Laboratory 9

Title of the Laboratory Exercise: Functionality implementation

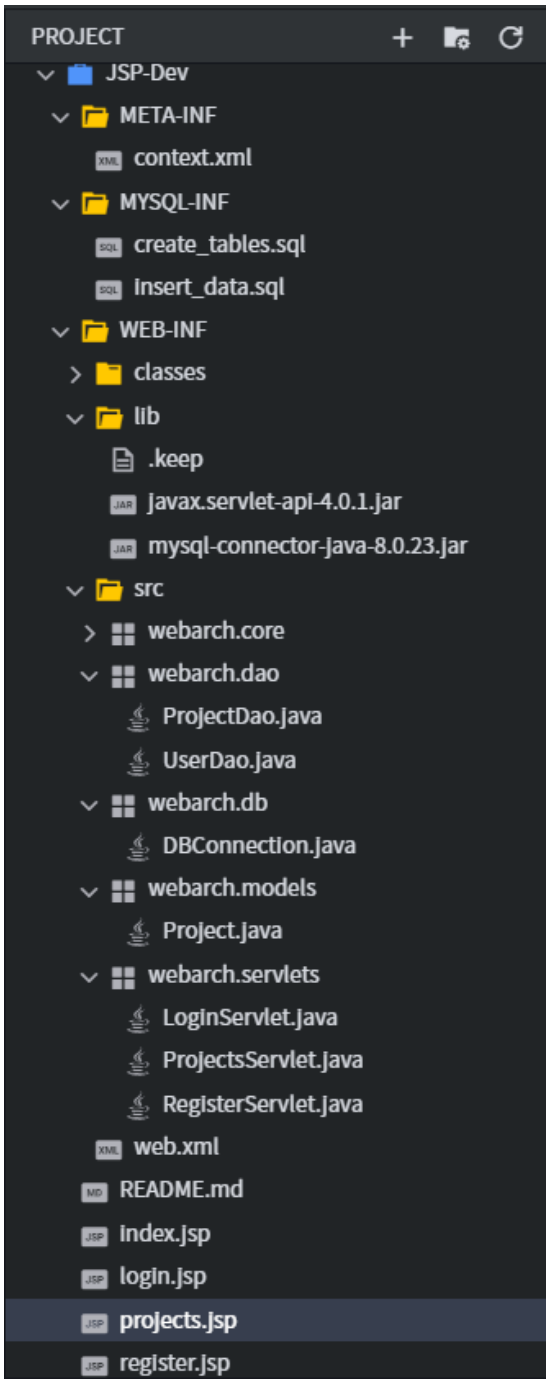
1. Introduction and Purpose of Experiment
2. Aim and Objectives

   Aim

3. Experimental Procedure
4. Calculations/Computations/Algorithms

NAME: SATYAJIT GHANA REG NO: 17ETCS002159

Project Structure



web.xml

```
<web-app>
    <servlet>
        <servlet-name>index</servlet-name>
        <jsp-file>/index.jsp</jsp-file>
    </servlet>
    <servlet-mapping>
        <servlet-name>index</servlet-name>
        <url-pattern>/</url-pattern>
    </servlet-mapping>

    <servlet>
        <servlet-name>register</servlet-name>
        <servlet-class>webarch.servlets.RegisterServlet</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>register</servlet-name>
        <url-pattern>/register</url-pattern>
    </servlet-mapping>

    <servlet>
        <servlet-name>login</servlet-name>
        <servlet-class>webarch.servlets.LoginServlet</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>login</servlet-name>
        <url-pattern>/login</url-pattern>
    </servlet-mapping>

    <servlet>
        <servlet-name>projects</servlet-name>
        <servlet-class>webarch.servlets.ProjectsServlet</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>projects</servlet-name>
        <url-pattern>/projects</url-pattern>
    </servlet-mapping>
</web-app>
```

