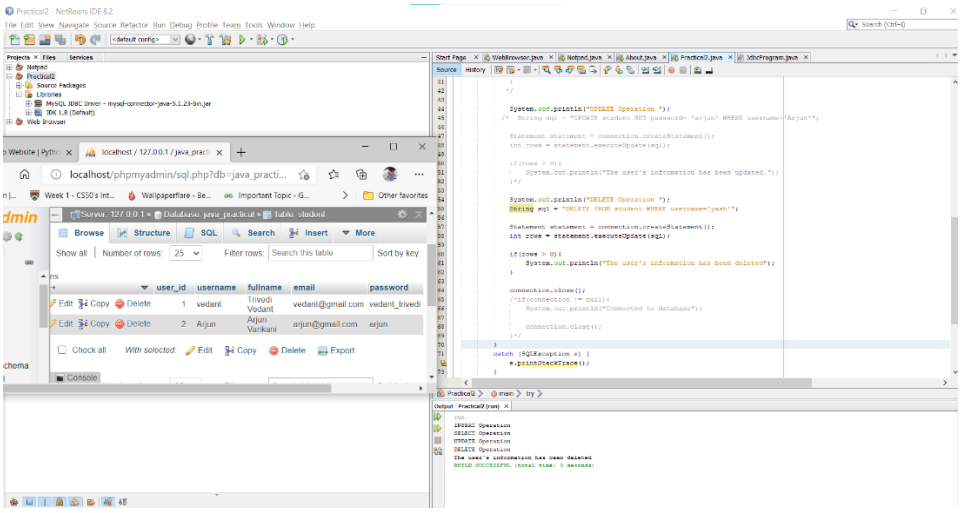
SELECT Query



UPDATE Query



DELETE Query



```
Code:

import java.sql.*;


public class Practical2 {

   public static void main(String[] args) {

      String jdbcURL = "jdbc:mysql://localhost:3306/java_practical";

      String dbUsername = "root";

      String dbPassword = "";



      try {

         Connection connection = DriverManager.getConnection(jdbcURL, dbUsername, dbPassword);


         System.out.println("INSERT Operation ");

        /* String sql = "INSERT INTO student (username, email, fullname, password) " + " VALUES
('Arjun', 'arjun@gmail.com', 'Arjun Vankani', 'arjun_vankani')";


         Statement statement = connection.createStatement();

         int rows = statement.executeUpdate(sql);
```

```
 if(rows > 0){
    System.out.println("A new user has been inserted successfully");
 }
 */





 System.out.println("SELECT Operation ");
/*  String sql = "SELECT * FROM student";

 Statement statement = connection.createStatement();
 ResultSet result = statement.executeQuery(sql);

 while(result.next()){
    String userId = result.getString("user_id");
    String username = result.getString("username");
    String fullname = result.getString("fullname");
    String email = result.getString("email");
    String password = result.getString("password");

    System.out.println(userId + " " + username + " " + fullname + " " + email + " " + password);
 }
*/

 System.out.println("UPDATE Operation ");
/*  String sql = "UPDATE student SET password= 'arjun' WHERE username='Arjun'";

 Statement statement = connection.createStatement();
 int rows = statement.executeUpdate(sql);
```

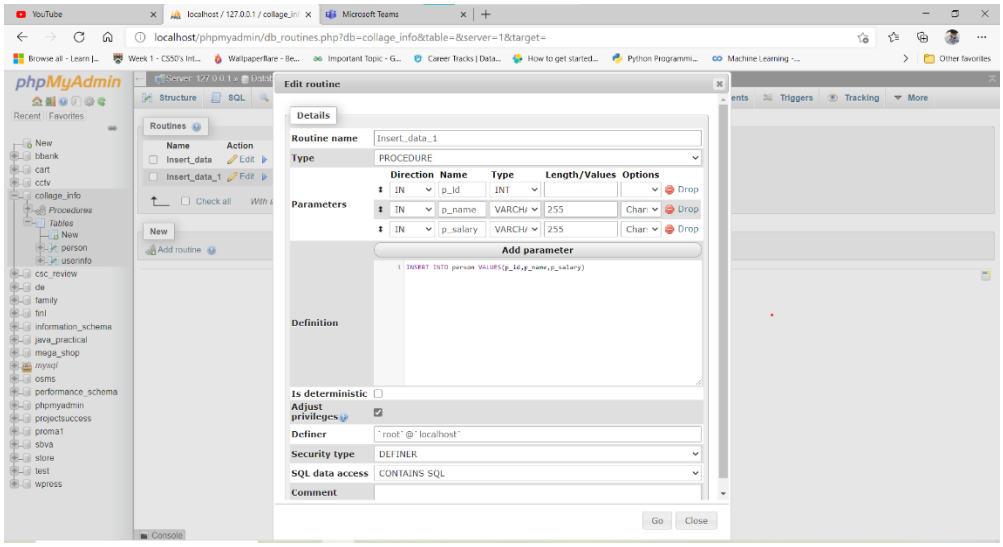
```
        if(rows > 0){
            System.out.println("The user's information has been updated.");
        }*/

        System.out.println("DELETE Operation ");
        /* String sql = "DELETE FROM student WHERE username='yash'";

        Statement statement = connection.createStatement();
        int rows = statement.executeUpdate(sql);

        if(rows > 0){
            System.out.println("The user's information has been deleted");
        }*/

        connection.close();
        /*if(connection != null){
            System.out.println("Connected to database");

            connection.close();
        }*/
    }
    catch (SQLException e) {
        e.printStackTrace();
    }
  }
}
```

Advanced Java Programming
(Lab Session-3)

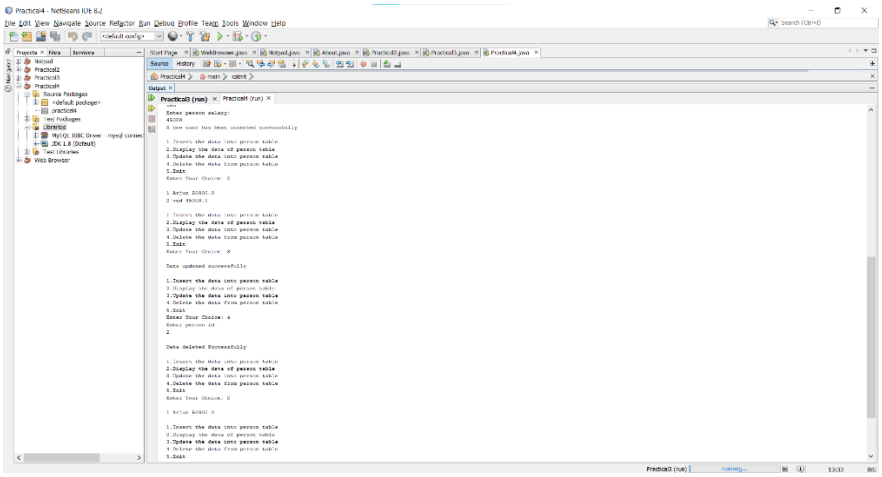
- **Practical No: 3)** **Write a JDBC Program to Fetch record from Person table having salary<15000.**
  Update data of selected records in underlying table with following condition.
  1. if salary is between 10000-11999 then add increment of 10%.
  2. if salary is between 12000-14999 then add increment of 11%.
  3. if salary is 15000 then add increment of 12%.

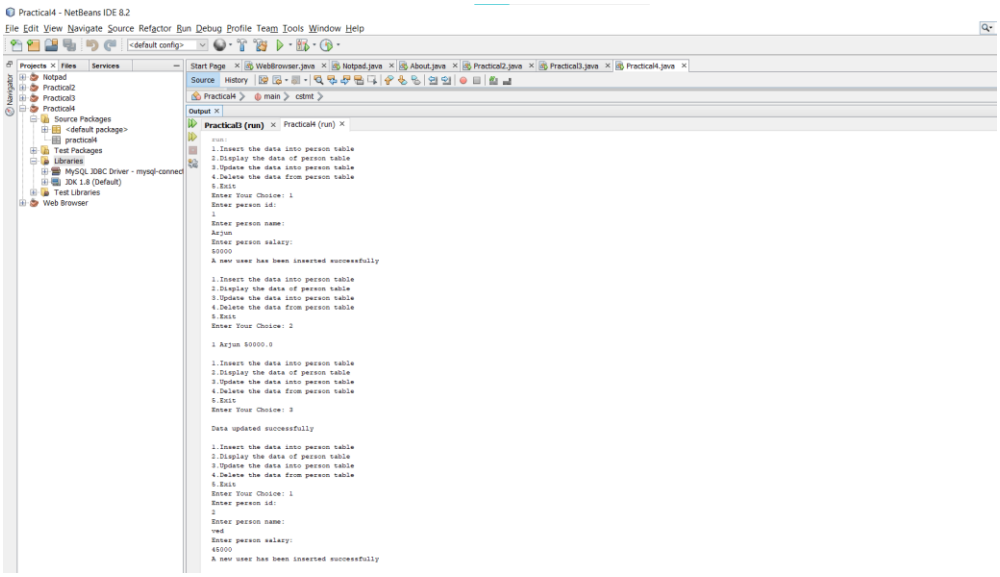
salary <= 15000

➢ Database Procedure



Output:





```
CODE:


import java.sql.*;
import java.util.Scanner;


public class Practical4 {
    public static void main(String[] args) {
        String jdbcURL = "jdbc:mysql://localhost:3306/collage_info";
        String dbUsername = "root";
        String dbPassword = "";
        Scanner scanner = new Scanner(System.in);
```

```
    Connection connection;
    Statement stmt;
    PreparedStatement pstmt;
    CallableStatement cstmt;
    ResultSet result;
    float old_salary, new_salary;

    try {
        connection = DriverManager.getConnection(jdbcURL, dbUsername, dbPassword);

        int x = 1;
        while (x == 1) {
            System.out.print(
                    "1.Insert the data into person table\n2.Display the data of person
table\n3.Update the data into person table\n4.Delete the data from person
table\n5.Exit\nEnter Your Choice: ");
            int choice = scanner.nextInt();

            if (choice == 1) {
                System.out.println("Enter person id: ");
                int id = scanner.nextInt();
                System.out.println("Enter person name: ");
                String name = scanner.next();
                System.out.println("Enter person salary: ");
                float salary = scanner.nextFloat();

                cstmt = connection.prepareCall("{call Insert_data_1(?,?,?)}");
                cstmt.setInt(1, id);
                cstmt.setString(2, name);
                cstmt.setFloat(3, salary);
```

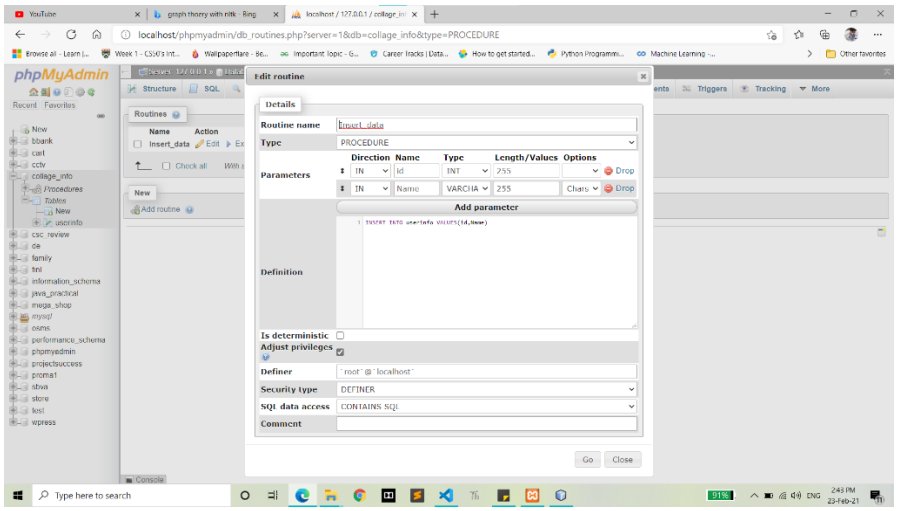
```
            cstmt.execute();
            System.out.println("A new user has been inserted successfully");
            cstmt.close();
            System.out.println();
        }

        else if (choice == 2) {
            String query = "SELECT * FROM person";

            stmt = connection.createStatement();
            result = stmt.executeQuery(query);

            System.out.println();
            while (result.next()) {
                int id = result.getInt("p_id");
                String name = result.getString("p_name");
                float salary = result.getFloat("p_salary");
                System.out.println(id + " " + name + " " + salary);
            }
            result.close();
            stmt.close();
            System.out.println();

        }

        else if (choice == 3) {
            String query = "SELECT * FROM person WHERE p_salary <= 15000";

            stmt = connection.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,
ResultSet.CONCUR_UPDATABLE);
```

```
            result = stmt.executeQuery(query);
            while (result.next()) {
              if (result.getFloat("p_salary") >= 10000 && result.getFloat("p_salary") <=
11999) {
                old_salary = result.getFloat("p_salary");
                new_salary = old_salary + ((old_salary * 10) / 100);
                result.updateFloat("p_salary", new_salary);
                result.updateRow();
              } else if (result.getFloat("p_salary") >= 12000 && result.getFloat("p_salary")
<= 14999) {
                old_salary = result.getFloat("p_salary");
                new_salary = old_salary + ((old_salary * 11) / 100);
                result.updateFloat("p_salary", new_salary);
                result.updateRow();
              } else if (result.getFloat("p_salary") == 15000) {
                old_salary = result.getFloat("p_salary");
                new_salary = old_salary + ((old_salary * 12) / 100);
                result.updateFloat("p_salary", new_salary);
                result.updateRow();
              } else {
                System.out.println("Salary is greater than 15000");
              }

            }
            System.out.println("\nData updated successfully\n");
            result.close();
            stmt.close();
          }

          else if (choice == 4) {
```

