

Department of Computer Science & Applications
(B.Sc. - Computer Science with Specialization in Cyber Security)

PRACTICAL RECORD

NAME :

REGISTER NO :

COURSE : B.Sc. - Computer Science with Specialization in
Cyber Security

SEMESTER / YEAR : IV / II

SUBJECT CODE : UCS23401J

SUBJECT NAME : ENTERPRISE JAVA PROGRAMMING

APRIL 2025



SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
Ramapuram, Chennai
FACULTY OF SCIENCE AND HUMANITIES

Department of Computer Science & Applications

(B.Sc. - Computer Science with Specialization in Cyber Security)

REGISTER NUMBER:

BONAFIDE CERTIFICATE

This is to certify that the bonafide work done by _____ in the subject

ENTERPRISE JAVA PROGRAMMING [UCS23401J] at SRM Institute of Science and Technology,

Ramapuram, Chennai in April 2025.

STAFF IN CHARGE

HEAD OF THE DEPARTMENT

Submitted for the University Practical Examination held at SRM Institute of Science and Technology,

Ramapuram, Chennai on _____.

INTERNAL EXAMINER 1

INTERNAL EXAMINER 2

INDEX

S.No	Date	List of Experiments	PageNo
1.		Create distributed applications using RMI	
2.		Create applications which can demonstrate the use of JDBC for Database Connectivity	
3.		Create Student applications using JDBC Database Connectivity	
4.		Develop web applications using Servlet.	
5.		Develop web applications using Servlet Request , Servlet Response	
6.		Program that demonstrates the use of session management in Servlet	
7.		Develop web Applications using JSP	
8.		Create applications using Include Directive JSP Include Action	
9.		Create a JSP based web application which allows the user to edit his/her database Information	
10.		An EJB application that demonstrates Session Bean. - Stateless Bean	
11.		An EJB application that demonstrates Session Bean. – Stateful Bean	
12.		An EJB application that demonstrates Entity Bean.	
13.		MVC Architecture : Implementing MVC with Request Dispatcher Data Sharing Approaches	
14.		Build a web application that displays “Hello World” using Spring Framework	
15.		Create our view which will be required to browse and upload a selected file.	

EX.NO: 1

DATE:

ADDITION OF TWO NUMBERS USING RMI

AIM:

To add two Numbers using RMI concepts

PROCEDURE:

1. Open four notepad and type the following code with the respective name as given in program.
2. Save all the four files inside the folder rmi.
3. Open command prompt, go to rmi folder and compile all the four files using javac command.
4. Compile the implementation program (AdderRemote.java) using the rmic command.
5. Open the rmi registry using the command "start rmiregistry". Minimize the rmi registry window.
6. Start the server using the command "java".
7. Open a new command prompt window, go to rmi folder and start the client using command "java".
8. Exit from all the command prompt window.

PROGRAM :

1. Adder.java

```
import java.rmi.*;
public interface Adder extends Remote
{
    public int add(int x,int y)throws RemoteException;
}
```

2. AdderRemote.java

```
import java.rmi.*; import java.rmi.server.*;
public class AdderRemote extends UnicastRemoteObject implements Adder{
    AdderRemote()throws RemoteException{
        super();
    }
    public int add(int x,int y)
    {
        return x+y;}
}
```

3. MyServer.java

```
import java.rmi.*;
import java.rmi.registry.*;
public class MyServer{
    public static void main(String args[]){
        try{
            Adder stub=new AdderRemote();
            Naming.rebind("myobj",stub); }
        catch(Exception e){
            System.out.println(e);
        }
    }
}
```

```
}
```

4. MyClient.java

```
import java.rmi.*;
public class
MyClient{
public static void main(String args[]){try{
Adder stub=(Adder)Naming.lookup("myobj");
System.out.println(stub.add(34,4));
}
catch(Exception e){}
}}
```

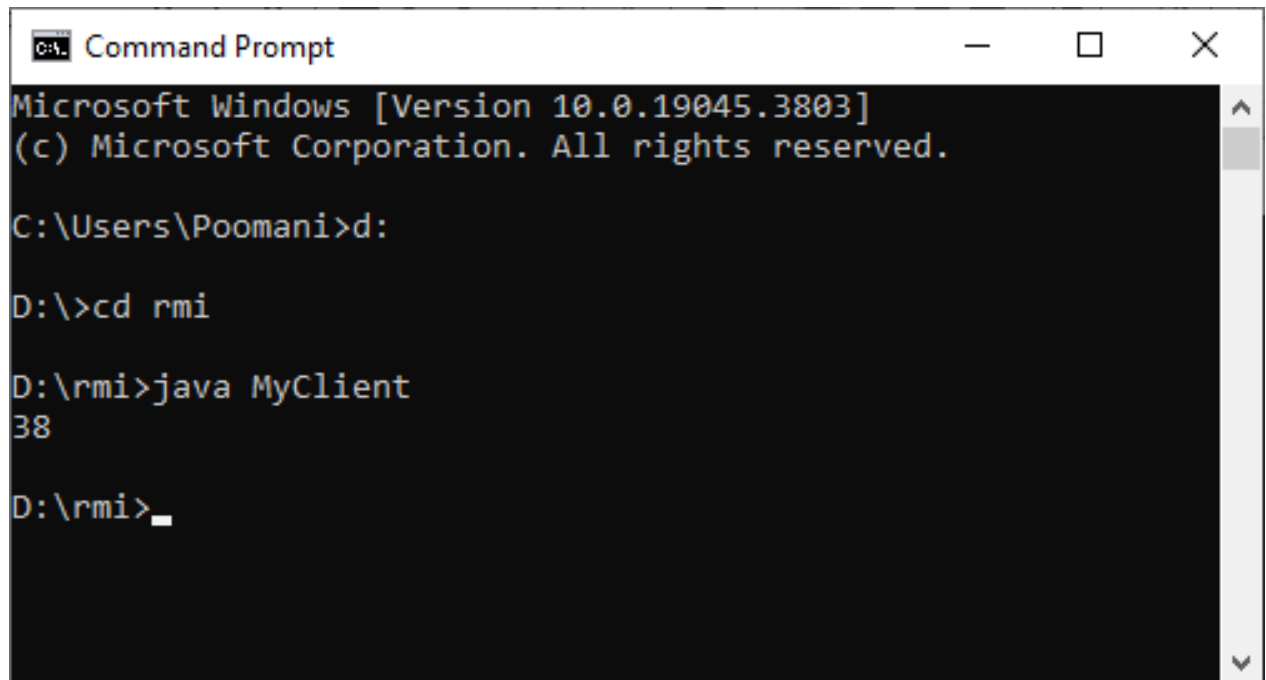
Steps to run this rmi example:

1. Compile all the java files
javac *.java
2. Create stub and skeleton object by rmic tool
rmic AdderRemote
3. Start rmi registry in one command prompt
start rmiregistry
4. Start the server in another command prompt
java MyServer
5. Start the client application in another command prompt
java MyClient

OUTPUT:



```
C:\> Command Prompt - Java MyServer
C:\Users\Poomani>d:
D:\>cd rmi
D:\rmi>javac *.java
D:\rmi>rmic AdderRemote.java
Warning: generation and use of skeletons and static stubs for JRMP
is deprecated. Skeletons are unnecessary, and static stubs have
been superseded by dynamically generated stubs. Users are
encouraged to migrate away from using rmic to generate skeletons and static
stubs. See the documentation for java.rmi.server.UnicastRemoteObject.
error: Class AdderRemote$java not found.
1 error
D:\rmi>start rmiregistry
D:\rmi>Java MyServer
```



```
Microsoft Windows [Version 10.0.19045.3803]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Poomani>d:

D:\>cd rmi

D:\rmi>java MyClient
38

D:\rmi>_
```

RESULT:

Thus the program has been executed successfully and output verified.

EX.NO: 2

DATE:

CREATE APPLICATIONS USING JDBC

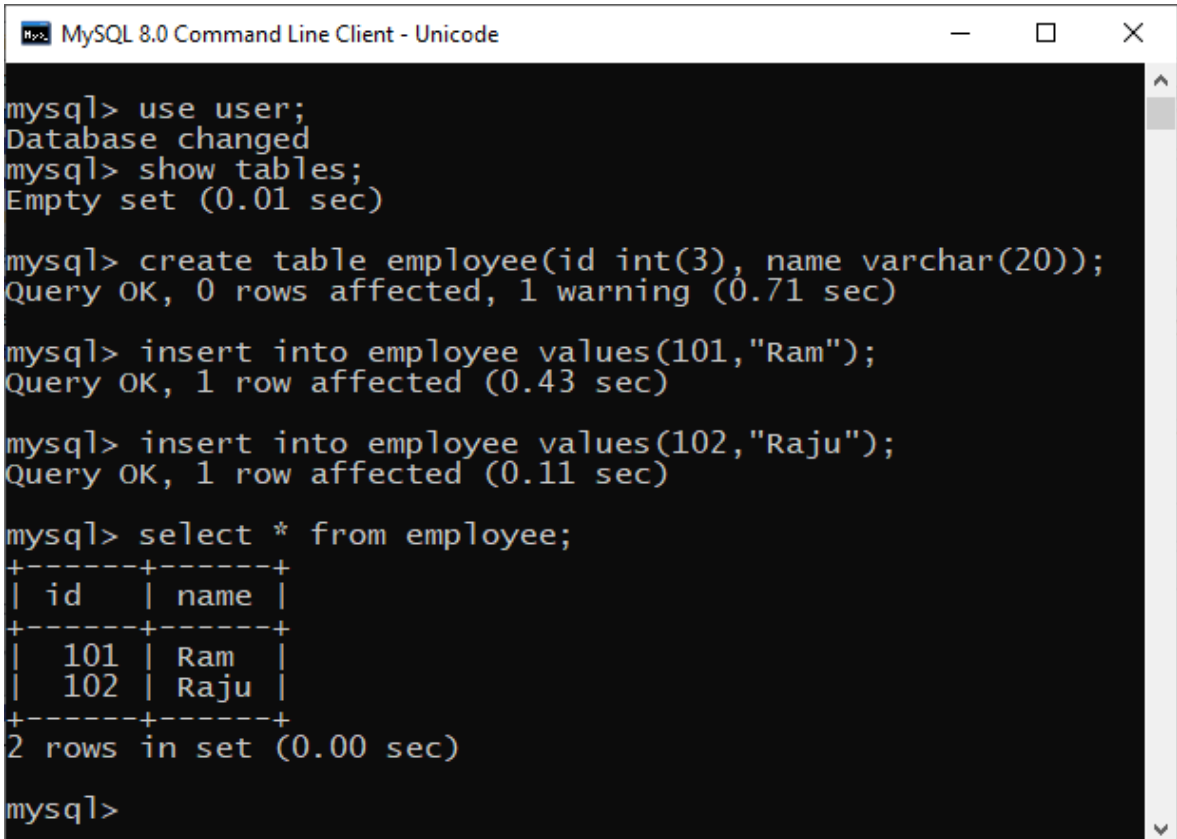
AIM:

To retrieve the records from employee table using Mysql Database and JDBC.

PROCEDURE:

1. Open mysql command line and create a database, table and insert two rows of data into the table.

- a. mysql> create database user;
- b. mysql> use user;
- c. mysql> show tables;
- d. mysql> create table employee(id int(3), name varchar(20));
- e. mysql> insert into employee values(101,"Ram");
- f. mysql> insert into employee values(102,"Raju");
- g. mysql> select * from employee;



```
MySQL 8.0 Command Line Client - Unicode
mysql> use user;
Database changed
mysql> show tables;
Empty set (0.01 sec)

mysql> create table employee(id int(3), name varchar(20));
Query OK, 0 rows affected, 1 warning (0.71 sec)

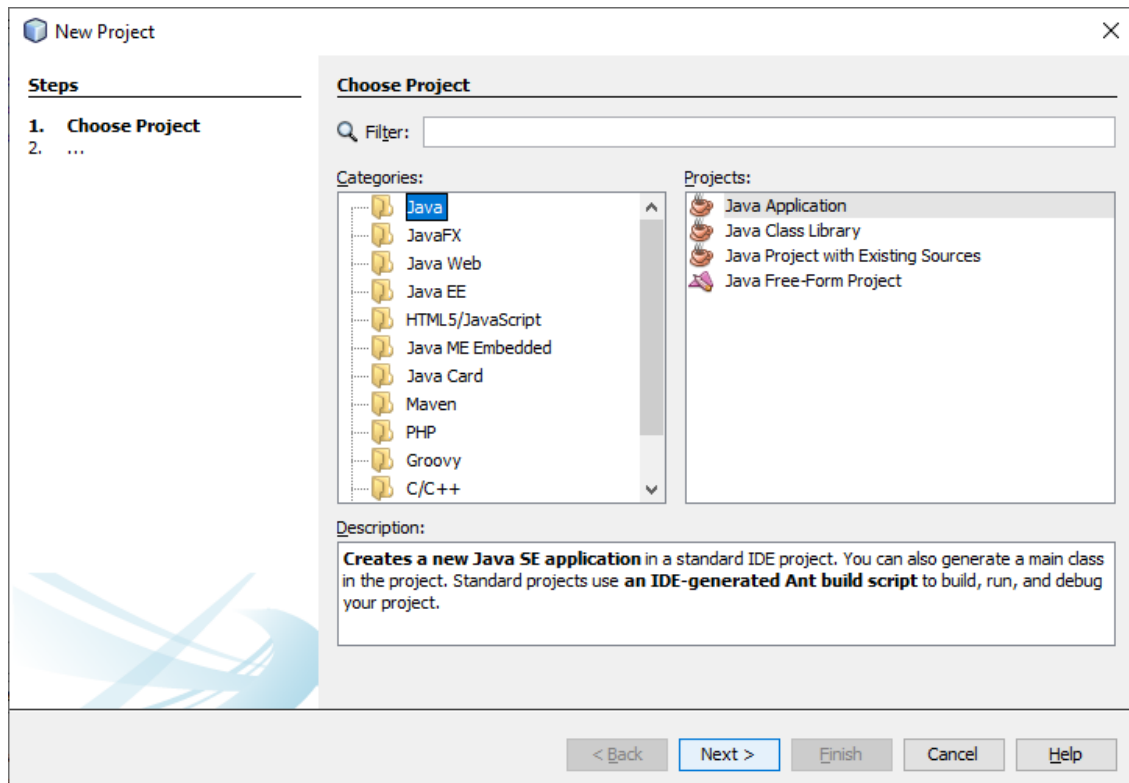
mysql> insert into employee values(101,"Ram");
Query OK, 1 row affected (0.43 sec)

mysql> insert into employee values(102,"Raju");
Query OK, 1 row affected (0.11 sec)

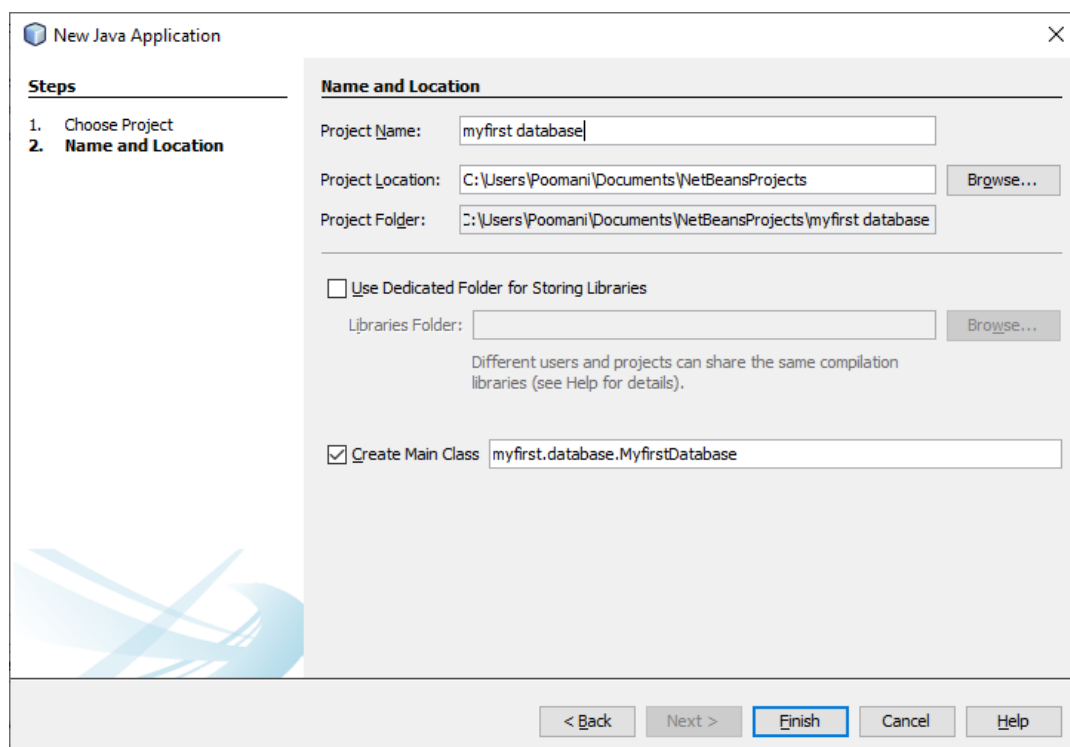
mysql> select * from employee;
+-----+-----+
| id  | name |
+-----+-----+
| 101 | Ram  |
| 102 | Raju |
+-----+-----+
2 rows in set (0.00 sec)

mysql>
```

2. Start Netbeans IDE , Go to file menu and select new project, The new project dialog box shown as below opens. From the Categories select the option java and from projects select java Application and click Next.