webarch.servlets.LoginServlet.java

```
package webarch.servlets;

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

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import webarch.dao.UserDao;
```

```
import javax.servlet.*;
import javax.servlet.http.*;

public class LoginServlet extends HttpServlet {
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        RequestDispatcher rd = request.getRequestDispatcher("login.jsp");
        rd.include(request, response);
    }
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        String username = request.getParameter("username");
        String password = request.getParameter("password");

        boolean loginStatus = UserDao.loginUser(username, password);

        if (loginStatus == true) {
            response.sendRedirect("/projects");
        } else {
            response.setContentType("text/html");
            PrintWriter pw = response.getWriter();
            pw.println("<script type=\"text/javascript\">");
            pw.println("alert('Username or Password incorrect ');");
            pw.println("</script>");
            RequestDispatcher rd = request.getRequestDispatcher("login.jsp");
            rd.include(request, response);
        }

    }
}
```

webarch.servlets.ProjectsServlet.java

```
package webarch.servlets;

import java.io.IOException;
import java.io.PrintWriter;
import java.util.ArrayList;
import java.util.List;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import webarch.dao.ProjectDao;
import webarch.models.Project;

import javax.servlet.*;
import javax.servlet.http.*;
```

```
public class ProjectsServlet extends HttpServlet {
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {

        List<Project> projects;

        if (request.getParameterMap().containsKey("projectname")) {
            String projectname = request.getParameter("projectname");
            projects = ProjectDao.findProjects(projectname);
        } else {
            projects = new ArrayList<>();
        }

        request.setAttribute("projects", projects);

        RequestDispatcher rd = request.getRequestDispatcher("projects.jsp");
        rd.include(request, response);

    }
}
```

webarch.servlets.RegisterServlet.java

```
package webarch.servlets;

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

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import webarch.dao.UserDao;

import javax.servlet.*;
import javax.servlet.http.*;

public class RegisterServlet extends HttpServlet {
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        RequestDispatcher rd = request.getRequestDispatcher("register.jsp");
        rd.include(request, response);
    }
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        String username = request.getParameter("username");
        String password = request.getParameter("password");
        String fullname = request.getParameter("fullname");
```

```
        String usnno = request.getParameter("usnno");
        String dept = request.getParameter("dept");
        String course = request.getParameter("course");

        boolean registerStatus = UserDao.registerUser(username, password, fullname, usnno, dept, course);


        response.setContentType("text/html");
        PrintWriter pw = response.getWriter();
        pw.println("<script type=\"text/javascript\">");
        pw.println("alert('Register Status: [ "+registerStatus+"] ');");
        pw.println("</script>");

        // if register success then send to login page
        // RequestDispatcher rd = request.getRequestDispatcher("login.jsp");

        RequestDispatcher rd = request.getRequestDispatcher("register.jsp");
        rd.include(request, response);
    }
}
```

webarch.models.Project.java

```
package webarch.models;

public class Project {
    private Integer id;
    private String projectLeaderRegno;
    private String projectName;
    private String mentorName;
    private String department;
    private String category;


    public Project(Integer id, String projectLeaderRegno, String projectName, String mentorName, String dep
artment, String category) {
        this.id = id;
        this.projectLeaderRegno = projectLeaderRegno;
        this.projectName = projectName;
        this.mentorName = mentorName;
        this.department = department;
        this.category = category;
    }

    public Integer getId() {
        return this.id;
    }

    public String getProjectLeaderRegno() {
        return this.projectLeaderRegno;
    }
```

```
    public String getProjectName() {
        return this.projectName;
    }

    public String getMentorName() {
        return this.mentorName;
    }

    public String getDepartment() {
        return this.department;
    }

    public String getCategory() {
        return this.category;
    }
}
```

webarch.db.DBConnection.java

```
package webarch.db;

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

public class DBConnection {
    private static Connection conn;

    public static Connection getDbConnection() {
        if (conn == null) {
            try {
                Class.forName("com.mysql.jdbc.Driver");
                conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/webarch", "root", "");
            } catch (SQLException e) {
                e.printStackTrace();
            } catch (ClassNotFoundException e) {
                e.printStackTrace();
            }
        }

        return conn;
    }
}
```