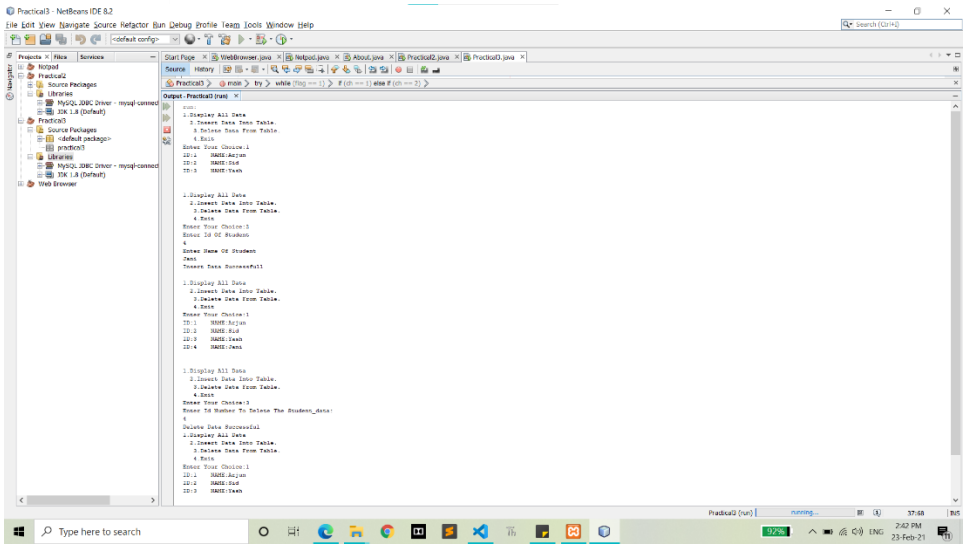
```
            System.out.println("Enter person id: ");
            int id = scanner.nextInt();

            String query = "DELETE FROM person WHERE p_id = ?";

            pstmt = connection.prepareStatement(query);
            pstmt.setInt(1, id);
            pstmt.executeUpdate();
            System.out.println("\nData deleted Successfully\n");
            pstmt.close();
          }
          else {
            x = 0;
          }
        }
        connection.close();
      }
      catch (SQLException e) {
        System.out.println(e);
      }
    }
}
```

Advanced Java Programming
(Lab Session-4)

- **Practical No: 4) Implement java application which demonstrates use of Statement, Prepared Statement and Callable Statement.**
  - **Database Procedure**



Output:



```
CODE:
import java.sql.*;
import java.util.*;
import java.util.Scanner;

class Practical3 {

        public static void main(String[] args) {
                Statement smt;
                PreparedStatement pre_smt;
                ResultSet res;
                CallableStatement call_smt;
                Scanner sc = new Scanner(System.in);
                try {
                        Class.forName("com.mysql.jdbc.Driver");
                        Connection con =
DriverManager.getConnection("jdbc:mysql://localhost/collage_info", "root", "");
```

```
int flag = 1;
while (flag == 1) {
System.out.print(
"1.Display All Data\n  2.Insert Data Into
Table.\n   3.Delete Data From Table.\n   4.Exit\nEnter Your Choice:");
int ch = sc.nextInt();
if (ch == 1) {
String fe_data = "Select * from userinfo";
smt = con.createStatement();
res = smt.executeQuery(fe_data);
while (res.next()) {
System.out.println("ID:" + res.getInt("id") +
"\tNAME:" + res.getString("Name"));
}
System.out.println();
System.out.println();

} else if (ch == 2) {
System.out.println("Enter Id Of Student");
int stu_id = sc.nextInt();
System.out.println("Enter Name Of Student");
String stu_name = sc.next();
call_smt = con.prepareCall("{call Insert_data(?,?)}");
call_smt.setInt(1, stu_id);
call_smt.setString(2, stu_name);
call_smt.execute();
System.out.print("Insert Data Successfull");
System.out.println();
System.out.println();
} else if (ch == 3) {
```

```
String fetch_data = "Select id FROM userinfo";
pre_smt = con.prepareStatement(fetch_data);
res = pre_smt.executeQuery();
int count = 0;
while (res.next()) {
        count++;
}
int check_arr[] = new int[count];
int i = 0;
fetch_data = "Select id FROM userinfo";
pre_smt = con.prepareStatement(fetch_data);
res = pre_smt.executeQuery();
while (res.next()) {
        int id = res.getInt("id");
        check_arr[i] = id;
        i++;
}
System.out.println("Enter Id Number To Delete The
Student_data:");
int temp_id = sc.nextInt();
for (int j = 0; j < check_arr.length; j++) {
        if (check_arr[j] == temp_id) {
                String qur = "Delete FROM userinfo
where id = ?";
                pre_smt = con.prepareStatement(qur);
                pre_smt.setInt(1, temp_id);
                pre_smt.execute();
                System.out.println("Delete Data
Successful");
                break;
```

```
                                        }
                                }
                        } else {
                                flag = 0;
                        }
                }
        } catch (Exception e) {
                System.out.print(e);
        }
    }
}
```

Advanced Java Programming (Lab Session-5)

Practical No: 5) Find Out Current Date and Time

CODE: CurrentDate.java

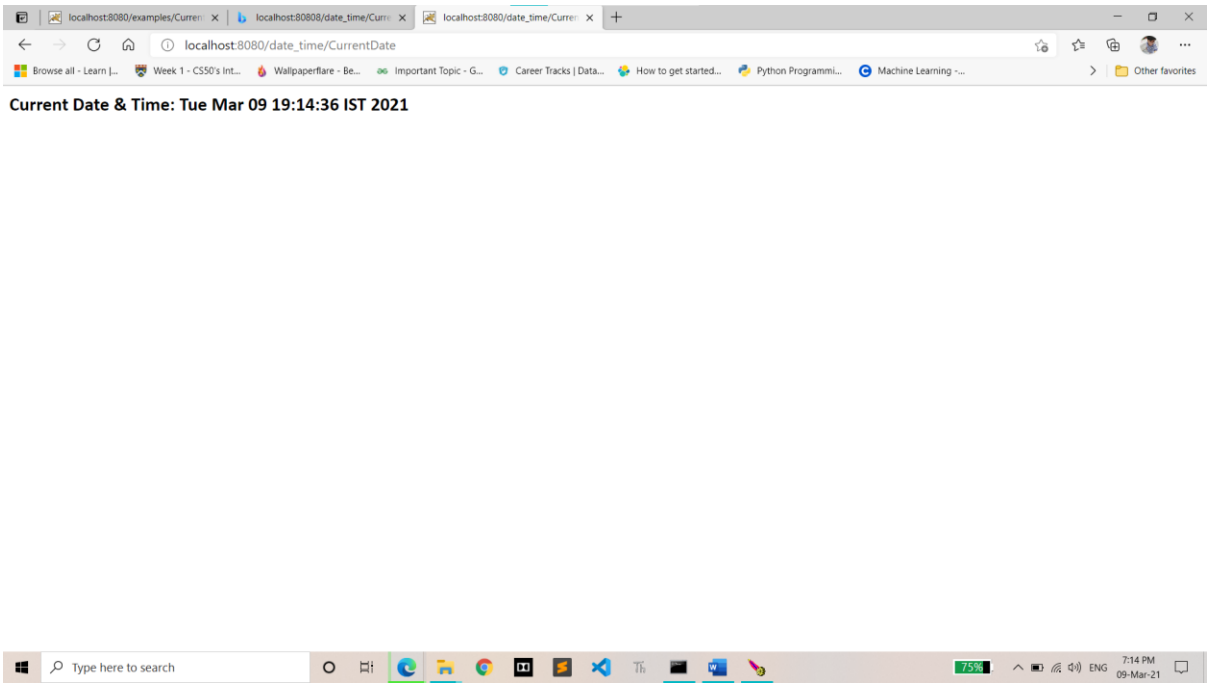
```
import java.io.*;
import java.util.Date;
import javax.servlet.*;
import javax.servlet.http.*;

// Extend HttpServlet class
public class CurrentDate extends HttpServlet {

  public void doGet(HttpServletRequest request, HttpServletResponse respons
e)
    throws ServletException, IOException {

    // Set response content type
    response.setContentType("text/html");

    PrintWriter out = response.getWriter();
    Date date = new Date();
    out.println("<h2>"+"Current Date & Time: " +date.toString()+"</h2>");
  }
}
```

Web.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<web-app>

        <servlet>

                <servlet-name>CurrentDate</servlet-name>

                <servlet-class>CurrentDate</servlet-class>

        </servlet>
```

```
        <servlet-mapping>
                <servlet-name>CurrentDate</servlet-name>
                <url-pattern>/CurrentDate</url-pattern>
        </servlet-mapping>
</web-app>
```

Output:



Advanced Java Programming (Lab Session-6)

- **Practical No: 6) Write a servlet program to validate user using username and password from database.**

Code:

JSPDB.jsp

```
<%@ page import="java.sql.*"%>
<%@ page import="java.util.*"%>
<%!
   Connection con;
   PreparedStatement ps1, ps2;
   public void jspInit()
   {
      try
      {
         Class.forName("com.mysql.jdbc.Driver");
         Connection con =
DriverManager.getConnection("jdbc:mysql://localhost/java_practical", "root",
"");
         //create statement object
         ps1 = con.prepareStatement("select count(*) from userlist where
username = ? and password=?");
         ps2 = con.prepareStatement("select * from userlist");
      }
      catch(Exception ex)
      {
         ex.printStackTrace();
```

```
    }
  }
%>
<%
  String param = request.getParameter("s1");
  if(param =="link")
  {
    ResultSet rs = ps2.executeQuery();
    out.println("<table>");
    while(rs.next())
    {
      out.println("<tr>");
      out.println("<td>"+rs.getString(1)+"</td>");
      out.println("<td>"+rs.getString(2)+"</td");
      out.println("</tr>");
    }
    out.println("</table>");
    rs.close();
  }
  else
  {
    //write jdbc code for authentication
    String user = request.getParameter("uname");
    String pass = request.getParameter("pwd");
    //set form data as param value
    ps1.setString(1,user);
```

```
      ps1.setString(2,pass);
      //excute the query
      ResultSet rs = ps1.executeQuery();
      int cnt = 0;
      if (rs.next())
         cnt = rs.getInt(1);
      if(cnt == 0)
         out.println("<div style='margin-left:400px'><b><i><font
color='#A0DBDB'>Invalid credential! <br> Please Try
again!</font></i></b></div>");
      else
      {
         out.println("<form style='margin-left:400px' ><fieldset style=
width:75%; >");
         out.println("<b><i><font color='#A0DBDB'>valid
credential..</font></i></b><br>");
         out.println("<b><i><font size=18 color='#ABD6BF' >Welcome to My
Page</font></i></b>");
         out.println("</fieldset></form>");
      }
   }
%>
<%!
   public void jspDestroy()
   {
      try
      {
         //colse
```

```
        ps1.close();
        ps2.close();
        con.close();
      }
      catch(Exception ex)
      {
        ex.printStackTrace();
      }
   }
%>
```

Index.html

```
<!doctype html>
<html lang="en">
   <head>
      <title>Welcome </title>
   </head>
   <body>
        <div style="margin-left:600px">
      <form method = "post" action = "JspDB.jsp" >
        <fieldset style="width:23%; background-color:#b3d1ff;">
           <h3><center> Login Page</center></h3>
           <hr>
           <table>
              <tr>
                 <td>Username:</td>
```

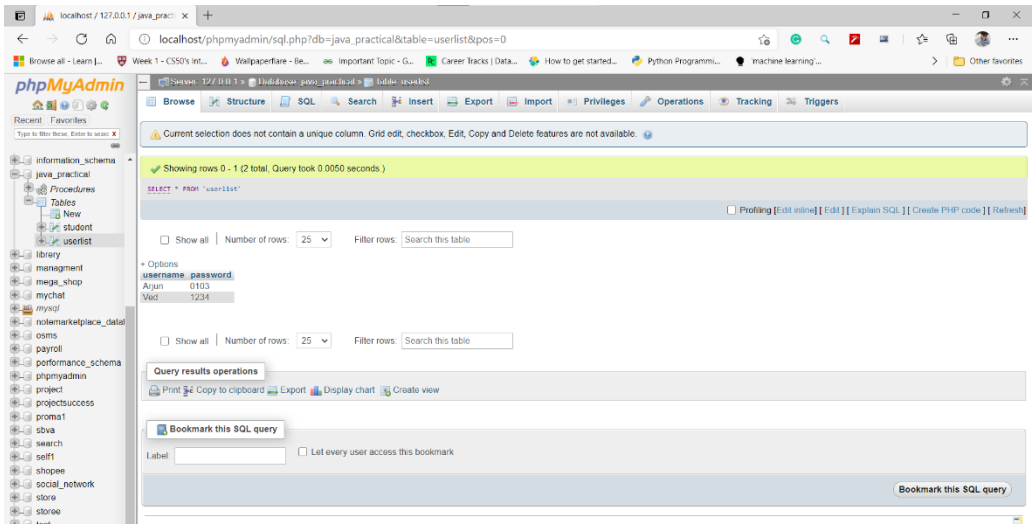
```
                <td> <input type = "text" name = "uname"></td>
            </tr>
            <tr>
                <td>Password:</td>
                <td><input type = "password" name = "pwd"></td>
            </tr>
            <tr>
                <td></td>
                <td><input type = "submit" value="check detail" name =
"s1"></td>
            </tr>
          </table>
        </fieldset>
      </form>
      <a href="JspDB.jsp? s1=link">Get all user detail</a>
        </div>
  </body>
</html>
```

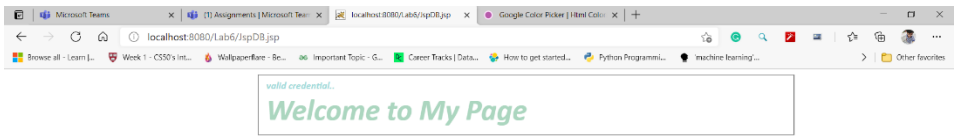
Web.xml

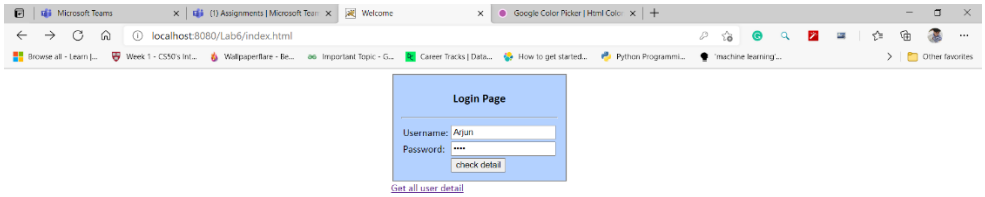
```
<web-app>
  <servlet>
    <servlet-name>xyz</servlet-name>
    <jsp-file>/input.html</jsp-file>
  </servlet>
  <servlet-mapping>
    <servlet-name>xyz</servlet-name>
```