3. Type the name of the project as “myfirstdatabase” and select a location to save the project and click Finish.



4. In the project tab right click the library folder of your application and select the option “Add JAR/Folder” and go to the location where we have saved our mysql connector JAR file. Select the JAR file and it will be added to your project.

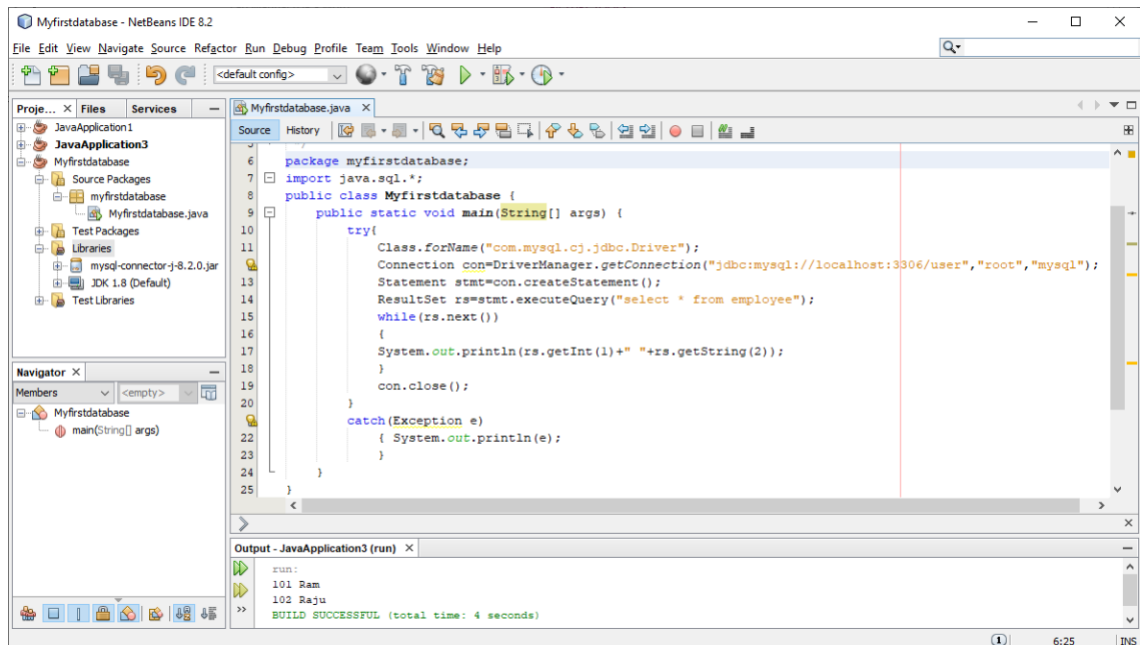
5. Type the following code in the main method.

6. Run the code by clicking the run tool or by selecting run option from run menu.

PROGRAM:

```
package myfirstdatabase;
import java.sql.*;
public class Myfirstdatabase
{
    public static void main(String[] args)
    {
        try
        {
            Class.forName("com.mysql.cj.jdbc.Driver");
            Connection
                con=DriverManager.getConnection("jdbc:mysql://localhost:3306/user","root","mysql");
            Statement stmt=con.createStatement();
            ResultSet rs=stmt.executeQuery("select * from employee");
            while(rs.next())
            {
                System.out.println(rs.getInt(1)+" "+rs.getString(2));
            }
            con.close();
        }
        catch(Exception e)
        {
            System.out.println(e);
        }
    }
}
```

OUTPUT:



RESULT:

Thus the program has been executed successfully and output verified.

EX.NO: 3

DATE:

CREATE STUDENT APPLICATIONS USING JDBC

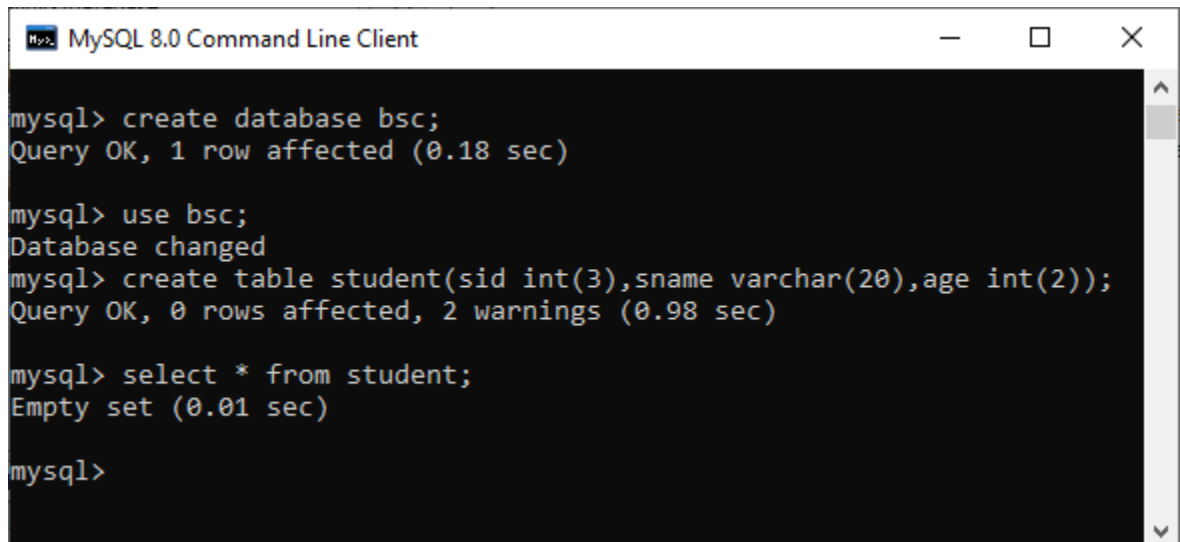
AIM:

To create a student application using MySQL Database and JDBC.

PROCEDURE :

1. Open mysql command line and create a database and table as given below:

- a. mysql> create database bsc;
- b. mysql> use bsc;
- c. mysql> create table student(sid int(3),sname varchar(20),age int(2));;
- d. mysql> select * from student;



```
MySQL 8.0 Command Line Client

mysql> create database bsc;
Query OK, 1 row affected (0.18 sec)

mysql> use bsc;
Database changed
mysql> create table student(sid int(3),sname varchar(20),age int(2));
Query OK, 0 rows affected, 2 warnings (0.98 sec)

mysql> select * from student;
Empty set (0.01 sec)

mysql>
```

2. Start Netbeans IDE , Go to file menu and select new project, The new project dialog box shown as below opens. From the Categories select the option java and from projects select java Application and click Next.

3. Type the name of the project as “studentJDBC” and select a location to save the project and click Finish.

4. In the project tab right click the library folder of your application and select the option “Add JAR/Folder” and go to the location where we have saved our mysql connector JAR file. Select the JAR file and it will be added to your project.

5. Type the following code in the main method.

6. Run the code by clicking the run tool or by selecting run option from run menu.

PROGRAM:

```
import
java.sql.*;
import
java.io.*;
public class studentJDBC
{
    public static void main(String args[])
    {
        int choice; String n,ch;int id,age; int rows;
        BufferedReader br=new BufferedReader(new
        InputStreamReader(System.in)); try
        {
            Class.forName("com.mysql.cj.jdbc.Driver");
            Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/bsc","root","mysql");
            Statement s=con.createStatement();
            do
            {
                System.out.println("choice");
                System.out.println("1.Insertion\t 2.Deletion \t 3.Select contents from table");
                choice=Integer.parseInt(br.readLine());
                switch(choice)
                {
                    case 1:
                    {
                        String ss="insert into student(sid,sname,age)
                        values(?,?,?)"; PreparedStatement
                        ps=con.prepareStatement(ss);
                        System.out.println("enter student id");
                        id=Integer.parseInt(br.readLine());
                        System.out.println("enter student name");
                        n=br.readLine();
                        System.out.println("enter student
                        age");
                        age=Integer.parseInt(br.readLine());
                        ps.setInt(1,id);
                        ps.setString(2,n);
                        ps.setInt(3,age);
                        rows=ps.executeUpdate();
                        System.out.println(rows+"rows
                        inserted"); break;
                    }
                    case 2:
                    {
                        String ss="delete from student where id=?";
                        PreparedStatement
                        ps=con.prepareStatement(ss);
                        System.out.println("enter student id to
                        delete"); id=Integer.parseInt(br.readLine());
                        ps.setInt(1,id);
```

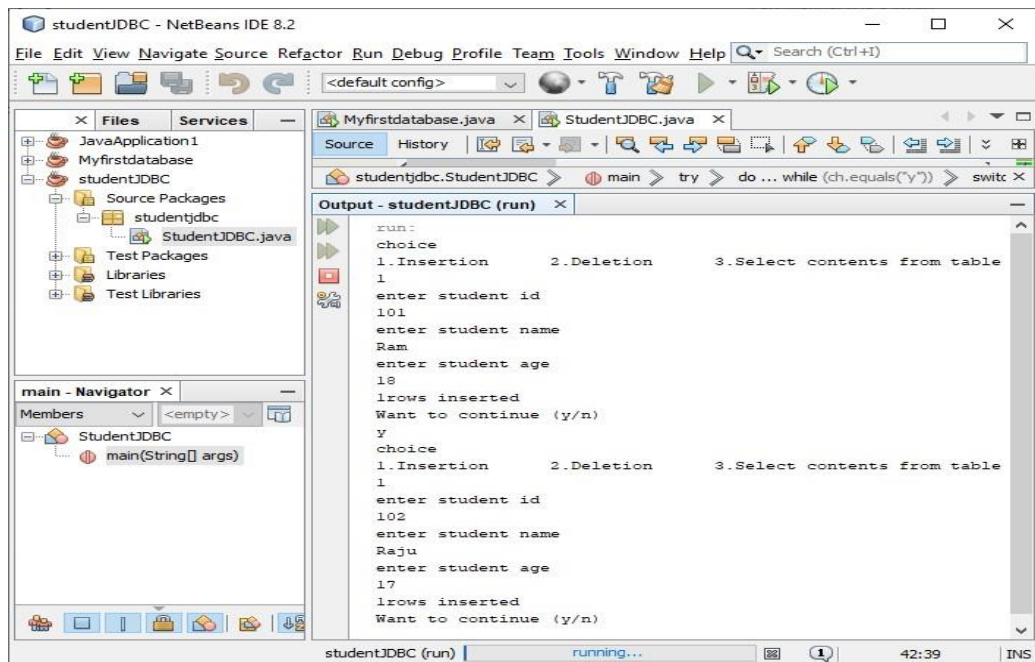
```

rows=ps.executeUpdate();
System.out.println(rows+"rows
deleted"); break;
}
case 3:
{
ResultSet rs=s.executeQuery("select * from student");
while(rs.next())
{
System.out.println("details from student
table"); System.out.println(rs.getInt(1));
System.out.println(rs.getString(2));
System.out.println(rs.getInt(3));
}
break;
}
default:
System.out.println("choose between 1 to 3");
}
System.out.println("Want to continue (y/n)");
ch=br.readLine();
}
while(ch.equals("y"));
s.close();
con.close();
}
catch(Exception e)
{
System.out.println("Exception caught"+e);
}
}
}
}

```

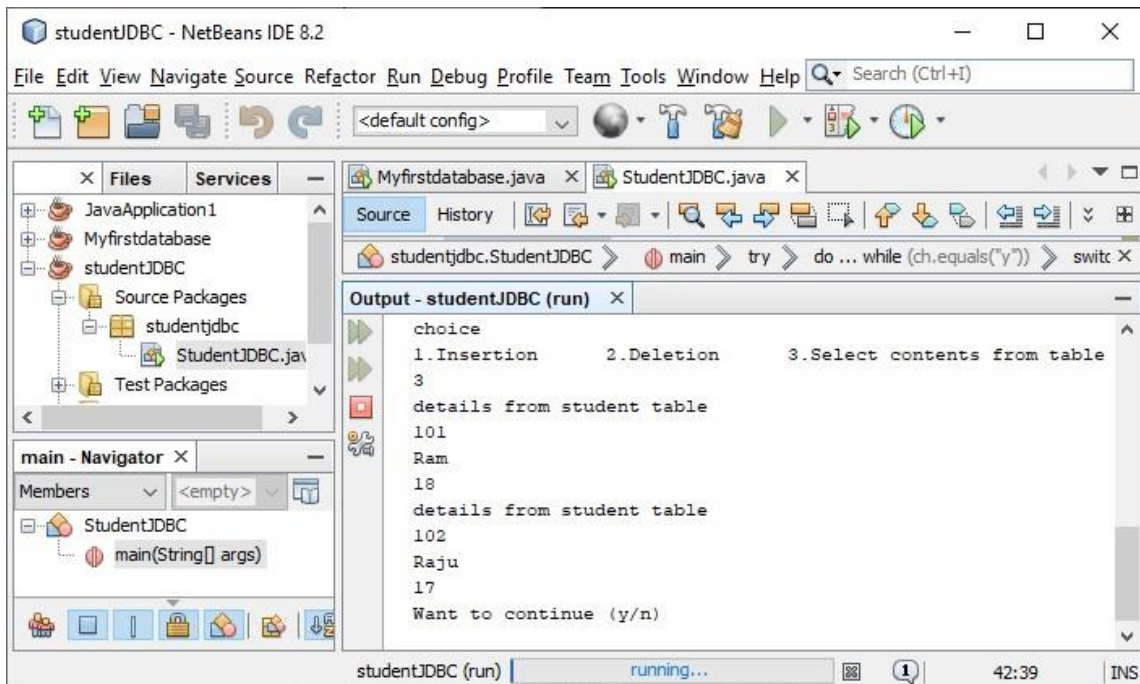
OUTPUT:

1. Inserting 2 rows :



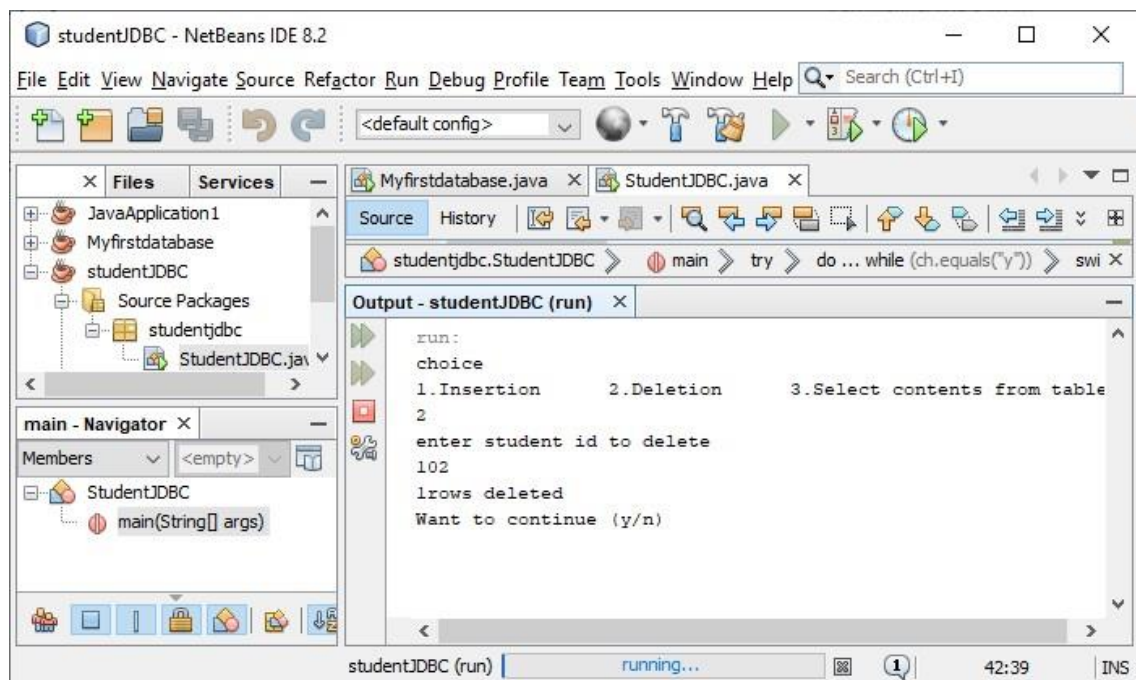
```
run:
choice
1.Insertion      2.Deletion      3.Select contents from table
1
enter student id
101
enter student name
Ram
enter student age
18
1rows inserted
Want to continue (y/n)
Y
choice
1.Insertion      2.Deletion      3.Select contents from table
1
enter student id
102
enter student name
Raju
enter student age
17
1rows inserted
Want to continue (y/n)
```

2. Displaying all rows:



```
choice
1.Insertion      2.Deletion      3.Select contents from table
3
details from student table
101
Ram
18
details from student table
102
Raju
17
Want to continue (y/n)
```

3. Deleting 1 row:



```
MySQL 8.0 Command Line Client

mysql> use bsc;
Database changed
mysql> create table student(sid int(3),sname varchar(20),age int(2));
Query OK, 0 rows affected, 2 warnings (0.98 sec)

mysql> select * from student;
Empty set (0.01 sec)

mysql> select * from student;
+-----+-----+-----+
| sid | sname | age |
+-----+-----+-----+
| 101 | Ram   | 18  |
| 102 | Raju  | 17  |
+-----+-----+-----+
2 rows in set (0.00 sec)

mysql> _
```

```
MySQL 8.0 Command Line Client

mysql> create database bsc;
Query OK, 1 row affected (0.18 sec)

mysql> use bsc;
Database changed
mysql> create table student(sid int(3),sname varchar(20),age int(2));
Query OK, 0 rows affected, 2 warnings (0.98 sec)

mysql> select * from student;
Empty set (0.01 sec)

mysql> select * from student;
+-----+-----+-----+
| sid | sname | age |
+-----+-----+-----+
| 101 | Ram   | 18  |
| 102 | Raju  | 17  |
+-----+-----+-----+
2 rows in set (0.00 sec)

mysql> select * from student;
+-----+-----+-----+
| sid | sname | age |
+-----+-----+-----+
| 101 | Ram   | 18  |
+-----+-----+-----+
1 row in set (0.00 sec)

mysql> _
```

RESULT:

Thus the program has been executed successfully and output verified.

EX.NO: 4

DATE:

DEVELOP WEB APPLICATIONS USING SERVLET

AIM:

Develop simple Login Application using Servlet.

PROCEDURE:

1. Start Netbeans IDE, Go to file menu and select new project, the new project dialog box shown as below opens. From the Categories select the option “java web” and from projects select “web Application” and click Next.
2. Type the name of the project as “LoginServlet” and select a location to save the project and click Next.
3. In the server select default server “GlassFish Server” and click “Next”. Click “Finish”.
4. Type the index.html code in the respective index .html file.
5. Right click the source packages folder from the workspace and select option “new” and select “servlet” and type the name of the servlet as “LoginServlet” click “Next”. Select the check box “Add information to deployment descriptor” and then click “Finish”. Type its code.
6. Run the index.html by right clicking the file and select run.

PROGRAM :

Index.html

```
<html>
  <head>
    <title>Login Form</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
  </head>
  <body>
    <form method="post" action="LoginServlet">
      Email ID:<input type="text" name="email"
    ><br/>
      Password:<input type="password" name="pass" /><br/>
      <input type="submit" value="login" />
    </form> </body> </html>
```

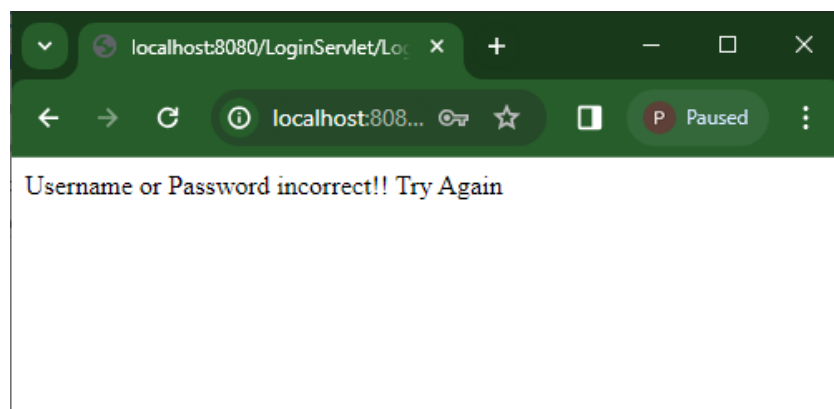
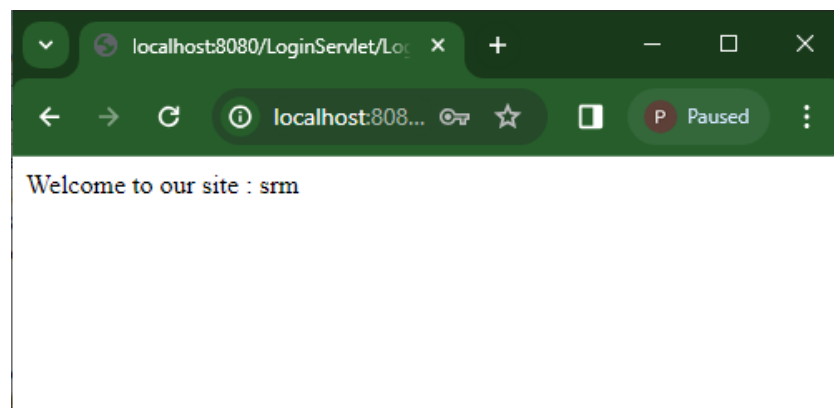
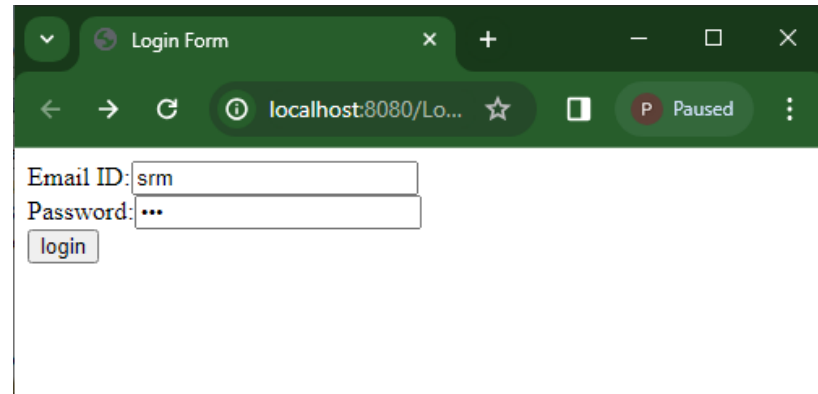
LoginServlet.java

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class LoginServlet extends HttpServlet
{
  protected void processRequest(HttpServletRequest request, HttpServletResponse response) throws
  ServletException, IOException
  {response.setContentType("text/html;charset=UTF-8");
  PrintWriter out = response.getWriter();
  String user = request.getParameter("email");
```



```
String pass = request.getParameter("pass");
if(user.equals("srm")&& pass.equals("123"))
{
out.println("Welcome to our site" + user);
}
else
{
out.println("Username or Password incorrect!! Try Again");
}}
```

OUTPUT:



RESULT:

Thus the program has been executed successfully and output verified.

EX.NO: 5

DATE:

TO CREATE USER LOGIN FORM USING SERVLET

AIM:

To create the user Login page and to Read the Form Data using Servlet

PROCEDURE:

- 1.Start Netbeans IDE, Go to file menu and select new project, the new project dialog box shown as below opens. From the Categories select the option “java web” and from projects select “web Application” and click Next.
2. Type the name of the project as “LoginServlet” and select a location to save the project and click Next.
3. In the server select default server “GlassFish Server” and click “Next”. Click “Finish”.
4. Type the index.html code in the respective index .html file.
5. Right click the source packages folder from the workspace and select option “new” and select “servlet” and type the name of the servlet as “GetForm” click “Next”. Select the check box “Add information to deployment descriptor” and then click “Finish”. Type its code.
6. Run the index.html by right clicking the file and select run.

PROGRAM:

index.html

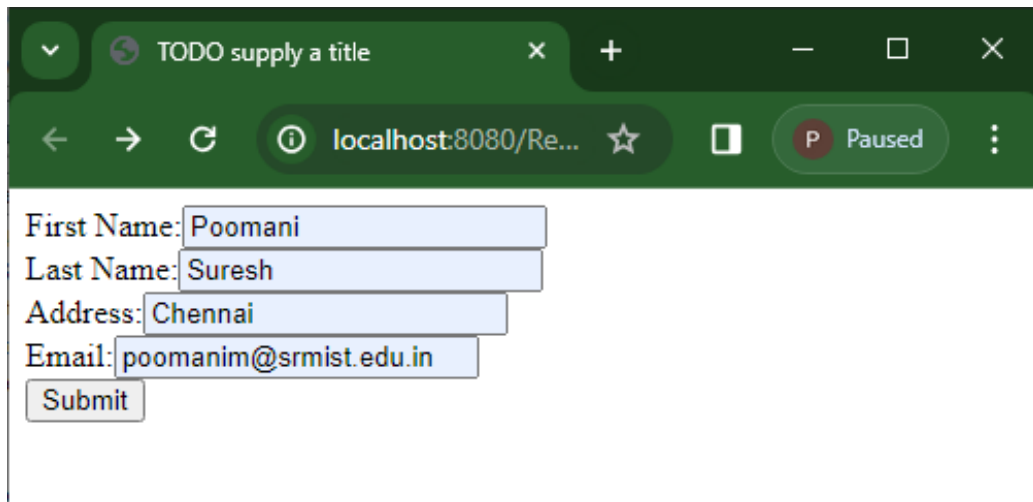
```
<html>
<head>
<title>TODO supply a title</title>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
</head>
<body>
<form action = "GetForm" method = "GET">
First Name: <input type = "text" name = "first_name"><br
/> Last Name: <input type = "text" name = "last_name"
/><br /> Address: <input type = "text" name =
"address"/><br /> Email: <input type = "text" name =
"email"/><br />
<input type = "submit" value = "Submit" />
</form>
</body>
</html>
```

GetForm.java

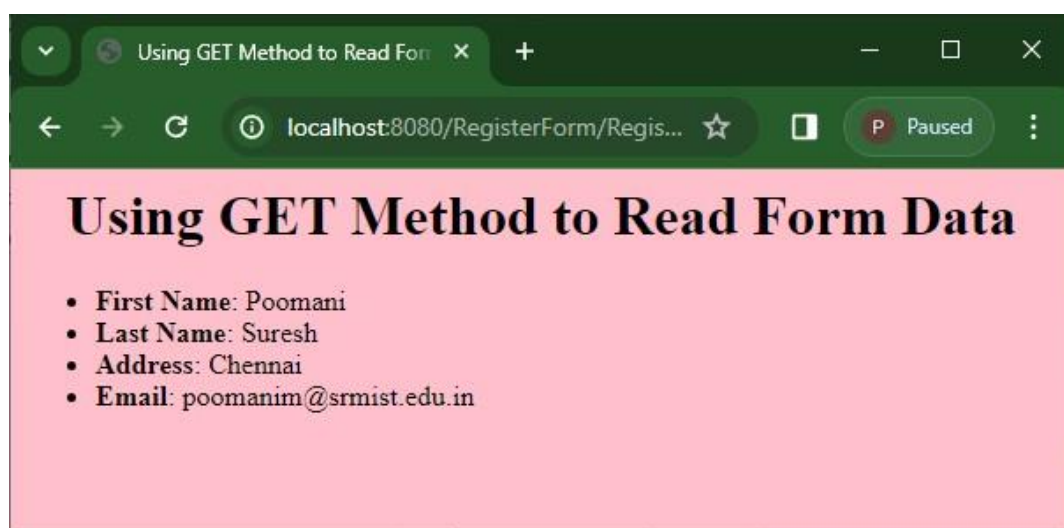
```
import java.io.*;
import javax.servlet.*; import
javax.servlet.http.*;
public class GetForm extends HttpServlet {
```

```
protected void processRequest(HttpServletRequest request, HttpServletResponse response)throws
ServletException, IOException
{
response.setContentType("text/html;charset=UTF-8");
PrintWriter out = response.getWriter();
String title = "Using GET Method to Read Form Data";
String docType = "<!doctype html public "-//w3c//dtd html 4.0 " + "transitional//en">\n";
out.println(docType + "<html>\n"
+ "<head><title>" + title + "</title></head>\n" + "<body bgcolor = \"pink\">\n"
+ "<h1 align = \"center\">" + title + "</h1>\n" + "<ul>\n" + " <li><b>First Name</b>: "
+ request.getParameter("first_name") + "\n" + "<li><b>Last Name</b>: "
+ request.getParameter("last_name") + "\n" + "<li><b>Address</b>: "
+ request.getParameter("address") + "\n" + "<li><b>Email</b>: "
+ request.getParameter("email") + "\n" + "</ul>\n" + "</body>" + "</html>");
}}
```

OUTPUT:



First Name: Poomani
Last Name: Suresh
Address: Chennai
Email: poomanim@srmist.edu.in
Submit



RESULT:

Thus the program has been executed successfully and output verified.

EX.NO: 6

DATE:

SESSION MANAGEMENT USING SERVLET

AIM:

To perform session management using servlet.

PROCEDURE:

1. Start Netbeans IDE, Go to file menu and select new project, the new project dialog box shown as below opens. From the Categories select the option “java web” and from projects select “web Application” and click Next.
2. Type the name of the project as “Sess” and select a location to save the project and click Next.
3. In the server select default server “GlassFish Server” and click “Next”. Click “Finish”.
4. Type the index.html code in the respective index .html file.
5. Right click the source packages folder from the workspace and select option “new” and select “servlet” and type the name of the servlet as “FirstServlet” click “Next”. Select the check box “Add information to deployment descriptor” and then click “Finish”. Type its code.
6. Repeat step 5 and create another servlet with the name “SecondServlet” and type its code.
7. Run the index.html by right clicking the file and select run.

PROGRAM:

FirstServlet.java