webarch.dao.ProjectDao.java

```
package webarch.dao;

import java.sql.Connection;
import java.sql.PreparedStatement;
```

```
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;

import webarch.db.DBConnection;
import webarch.models.Project;

public class ProjectDao {
    public static List<Project> findProjects(String projectName) {
        Connection conn = DBConnection.getDbConnection();

        List<Project> projects = new ArrayList<>();

        try {

            PreparedStatement stmt = conn.prepareStatement("SELECT * FROM PROJEKT WHERE project_name LIKE ?
");
            stmt.setString(1, "%"+projectName+"%");

            ResultSet rs = stmt.executeQuery();



            while (rs.next()) {
                projects.add(new Project(rs.getInt(1), rs.getString(2), rs.getString(3), rs.getString(4), r
s.getString(5), rs.getString(6)));
            }

            return projects;


        } catch (SQLException e) {
            e.printStackTrace();
        }

        return projects;
    }
}
```

webarch.dao.UserDao.java

```
package webarch.dao;

import java.math.BigInteger;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
```

```java
import java.sql.Statement;

import webarch.db.DBConnection;

public class UserDao {

    public static String getMd5(String input) {
        try {

            // Static getInstance method is called with hashing MD5
            MessageDigest md = MessageDigest.getInstance("MD5");

            // digest() method is called to calculate message digest
            //  of an input digest() return array of byte
            byte[] messageDigest = md.digest(input.getBytes());

            // Convert byte array into signum representation
            BigInteger no = new BigInteger(1, messageDigest);

            // Convert message digest into hex value
            String hashtext = no.toString(16);
            while (hashtext.length() < 32) {
                hashtext = "0" + hashtext;
            }
            return hashtext;
        }

        // For specifying wrong message digest algorithms
        catch (NoSuchAlgorithmException e) {
            throw new RuntimeException(e);
        }
    }

    public static boolean loginUser(String username, String password) {
        Connection conn = DBConnection.getDbConnection();

        try {
            PreparedStatement stmt = conn.prepareStatement("SELECT id, hashed_password FROM `STUDENT_LOGIN`
 WHERE `user_name` = ? LIMIT 1");
            stmt.setString(1, username);

            ResultSet rs =  stmt.executeQuery();

            if (rs.next()) {
                String hashedPassword = rs.getString(2);

                if (hashedPassword.equals(getMd5(password))) {

                    System.out.println("Login Success for User: " + username);

                    return true;
```

```
                }
            }

            return false;

        } catch (SQLException e) {
            e.printStackTrace();
        }

        return false;
    }

    public static boolean registerUser(String username, String password, String fullname, String usnno, Str
ing dept, String course ) {
        Connection conn = DBConnection.getDbConnection();

        try {
            PreparedStatement stmt = conn.prepareStatement("INSERT INTO `STUDENT_LOGIN` (`user_name`, `hash
ed_password`) VALUES(?, ?)", Statement.RETURN_GENERATED_KEYS);
            stmt.setString(1, username);
            stmt.setString(2, getMd5(password));

            stmt.executeUpdate();

            ResultSet rs = stmt.getGeneratedKeys();
            if (rs.next()) {
                Integer id = rs.getInt(1);

                PreparedStatement stmt2 = conn.prepareStatement("INSERT INTO STUDENT( id, reg_no, name, dep
artment, course, contact_no ) VALUES (?, ?, ?, ?, ?, ?)");
                stmt2.setInt(1, id);
                stmt2.setString(2, usnno);
                stmt2.setString(3, fullname);
                stmt2.setString(4, dept);
                stmt2.setString(5, course);
                stmt2.setString(6, "9999999999"); // hardcoded value for now

                int count = stmt2.executeUpdate();

                if (count > 0)
                    return true;
                else
                    return false;
            }
            rs.close();
        } catch (SQLException e) {
            e.printStackTrace();
            return false;
        }

        return false;
```

```
    }
}
```