```
    <url-pattern>/test</url-pattern>
  </servlet-mapping>
</web-app>
```

Output:







Advanced Java Programming (Lab Session-7)

- **Practical No: 7) Create an application to manage session of user using Http Session**

Code:

Loginservlet.java

```
package com.session;


import java.io.IOException;
import java.io.PrintWriter;


import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.Cookie;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;


/**
 * Servlet implementation class LoginServlet
 */
@WebServlet("/LoginServlet")
public class LoginServlet extends HttpServlet {
	private static final long serialVersionUID = 1L;
	private final String userID = "arjun";
	private final String password = "0103";
```

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


		// get request parameters for userID and password
		String user = request.getParameter("user");
		String pwd = request.getParameter("pwd");


		if(userID.equals(user) && password.equals(pwd)){
			HttpSession session = request.getSession();
			session.setAttribute("user", "Arjun");
			//setting session to expiry in 30 mins
			session.setMaxInactiveInterval(30*60);
			Cookie userName = new Cookie("user", user);
			userName.setMaxAge(30*60);
			response.addCookie(userName);
			response.sendRedirect("LoginSuccess.jsp");
		}else{
			RequestDispatcher rd =
getServletContext().getRequestDispatcher("/login.html");
			PrintWriter out= response.getWriter();
			out.println("<font color=red>Either user name or password is
wrong.</font>");
			rd.include(request, response);
		}
	}
}
```

Logoutservlet.java

```
package com.session;


import java.io.IOException;


import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.Cookie;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;


/**
 * Servlet implementation class LogoutServlet
 */
@WebServlet("/LogoutServlet")
public class LogoutServlet extends HttpServlet {
	private static final long serialVersionUID = 1L;


    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
	response.setContentType("text/html");
	Cookie[] cookies = request.getCookies();
	if(cookies != null){
	for(Cookie cookie : cookies){
		if(cookie.getName().equals("JSESSIONID")){
			System.out.println("JSESSIONID="+cookie.getValue());
			break;
		}
	}
```

```
        }
        //invalidate the session if exists
        HttpSession session = request.getSession(false);
        System.out.println("User="+session.getAttribute("user"));
        if(session != null){
                session.invalidate();
        }
        response.sendRedirect("login.html");
  }
}
```

Web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://java.sun.com/xml/ns/javaee"
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd" id="WebApp_ID"
version="3.0">
  <display-name>ServletHttpSessionExample</display-name>
  <welcome-file-list>
    <welcome-file>login.html</welcome-file>
  </welcome-file-list>
</web-app>
```

Login.html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="US-ASCII">
<title>Login Page</title>
</head>
<body>

<form action="LoginServlet" method="post">

Username: <input type="text" name="user">
<br>
Password: <input type="password" name="pwd">
<br>
<input type="submit" value="Login">
</form>
</body>
</html>
```

LoginSuccesful.jsp

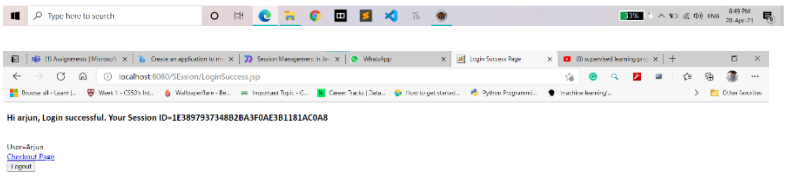
```
<%@ page language="java" contentType="text/html; charset=US-ASCII"
    pageEncoding="US-ASCII"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=US-ASCII">
<title>Login Success Page</title>
</head>
<body>
<%
//allow access only if session exists
String user = null;
if(session.getAttribute("user") == null){
      response.sendRedirect("login.html");
}else user = (String) session.getAttribute("user");
String userName = null;
String sessionID = null;
Cookie[] cookies = request.getCookies();
if(cookies !=null){
for(Cookie cookie : cookies){
      if(cookie.getName().equals("user")) userName = cookie.getValue();
      if(cookie.getName().equals("JSESSIONID")) sessionID =
cookie.getValue();
}
}
%>
<h3>Hi <%=userName %>, Login successful. Your Session ID=<%=sessionID
%></h3>
<br>
User=<%=user %>
<br>
<a href="CheckoutPage.jsp">Checkout Page</a>
<form action="LogoutServlet" method="post">
<input type="submit" value="Logout" >
</form>
</body>
</html>
```

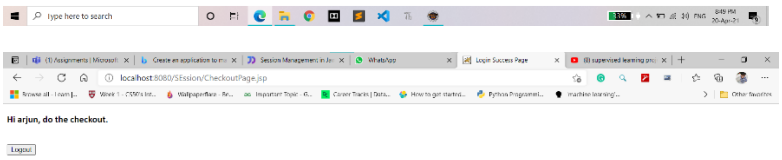
CheckoutPage.jsp

```
<%@ page language="java" contentType="text/html; charset=US-ASCII"
    pageEncoding="US-ASCII"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=US-ASCII">
<title>Login Success Page</title>
</head>
<body>
<%
//allow access only if session exists
if(session.getAttribute("user") == null){
      response.sendRedirect("login.html");
}
String userName = null;
String sessionID = null;
Cookie[] cookies = request.getCookies();
```

```
if(cookies !=null){
for(Cookie cookie : cookies){
      if(cookie.getName().equals("user")) userName = cookie.getValue();
}
}
%>
<h3>Hi <%=userName %>, do the checkout.</h3>
<br>
<form action="LogoutServlet" method="post">
<input type="submit" value="Logout" >
</form>
</body>
</html>
```

Output:





Advanced Java Programming (Lab Session-8)

Practical-8) Write a JSP application to manage User Profile. User must be able to View, Edit, Delete information from application.

Code:

Index.html

```
<!DOCTYPE html>
<html>
<head>
   <title>User Profile</title>
</head>
<body>
   <div style="text-align: center;">
        <div class='jumbotron'style='background-color: #E1FFBF'>
                <br>
      <button><a href='login.jsp'>Login</a></button><br><br>
      <button><a
href='registration.jsp'>Registration</a></button><br><br>
   </div>
   </div>
</body>
</html>
```

Login.jsp

```
<%@ page language="java" contentType="text/html"%>
<%
   if(request.getParameter("logout")==null){
      session.setAttribute("id","-1");
   }

%>
<!DOCTYPE html>
<html>
   <head>

      <title>Login</title>
   </head>
   <body >
      <div style="text-align: center;">
      <div class='jumbotron'style='background-color: #E1FFBF'>
         <br>
      <div>
      <form action='login_connect.jsp' method="post">
         <h1>Login Page</h1>
         <table style="margin-left: 600px;">
            <tr>
               <td >Username:-</td>
```

```
            <td><input type='text' name='user'></td>
          </tr>
          <tr>
            <td>Password:-</td>
            <td><input type='password' name='pass'></td>
          </tr>
        </table><br>
        <button type="Submit">Submit</button><br><br>
      </form>
    </div>
</div></div>
    </body>
</html>
```

Register.jsp

```
<%@ page language="java" contentType="text/html"%>
<!DOCTYPE html>
<html>
    <head>
      <title>Registration</title>
    </head>
    <body >
      <div>
        <div style="text-align: center;">
```

```
<div class='jumbotron'style='background-color: #E1FFBF'>
  <br>
<form action='reg_connect.jsp' method="post">
  <h1>Registration Page</h1>
  <table style="margin-left: 600px;">
    <tr>
      <td>Username:-</td>
      <td><input type='text' name='user'></td>
    </tr>
    <tr>
      <td>Password:-</td>
      <td><input type='password' name='pass'></td>
    </tr>
    <tr>
      <td>Email:-</td>
      <td><input type='email' name='email'></td>
    </tr>
    <tr>
      <td>Contact No:-</td>
      <td><input type='number' name='cont_no'></td>
    </tr>
  </table>
  <button type="Submit">Submit</button><br><br>
</form>
```

```
  </div>
</div></div>
  </body>
</html>
```

Login_connect.jsp

```
<%@ page import="java.sql.*" %>
<%
  String name = request.getParameter("user");
  String password = request.getParameter("pass");
  try{
    Class.forName("com.mysql.jdbc.Driver");
    Connection con =
DriverManager.getConnection("jdbc:mysql://localhost/student","r
oot","");
    PreparedStatement ps = con.prepareStatement("select * from
register where name=? and password=?");
    ps.setString(1, name);
    ps.setString(2, password);
    ResultSet rs=ps.executeQuery();
    if(rs.next()){
      String id = String.valueOf(rs.getInt(1));
      session.setAttribute("id",id);
      response.sendRedirect("view_profile.jsp");
    }
```

```
    else{
        response.sendRedirect("login.jsp");
    }
  }
  catch(Exception e){
    out.println(e);
  }
%>
```

Register_connect.jsp

```
<%@ page import="java.sql.*"%>
<%
  String name=request.getParameter("user");
  String password=request.getParameter("pass");
  String email=request.getParameter("email");
  String cont_no = request.getParameter("cont_no");
  try{
    Class.forName("com.mysql.jdbc.Driver");
    Connection con =
DriverManager.getConnection("jdbc:mysql://localhost/student","r
oot","");


    Statement st = con.createStatement();
```

```
        String sqr = "insert into
register(name,password,email,cont_no)
values('"+name+"','"+password+"','"+email+"','"+cont_no+"')";

        int x = st.executeUpdate(sqr);

        if(x>0){

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

            ps.setString(1, name);

            ps.setString(2, password);

            ResultSet rs=ps.executeQuery();

            if(rs.next()){

                String id = String.valueOf(rs.getInt(1));

                session.setAttribute("id",id);

                response.sendRedirect("view_profile.jsp");

            }

            response.sendRedirect("view_profile.jsp");

        }

        else{

            response.sendRedirect("registration.jsp");

        }

    }

    catch(Exception e){

        out.println(e);

    }

%>
```

View_profile.jsp

```
<%@ page import="java.sql.*" %>
<%
   String s = (String)session.getAttribute("id");
   if(s=="-1"){
      response.sendRedirect("login.jsp");
   }
%>
<%
   try{
      Class.forName("com.mysql.jdbc.Driver");
      Connection
con=DriverManager.getConnection("jdbc:mysql://localhost/studen
t","root","");
      String id = (String)session.getAttribute("id");
      int id1 = Integer.parseInt(id);
      if(request.getParameter("update")!=null){
         String Name,Email,Cont;
         out.println(request.getParameter("update"));
         PreparedStatement ps1 = con.prepareStatement("select *
from register where id=?");
         ps1.setInt(1, id1);
         ResultSet rs = ps1.executeQuery();
         if(rs.next()){
            Name = rs.getString(2);
```

```
            Email = rs.getString(4);
            Cont = rs.getString(5);
            if(request.getParameter("name_update")!=""){
               Name = request.getParameter("name_update");
            }
            if(request.getParameter("email_update")!=""){
               Email = request.getParameter("email_update");
            }

            if(request.getParameter("cont_update")!=""){
               Cont = request.getParameter("cont_update");
            }
            PreparedStatement ps2 = con.prepareStatement("Update
register set name=?,email=?,cont_no=? where id=?");
            ps2.setString(1,Name);
            ps2.setString(2,Email);
            ps2.setString(3,Cont);
            ps2.setInt(4,id1);
            ps2.executeUpdate();
         }
      }
      PreparedStatement ps = con.prepareStatement("select * from
register where id=?");
      ps.setInt(1, id1);
```

```
        ResultSet rs = ps.executeQuery();
        if(rs.next()){
            out.println(" Name :"+rs.getString(2)+"<br>"+" email
:"+rs.getString(4)+"<br>"+" contact No : "+rs.getString(5));
        }


    }
    catch(Exception e){
            out.println(e);
    }
%>
<%@ page language="java" contentType="text/html"%>
<!DOCTYPE html>
<html>
    <head>
        <title>Registration</title>
    </head>
    <body >
        <br><br>
        <button onclick="location.href='edit_profile.jsp'">Edit
Profile</button><br><br>
        <button onclick="location.href='delete_profile.jsp'">Delete
Profile</button><br><br>
        <button name='logout' onclick="location.href='login.jsp'">Log
Out</button><br><br>
```

```
  </body>
</html>
```