```
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;
public class FirstServlet extends
HttpServlet {
protected void processRequest(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException
{
response.setContentType("text/html");
PrintWriter out = response.getWriter();
String
n=request.getParameter("userName");
out.print("Welcome "+n);
out.print("<a
href='SecondServlet?uname="+n+"'>visit</a>");
out.close();
}
protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException,IOException
{
processRequest(request, response);
```

```
}
```

Second Servlet.java

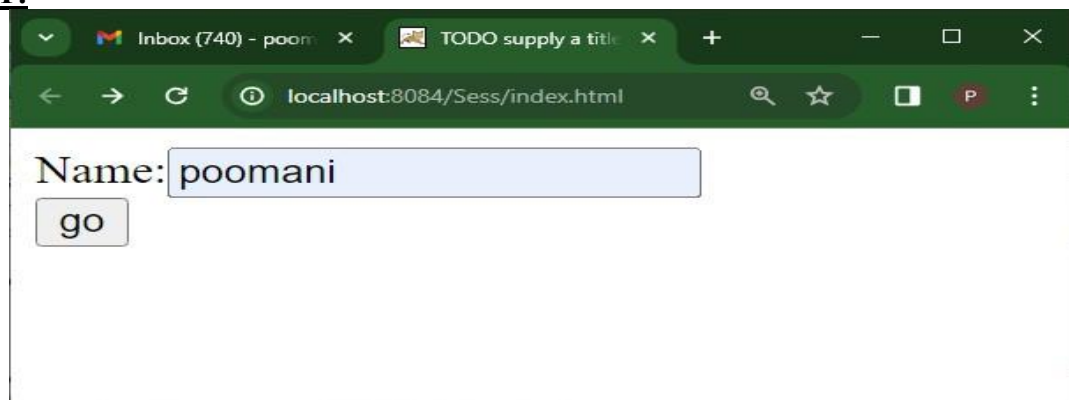
```
import
java.io.IOException;
import
java.io.PrintWriter;
import
javax.servlet.ServletException;
import
javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
public class SecondServlet extends HttpServlet{
protected void processRequest(HttpServletRequest request, HttpServletResponse response)throws
ServletException, IOException {

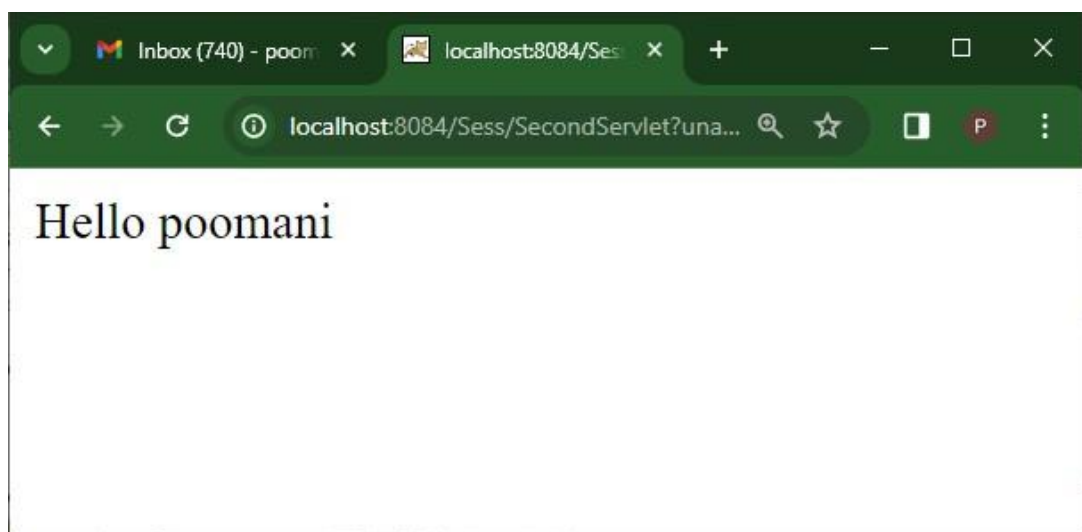
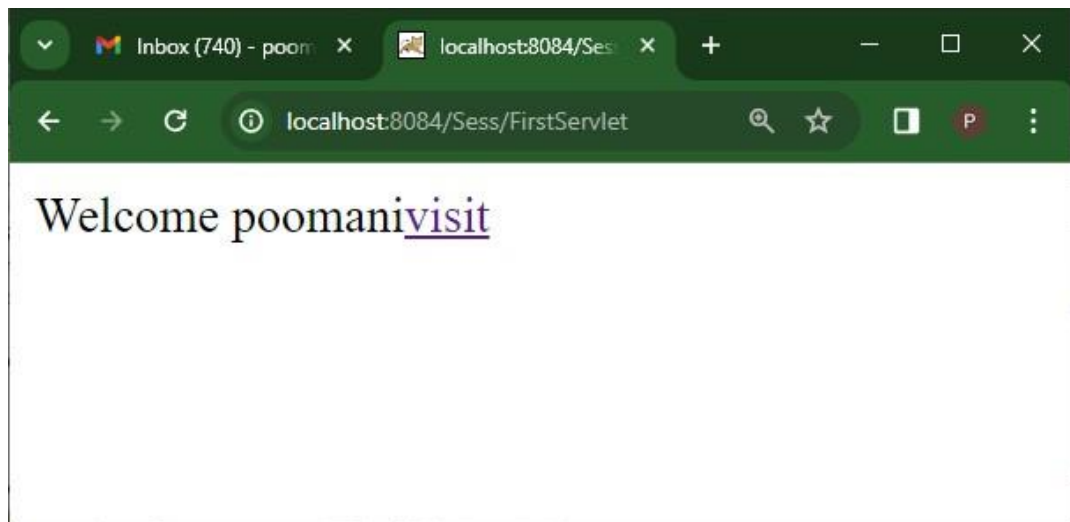
response.setContentType("text/html");
PrintWriter out = response.getWriter();
String n=request.getParameter("uname");
out.print("Hello "+n);
out.close();
}
protected void doPost (HttpServletRequest request, HttpServletResponse response)throws
ServletException, IOException {
processRequest(request, response);
}}
```

Index.html

```
<html>
<body>
<form action="FirstServlet" method="post">
Name:<input type="text"
name="userName"/><br/>
<input type="submit" value="go"/>
</form>
</body>
</html>
```

OUTPUT:





RESULT:

Thus the program has been executed successfully and output verified.

EX.NO: 7

DATE:

WEB APPLICATION USING JSP

AIM :

To develop a web application to select the author of a text book using JSP.

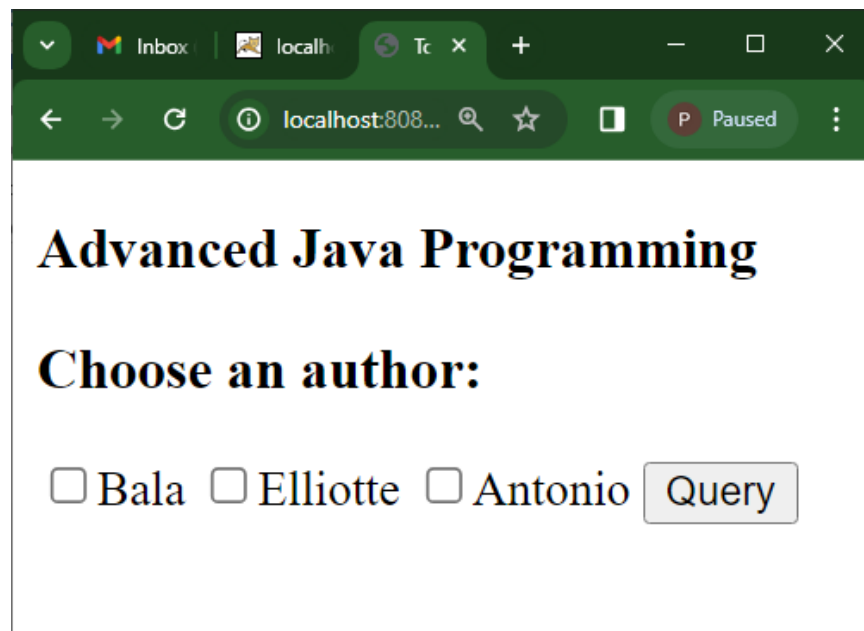
PROCEDURE:

1. Start Netbeans IDE, Go to file menu and select new project, the new project dialog box shown as below opens. From the Categories select the option “java web” and from projects select “web Application” and click Next.
2. Type the name of the project as “FirstJSP” and select a location to save the project and click Next.
3. In the server select default server “GlassFish Server” and click “Next”. Click “Finish”.
4. Right click the Web Pages folder from the workspace and select option “new” and select “JSP” and type the name of the file as “First” and then click “Finish”. Type its code.
5. Run the first.jsp by right clicking the file and select run.

PROGRAM:

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>To Design the Web application using JSP </title>
</head>
<body>
<h3>Advanced Java Programming</h3>
<h3>Choose an author:</h3>
<form method="get">
<input type="checkbox" name="author" value="Balaguruswamy">Bala
<input type="checkbox" name="author" value="Elliotte Rusty Harold">Elliotte
<input type="checkbox" name="author" value="Antonio Goncalves">Antonio
<input type="submit" value="Query">
</form>
<% String[] authors =
request.getParameterValues("author"); if (authors!= null)
{ %>
<h3>You have selected author(s):</h3>
<ul>
<% for (int i = 0; i < authors.length; ++i) { %>
<li><%= authors[i] %></li>
<% } %>
</ul>
<a href="<%= request.getRequestURI() %>">BACK</a>
<% } %>
</body>
</html>
```

OUTPUT:

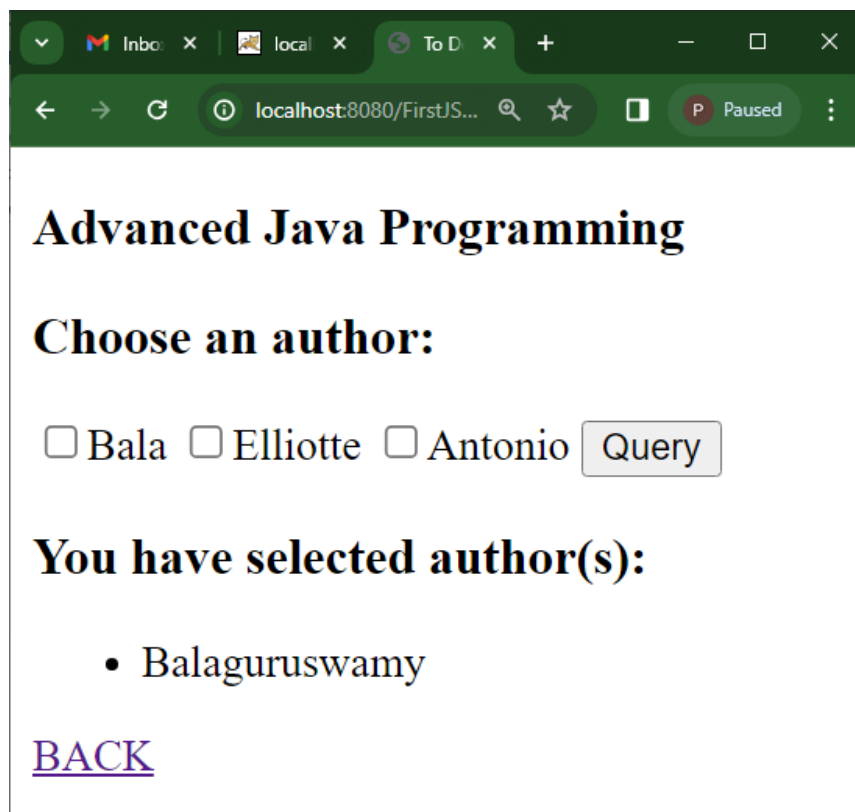


A screenshot of a web browser window. The address bar shows 'localhost:8080...'. The page content includes the title 'Advanced Java Programming' in a large, bold, black serif font. Below it is the text 'Choose an author:' in a bold, black serif font. Underneath are three radio buttons followed by the names 'Bala', 'Elliott', and 'Antonio'. To the right of these is a button labeled 'Query'.

Advanced Java Programming

Choose an author:

☐ Bala ☐ Elliott ☐ Antonio



A screenshot of a web browser window showing the result of a query. The address bar shows 'localhost:8080/FirstJS...'. The page content includes the title 'Advanced Java Programming' in a large, bold, black serif font. Below it is the text 'Choose an author:' in a bold, black serif font. Underneath are three radio buttons followed by the names 'Bala', 'Elliott', and 'Antonio'. To the right of these is a button labeled 'Query'. Below this is the text 'You have selected author(s):' in a bold, black serif font. Underneath is a bulleted list with one item: 'Balaguruswamy'. At the bottom is a link labeled 'BACK' in a purple, underlined serif font.

Advanced Java Programming

Choose an author:

☐ Bala ☐ Elliott ☐ Antonio

You have selected author(s):

- Balaguruswamy

[BACK](#)

RESULT:

Thus the program has been executed successfully and output verified.

EX.NO: 8

DATE:

CREATE APPLICATIONS USING INCLUDE DIRECTIVE

AIM:

To create web applications using include Directive JSP Include Action

PROCEDURE :

1. Start Netbeans IDE, Go to file menu and select new project, the new project dialog box shown as below opens. From the Categories select the option “java web” and from projects select “web Application” and click Next.
2. Type the name of the project as “FirstJSP” and select a location to save the project and click Next.
3. In the server select default server “GlassFish Server” and click “Next”. Click “Finish”.
4. Right click the Web Pages folder from the workspace and select option “new” and select “JSP” and type the name of the file as “Date” and then click “Finish”. Type its code.
5. Repeat the step 4 and name the file as “Order.jsp” and type its code.
6. Repeat the step 4 and name the file as “includedir.jsp” and type its code.
7. Run the includedir.jsp by right clicking the file and select run.

PROGRAM:

Date.jsp

```
<html>
<head>
<title>Date Display</title>
</head>
<body>
<%-- JSP comments --%>
<%@page import="java.util.Date"%>
<%! Date date; %>
<% date = new Date(); %>
<b> System date and time: </b> <%= date %>
</body>
</html>
```

Order.jsp

```
<HTML>
<HEAD>
<TITLE>A Catalog Order Form</TITLE>
</HEAD>
<BODY>
<H1> Invoice Form</H1>
<%! String item[] = {"Pen", "Sketch", "Marker"};
double price[] = {20.50, 10.50, 15.50};
int quantity[] = {10, 15, 25}; %>
<TABLE BORDER="1" >
<TR><TD>Item</TD>
<TD>Price</TD>
<TD>Quantity</TD>
<TD>Total Price</TD>
</TR>
```

```

<% for (int i=0; i<3; i++) { %>
<TR><TD><%= item[i] %></TD>
<TD><%= price[i] %></TD>
<TD><%= quantity[i] %></TD>
<TD><%= price[i] * quantity[i] %></TD>
</TR>
<% } %>
</TABLE>
</BODY>
</HTML>

```

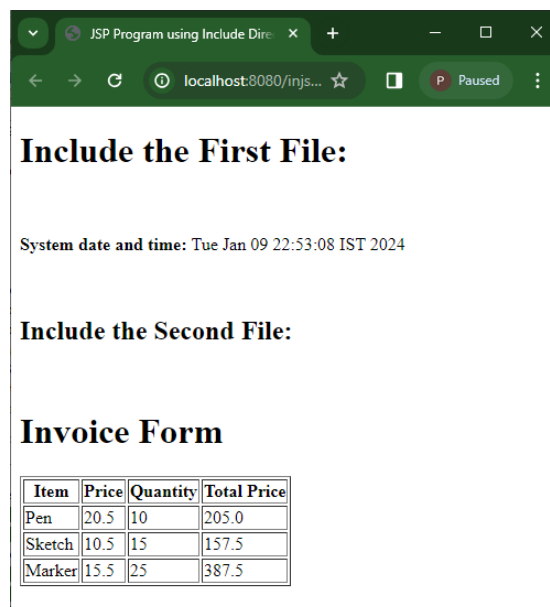
Includedir.jsp

```

<html>
<head>
<title>JSP Program using Include Directive</title>
</head>
<body>
<h1>Include the First File:</h1>
<br><br>
<%@ include file="Date.jsp"%>
<br><br><br>
<h2>Include the Second File:</h2>
<br>
<%@ include file="Order.jsp"%>
</body>
</html>

```

OUTPUT:



RESULT:

Thus the program has been executed successfully and output verified.