register.jsp

```
<%@ page contentType = "text/html;charset=utf-8" %>
<html>
    <head>
    <title>Register</title>

    <script>

        function validateForm() {
            var username = document.regform.username.value;
            var password = document.regform.password.value;
            var fullname = document.regform.fullname.value;
            var usnno    = document.regform.usnno.value;

            if (username == "") {
                alert("username cannot be blank");
                return false;
            } else if (password == "") {
                alert("password cannot be blank");
                return false;
            } else if (fullname == "") {
                alert("fullname cannot be blank");
                return false;
            } else if (usnno == "") {
                alert("usnno cannot be blank");
                return false;
            } else if (password.length < 6) {
                alert("password must be at least 6 characters long");
                return false;
            }

            return true;
        }

    </script>
    </head>

    <body style="text-align:center; margin:auto">
        <h1>Register</h1>
        <br/>
        <center>
            <form align="center" name="regform" action="register" method="post" onsubmit="return validateFo
rm();">
                <table>
                    <tr>
                        <td>username</td>
```

```
                    <td><input type="text" name="username" /></td>
                </tr>
                <tr>
                    <td>password</td>
                    <td><input type="password" name="password" /></td>
                </tr>
                <tr>
                    <td>full name</td>
                    <td><input type="text" name="fullname" /></td>
                </tr>
                <tr>
                    <td>usn no</td>
                    <td><input type="text" name="usnno" /></td>
                </tr>
                <tr>
                    <td>dept</td>
                    <td>
                        <select name="dept">
                          <option value="CSE">CSE</option>
                          <option value="EEE">EEE</option>
                          <option value="ECE">ECE</option>
                          <option value="CIVIL">CIVIL</option>
                        </select>
                    </td>
                </tr>
                <tr>
                    <td>course</td>
                    <td>
                        <select name="course">
                          <option value="B.Tech">B.Tech</option>
                          <option value="M.Tech">M.Tech</option>
                        </select>
                    </td>
                </tr>
            </table>
            <br/>
            <input type="submit" value="register" />
        </form>
    </center>
</body>
</html>
```

login.jsp

```
<%@ page contentType = "text/html;charset=utf-8" %>
<html>
    <head>
    <title>Login</title>
    </head>

    <body style="text-align:center; margin:auto">
        <h1>Login to RUAS LMS</h1>
```

```
        <br/>
        <center>
            <form align="center" action="login" method="post">
                <table>
                    <tr>
                        <td>username</td>
                        <td><input type="text" name="username" /></td>
                    </tr>
                    <tr>
                        <td>password</td>
                        <td><input type="password" name="password" /></td>
                    </tr>
                </table>
                <br/>
                <input type="submit" value="login" />
            </form>
        </center>
    </body>
</html>
```