Edit_profile.jsp

```
<%@ page import="java.sql.*" %>
<%
  String s=(String)session.getAttribute("id");
  if(s=="-1"){
    response.sendRedirect("login.jsp");
  }
%>
<%
  try{
    Class.forName("com.mysql.jdbc.Driver");
    Connection con =
DriverManager.getConnection("jdbc:mysql://localhost/student","r
oot","");
    String id = (String)session.getAttribute("id");
    int id1 = Integer.parseInt(id);
    PreparedStatement ps = con.prepareStatement("select * from
register where id=?");
    ps.setInt(1, id1);
    ResultSet rs = ps.executeQuery();
  }
  catch(Exception e){
```

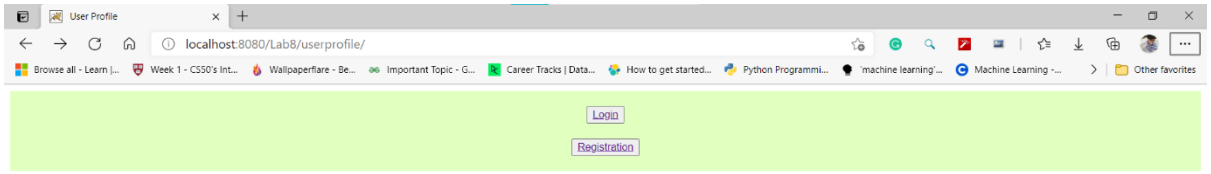
```
    out.println(e);
}
%>
<%@ page language="java" contentType="text/html"%>
<!DOCTYPE html>
<html>
<head>
  <title>Registration</title>
</head>
  <body>
  <form action="view_profile.jsp" method='post'>
    Name:-<input type='text' placeholder="Enter the Name" name='name_update'/><br><br>
    Email:-<input type='email' placeholder="Enter the Email" name="email_update"/><br><br>
    Contact No:-<input type="number" placeholder="Enter the Contact No" name="cont_update"/><br><br>
    <button name="update" type="submit">Update</button>
  </form>
  </body>
</html>
```

Delet_profile.jsp

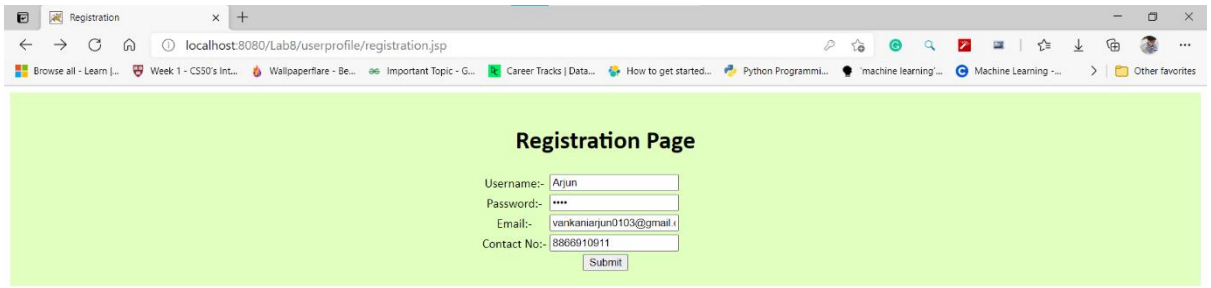
```
<%@ page import="java.sql.*" %>
<%
   try{
      Class.forName("com.mysql.jdbc.Driver");
      Connection con =
DriverManager.getConnection("jdbc:mysql://localhost/student","r
oot","");
      String id = (String)session.getAttribute("id");
      int id1 = Integer.parseInt(id);
      PreparedStatement ps = con.prepareStatement("Delete from
register where id=?");
      ps.setInt(1, id1);
      int x=ps.executeUpdate();
      if(x>0){
         response.sendRedirect("registration.jsp");
      }
      else{
         response.sendRedirect("view_profile.jsp");
      }
   }
   catch(Exception e){
      out.println(e);
   }
%>
```

Output:

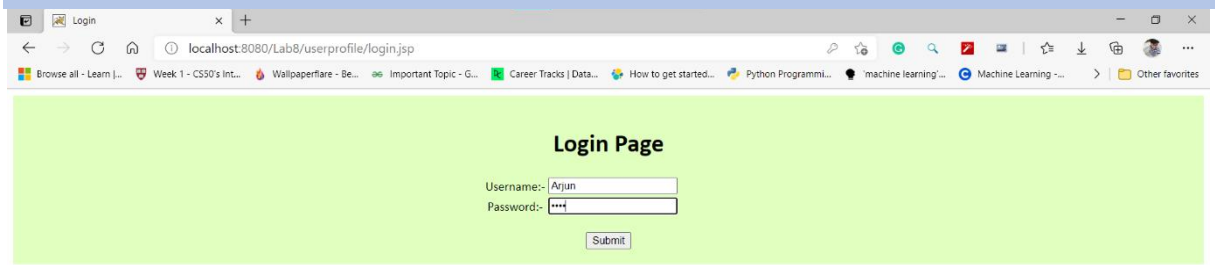
Index Page



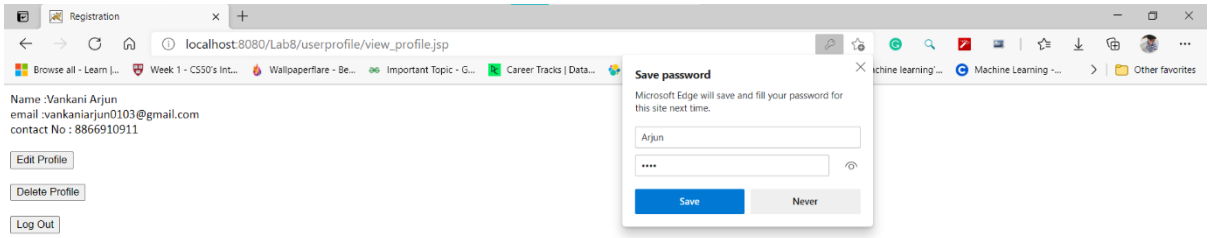
Register Page



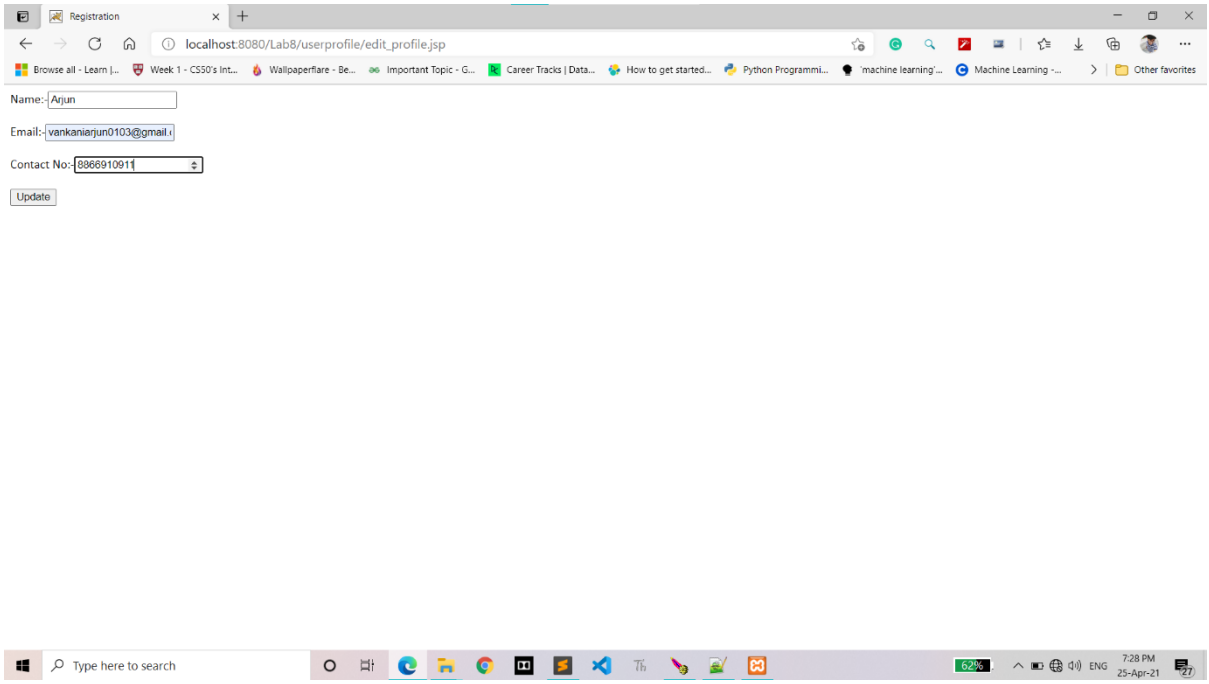
Login Page



Profile



Edit Profile Page



Advanced Java Programming (Lab Session-9)

Practical No: 9) Write a JSP application to manage User Profile. User must be able to View, Edit, Delete information from application.

Code:

UserDAO.java

```
package com.xadmin.usermanagement.dao;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;

import com.xadmin.usermanagement.model.User;

public class USerDAO {
	private String jdbcURL =
"jdbc:mysql://localhost:3306/Managment?useSSL=false";
	private String jdbcUsername = "root";
	private String jdbcPassword = "";

	private static final String INSERT_USERS_SQL = "INSERT INTO users" +
"  (name, email, country) VALUES "
		+ " (?, ?, ?);";

	private static final String SELECT_USER_BY_ID = "select
id,name,email,country from users where id =?";
	private static final String SELECT_ALL_USERS = "select * from users";
	private static final String DELETE_USERS_SQL = "delete from users
where id = ?;";
	private static final String UPDATE_USERS_SQL = "update users set name
= ?,email= ?, country =? where id = ?;";

	public USerDAO() {
	}

	protected Connection getConnection() {
		Connection connection = null;
		try {
			Class.forName("com.mysql.jdbc.Driver");
			connection = DriverManager.getConnection(jdbcURL,
jdbcUsername, jdbcPassword);
		} catch (SQLException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		} catch (ClassNotFoundException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
```

```
		}
		return connection;
	}

	public void insertUser(User user) throws SQLException {
		System.out.println(INSERT_USERS_SQL);
		// try-with-resource statement will auto close the connection.
		try (Connection connection = getConnection();
				PreparedStatement preparedStatement =
connection.prepareStatement(INSERT_USERS_SQL)) {
			preparedStatement.setString(1, user.getName());
			preparedStatement.setString(2, user.getEmail());
			preparedStatement.setString(3, user.getCountry());
			System.out.println(preparedStatement);
			preparedStatement.executeUpdate();
		} catch (SQLException e) {
			printSQLException(e);
		}
	}

	public User selectUser(int id) {
		User user = null;
		// Step 1: Establishing a Connection
		try (Connection connection = getConnection();
				// Step 2:Create a statement using connection
object
			PreparedStatement preparedStatement =
connection.prepareStatement(SELECT_USER_BY_ID);) {
			preparedStatement.setInt(1, id);
			System.out.println(preparedStatement);
			// Step 3: Execute the query or update query
			ResultSet rs = preparedStatement.executeQuery();

			// Step 4: Process the ResultSet object.
			while (rs.next()) {
				String name = rs.getString("name");
				String email = rs.getString("email");
				String country = rs.getString("country");
				user = new User(id, name, email, country);
			}
		} catch (SQLException e) {
			printSQLException(e);
		}
		return user;
	}

	public List<User> selectAllUsers() {

		// using try-with-resources to avoid closing resources (boiler
plate code)
		List<User> users = new ArrayList<>();
		// Step 1: Establishing a Connection
		try (Connection connection = getConnection();

				// Step 2:Create a statement using connection
object
			PreparedStatement preparedStatement =
connection.prepareStatement(SELECT_ALL_USERS);) {
			System.out.println(preparedStatement);
			// Step 3: Execute the query or update query
			ResultSet rs = preparedStatement.executeQuery();
```

```
                // Step 4: Process the ResultSet object.
                while (rs.next()) {
                        int id = rs.getInt("id");
                        String name = rs.getString("name");
                        String email = rs.getString("email");
                        String country = rs.getString("country");
                        users.add(new User(id, name, email, country));
                }
        } catch (SQLException e) {
                printSQLException(e);
        }
        return users;
    }

    public boolean deleteUser(int id) throws SQLException {
        boolean rowDeleted;
        try (Connection connection = getConnection();
                        PreparedStatement statement =
connection.prepareStatement(DELETE_USERS_SQL);) {
                statement.setInt(1, id);
                rowDeleted = statement.executeUpdate() > 0;
        }
        return rowDeleted;
    }

    public boolean updateUser(User user) throws SQLException {
        boolean rowUpdated;
        try (Connection connection = getConnection();
                        PreparedStatement statement =
connection.prepareStatement(UPDATE_USERS_SQL);) {
                System.out.println("updated USer:"+statement);
                statement.setString(1, user.getName());
                statement.setString(2, user.getEmail());
                statement.setString(3, user.getCountry());
                statement.setInt(4, user.getId());

                rowUpdated = statement.executeUpdate() > 0;
        }
        return rowUpdated;
    }

    private void printSQLException(SQLException ex) {
        for (Throwable e : ex) {
                if (e instanceof SQLException) {
                        e.printStackTrace(System.err);
                        System.err.println("SQLState: " + ((SQLException)
e).getSQLState());
                        System.err.println("Error Code: " + ((SQLException)
e).getErrorCode());
                        System.err.println("Message: " + e.getMessage());
                        Throwable t = ex.getCause();
                        while (t != null) {
                                System.out.println("Cause: " + t);
                                t = t.getCause();
                        }
                }
        }
    }

}
```