EX.NO: 9

DATE:

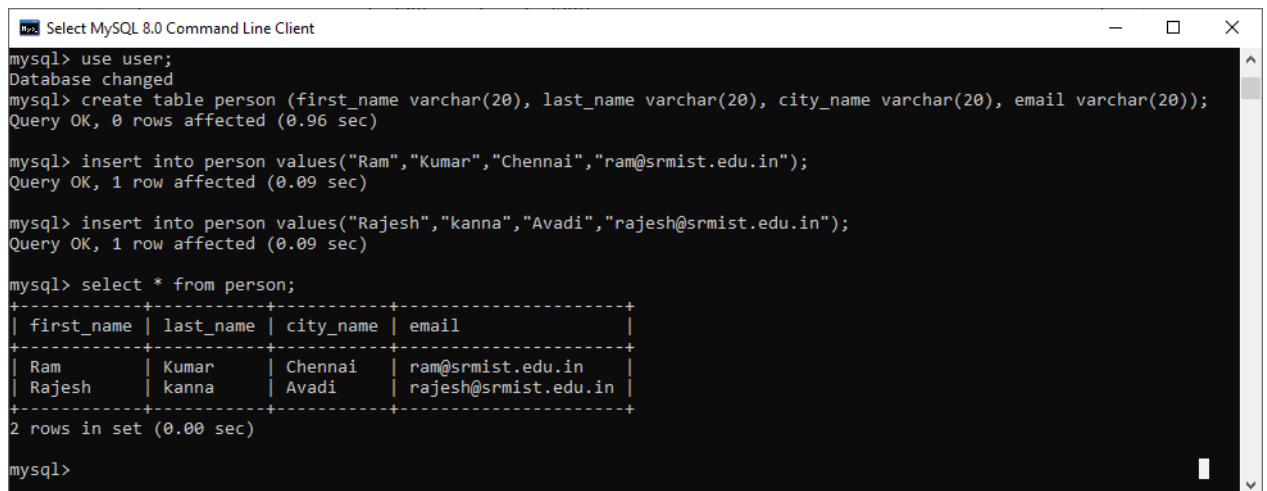
CREATE A JSP BASED WEB APPLICATION USING DATABASE

AIM:

To write a JSP Program for retrieving the Employee Details through JDBC Connectivity.

PROCEDURE:

1. Open mysql command line and create a database and table as given below:
 - a. `mysql> create database user;`
 - b. `mysql> use user;`
 - c. `mysql> create table person(first_name varchar(20), last_name varchar(20), city_name varchar(20), email varchar(20));`
 - d. `mysql> insert into person values("Ram","Kumar","Chennai","ram@srmist.edu.in");`
 - e. `mysql> insert into person values("Rajesh","kanna","Avadi","rajesh@srmist.edu.in");`
 - f. `mysql> select * from person;`



```
Select MySQL 8.0 Command Line Client
mysql> use user;
Database changed
mysql> create table person (first_name varchar(20), last_name varchar(20), city_name varchar(20), email varchar(20));
Query OK, 0 rows affected (0.96 sec)

mysql> insert into person values("Ram","Kumar","Chennai","ram@srmist.edu.in");
Query OK, 1 row affected (0.09 sec)

mysql> insert into person values("Rajesh","kanna","Avadi","rajesh@srmist.edu.in");
Query OK, 1 row affected (0.09 sec)

mysql> select * from person;
+-----+-----+-----+-----+
| first_name | last_name | city_name | email |
+-----+-----+-----+-----+
| Ram        | Kumar    | Chennai  | ram@srmist.edu.in |
| Rajesh     | kanna    | Avadi    | rajesh@srmist.edu.in |
+-----+-----+-----+-----+
2 rows in set (0.00 sec)

mysql>
```

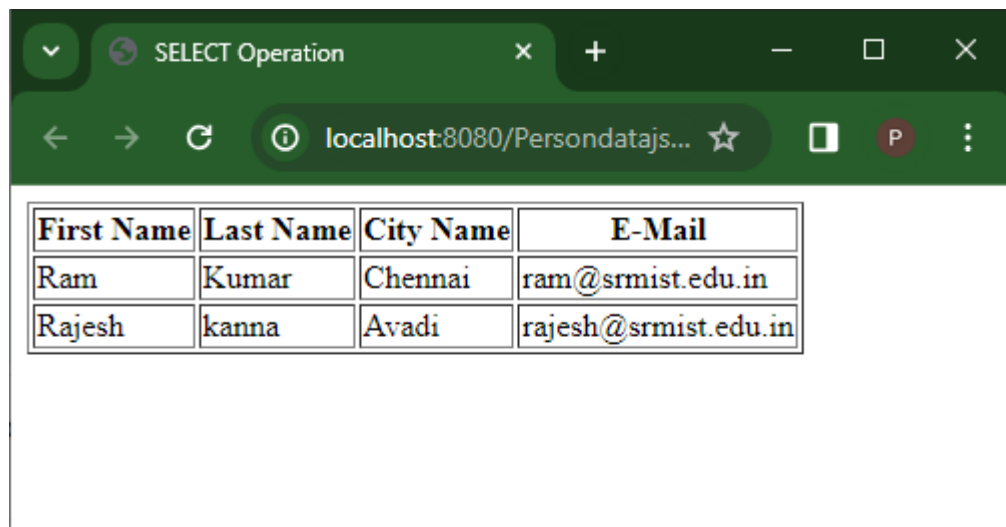
2. Start Netbeans IDE, Go to file menu and select new project, the new project dialog box opens. From the Categories select the option Java Web and from projects select Java Web Application and click Next.
3. Type the name of the project as “persondatajsp” and select a location to save the project and click Finish.
4. In the project tab right click the library folder of your application and select the option “Add JAR/Folder” and go to the location where we have saved our mysql connector JAR file. Select the JAR file and it will be added to your project.
5. Right Click the project name and select the option new and JSP and give name for jsp file and click finish.
6. Type the following code and run the code by clicking the run tool or by selecting run option from run menu.

PROGRAM:

Persondata.jsp

```
<%@ page import = "java.io.*,java.util.*,java.sql.*"%>
<%@ page import = "javax.servlet.http.*,javax.servlet.*" %>
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>SELECT Operation</title>
    </head>
    <body>
        <table border="1">
            <th>First Name</th><th>Last Name</th><th>City Name</th><th>E-Mail</th>
        <%
            Connection
            con; try
            {
                Class.forName("com.mysql.cj.jdbc.Driver")
                ; con =
                DriverManager.getConnection("jdbc:mysql://localhost:3306/user?useSSL=false&allowPublicKey
                Retrieval=true", "root" ,"mysql");
                if(con!=null) {
                    out.println("Successfully connected to " + "MySQL server using TCP/IP..." + "<br>");
                    Statement stmt = con.createStatement();
                    ResultSet rs = stmt.executeQuery("select * from
                    person"); while (rs.next())
                    {%>
                        <tr>
                            <td><%=rs.getString(1)%></td>
                            <td><%=rs.getString(2)%></td>
                            <td><%=rs.getString(3)%></td>
                            <td><%=rs.getString(4)%></td>
                        </tr>
                    <%}
                }
            con.close();
        }
        catch (Exception e) {
            out.println("Exception: " + e.getMessage());
        }
    %>
        </table>
    </body>
</html>
```

OUTPUT



A screenshot of a web browser window. The title bar shows 'SELECT Operation'. The address bar shows 'localhost:8080/Persondatajs...'. The main content area displays a table with four columns: 'First Name', 'Last Name', 'City Name', and 'E-Mail'. The table contains two rows of data.

First Name	Last Name	City Name	E-Mail
Ram	Kumar	Chennai	ram@srmist.edu.in
Rajesh	kanna	Avadi	rajesh@srmist.edu.in

RESULT:

Thus the program has been executed successfully and output verified.

EX.NO: 10

DATE:

AN EJB APPLICATION THAT DEMONSTRATES SESSION BEAN -STATELESS BEAN

AIM:

To develop an EJB application that demonstrates Stateless Session

Bean. **PROCEDURE:**

1. Start Netbeans IDE, Go to file menu and select new project, the new project dialog box opens. From the Categories select the option JavaEE and from projects select Enterprise Application and click Next, type the name as “Myfirst”, select location click next and finish.
2. Right click the source packages in “Myfirst-ejb” select new and select “session bean” and give a name for it as “calcbean” and package name as “SessionBean” select session as “stateless” and select checkbox “local” and click Finish.
3. Inside the code of the class rightclick and select insertcode and select Add Business Method. A dialog box opens give the name of the method as “addition” and for return type click browse and select integer (java.lang) and click ok. Click the add button and enter the parameter name as “a” and type as int and click add button and enter parameter name as “b” and type as int and click ok. Now change the return value in the code.
4. Right Click the “Myfirst-war” file and select the option new and select the option servlet and give name “calcservlet” and package as “SessionBean” click Next. Select the Check box “Add information to deployment descriptor (web.xml) and click Finish.
5. Right click in the next line of the class of the servlet code and select the option insert code and then select the option “call Enterprise Bean” and select “calcbean” from “myfirst-ejb” file.
6. Type the code which are given in bolded case.
7. Right Click the “myfirst-war” file and select the option new and select the option jsp and give name “calcjsp” click Finish. Type the code of the jsp file.
8. Right click the Application from workspace and select the option deploy. Right click the application and select the option Run.

PROGRAM:

calcbeanLocal.java

```
package  
SessionBean;  
import  
javax.ejb.Local;  
@Local  
public interface calcbeanLocal {  
    Integer addition(int a, int b);  
}
```

calcbean.java

```

package SessionBean;
import
javax.ejb.Stateless;
@Stateless
public class calcbean implements calcbeanLocal
{
    @Override
    public Integer addition(int a, int
        b) { return (a+b);
    }
}

```

clacservlet.java

```

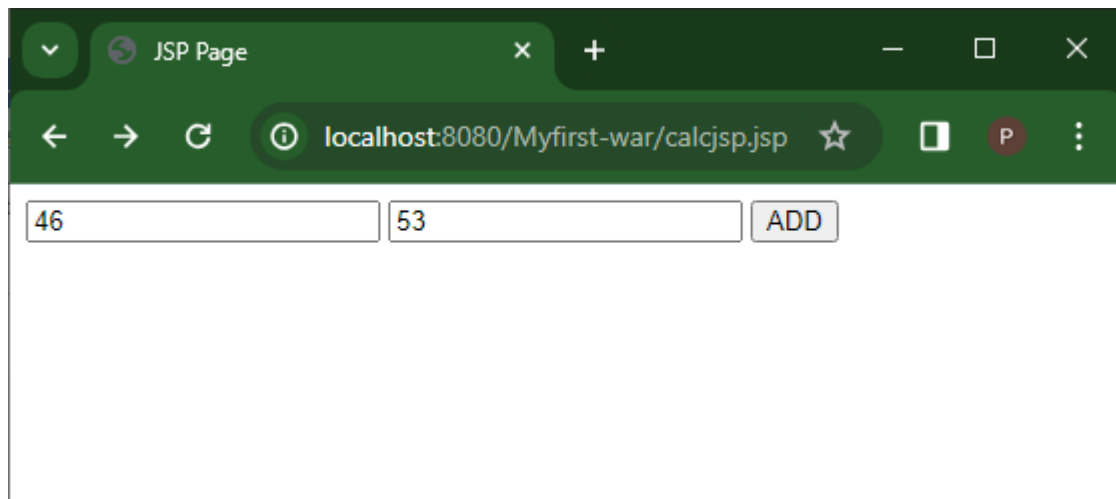
package SessionBean;
import
java.io.IOException;
import
java.io.PrintWriter;
import javax.ejb.EJB;
import
javax.servlet.ServletException;
import
javax.servlet.http.HttpServlet;
import
javax.servlet.http.HttpServletRequest;
import
javax.servlet.http.HttpServletResponse;
public class clacservlet extends HttpServlet
{
    @EJB
    private calcbeanLocal calcbean;
    protected void processRequest(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-
            8"); try (PrintWriter out = response.getWriter()) {
            out.println("<!DOCTYPE
                html>"); out.println("<html>");
            out.println("<head>");
            out.println("<title>Servlet clacservlet</title>");
            out.println("</head>");
            out.println("<body>");
            int
            a=Integer.parseInt(request.getParameter("t1"));
            int
            b=Integer.parseInt(request.getParameter("t2"));
            out.println("<h1>Sum = " + calcbean.addition(a, b) + "</h1>");
            out.println("</body>");
            out.println("</html>");
        }
    }
}

```

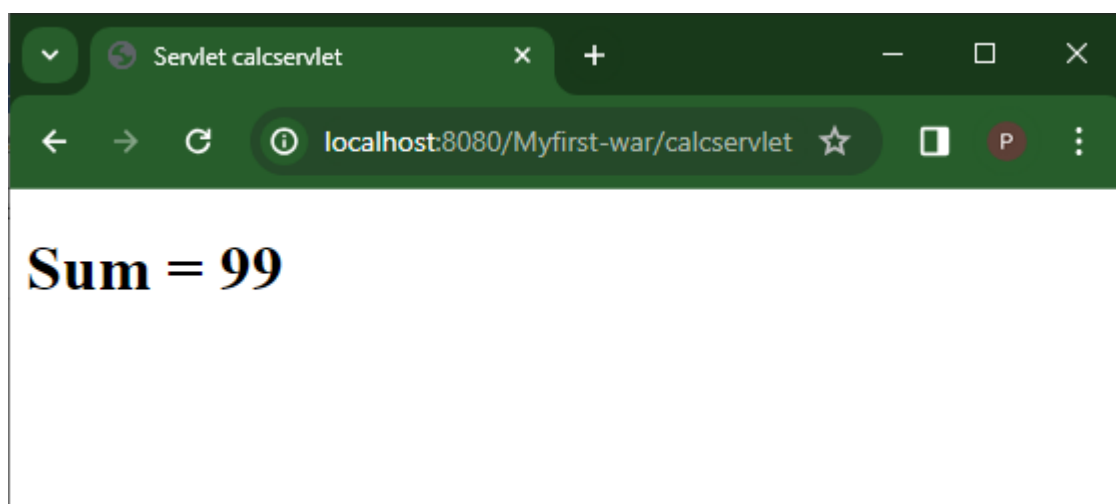
calcisp.java

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>JSP Page</title>
  </head>
  <body>
    <form action="calcservlet" method="POST">
      <input type="text" name="t1">
      <input type="text" name="t2">
      <input type="Submit" value="ADD">
    </form>
  </body>
</html>
```

OUTPUT:



A screenshot of a web browser window. The title bar shows 'JSP Page'. The address bar shows 'localhost:8080/Myfirst-war/calcjsp.jsp'. The page content consists of two text input fields, the first containing '46' and the second containing '53', followed by an 'ADD' button.



A screenshot of a web browser window. The title bar shows 'Servlet calcservlet'. The address bar shows 'localhost:8080/Myfirst-war/calcservlet'. The page content displays 'Sum = 99' in a large, bold, black serif font.

RESULT:

Thus the program has been executed successfully and output verified.

EX.NO: 11

DATE:

AN EJB APPLICATION THAT DEMONSTRATES SESSION BEAN-STATEFULL BEAN

AIM:

To develop an EJB application that demonstrates Stateful Session Bean.

PROCEDURE:

1. Start Netbeans IDE, Go to file menu and select new project, the new project dialog box opens. From the Categories select the option JavaEE and from projects select Enterprise Application and click Next, type the name as "MySecond", select location click next and finish.
2. Right click the source packages in "MySecond-ejb" select new and select "session bean" and give a name for it as "hellobean" and package name as "Hello" select session as "stateful" and select checkbox "local" and click Finish.
3. Inside the code of the class rightclick and select insertcode and select Add Business Method. A dialog box opens give the name of the method as "Helloname" and for return type click browse and select String (java.lang) and click ok. Click the add button and enter the parameter name as "name" and type as "String" and click ok. Now type the return value in the code.
4. Right Click the "MySecond-war" file and select the option new and select the option servlet and give name "Helloservlet" and package as "Hello" click Next. Select the Check box "Add information to deployment descriptor (web.xml) and click Finish.
5. Right click in the next line of the class of the servlet code and select the option insert code and then select the option "call Enterprise Bean" and select "hellobean" from "MySecond- ejb" file.
6. Type the code which are given in bolded case.
7. Type the code of the "index.html" file.
8. Right click the Application from workspace and select the option deploy. Right click the application and select the option Run.

PROGRAM:

hellobean.java