projects.jsp

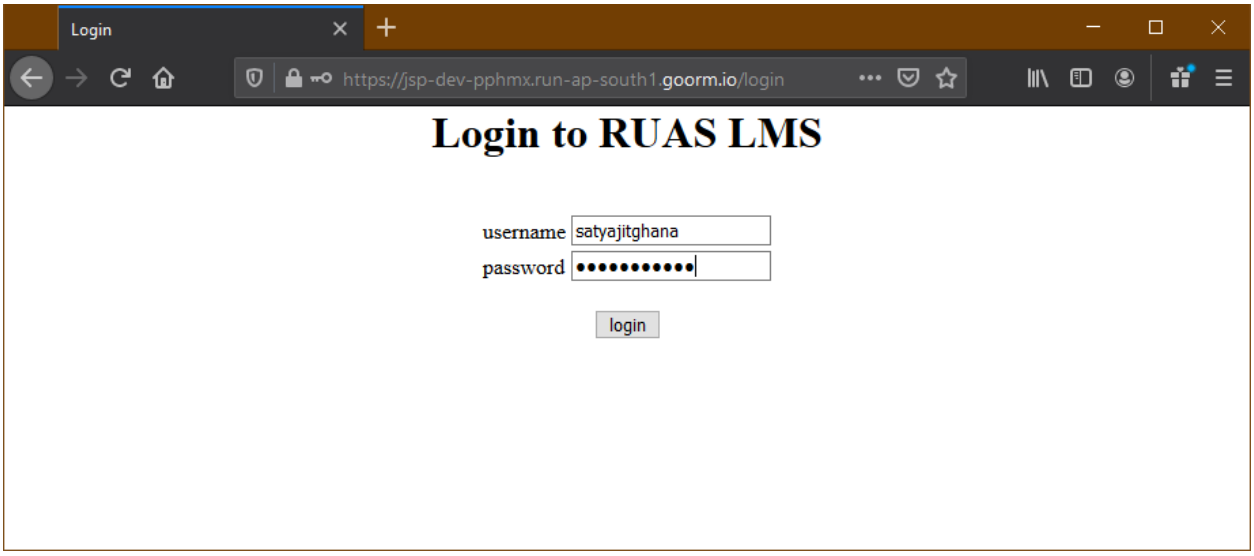
```
<%@ page contentType = "text/html;charset=utf-8" %>
<%@ page import="java.util.List" %>
<%@ page import="webarch.models.Project" %>
<html>
    <head>
    <title>Project</title>
    </head>

    <body style="text-align:center; margin:auto">
        <h1>Search for Project Details</h1>
        <br/>
        <center>
            <form align="center" action="projects" method="get">
                <table>
                    <tr>
                        <td>project name</td>
                        <td>
                            <input type="text" name="projectname" />
                        </td>
                    </tr>
                </table>
                <br />
                <input type="submit" value="search" />
            </form>
            <br />
            <h2>Results</h2>
            <table>
                <tr>
                    <th>Id</th>
                    <th>Project Leader</th>
```

```
                    <th>Project Name</th>
                    <th>Mentor Name</th>
                    <th>Department</th>
                    <th>Category</th>
                </tr>
                <%
                    List<Project> projects = (List<Project>) request.getAttribute("projects");
                    if (projects != null) {
                        if (projects.size() == 0) {
                %>
                            <h4> No results found</h4>
                <%
                        } else {
                            for (Project project: projects) {
                %>
                                <tr>
                                    <td>
                                        <%out.write(project.getId().toString());%>
                                    </td>
                                    <td>
                                        <%out.write(project.getProjectLeaderRegno());%>
                                    </td>
                                    <td>
                                        <%out.write(project.getProjectName());%>
                                    </td>
                                    <td>
                                        <%out.write(project.getMentorName());%>
                                    </td>
                                    <td>
                                        <%out.write(project.getDepartment());%>
                                    </td>
                                    <td>
                                        <%out.write(project.getCategory());%>
                                    </td>
                                </tr>
                <%
                            }
                        }
                    }
                %>
            </table>
        </center>
    </body>
</html>
```