User.java

```
package com.xadmin.usermanagement.model;
public class User {
	protected int id;
	protected String name;
	protected String email;
	protected String country;

	public User() {
	}

	public User(String name, String email, String country) {
		super();
		this.name = name;
		this.email = email;
		this.country = country;
	}

	public User(int id, String name, String email, String country) {
		super();
		this.id = id;
		this.name = name;
		this.email = email;
		this.country = country;
	}

	public int getId() {
		return id;
	}
	public void setId(int id) {
		this.id = id;
	}
	public String getName() {
		return name;
	}
	public void setName(String name) {
		this.name = name;
	}
	public String getEmail() {
		return email;
	}
	public void setEmail(String email) {
		this.email = email;
	}
	public String getCountry() {
		return country;
	}
	public void setCountry(String country) {
		this.country = country;
	}
}
```

Userservlet.java

```
package com.xadmin.usermanagement.web;
import java.io.IOException;
import java.sql.SQLException;
import java.util.List;
```

```
import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import com.xadmin.usermanagement.dao.USerDAO;
import com.xadmin.usermanagement.model.User;



@WebServlet("/")
public class UserServlet extends HttpServlet {
	private static final long serialVersionUID = 1L;
	private USerDAO userDAO;

	public void init() {
		userDAO = new USerDAO();
	}

	protected void doPost(HttpServletRequest request, HttpServletResponse
response)
			throws ServletException, IOException {
		doGet(request, response);
	}

	protected void doGet(HttpServletRequest request, HttpServletResponse
response)
			throws ServletException, IOException {
		String action = request.getServletPath();

		try {
			switch (action) {
			case "/new":
				showNewForm(request, response);
				break;
			case "/insert":
				insertUser(request, response);
				break;
			case "/delete":
				deleteUser(request, response);
				break;
			case "/edit":
				showEditForm(request, response);
				break;
			case "/update":
				updateUser(request, response);
				break;
			default:
				listUser(request, response);
				break;
			}
		} catch (SQLException ex) {
			throw new ServletException(ex);
		}
	}

	private void listUser(HttpServletRequest request, HttpServletResponse
response)
```

```
                    throws SQLException, IOException, ServletException {
            List<User> listUser = userDAO.selectAllUsers();
            request.setAttribute("listUser", listUser);
            RequestDispatcher dispatcher =
request.getRequestDispatcher("user-list.jsp");
            dispatcher.forward(request, response);
      }

      private void showNewForm(HttpServletRequest request,
HttpServletResponse response)
                    throws ServletException, IOException {
            RequestDispatcher dispatcher =
request.getRequestDispatcher("user-form.jsp");
            dispatcher.forward(request, response);
      }

      private void showEditForm(HttpServletRequest request,
HttpServletResponse response)
                    throws SQLException, ServletException, IOException {
            int id = Integer.parseInt(request.getParameter("id"));
            User existingUser = userDAO.selectUser(id);
            RequestDispatcher dispatcher =
request.getRequestDispatcher("user-form.jsp");
            request.setAttribute("user", existingUser);
            dispatcher.forward(request, response);

      }

      private void insertUser(HttpServletRequest request,
HttpServletResponse response)
                    throws SQLException, IOException {
            String name = request.getParameter("name");
            String email = request.getParameter("email");
            String country = request.getParameter("country");
            User newUser = new User(name, email, country);
            userDAO.insertUser(newUser);
            response.sendRedirect("list");
      }

      private void updateUser(HttpServletRequest request,
HttpServletResponse response)
                    throws SQLException, IOException {
            int id = Integer.parseInt(request.getParameter("id"));
            String name = request.getParameter("name");
            String email = request.getParameter("email");
            String country = request.getParameter("country");

            User book = new User(id, name, email, country);
            userDAO.updateUser(book);
            response.sendRedirect("list");
      }

      private void deleteUser(HttpServletRequest request,
HttpServletResponse response)
                    throws SQLException, IOException {
            int id = Integer.parseInt(request.getParameter("id"));
            userDAO.deleteUser(id);
            response.sendRedirect("list");

      }
}
```

Web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://xmlns.jcp.org/xml/ns/javaee"
xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd" id="WebApp_ID"
version="3.1">
  <display-name>UserMangement</display-name>
  <welcome-file-list>
    <welcome-file>user-list.jsp</welcome-file>
  </welcome-file-list>
  <error-page>
    <exception-type>java.lang.Exception</exception-type>
    <location>/Error.jsp</location>
  </error-page>
</web-app>
```

Userform-servlet.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
      pageEncoding="UTF-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<html>
<head>
<title>User Management Application</title>
<link rel="stylesheet"
      href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstra
p.min.css"
      integrity="sha384-
ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T"
      crossorigin="anonymous">
</head>
<body style="background-color: #DFE5E8">

      <header>
            <nav class="navbar navbar-expand-md navbar-dark"
                  style="background-color: #8DCCCC">
                  <div>
                        <a href="https://www.xadmin.net" class="navbar-
brand"> User Management Application </a>
                  </div>

                  <ul class="navbar-nav">
                        <li><a href="<%=request.getContextPath()%>/list"
                              class="nav-link">Users</a></li>
                  </ul>
            </nav>
      </header>
      <br>
      <div style="background-color: #F0EBEB" class="container col-md-5">
            <div class="card">
                  <div class="card-body">
                        <c:if test="${user != null}">
                              <form action="update" method="post">
                        </c:if>
                        <c:if test="${user == null}">
                              <form action="insert" method="post">
                        </c:if>
```

```
                        <caption>
                              <h2>
                                    <c:if test="${user != null}">
                              Edit User
                        </c:if>
                                    <c:if test="${user == null}">
                              Add New User
                        </c:if>
                              </h2>
                        </caption>

                        <c:if test="${user != null}">
                              <input type="hidden" name="id" value="<c:out
value='${user.id}' />" />
                        </c:if>

                        <fieldset class="form-group">
                              <label>User Name</label> <input type="text"
                                    value="<c:out value='${user.name}' />"
class="form-control"
                                    name="name" required="required">
                        </fieldset>

                        <fieldset class="form-group">
                              <label>User Email</label> <input type="text"
                                    value="<c:out value='${user.email}' />"
class="form-control"
                                    name="email">
                        </fieldset>

                        <fieldset class="form-group">
                              <label>User Country</label> <input
type="text"
                                    value="<c:out value='${user.country}'
/>" class="form-control"
                                    name="country">
                        </fieldset>

                        <button type="submit" class="btn btn-
success">Save</button>
                        </form>
                  </div>
            </div>
      </div>
</body>
</html>
```

Userlist.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
      pageEncoding="UTF-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<html>
<head>
<title>User Management Application</title>
<link rel="stylesheet"
      href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstra
p.min.css"
      integrity="sha384-
ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T"
```

```
        crossorigin="anonymous">
</head>
<body style="background-color: #DFE5E8">

        <header>
                <nav class="navbar navbar-expand-md navbar-dark"
                        style="background-color: #8DCCCC">
                        <div>
                                <a href="https://www.xadmin.net" class="navbar-
brand"> User
                                        Management Application </a>
                        </div>

                        <ul class="navbar-nav">
                                <li><a href="<%=request.getContextPath()%>/list"
                                        class="nav-link">Users</a></li>
                        </ul>
                </nav>
        </header>
        <br>

        <div class="row">
                <!-- <div class="alert alert-success"
*ngIf='message'>{{message}}</div> -->

                <div class="container">
                        <h3 class="text-center">List of Users</h3>
                        <hr>
                        <div class="container text-left">

                                <a href="<%=request.getContextPath()%>/new"
class="btn btn-success" style="background-color:#86CF9F">Add
                                        New User</a>
                        </div>
                        <br>
                        <table class="table table-bordered">
                                <thead>
                                        <tr>
                                                <th>ID</th>
                                                <th>Name</th>
                                                <th>Email</th>
                                                <th>Country</th>
                                                <th>Actions</th>
                                        </tr>
                                </thead>
                                <tbody>

                                        <c:forEach var="user" items="${listUser}">

                                                <tr>
                                                        <td><c:out value="${user.id}"
/></td>
                                                        <td><c:out value="${user.name}"
/></td>
                                                        <td><c:out value="${user.email}"
/></td>
                                                        <td><c:out
value="${user.country}" /></td>
                                                        <td><a href="edit?id=<c:out
value='${user.id}' />">Edit</a>
                                                                     <a
```

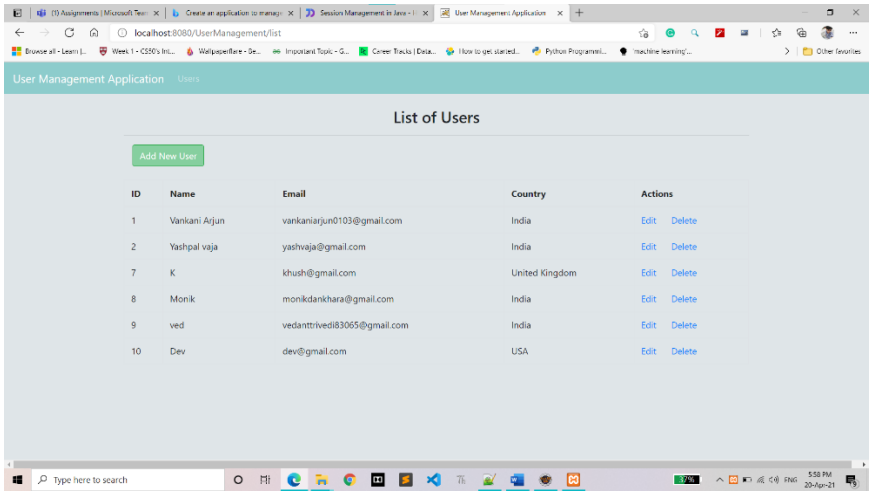
```
                                                    href="delete?id=<c:out
value='${user.id}' />">Delete</a></td>
                                        </tr>
                                    </c:forEach>

                                </tbody>

                            </table>
                        </div>
                    </div>
</body>
</html>
```

Eroor.jsp

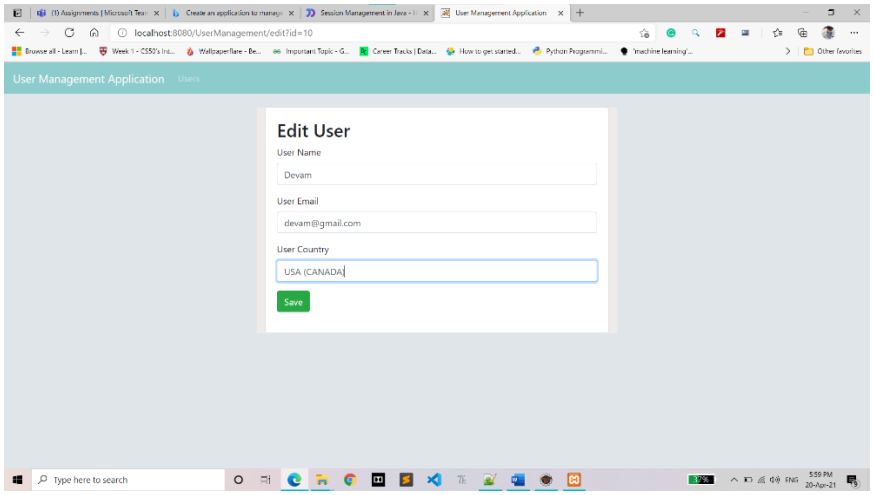
```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8" isErrorPage="true" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
      "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Error</title>
</head>
<body>
      <center>
            <h1>Error</h1>
            <h2><%=exception.getMessage() %><br/> </h2>
      </center>
</body>
</html>
```

Output:

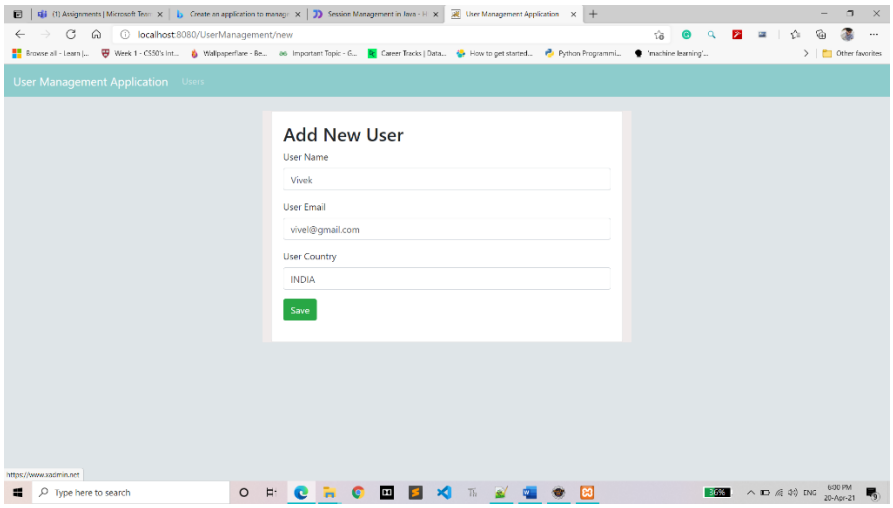
View List



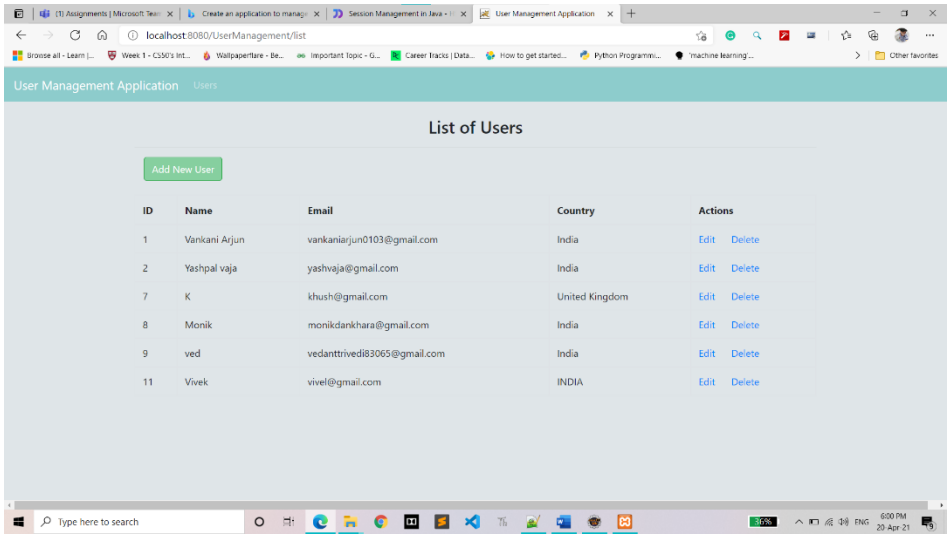
Edit User:



Add New User:



User Updating list:



Advanced Java Programming (Lab Session-10)

Practical-10) Mini Project Based on Spring and Hibernate

Code:

1) Hibernate Project only

Hibernateutil.java