```
package Hello;
import
javax.ejb.Stateful;
@Stateful
public class hellobean implements
    hellobeanLocal { @Override
    public String Helloname(String name) {
        return "name";
    }
}
```

hellobeanLocal.java

```
package Hello;
```

```
import
javax.ejb.Local;
@Local
public interface hellobeanLocal {
    String Helloname(String name);
}
```

Helloservlet.java

```
package Hello;
import java.io.IOException;
import java.io.PrintWriter;
import
java.util.logging.Level;
import
java.util.logging.Logger;
import
javax.naming.Context;
import javax.naming.InitialContext;
import
javax.naming.NamingException;
import
javax.servlet.ServletException;
import
javax.servlet.http.HttpServlet;
import
javax.servlet.http.HttpServletRequest;
import
javax.servlet.http.HttpServletResponse;
public class Helloservlet extends
HttpServlet {
    hellobeanLocal hellobean = lookuphellobeanLocal();
    protected void processRequest(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        String n=request.getParameter("name");
        try (PrintWriter out =
            response.getWriter()) {
            out.println("<!DOCTYPE html>");
            out.println("<html>");
            out.println("<head>");
            out.println("<title>Servlet Helloservlet</title>");
            out.println("</head>");
            out.println("<body>");
            out.println("<h1>Hello " + n +
"</h1>"); out.println("</body>");
            out.println("</html>");
        }
    }
}
```

```

private hellobeanLocal
lookuphellobeanLocal() { try {
    Context c = new InitialContext();
    return (hellobeanLocal) c.lookup("java:global/MySecond/MySecond-
ejb/hellobean!Hello.hellobeanLocal");
} catch (NamingException ne) {
    Logger.getLogger(getClass().getName()).log(Level.SEVERE, "exception caught", ne);
    throw new RuntimeException(ne);
}
}
}
}

```

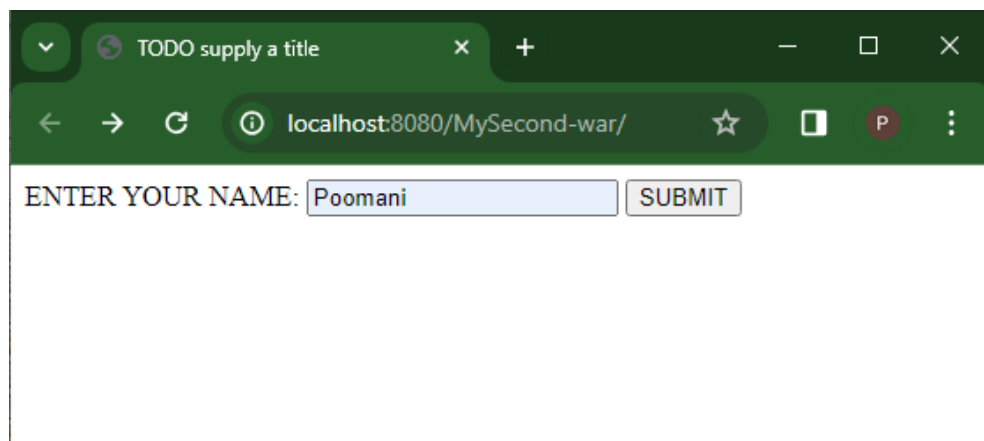
Index.html

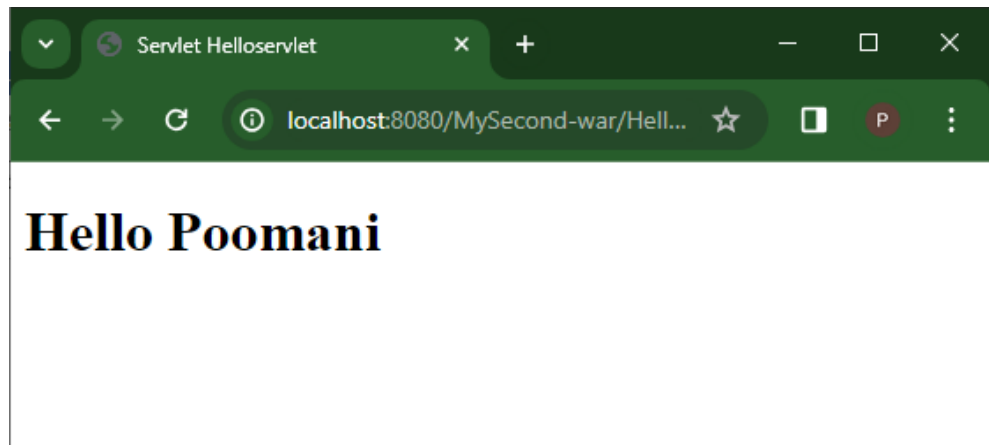
```

<html>
<head>
<title>TODO supply a title</title>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
</head>
<body>
<form action="HelloServlet" method="POST">
    ENTER YOUR NAME: <input type="text" name="name">
    <input type="submit" value="SUBMIT">
</form>
</body>
</html>

```

OUTPUT:





RESULT:

Thus the program has been executed successfully and output verified.

EX.NO: 12

DATE:

AN EJB APPLICATION THAT DEMONSTRATES ENTITY BEAN

AIM:

To create an EJB application that demonstrates Entity Bean.

PROCEDURE :

Create datasource in netbeans:

1. Open netbeans IDE and go to services tab in the workspace. Right Click the javaDB inside the databases and start the server. Right click again and select option create database and give a name for the database as "mine" and enter username and password and confirm password and click OK.
2. The database will be created under the driver url. Right click and create new table in that new database. Created two fields one is regno with varchar datatype of size 6 with primary key checked and other field name with varchar datatype of size 20.

Create Entity Bean Application

3. Start Netbeans IDE, Go to file menu and select new project, the new project dialog box shown as below opens. From the Categories select the option "java web" and from projects select "web Application" and click Next.
4. Enter the name of the Application as "Entitystudent" and click next and click finish
5. Right click the application select the option other and then select the option "persistence" and then select the option " Entity classes from database" and Give new name for datasource and select the driver we created, it automatically displays the tables available in that database. Select the table and click the add button. Enter the package name "stud" and click next and finish.
6. Right click the project select the option "session bean for entity class" and select the entity class and select the button add and click next now select the option local and click finish.
7. Add a servlet Page by right clicking the Web Pages and selecting New → servlet. Enter name Studserv and select package stud and click next select Add deployment descriptor and click finish.
8. Right click next to the class line in the servlet file and select insert code option and select the option "call enterprise bean" and select our entity bean.
9. Type the code given in bold case.
10. Right click the project and deploy and then right click and run the application.

PROGRAM:

StudentFacade.java

```
package stud;
import javax.ejb.Stateless;
import javax.persistence.EntityManager;
import
javax.persistence.PersistenceContext;
@Stateless
public class StudentFacade extends AbstractFacade<Student> implements StudentFacadeLocal
{
    @PersistenceContext(unitName = "EntityStudentPU")
    private EntityManager em;
    @Override
    protected EntityManager
        getEntityManager() { return em; }
```

```

public StudentFacade() {
    super(Student.class);
}
@Override
public void add(String regno, String name)
{
    Student s1=new
    Student();
    s1.setRegno(regno);
    s1.setName(name);
    persist(s1);
}
@Override
public void persist(Object obj)
{
    em.persist(obj);
}
}

```

Student.java

```

package stud;

import java.io.Serializable;
import
javax.persistence.Basic;
import
javax.persistence.Column;
import
javax.persistence.Entity;
import javax.persistence.Id;
import
javax.persistence.NamedQueries;
import
javax.persistence.NamedQuery;
import javax.persistence.Table;
import javax.validation.constraints.NotNull;
import javax.validation.constraints.Size;
import
javax.xml.bind.annotation.XmlRootElement;
@Entity
@Table(name =
"STUDENT")
@XmlRootElement
@NamedQueries({
    @NamedQuery(name = "Student.findAll", query = "SELECT s FROM Student s")
    , @NamedQuery(name = "Student.findByRegno", query = "SELECT s FROM Student s
WHERE s.regno = :regno")
    , @NamedQuery(name = "Student.findByName", query = "SELECT s FROM Student s
WHERE s.name = :name")})

```

```

public class Student implements Serializable {
    private static final long serialVersionUID =
        1L; @Id
    @Basic(optional =
        false) @NotNull
    @Size(min = 1, max = 6)
    @Column(name =
        "REGNO") private String
        regno;
    @Size(max = 20)
    @Column(name =
        "NAME") private String
        name;
    public Student() {
    }
    public Student(String regno) {
        this.regno = regno;
    }
    public String getRegno() {
        return regno;
    }
    public void setRegno(String regno) {
        this.regno = regno;
    }
    public String getName() {
        return name;
    }
    public void setName(String
        name) { this.name = name;
    }
    @Override
    public int
        hashCode() { int
            hash = 0;
            hash += (regno != null ? regno.hashCode() :
                0); return hash;
        }
    @Override
    public boolean equals(Object object) {
        if (!(object instanceof Student)) {
            return false;
        }
        Student other = (Student) object;
        if ((this.regno == null && other.regno != null) || (this.regno != null &&
            !this.regno.equals(other.regno))) {
            return false;
        }
    }
}

```

```

        return true;
    }
    @Override
    public String toString() {
        return "stud.Student[ regno=" + regno + " ]";
    }
}

```

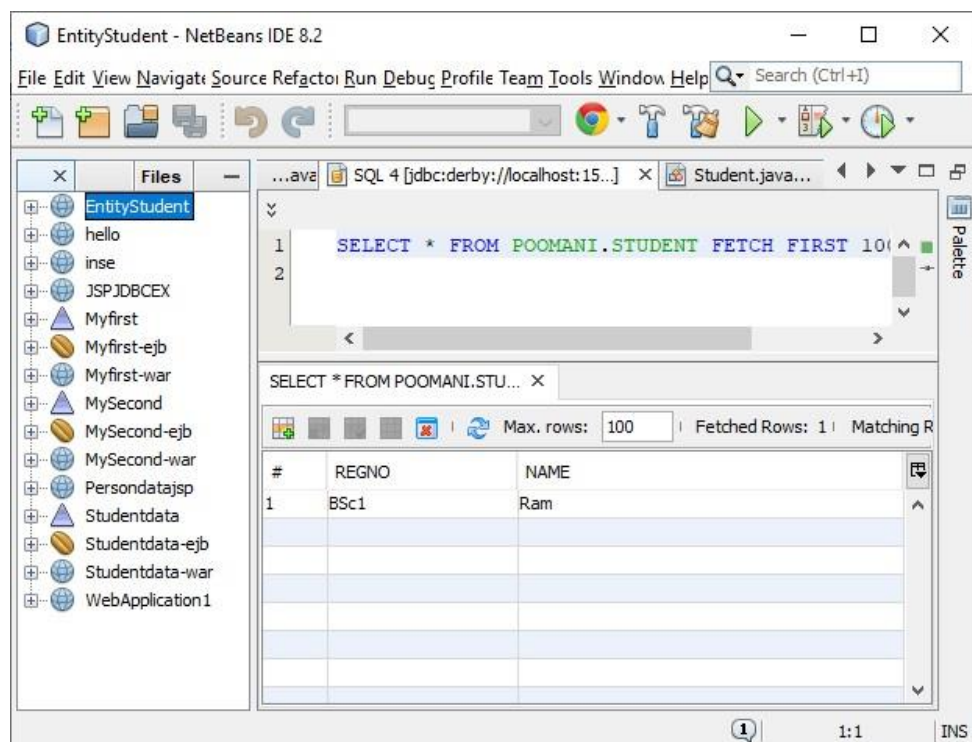
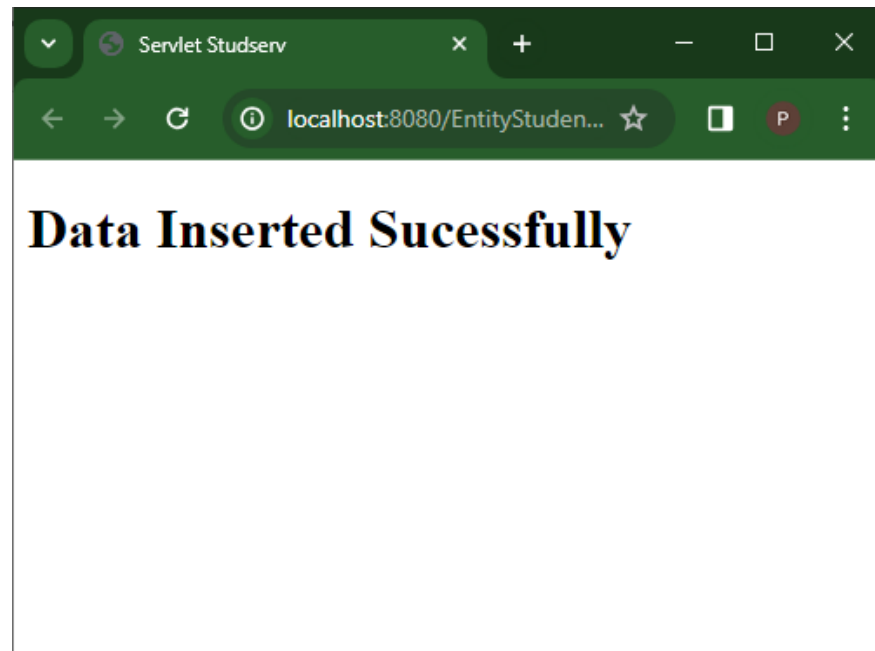
Studserv.java