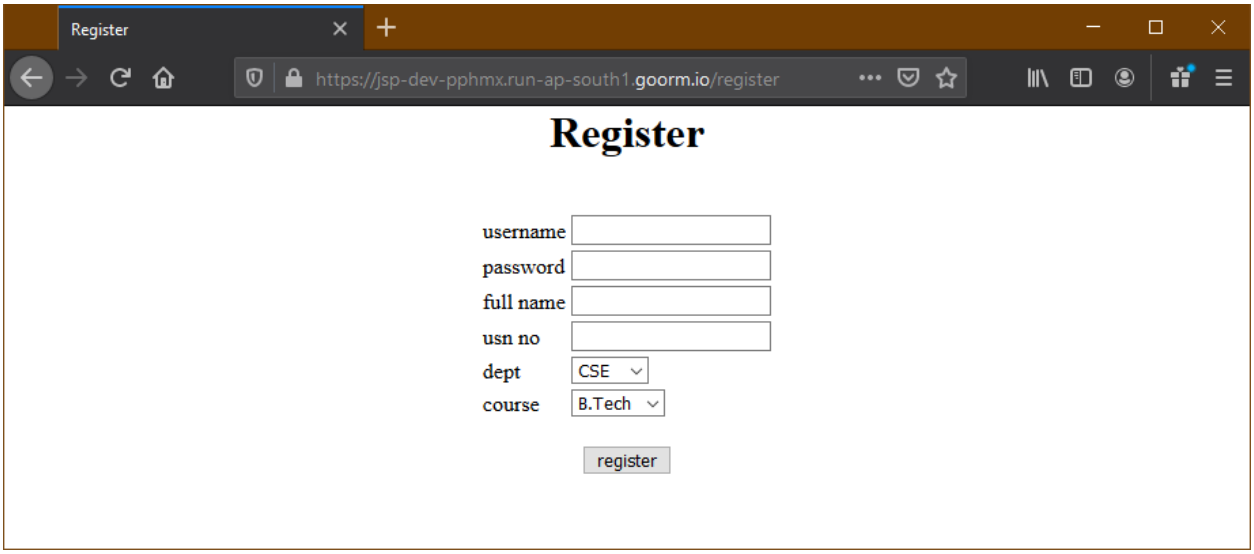
5. Presentation of Results

NAME: SATYAJIT GHANA REG NO: 17ETCS002159

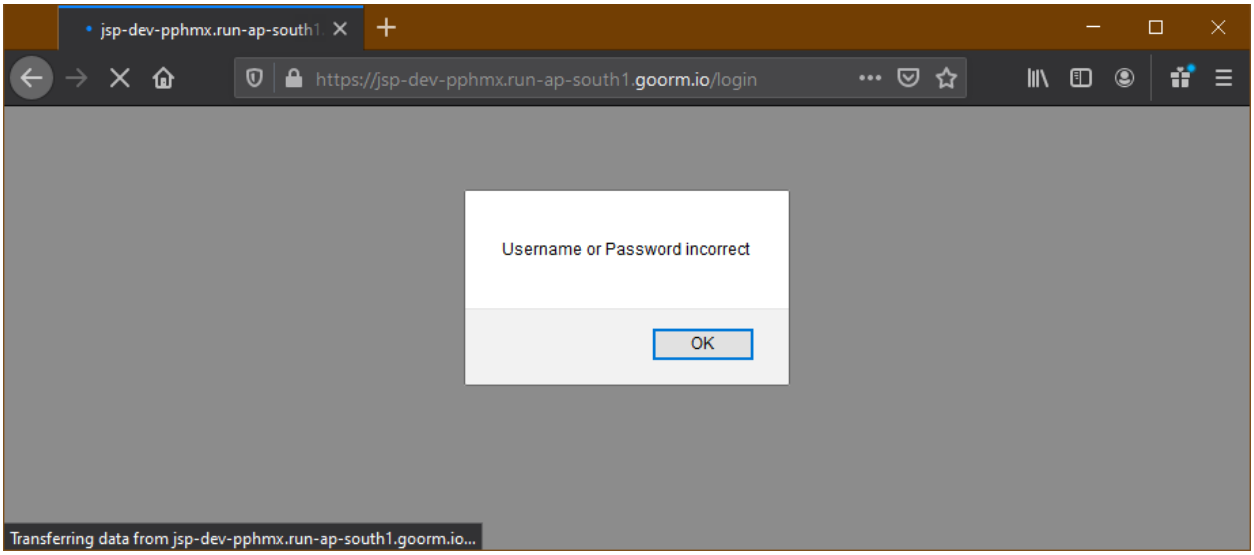
Login Page (On Success redirects to Search Page)