```
package net.roseindia;


import org.hibernate.SessionFactory;


import org.hibernate.cfg.Configuration;
import org.hibernate.service.ServiceRegistry;
import org.hibernate.service.ServiceRegistryBuilder;



public class HibernateUtil {
	private static final SessionFactory sessionFactory;

	private static ServiceRegistry serviceRegistry;

	static {

		try {
```

```
                    Configuration configuration = new Configuration();
                    configuration.configure();
                    serviceRegistry = new
ServiceRegistryBuilder().applySettings(

        configuration.getProperties()).buildServiceRegistry();
                    sessionFactory =
configuration.buildSessionFactory(serviceRegistry);

            } catch (Throwable th) {

                    System.err.println("Enitial SessionFactory creation
failed" + th);

                    throw new ExceptionInInitializerError(th);

            }

        }

        public static SessionFactory getSessionFactory() {

            return sessionFactory;
        }
```

```
}
```

createData.java

```
package net.roseindia;

import org.hibernate.Session;
import org.hibernate.SessionFactory;

import net.roseindia.model.*;

public class CreateData {
	public static void main(String[] args) throws Exception {

		SessionFactory sessFact =
HibernateUtil.getSessionFactory();
		Session session = sessFact.getCurrentSession();
		org.hibernate.Transaction tr =
session.beginTransaction();
		Employee emp = new Employee();
		emp.setEmpName("Arjun Vankani");
		emp.setEmpMobileNos("8866910911");
		emp.setEmpAddress("Delhi - India");
		session.save(emp);
		tr.commit();
```

```
            System.out.println("Successfully inserted");
            sessFact.close();
        }

}
```

Hibernate.cfg.xml

```
<?xml version='1.0' encoding='utf-8'?>
<!DOCTYPE hibernate-configuration PUBLIC
"-//Hibernate/Hibernate Configuration DTD//EN"
"http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">

<hibernate-configuration>
<session-factory>
<property
name="hibernate.connection.driver_class">com.mysql.jdbc.Driver</property>
<property
name="hibernate.connection.url">jdbc:mysql://localhost/hibernate4</property
>
<property name="hibernate.connection.username">root</property>
<property name="hibernate.connection.password"></property>
<property name="hibernate.connection.pool_size">10</property>
<property name="show_sql">true</property>
<property name="dialect">org.hibernate.dialect.MySQLDialect</property>
<property name="hibernate.current_session_context_class">thread</property>

<mapping class="net.roseindia.model.Employee" />

</session-factory>
</hibernate-configuration>
```

Employee.java

```
package net.roseindia.model;

import java.io.Serializable;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.Id;

import javax.persistence.Table;
```

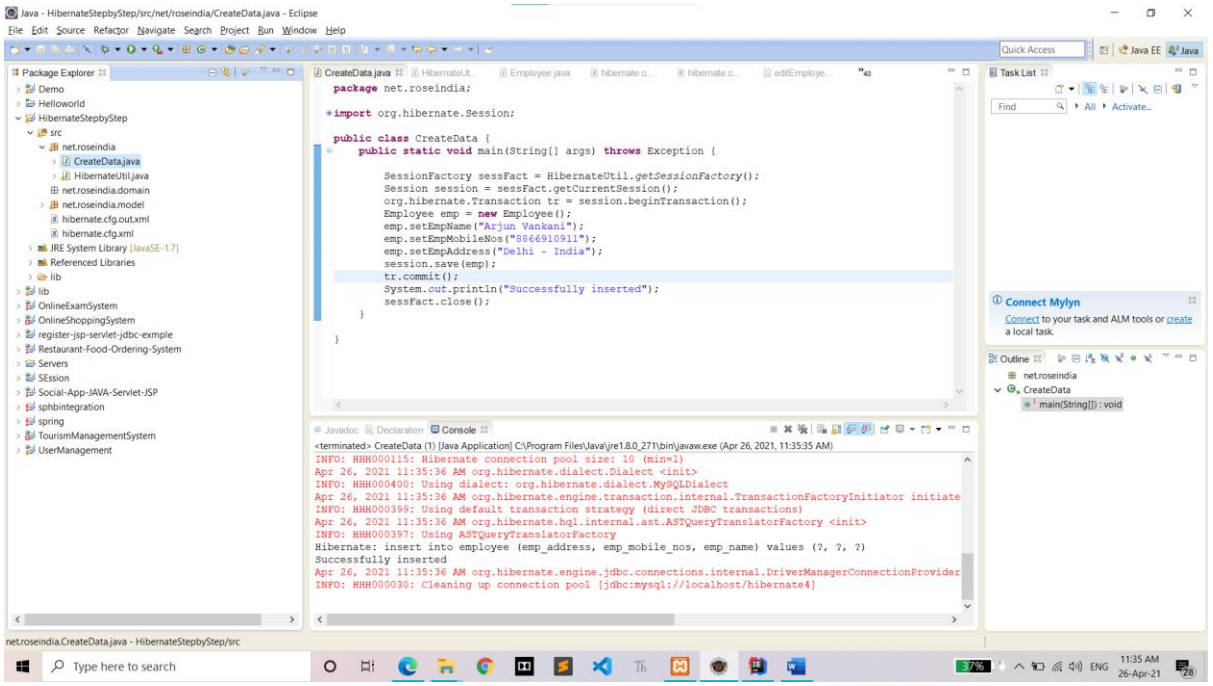
```
@Entity
@Table(name = "employee")
public class Employee implements Serializable{

	@Id
	@GeneratedValue
	@Column(name="id")
	private int id;

	@Column(name="emp_name")
	private String empName;

	@Column(name="emp_address")
	private String empAddress;

	@Column(name="emp_mobile_nos")
	private String empMobileNos;

	public int getId() {
		return id;
	}
	public void setId(int id) {
		this.id = id;
```

```
}

public String getEmpName() {
        return empName;
}

public void setEmpName(String empName) {
        this.empName = empName;
}

public String getEmpAddress() {
        return empAddress;
}

public void setEmpAddress(String empAddress) {
        this.empAddress = empAddress;
}

public String getEmpMobileNos() {
        return empMobileNos;
}
public void setEmpMobileNos(String empMobileNos) {
        this.empMobileNos = empMobileNos;
}
```

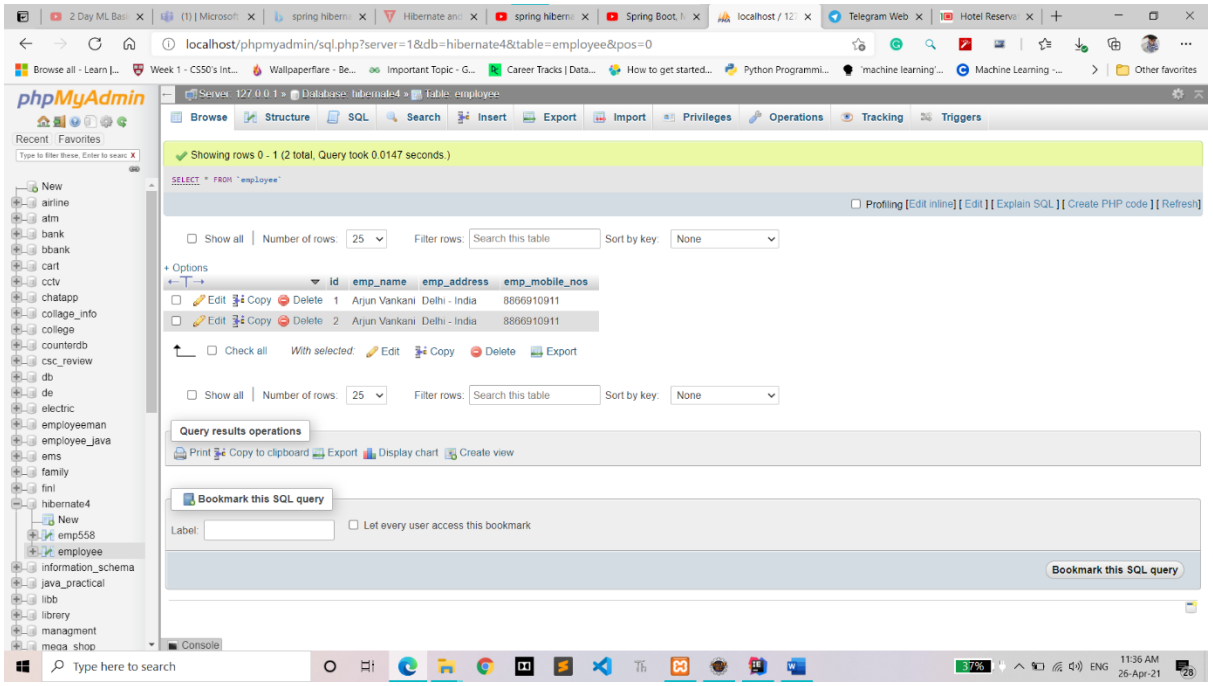
```
}
```

Output:

Hibernate data input



Database Successfully Added



2) Hotel Reservation Application Spring MVC Hibernate Project

Code:

HotelReservationController.java

```
package com.mindtree.hotelreservation.controller;

import java.util.ArrayList;
import java.util.Date;
import java.util.List;
import javax.annotation.PostConstruct;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.hibernate.HibernateException;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.servlet.ModelAndView;
import com.mindtree.hotelreservation.Dto.ReservationDto;
import com.mindtree.hotelreservation.Dto.UserDto;
import com.mindtree.hotelreservation.entity.Hotel;
import com.mindtree.hotelreservation.entity.RegisteredUser;
import com.mindtree.hotelreservation.entity.Reservation;
import com.mindtree.hotelreservation.service.HotelReservationService;
import com.mindtree.hotelreservation.util.StringToDateConversion;

@Controller
public class HotelReservationController {
    @Autowired
    HotelReservationService hotelReservationService;// = new
                                                    // HotelReservationServiceImpl();
    @PostConstruct
    public void display() {
        System.out.println("hotelReservationService created");
    }

    public HotelReservationService getHotelReservationService() {
        return hotelReservationService;
    }

    public void setHotelReservationService(HotelReservationService hotelReservationService) {
        this.hotelReservationService = hotelReservationService;
    }

    @RequestMapping("/searchHotels")
    public String searchHotels() {
        return "SearchHotels";
    }

    @RequestMapping("/getRegistrationPage")
    public ModelAndView getRegistrationPage() {
        return new ModelAndView("userregistration", "registeredUser", new RegisteredUser());
    }
    /*
     * @RequestMapping("/searchHotels") public ModelAndView searchHotels() {
     * ModelAndView modelView = new ModelAndView("searchHotels"); List<String>
     * hotelNames = new ArrayList<String>(); List<Hotel> hotels =
     * hotelReservationService.getAllHotels(); for(Hotel hotel: hotels) {
```

```
 * hotelNames.add(hotel.getName()); } modelView.addObject("hotelNames",
 * hotelNames); return modelView; }
 */

@RequestMapping("/getHotels")
public ModelAndView getHotels(HttpServletRequest request, HttpServletResponse response) {

    ModelAndView modelView = null;
    System.out.println("get Hotels in controller is called successfully");
    String searchHotel = request.getParameter("searchHotel");
    System.out.println(searchHotel);
    if (searchHotel.isEmpty()) {
        modelView = new ModelAndView("SearchHotels");
        modelView.addObject("errorMessage", "Search String is empty, Please enter a valid search");
        return modelView;
    }
    modelView = new ModelAndView("Hotels");
    List<Hotel> hotels = null;
    try {
        hotels = hotelReservationService.getAllHotels();
    } catch (HibernateException hEx) {
        modelView = new ModelAndView("searchhotelspage");
        modelView.addObject("errorMessage", "Please make sure Database is connected");
        return modelView;
    }