```

package stud;
import
java.io.IOException;
import
java.io.PrintWriter;
import javax.ejb.EJB;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import
javax.servlet.http.HttpServletRequest;
import
javax.servlet.http.HttpServletResponse;
public class Studserv extends HttpServlet {
    @EJB
    private StudentFacadeLocal studentFacade;
    protected void processRequest(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-
        8"); try (PrintWriter out = response.getWriter()) {
            out.println("<!DOCTYPE
            html>"); out.println("<html>");
            out.println("<head>");
            out.println("<title>Servlet
            Studserv</title>"); out.println("</head>");
            out.println("<body>");
            studentFacade.add("BSc1", "Ram");
            out.println("<h1>Data Inserted Sucessfully </h1>");
            out.println("</body>");
            out.println("</html>");
        }
    }
}
}

```


OUTPUT:



RESULT:

Thus the program has been executed successfully and output verified.

EX.NO: 13

DATE:

IMPLEMENTING MVC WITH REQUEST DISPATCHER

AIM:

Develop Login application to implement MVC with Request Dispatcher.

PROCEDURE:

1. Start Netbeans IDE, Go to file menu and select new project, the new project dialog box shown as below opens. From the Categories select the option “java web” and from projects select “web Application” and click Next.
2. Type the name of the project as “Login” and select a location to save the project and click Next.
3. In the server select default server “GlassFish Server” and click “Next”. Click “Finish”.
4. Type the index.html code in the respective index.html file.
5. Right click the source packages folder from the workspace and select option “new” and select “Servlet” and type the name of the servlet as “ControllerServlet” and in package type “com.mine” click “Next”. Select the check box “Add information to deployment descriptor” and then click “Finish”. Type its code.
6. Again Right click the source packages folder from the workspace and select option “new” and select “java class” and type the name of the class as “LoginBean” and select package as “com.mine” click “Next”. Create the variables and generate getter and setter methods and type code of validate method.
7. Again Right click the source packages folder from the workspace and select option “new” and select “jsp” and type the name of the class as “login-sucess” and click “Next”. Type its code.
8. Again Right click the source packages folder from the workspace and select option “new” and select “jsp” and type the name of the class as “login-error” and click “Next”. Type its code.
9. Build and Run the project.

PROGRAM:

Index.html

```
<html>
  <head>
    <title>TODO supply a title</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
  </head>
  <body>
    <form action="ControllerServlet" method="post">
      Name:<input type="text" name="name"><br>
      Password:<input type="password"
      name="password"><br>
      <input type="submit" value="login">
    </form>
  </body>
</html>
```

ControllerServlet.java

```

import
java.io.IOException;
import
java.io.PrintWriter;
import
javax.servlet.RequestDispatcher;
import
javax.servlet.ServletException;
import
javax.servlet.http.HttpServlet;
import
javax.servlet.http.HttpServletRequest;
import
javax.servlet.http.HttpServletResponse;
@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    response.setContentType("text/html");
    PrintWriter out=response.getWriter();
    String
    name=request.getParameter("name");
    String
    password=request.getParameter("password");
    out.println("hello");
    LoginBean bean=new
    LoginBean();
    bean.setName(name);
    bean.setPassword(password);
    request.setAttribute("bean",bean);
    boolean status=bean.validate();
    if(status){
        RequestDispatcher rd=request.getRequestDispatcher("login-
        success.jsp"); rd.forward(request, response);
    }
    else{
        RequestDispatcher rd=request.getRequestDispatcher("login-
        error.jsp"); rd.forward(request, response);
    }
}

```

```
package com.mine;
public class
LoginBean {
    private String
    name,password; public
    String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public String getPassword()
    { return password;
    }
    public void setPassword(String
    password) { this.password =
    password;
    }
    public boolean validate()
    {
        return password.equals("admin");
    }
}
```

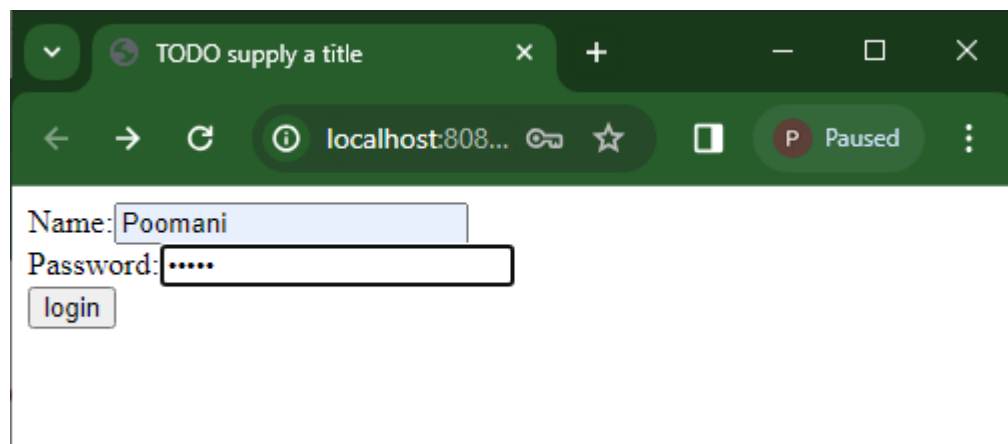
login-sucess.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@page import="com.mine.LoginBean"%>
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>JSP Page</title>
  </head>
  <body>
    <p>You are successfully logged in!</p>
    <% LoginBean
      bean=(LoginBean)request.getAttribute("bean");
      out.print("Welcome, "+bean.getName()); %>
  </body>
</html>
```

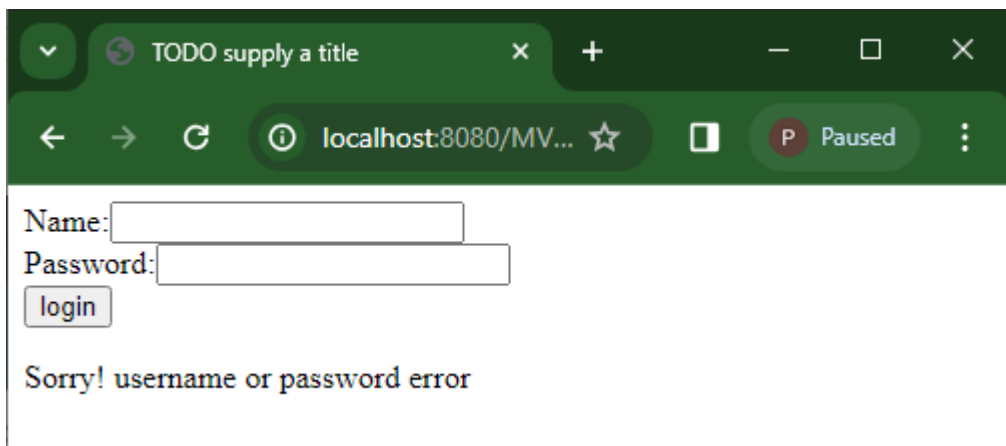
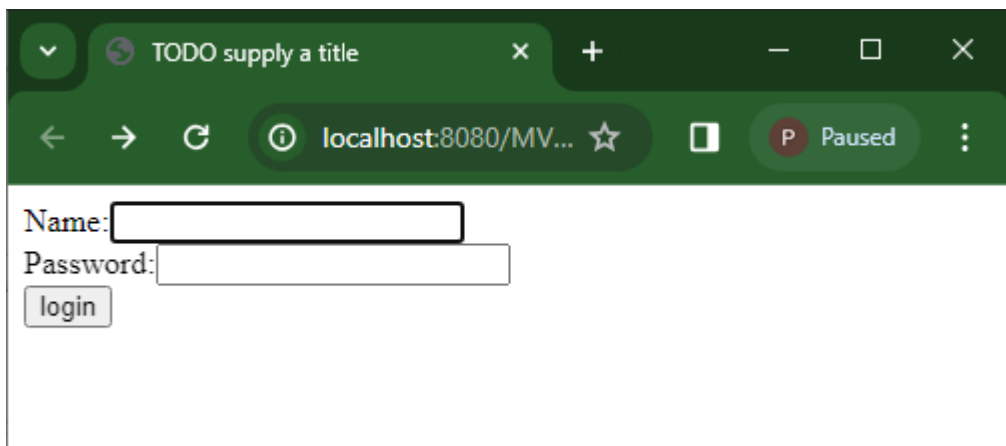
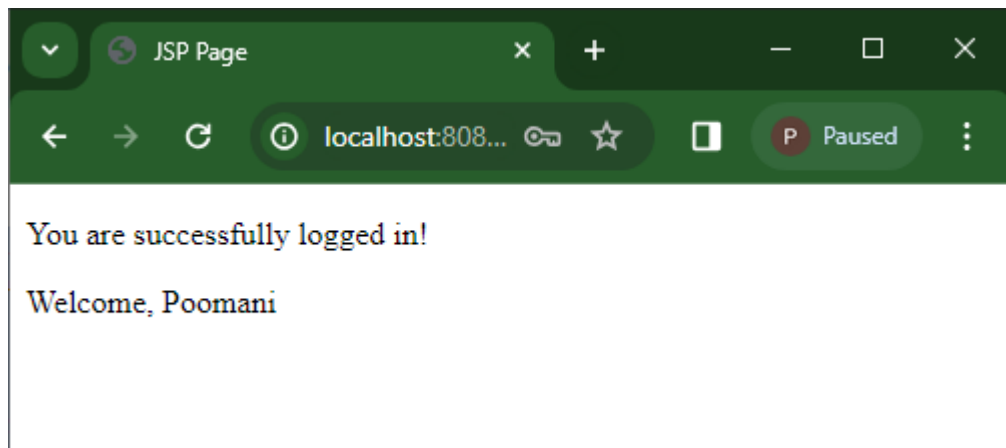
login-error.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@ include file="index.html" %>
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>JSP Page</title>
  </head>
  <body>
    <p>Sorry! username or password error</p>
  </body>
</html>
```

Output:



The screenshot shows a web browser window with a dark green header. The address bar displays 'localhost:808...'. The page content includes a login form with two input fields: 'Name:' containing 'Poomani' and 'Password:' containing five dots. Below the password field is a 'login' button.



RESULT:

Thus the program has been executed successfully and output verified.

EX.NO: 14

DATE:

TO BUILD A WEB APPLICATION USING STRUTS

AIM:

To build a web application that collects the user's name and displays "Hello World" followed by the user name.

PROCEDURE:

1. Start Netbeans IDE, Go to file menu and select new project, the new project dialog box shown as below opens. From the Categories select the option "java web" and from projects select "web Application" and click Next.
2. Type the name of the project as "LoginStruts" and select a location to save the project and click Next.
3. In the server select default server "GlassFish Server" and click "Next". Select the check box "Struts 1.3.10" and then Click "Finish".
4. Right click the source packages folder from the workspace and select option "new" and select "JSP" and type the name of the file as "login" and click "Finish". Repeat the same step and create two more JSP files with names "success" and "failure". Edit its code.
5. Again Right click the source packages folder from the workspace and select option "new" and select "ActionFormClass" and type the name of the class as "LoginBean" and select package as "com.myapp.struts" click "Next". Create the variables and generate getter and setter methods and comment the code lines inside the validate method.
6. Again Right click the source packages folder from the workspace and select option "new" and select "ActionClass" and type the name of the class as "loginform" and click "Next". Type its code.
7. In the struts-config.xml file add the forwards for success page and failure page. Edit the web.xml file by setting login.jsp as start page.
8. Build and Run the project.

PROGRAM:

login.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>JSP Page</title>
</head>
<body>
<form action="loginform.do" method="post">
  Username <input type="text"
  name="uname"/><br/>
  Password <input type="password" name="upass"/><br/>
  <input type="submit" value="LOGIN"/>
</form>
</body>
</html>
```

success.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>JSP Page</title>
  </head>
  <body>
    <h1>Login Success.Welcome to Your Page</h1>
  </body>
</html>
```

failure.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>JSP Page</title>
  </head>
  <body>
    <h1>Login Failed . No Authentication</h1>
  </body>
</html>
```

LoginBean.java

```
package com.myapp.struts;
import
javax.servlet.http.HttpServletRequest;
import
org.apache.struts.action.ActionErrors;
import
org.apache.struts.action.ActionMapping;
import
org.apache.struts.action.ActionMessage;
public class LoginBean extends org.apache.struts.action.ActionForm {
  private String uname;
  private String upass;
  public String
  getUname() {
    return uname;
  }
  public void setUname(String uname) {
    this.uname = uname;
  }
  public String getUpass()
  { return upass;
  }
  public void setUpass(String upass) {
```



```

    this.upass = upass;
}

public LoginBean()
{ super();
}

public ActionErrors validate(ActionMapping mapping, HttpServletRequest request) {
    ActionErrors errors = new ActionErrors();
    /*if (getName() == null || getName().length() < 1) {
        errors.add("name", new
            ActionMessage("error.name.required"));
        // TODO: add 'error.name.required' key to your resources
    }*/
    return errors;
}
}

```

Loginform.java

```

package com.myapp.struts;

import
javax.servlet.http.HttpServletRequest;
import
javax.servlet.http.HttpServletResponse;
import org.apache.struts.action.ActionForm;
import
org.apache.struts.action.ActionForward;
import
org.apache.struts.action.ActionMapping;
public class loginform extends org.apache.struts.action.Action
{ private static final String SUCCESS = "success";
  private static final String FAILURE =
    "failure"; @Override
  public ActionForward execute(ActionMapping mapping, ActionForm form,
    HttpServletRequest request, HttpServletResponse response)
    throws Exception {
    LoginBean lb= (LoginBean) form;
    if(lb.getUsername().equals("poomani") && lb.getUpass().equals("suresh"))
    {
        return mapping.findForward(SUCCESS);
    }
    else
    {
        return mapping.findForward(FAILURE);
    }
  }
}

```

struts-config.xml

```

<?xml version="1.0" encoding="UTF-8" ?>

```

```

<!DOCTYPE struts-config PUBLIC
    "-//Apache Software Foundation//DTD Struts Configuration
    1.3//EN" "http://jakarta.apache.org/struts/dtds/struts-
    config_1_3.dtd">
<struts-config>
    <form-beans>
        <form-bean name="LoginBean" type="com.myapp.struts.LoginBean"/>
    </form-beans>
    <global-exceptions>
    </global-exceptions>
    <global-forwards>
        <forward name="failure" path="/failure.jsp"/>
        <forward name="success" path="/success.jsp"/>
        <forward name="welcome" path="/Welcome.do"/>
    </global-forwards>
    <action-mappings>
        <action name="LoginBean" path="/loginform" scope="request"
type="com.myapp.struts.loginform" validate="false"/>
        <action path="/Welcome" forward="/welcomeStruts.jsp"/>
    </action-mappings>
    <controller processorClass="org.apache.struts.tiles.TilesRequestProcessor"/>
    <message-resources parameter="com/myapp/struts/ApplicationResource"/>