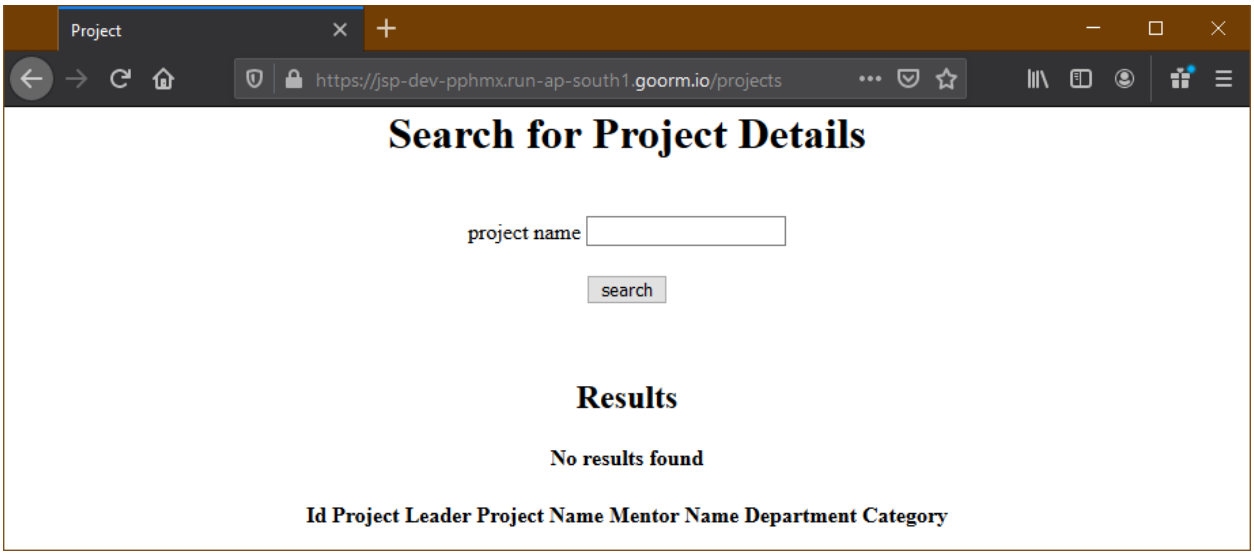
Register Page (On Success redirects to Login Page)



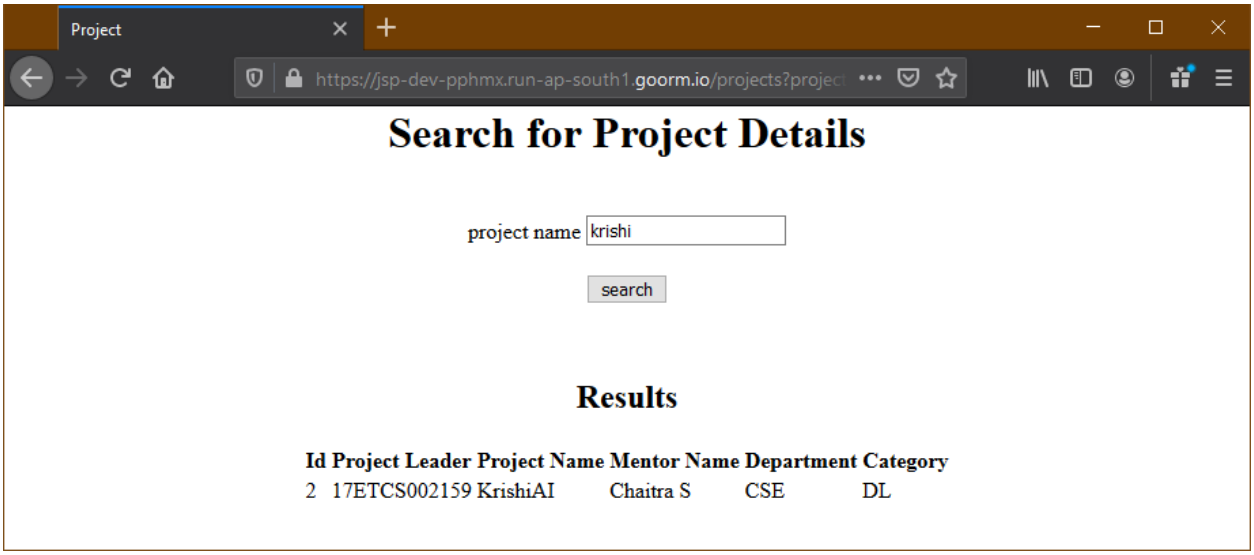
Login Page (Error)