    List<Hotel> matchedHotels = new ArrayList<Hotel>();
    for (Hotel hotelI : hotels) {
        System.out.println(hotelI.getName());
        if (hotelI.getName().matches("(.*)" + searchHotel + "(.*)")) {
            matchedHotels.add(hotelI);
            System.out.println("Added to matched hotels " + hotelI.getName());
        }
    }
    if (matchedHotels.isEmpty()) {
        modelView = new ModelAndView("SearchHotels");
        modelView.addObject("errorMessage", "Sorry, Your search doesn't match any of the hotels");
        return modelView;
    }
    modelView.addObject("hotels", matchedHotels);

    return modelView;
}

@RequestMapping("/getHotel")
public ModelAndView getHotel(@RequestParam("id") int id) {
    System.out.println(id);
    System.out.println("get Hotel in controller is called successfully");
    ModelAndView modelView = new ModelAndView("Hotel");
    Hotel hotel = null;
    try {
        hotel = hotelReservationService.getHotelById(id);
    } catch (HibernateException hEx) {
        modelView = new ModelAndView("hotels");
        modelView.addObject("hotel",
                "U have done something wrong (Database disconnected) you Please make sure Database is con
nected");
        return modelView;
    }
    modelView.addObject("hotel", hotel);
    return modelView;
}

@RequestMapping("/bookHotelAfterLogin")
public ModelAndView bookHotelAfterLogin(@RequestParam("id") int id, HttpSession session) {
    System.out.println(id);
```

```
        System.out.println("get Hotel in controller is called successfully");
        ModelAndView modelView = new ModelAndView("userLogin", "userDto", new UserDto());
        Hotel hotel = null;
        try {
            session.setAttribute("hotelId", id);
            hotel = hotelReservationService.getHotelById(id);
        } catch (HibernateException hEx) {
            modelView = new ModelAndView("hotel");
            modelView.addObject("errorMessage",
                    "U have done something wrong (Database disconnected) you Please make sure Database is con
nected");
            return modelView;
        }
        modelView.addObject("hotel", hotel);
        return modelView;
    }

    @RequestMapping("/bookHotel")
    public ModelAndView bookHotel(@ModelAttribute UserDto userDto, BindingResult result, HttpSession session)
 {
        ModelAndView modelView = null;
        System.out.println("Validation Errors " + result.hasErrors());
        Boolean isErrors = false;
        int userNameLength = userDto.getUserName().length();
        int passwordLength = userDto.getPassword().length();
        if (userNameLength < 1 || userNameLength > 20 || passwordLength < 1 || passwordLength > 20)
            isErrors = true;
        if (result.hasErrors() || isErrors) {
            System.out.println("spring validation started");
            modelView = new ModelAndView("userLogin");
            modelView.addObject("errorMessage", "Invalid user name or password! enter again");
            return modelView;
        }

        String userName = userDto.getUserName();
        String password = userDto.getPassword();
        if (!hotelReservationService.isValidUser(userName)) {
            System.out.println("is User Name valid in controller");
            modelView = new ModelAndView("userLogin");
            modelView.addObject("errorMessage", "User name doesn't exist! enter again");
            return modelView;
        }
        Hotel hotel = null;
        RegisteredUser user = null;
        try {
            hotel = hotelReservationService.getHotelById((int) session.getAttribute("hotelId"));
            user = hotelReservationService.getUserByName(userName);
        } catch (HibernateException hEx) {
            modelView = new ModelAndView("userLogin");
            modelView.addObject("errorMessage",
                    "U have done something wrong (Database disconnected) you Please make sure Database is con
nected");
            return modelView;
        }
        Reservation reservation = new Reservation();
        reservation.setHotelId(hotel.getId());
        reservation.setUserId(user.getId());
        System.out.println(userDto.getUserName());
        System.out.println(userDto.getPassword());
        System.out.println(userDto.getHotelId());
        boolean status = false;
        try {
            status = hotelReservationService.authenticateUser(userName, password);
        } catch (HibernateException hEx) {
            modelView = new ModelAndView("userLogin");
            modelView.addObject("errorMessage",
```

```
                    "U have done something wrong (Database disconnected) you Please make sure Database is con
nected");
            return modelView;
        }
        System.out.println(status);

        if (status == false) {
            modelView = new ModelAndView("userLogin");
            modelView.addObject("errorMessage", "User name or password is wrong! enter again");
        } else {
            modelView = new ModelAndView("reservation", "reservation", new Reservation());
            modelView.addObject("reservationObj", reservation);
            modelView.addObject("hotel", hotel);
            modelView.addObject("user", user);

        }
        return modelView;
    }

    @RequestMapping("/showBookedHotels")
    public ModelAndView showBookedHotels(HttpServletRequest request, HttpServletResponse response) {
        ModelAndView modelView = null;
        System.out.println(request.getParameter("checkInDate"));
        System.out.println(request.getParameter("checkOutDate"));
        System.out.println();
        System.out.println();
        System.out.println("showBookedHotels in controller");
        Reservation reservation = new Reservation();
        int hotelId = Integer.parseInt(request.getParameter("hotelId"));
        String userName = request.getParameter("userName");
        String checkInDateString = request.getParameter("checkInDate");
        String checkOutDateString = request.getParameter("checkOutDate");
        Boolean isError = (checkInDateString.length() < 1) || (checkOutDateString.length() < 1);
        System.out.println(isError);
        if (checkInDateString.isEmpty() || checkOutDateString.isEmpty() || isError) {
            modelView = new ModelAndView("reservation");
            modelView.addObject("errorMessage", "Please fill all the fields");
            modelView.addObject("reservationObj", reservation);
            modelView.addObject("hotel", hotelReservationService.getHotelById(hotelId));
            modelView.addObject("user", hotelReservationService.getUserByName(userName));
            return modelView;
        }
        System.out.println(checkInDateString.length());
        System.out.println(checkOutDateString.length());
        // Hotel hotel = reservationServiceImpl.getHotelById(hotelId);
        RegisteredUser user = hotelReservationService.getUserByName(userName);
        Date checkInDate = StringToDateConversion.getDayMonthYearFromStringDate(checkInDateString);
        Date checkOutDate = StringToDateConversion.getDayMonthYearFromStringDate(checkOutDateString);
        System.out.println(checkInDate);
        System.out.println(checkOutDate);
        System.out.println(new Date());
        // new Scanner(System.in).nextLine();
        if (checkInDate.before(new Date())) {
            System.out.println("Before current date");
            modelView = new ModelAndView("reservation");
            modelView.addObject("errorMessage", "CheckInDate cant be a past date " + "or before current date"
);
            modelView.addObject("reservationObj", reservation);
            modelView.addObject("hotel", hotelReservationService.getHotelById(hotelId));
            modelView.addObject("user", hotelReservationService.getUserByName(userName));
            return modelView;
        } else if (checkOutDate.before(checkInDate)) {
            modelView = new ModelAndView("reservation");
            System.out.println("Before checkin date");
            modelView.addObject("errorMessage", "CheckOutDate can't be before CheckInDate");
            modelView.addObject("reservationObj", reservation);
```

```
            modelView.addObject("hotel", hotelReservationService.getHotelById(hotelId));
            modelView.addObject("user", hotelReservationService.getUserByName(userName));
            return modelView;
        }
        reservation.setUserId(user.getId());
        reservation.setHotelId(hotelId);
        reservation.setCheckInDate(checkInDate);
        reservation.setCheckOutDate(checkOutDate);

        modelView = new ModelAndView("showBookedHotels");

        hotelReservationService.addReservation(reservation);
        List<Reservation> reservationsById = null;
        try {
            reservationsById = hotelReservationService.getReservationsByUserId(user.getId());
        } catch (HibernateException hEx) {
            modelView = new ModelAndView("showBookedHotels");
            modelView.addObject("errorMessage",
                    "U have done something wrong (Database disconnected) you Please make sure Database is con
nected");
            return modelView;
        }
        List<ReservationDto> reservationsOfUser = new ArrayList<ReservationDto>();
        for (Reservation reservtn : reservationsById) {
            reservationsOfUser.add(new ReservationDto(user, hotelReservationService.getHotelById(reservtn.get
HotelId()),
                    reservtn.getCheckInDate(), reservtn.getCheckOutDate()));
        }
        modelView.addObject("reservations", reservationsOfUser);
        modelView.addObject("user", user);
        return modelView;
    }

    @RequestMapping(value = "/createUser")
    public ModelAndView createUser(@ModelAttribute RegisteredUser registeredUser, BindingResult result) {
        ModelAndView modelView = null;
        Boolean isErrors = false;
        int userNameLength = registeredUser.getUserName().length();
        int passwordLength = registeredUser.getPassword().length();
        if (userNameLength < 1 || userNameLength > 20 || passwordLength < 1 || passwordLength > 20)
            isErrors = true;
        if (result.hasErrors() || isErrors) {
            System.out.println("validation started");
            modelView = new ModelAndView("userregistration");
            modelView.addObject("errorMessage",
                    "Invalid user name or password!" + " enter again" + "\n" + "Didn't match the Requirements
");
            return modelView;
        }
        if (hotelReservationService.isValidUser(registeredUser.getUserName())) {
            System.out.println("server side validations started");
            modelView = new ModelAndView("userregistration");
            modelView.addObject("errorMessage", "User name already existed" + "\nPlease choose other User nam
e");
            return modelView;
        }
        hotelReservationService.createUser(registeredUser);
        modelView = new ModelAndView("userLogin", "userDto", new UserDto());
        modelView.addObject("successMessage","Successfully Registered! Now you can login"
                + "with your user name");
        return modelView;

    }
}
```

Hotelreservationdao.java

```
package com.mindtree.hotelreservation.dao;

import java.util.List;
import com.mindtree.hotelreservation.entity.Hotel;
import com.mindtree.hotelreservation.entity.RegisteredUser;
import com.mindtree.hotelreservation.entity.Reservation;

public interface HotelReservationDAO {

    List<Hotel> getAllHotels();

    String getPasswordByUserName(String userName);

    RegisteredUser getUserByName(String userName);

    boolean isValidUser(String userName);

    Hotel getHotelById(int hotelId);

    void addReservation(Reservation reservation);

    List<Reservation> getReservationsByUserId(int userId);

    void createUser(RegisteredUser user);
}
```

Index.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

            pageEncoding="ISO-8859-1"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>




<style>

html

{

background-color:rgb(256,256,256);

}

body

{

background-color:rgb(232,232,232);

margin:10% 20%;
```

```
border:2px solid black;
font-family:"Times New Roman";
position:relative;
}
.dottedborder
{
border:1px dotted black;
}
h3
{
background-color:rgb(112,112,112);
}
header
{
height:138px;
margin:2px;
padding:2px;
display:block;
postion:relative;
}
footer
{
}
div
{
margin:2px;
display:block;
position:relative;
}
.left
{
```

```
float:left;
}
.right
{
float:right;
}
.righttopcontent
{
}
table
{
margin-left:50px;
border:1px solid black;
}
th
{
text-align:left;
background-color:rgb(200,200,200);
text-transform:uppercase;
}
table,th,tr,td
{
border:1px solid black;
}
td {
    padding: 10px;
}
.sideheading
{
text-transform:uppercase;
font-weight:bold;
```

```
float:bottom;
}
.center
{
text-align:center;
font-weight:bold;
}
hr
{
height:5px;
color:rgb(0,0,0);
}
.liststyle
{
list-style-type:none;
}
.subsubhead
{
text-transform:uppercase;
display:inline-block;
width:300px;
font-weight:600;
}

a:link, a:visited {
    color: white;
    padding: 14px 25px;
    text-align: center;
    text-decoration: none;
    display: inline-block;
}
```

```
a:hover, a:active {
   background-color: rgb(112,112,112);
}

.mainheading
{
text-transform:uppercase;
font-size:x-large;
text-align:center;
}

</style>
<title>HotelReservation</title>
</head>

<body>
        <center>
                <form>
                        <h1>Welcome To Hotel
                                Booking Application</h1>
                        <br> <br> <br> <br> <br> <br>
                        <table>
                                <tr>
                                        <td colspan="1"><a href="searchHotels"><h2 align="center"
                                                                style="color: black">Search
Hotels</h2></a></td>
                                        <%-- <td><h3 style="color: #48D1CC;">${msg}</h3></td> --
%>
                                </tr>
                        </table>
                </form>
```

```
        </center>
</html>
```

Pom.xml

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
        <modelVersion>4.0.0</modelVersion>
        <groupId>HotelReservation</groupId>
        <artifactId>HotelReservation</artifactId>
        <version>0.0.1-SNAPSHOT</version>
        <packaging>war</packaging>
        <properties>
                <org.springframework.version>4.2.5.RELEASE</org.springframework.version>
                <log4j.version>1.2.17</log4j.version>
        </properties>
        <build>
                <plugins>
                        <plugin>
                                <artifactId>maven-compiler-plugin</artifactId>
                                <version>3.3</version>
                                <configuration>
                                        <source>1.7</source>
                                        <target>1.7</target>
                                </configuration>
                        </plugin>
                        <plugin>
                                <artifactId>maven-war-plugin</artifactId>
                                <version>2.6</version>
                                <configuration>
                                        <warSourceDirectory>WebContent</warSourceDirectory>
```

```
                <failOnMissingWebXml>false</failOnMissingWebXml>
            </configuration>
        </plugin>
    </plugins>
</build>

<dependencies>
    <dependency>
        <groupId>mysql</groupId>
        <artifactId>mysql-connector-java</artifactId>
        <version>5.1.22</version>
    </dependency>
    <dependency>
        <groupId>commons-dbcp</groupId>
        <artifactId>commons-dbcp</artifactId>
        <version>1.4</version>
    </dependency>
    <dependency>
        <groupId>org.hibernate</groupId>
        <artifactId>hibernate-validator</artifactId>
        <version>5.2.4.Final</version>
    </dependency>
    <dependency>
        <groupId>org.hibernate</groupId>
        <artifactId>hibernate-entitymanager</artifactId>
        <version>5.0.0.Final</version>
    </dependency>
    <dependency>
        <groupId>javax.servlet</groupId>
        <artifactId>javax.servlet-api</artifactId>
        <version>3.0.1</version>
```