```

struts.xml

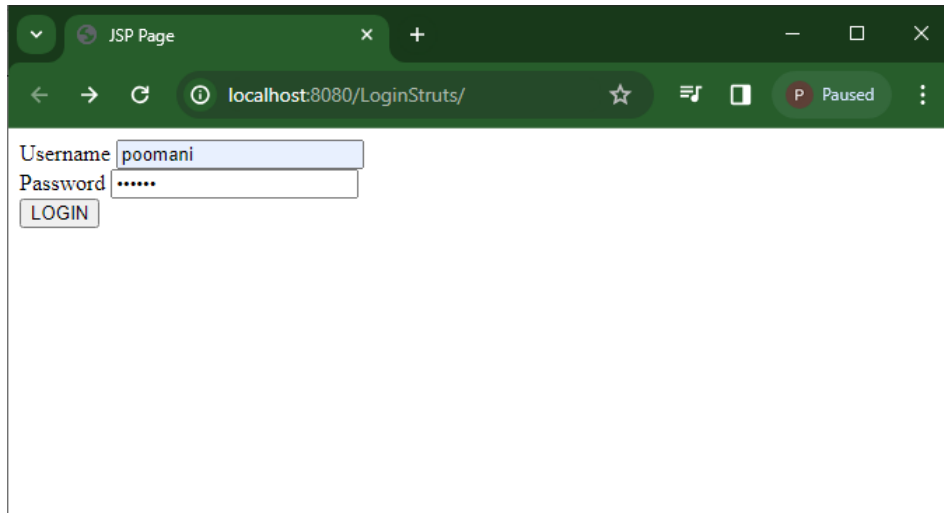
```

<?xml version="1.0" encoding="UTF-8"?>
<web-app version="3.1" xmlns="http://xmlns.jcp.org/xml/ns/javaee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd">
    <servlet>
        <servlet-name>action</servlet-name>
        <servlet-class>org.apache.struts.action.ActionServlet</servlet-class>
        <init-param>
            <param-name>config</param-name>
            <param-value>/WEB-INF/struts-config.xml</param-value>
        </init-param>
        <init-param>
            <param-name>debug</param-name>
            <param-value>2</param-value>
        </init-param>
        <init-param>
            <param-name>detail</param-name>
            <param-value>2</param-value>
        </init-param>
        <load-on-startup>2</load-on-startup>
    </servlet>
    <servlet-mapping>
        <servlet-name>action</servlet-name>
        <url-pattern>*.do</url-pattern>
    </servlet-mapping>
    <session-config>
        <session-
            timeout> 30
        </session-timeout>
    </session-config>

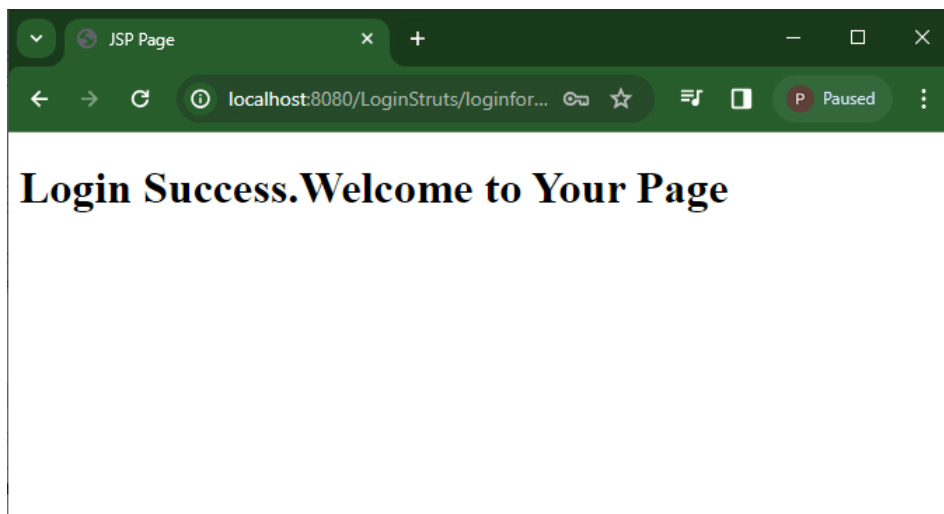
```

```
<welcome-file-list>  
  <welcome-file>login.jsp</welcome-file>  
</welcome-file-list>  
</web-app>
```

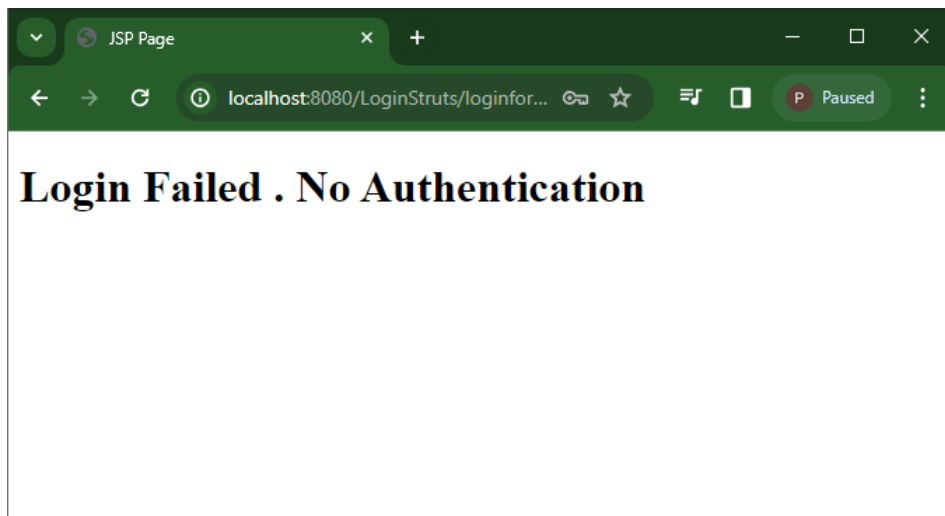
OUTPUT:



A screenshot of a web browser window with a dark green theme. The address bar shows 'localhost:8080/LoginStruts/'. The page content includes a login form with two input fields: 'Username' containing 'poomani' and 'Password' containing six dots. Below the password field is a 'LOGIN' button.



A screenshot of a web browser window with a dark green theme. The address bar shows 'localhost:8080/LoginStruts/loginfor...'. The page content displays a large, bold, black text message: 'Login Success.Welcome to Your Page'.



RESULT:

Thus the program has been executed successfully and output verified.

EX.NO: 14

REG.NO :

DATE:

NAME:

TO BUILD A WEB APPLICATION USING SPRING

Aim:

To build a web application that displays “Hello World” using Spring

Procedure:

Step 1 : Step 1: Create a Java Project with Maven

The very initial step is to create a simple Java Maven project using Spring Tool Suite IDE. Then follow the below options: **File > New > Maven Project > Select Create a Simple Project.**

Step 2 : Add Spring Dependency

Step 3 - Create Source Files

Step 4 - Create Bean Configuration Files

Step 5 - Running the Program

To do this, we should open the Main.java file then right click on that class, after that we can either use Run option or we can use the short cut key Ctrl+F11 to compile and run the application. At last we will be able to see the final output in the console as below:

Program:

pom.xml

```
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-core</artifactId>
  <version>4.0.0.RELEASE</version>
</dependency>
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-context</artifactId>
  <version>4.0.0.RELEASE</version>
</dependency>
```

HelloWorld.java

```
package spring_example;
```

```
public class HelloWorld {
```

```
    private String message;
```

```
    // GetterSetter for variable
```

```
    public void setMessage(String message){
        this.message = message;
```

```

public void getMessage(){
    System.out.println("Message : " + message);
}
}

```

Bean.xml

```

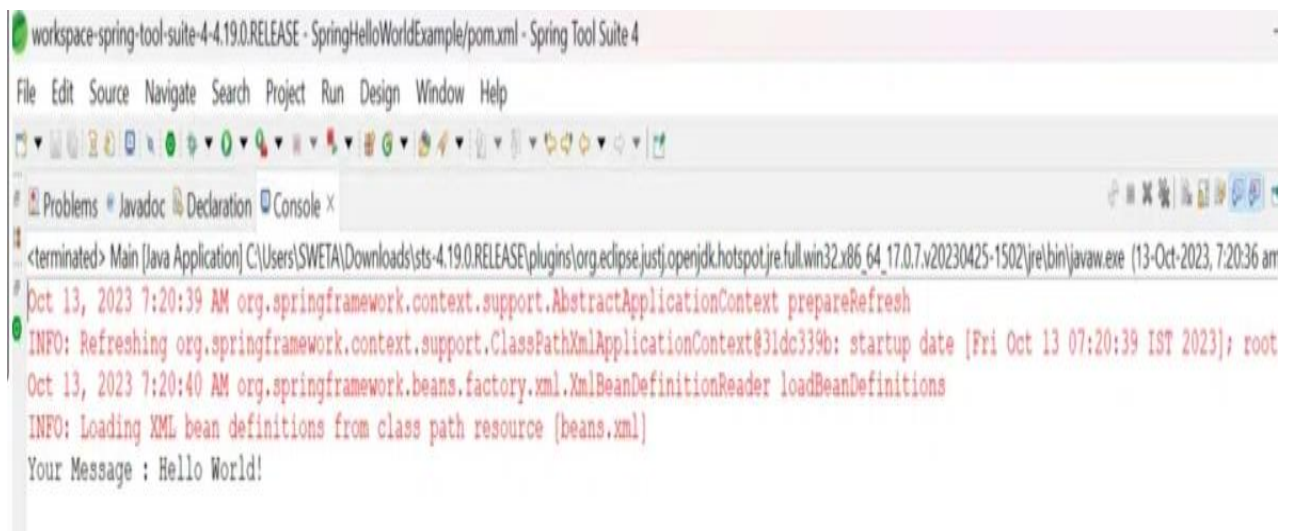
<?xml version = "1.0" encoding = "UTF-8"?>
<beans xmlns = "http://www.springframework.org/schema/beans"
    xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation = "http://www.springframework.org/schema/beans
    http://www.springframework.org/schema/beans/spring-beans-3.0.xsd">

    <bean id = "HelloWorld" class = "spring_example.HelloWorld">
        <property name = "message" value = "Hello World!"/>
    </bean>

</beans>

```

Output:



```

workspace-spring-tool-suite-4-4.19.0.RELEASE - SpringHelloWorldExample/pom.xml - Spring Tool Suite 4
File Edit Source Navigate Search Project Run Design Window Help
<terminated> Main [Java Application] C:\Users\SWETA\Downloads\sts-4.19.0.RELEASE\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_17.0.7.v20230425-1502\jre\bin\javaw.exe (13-Oct-2023, 7:20:36 am)
Oct 13, 2023 7:20:39 AM org.springframework.context.support.AbstractApplicationContext prepareRefresh
INFO: Refreshing org.springframework.context.support.ClassPathXmlApplicationContext@31dc339b: startup date [Fri Oct 13 07:20:39 IST 2023]; root
Oct 13, 2023 7:20:40 AM org.springframework.beans.factory.xml.XmlBeanDefinitionReader loadBeanDefinitions
INFO: Loading XML bean definitions from class path resource [beans.xml]
Your Message : Hello World!

```

RESULT:

Thus the program has been executed successfully and output verified.

EX.NO: 15

REG.NO :

DATE:

NAME:

TO UPLOAD A SELECTED FILE

Aim:

Create our view which will be required to browse and upload a selected file.

Procedure:

Step 1: Create an index.jsp with plain HTML upload form that allows the user to upload a file

Step 2: Create a simple jsp file success.jsp to display the outcome of our file upload in case it becomes success.

Step 3: Create error.jsp in case there is some error in uploading the file

Step 4: Create a Java class called uploadFile.java which will take care of uploading file and storing that file at a secure location

Step 5: Add The Struts Configuration in struts.xml

Step 6: Add <interceptor-ref> tag inside <action> in struts.xml

Step 7: Create The URL Action

Step 8: Build the WAR File and Run The Application

PROGRAM:

Index.jsp

```
<%@ page language = "java" contentType = "text/html; charset = ISO-8859-1"
    pageEncoding = "ISO-8859-1"%>
<%@ taglib prefix = "s" uri = "/struts-tags"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">
<html>
    <head>
        <title>File Upload</title>
    </head>
    <body>
        <form action = "upload" method = "post" enctype = "multipart/form-
            data"> <label for = "myFile">Upload your file</label>
```

```
<input type = "file" name = "myFile" />
<input type = "submit" value = "Upload"/>
</form>
</body>
</html>
```

success.jsp

```
<%@ page contentType = "text/html; charset = UTF-8"
%> <%@ taglib prefix = "s" uri = "/struts-tags" %>
<html>
  <head>
    <title>File Upload Success</title>
  </head>
  <body>
    You have successfully uploaded <s:property value =
    "myFileFileName"/> </body>
</html>
```

error.jsp

```
<%@ page contentType = "text/html; charset = UTF-8"
%> <%@ taglib prefix = "s" uri = "/struts-tags" %>
<html>
  <head>
    <title>File Upload Error</title>
  </head>
  <body>
    There has been an error in uploading the file.
  </body>
</html>
```

uploadFile.java

```
package com.srm.struts2;
import java.io.File;
import org.apache.commons.io.FileUtils;
import java.io.IOException;
```



```
import com.opensymphony.xwork2.ActionSupport;

public class uploadFile extends ActionSupport {
    private File myFile;
    private String myFileContentType;
    private String myFileFileName;
    private String destPath;
    public String execute() {
        /* Copy file to a safe location */
        destPath = "C:/apache-tomcat-6.0.33/work/";
        try {
            System.out.println("Src File name: " + myFile);
            System.out.println("Dst File name: " + myFileFileName);

            File destFile = new File(destPath, myFileFileName);
            FileUtils.copyFile(myFile, destFile); }

        catch(IOException e) {
            e.printStackTrace();
            return ERROR;
        }
        return SUCCESS;
    }
    public File getMyFile() {
        return myFile;
    }
}
```

```

public void setMyFile(File myFile) {
    this.myFile = myFile;
}

public String getMyFileContentType() {
    return myFileContentType;
}

public void setMyFileContentType(String myFileContentType)
{ this.myFileContentType = myFileContentType;
}

public String getMyFileFileName() {
    return myFileFileName;
}

public void setMyFileFileName(String myFileFileName)
{ this.myFileFileName = myFileFileName;
}
}

```

struts.xml

```

<?xml version = "1.0" Encoding = "UTF-8"?>
<!DOCTYPE struts PUBLIC
    "-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"
    "http://struts.apache.org/dtds/struts-2.0.dtd">
<struts>
    <constant name = "struts.devMode" value = "true" />
    <constant name = "struts.multipart.maxSize" value = "1000000"
/> <package name = "helloworld" extends = "struts-default">
    <action name = "upload" class = "com.srm.struts2.uploadFile">
        <result name = "success">/success.jsp</result>
        <result name = "error">/error.jsp</result>
    </action>
</package>
</struts>

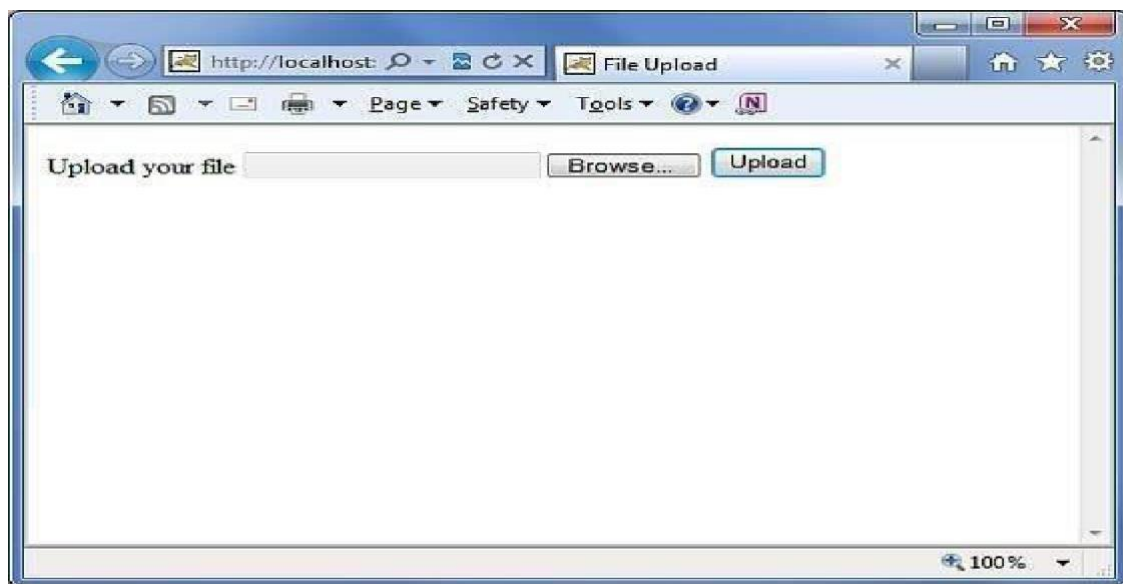
```

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"
  xmlns:web = "http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
  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>Struts 2</display-name>
  <welcome-file-list>
    <welcome-file>index.jsp</welcome-file>
  </welcome-file-list>
  <filter>
    <filter-name>struts2</filter-name>
    <filter-class>
      org.apache.struts2.dispatcher.FilterDispatcher
    </filter-class>
  </filter>

  <filter-mapping>
    <filter-name>struts2</filter-name>
    <url-pattern>/*</url-pattern>
  </filter-mapping>
</web-app>
```

Output:



RESULT:

Thus the program has been executed successfully and output verified.