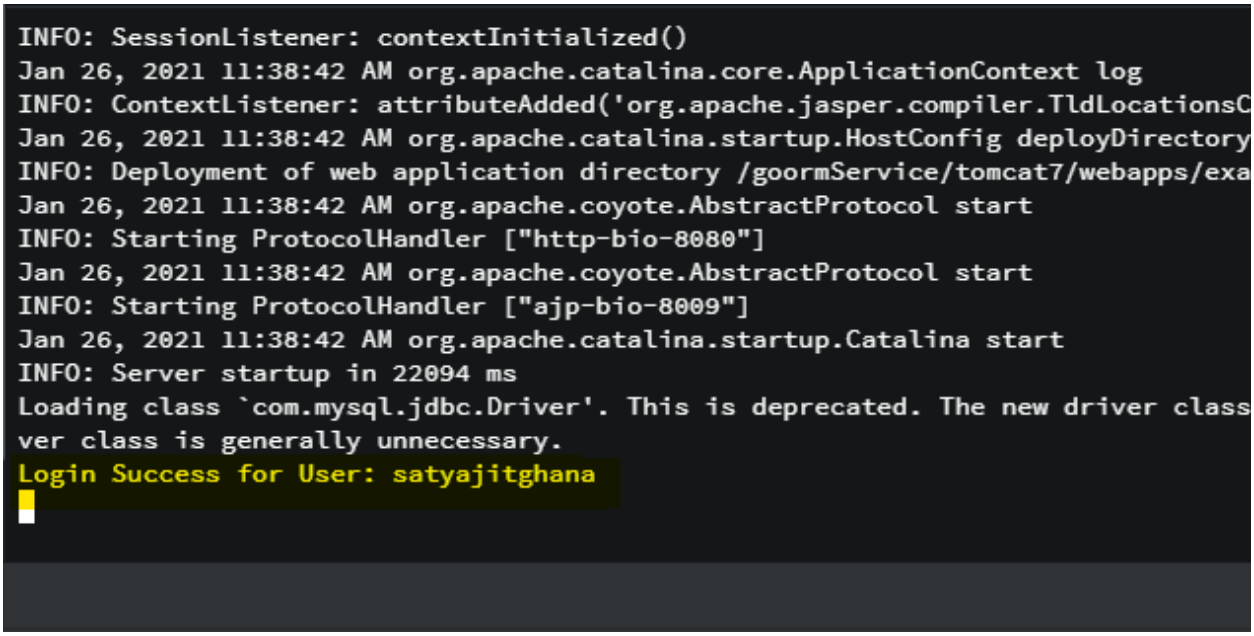
Search Page



Search for a term



Logs



6. Analysis and Discussions

Servlet can accept all protocol requests, including HTTP, while JSP can only accept HTTP requests.

In MVC architecture, servlet works as a controller while JSP works as a view for displaying output.

Servlet should be used when there is more data processing involved whereas, JSP is generally used when there is less involvement of data processing.

We learnt how to use Servlets and connect them with JSPs, and make a complete working website flow. Java makes it easy to connect web pages, and with good UX as well.

7. Conclusions

We were able to create a Project Exhibition Management Website with functionalities like Login, Register and Searching for a Registered Project Details.

8. Comments

a. Limitations of Experiments

Although Servlets are easy to use, they are being deprecated, the times are changing and people are moving towards robust and much easier to use frameworks like React, Vue, Svelte and Angular. They make it really easy to do routing, managing state and using the latest JS capabilities of the browser.

b. Limitations of Results

The website implemented here, does not use dynamic loading, i.e. the buttons invoke a request and we get the result, the whole page is refreshed, and not just the required parts of the website. This makes the UX really not that usable.

c. Learning happened

We learnt how to implement the functionalities of a website, from designing, to creating the DB and finally creating the JSP, Servlets and connecting all of these together.

d. Recommendations

It would be really beneficial to use something like React or Vue, as they are really evolved and meant for the modern browsers.

Component	Max Marks	Marks Obtained
Viva	6	
Results	7	
Documentation	7	
Total	20	