```
        <scope>provided</scope>
    </dependency>
    <dependency>
        <groupId>javax.servlet</groupId>
        <artifactId>jstl</artifactId>
        <version>1.2</version>
    </dependency>
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-core</artifactId>
        <version>${org.springframework.version}</version>
    </dependency>

    <!-- Expression Language (depends on spring-core) Define this if you use
        Spring Expression APIs (org.springframework.expression.*) -->
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-expression</artifactId>
        <version>${org.springframework.version}</version>
    </dependency>

    <!-- Bean Factory and JavaBeans utilities (depends on spring-core) Define
        this if you use Spring Bean APIs (org.springframework.beans.*) -->
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-beans</artifactId>
        <version>${org.springframework.version}</version>
    </dependency>

    <!-- Aspect Oriented Programming (AOP) Framework (depends on spring-core,
```

```
                spring-beans) Define this if you use Spring AOP APIs
(org.springframework.aop.*) -->
        <dependency>
                <groupId>org.springframework</groupId>
                <artifactId>spring-aop</artifactId>
                <version>${org.springframework.version}</version>
        </dependency>

        <!-- Application Context (depends on spring-core, spring-expression, spring-aop,
                spring-beans) This is the central artifact for Spring's Dependency Injection
                Container and is generally always defined -->
        <dependency>
                <groupId>org.springframework</groupId>
                <artifactId>spring-context</artifactId>
                <version>${org.springframework.version}</version>
        </dependency>

        <!-- Various Application Context utilities, including EhCache, JavaMail,
                Quartz, and Freemarker integration Define this if you need any of these
integrations -->
        <dependency>
                <groupId>org.springframework</groupId>
                <artifactId>spring-context-support</artifactId>
                <version>${org.springframework.version}</version>
        </dependency>

        <!-- Transaction Management Abstraction (depends on spring-core, spring-beans,
                spring-aop, spring-context) Define this if you use Spring Transactions or
                DAO Exception Hierarchy
(org.springframework.transaction.*/org.springframework.dao.*) -->
        <dependency>
                <groupId>org.springframework</groupId>
```

```
                <artifactId>spring-tx</artifactId>
                <version>${org.springframework.version}</version>
        </dependency>

        <!-- JDBC Data Access Library (depends on spring-core, spring-beans, spring-context,
                spring-tx) Define this if you use Spring's JdbcTemplate API
(org.springframework.jdbc.*) -->
        <dependency>
                <groupId>org.springframework</groupId>
                <artifactId>spring-jdbc</artifactId>
                <version>${org.springframework.version}</version>
        </dependency>

        <!-- Object-to-Relation-Mapping (ORM) integration with Hibernate, JPA,
                and iBatis. (depends on spring-core, spring-beans, spring-context, spring-tx)
                Define this if you need ORM (org.springframework.orm.*) -->
        <dependency>
                <groupId>org.springframework</groupId>
                <artifactId>spring-orm</artifactId>
                <version>${org.springframework.version}</version>
        </dependency>

        <!-- Object-to-XML Mapping (OXM) abstraction and integration with JAXB,
                JiBX, Castor, XStream, and XML Beans. (depends on spring-core, spring-
beans,
                spring-context) Define this if you need OXM (org.springframework.oxm.*) --
>
        <dependency>
                <groupId>org.springframework</groupId>
                <artifactId>spring-oxm</artifactId>
                <version>${org.springframework.version}</version>
        </dependency>
```

```
<!-- Web application development utilities applicable to both Servlet and
        Portlet Environments (depends on spring-core, spring-beans, spring-context)
        Define this if you use Spring MVC, or wish to use Struts, JSF, or another
        web framework with Spring (org.springframework.web.*) -->
<dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-web</artifactId>
        <version>${org.springframework.version}</version>
</dependency>

<!-- Spring MVC for Servlet Environments (depends on spring-core, spring-beans,
        spring-context, spring-web) Define this if you use Spring MVC with a Servlet
        Container such as Apache Tomcat (org.springframework.web.servlet.*) -->
<dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-webmvc</artifactId>
        <version>${org.springframework.version}</version>
</dependency>

<!-- Spring MVC for Portlet Environments (depends on spring-core, spring-beans,
        spring-context, spring-web) Define this if you use Spring MVC with a Portlet
        Container (org.springframework.web.portlet.*) -->
<dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-webmvc-portlet</artifactId>
        <version>${org.springframework.version}</version>
</dependency>

<!-- Support for testing Spring applications with tools such as JUnit and
        TestNG This artifact is generally always defined with a 'test' scope for
```

```
        the integration testing framework and unit testing stubs -->
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-test</artifactId>
        <version>${org.springframework.version}</version>
        <scope>test</scope>
    </dependency>
    <dependency>
        <groupId>javax.persistence</groupId>
        <artifactId>persistence-api</artifactId>
        <version>1.0</version>
    </dependency>
    <dependency>
        <groupId>org.springframework.security</groupId>
        <artifactId>spring-security-web</artifactId>
        <version>4.0.4.RELEASE</version>
    </dependency>
    <dependency>
        <groupId>org.jboss.spec.javax.transaction</groupId>
        <artifactId>jboss-transaction-api_1.2_spec</artifactId>
        <version>1.0.0.Final</version>
    </dependency>
    <dependency>
        <groupId>log4j</groupId>
        <artifactId>log4j</artifactId>
        <version>${log4j.version}</version>
        <scope>runtime</scope>
    </dependency>

    <!-- Mockito dependencies -->
    <dependency>
```

```
                <groupId>junit</groupId>
                <artifactId>junit</artifactId>
                <version>4.12</version>
            </dependency>
            <dependency>
                <groupId>org.mockito</groupId>
                <artifactId>mockito-all</artifactId>
                <version>1.9.5</version>
            </dependency>
        </dependencies>

</project>
```

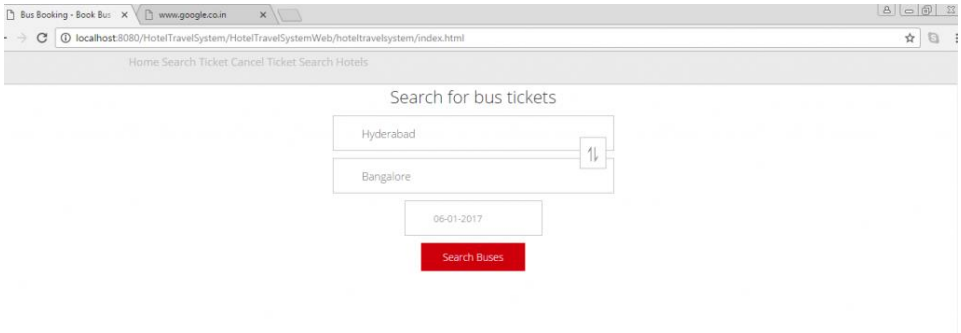
Web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xmlns="http://java.sun.com/xml/ns/javaee"
        xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd"
        id="WebApp_ID" version="3.0">



        <welcome-file-list>
            <welcome-file>index.jsp</welcome-file>
        </welcome-file-list>



        <servlet>
            <servlet-name>dispatcher</servlet-name>
            <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
            <init-param>
```

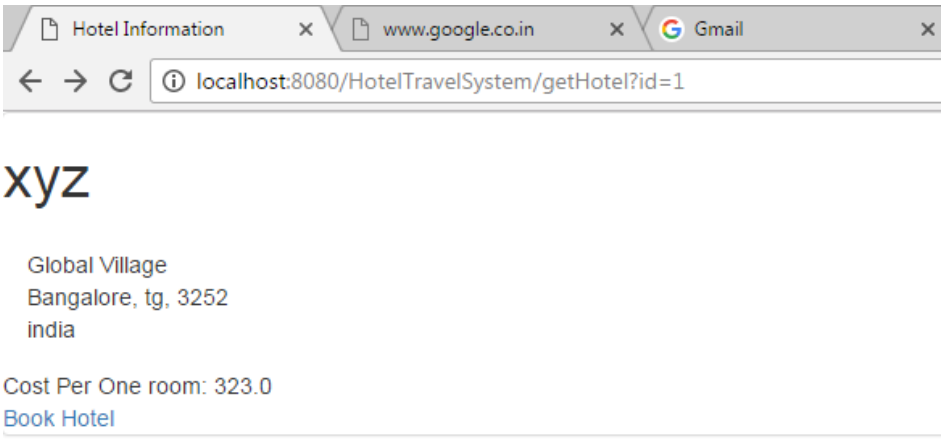
```
                        <param-name>contextConfigLocation</param-name>
                        <param-value>/WEB-INF/dispatcher-servlet.xml</param-value>
                </init-param>
                <load-on-startup>1</load-on-startup>

        </servlet>

        <servlet-mapping>
                <servlet-name>dispatcher</servlet-name>
                <url-pattern>/</url-pattern>
        </servlet-mapping>

        <context-param>
                <param-name>contextConfigLocation</param-name>
                <param-value>/WEB-INF/applicationContext.xml</param-value>
        </context-param>

        <listener>
                <listener-class>org.springframework.web.context.ContextLoaderListener</listener-
class>
        </listener>

</web-app>
```

Output:

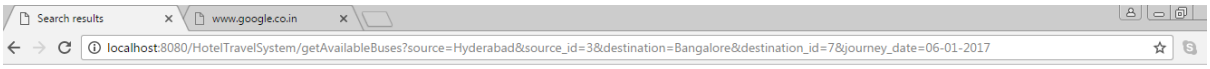
Searching Page (index page)



Show result



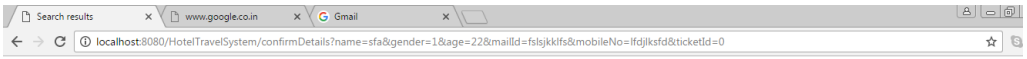
Availability of Bus



Available Buses

Sorry! No Buses Available for your search

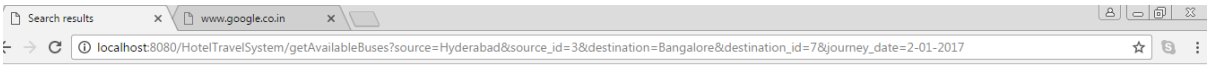
Bus No	Source	Destination	Book	cost



Hotels

your ticket with Id ecUBr is confirmed You can try the hotel which is based on your destination

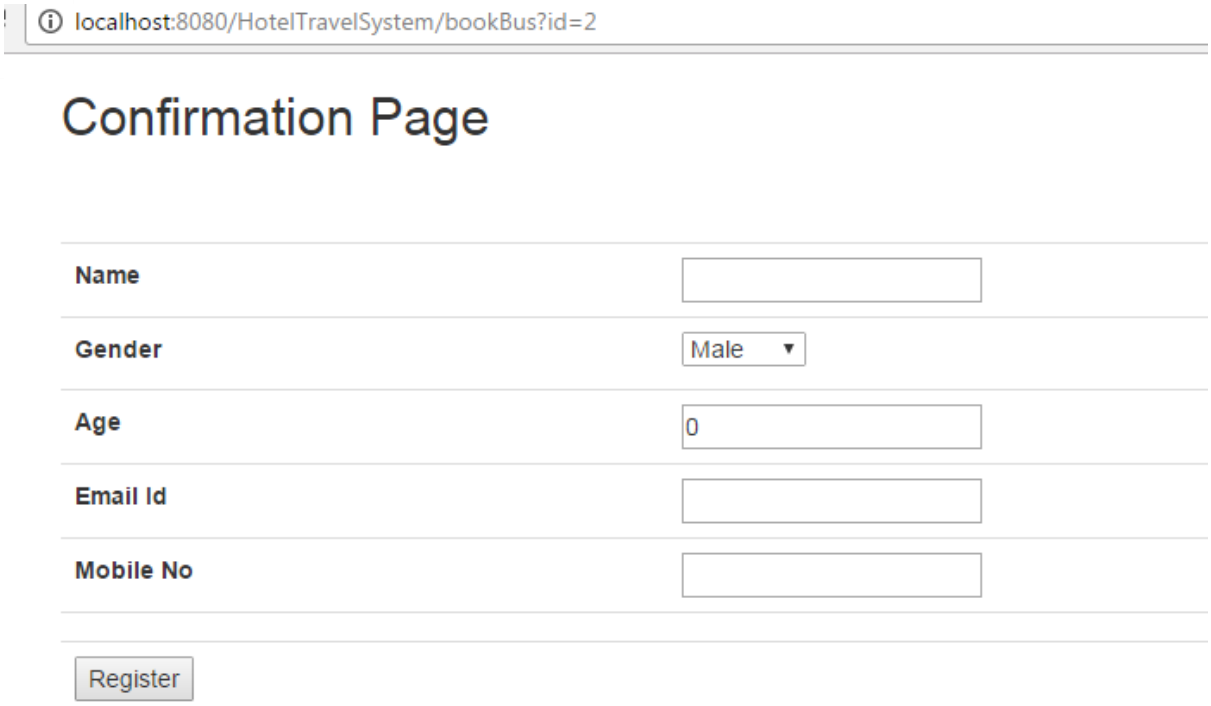
Name	Area	City,State	Zip	Action
xyz	Global Village tg	Bangalore tg	5534	View Hotel



vailable Buses

ius No	Source	Destination	Book	cost
2345	Hyderabad	Bangalore	1477	Book
432	Hyderabad	Bangalore	2532	Book
532	Hyderabad	Bangalore	549	Book

Booking Bus (confirm Page)



Second Project Reference from

: [Hotel Reservation Application Spring MVC Hibernate Project – 1000 Projects]

Thank you so much