ADVANCED JAVA PROGRAMMING

AMITY SCHOOL OF ENGINEERING AND TECHNOLOGY

NAME: **JAYANT BHALLA**

BRANCH: **B.TECH- 6CSE10-X**

ENROLLMENT NO.: **A12405215021**

BATCH: **2015-19**

FACULTY: **DR. ANIL KR. GIRI**

SIGNATURE: ______________________

INDEX

Experiment –1

Objective: Write a program to provide database connectivity using Database Driver to a employee table to insert, update delete data using Servlets.

Equipment/Software Used:

S.no.	Hardware	Software
1.	I7 processor	Os(windows 8)
2.	8 gb ram	Microsoft word
3.	Keyboard	Turbo c/c++
4.	Mouse	
5.	Monitor	
6.	Printer	

Theory-

Java JDBC is a java API to connect and execute query with the database. JDBC API uses jdbc drivers to connect with the database.

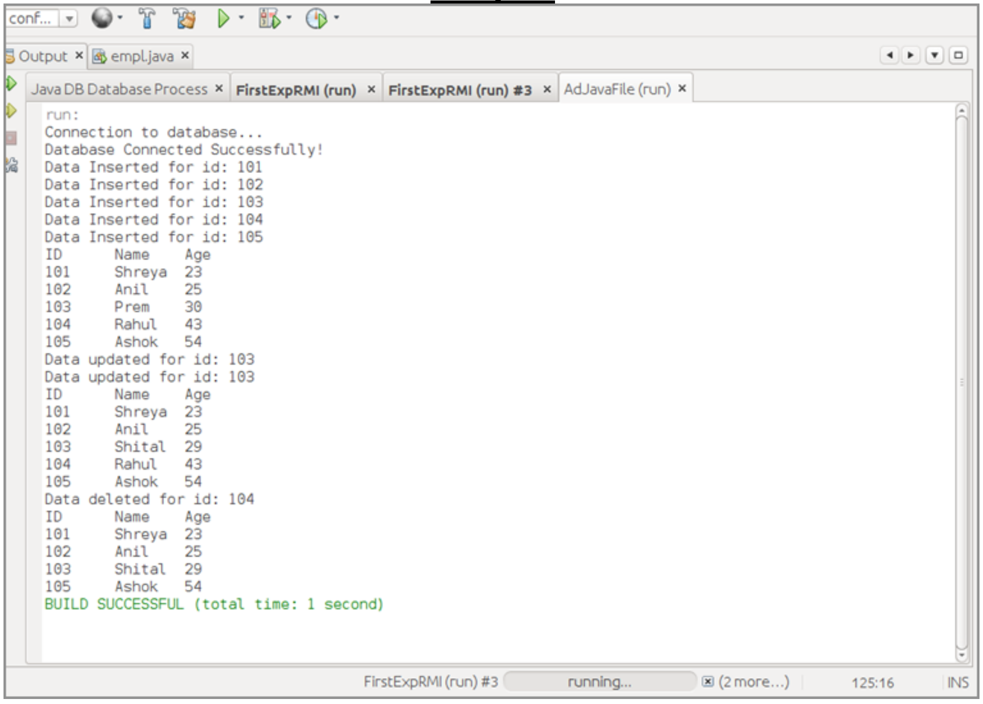
Before JDBC, ODBC API was the database API to connect and execute query with the database. But, ODBC API uses ODBC driver which is written in C language (i.e. platform dependent and unsecured). That is why Java has defined its own API (JDBC API) that uses JDBC drivers (written in Java language).

Source Code:

```
package exp1;
import java.sql.*;
import org.apache.derby.jdbc.EmbeddedDriver;
public class empl {
private static final String dbURL = “jdbc:derby://localhost:1527/Employee";
private static final String user = “root";
private static final String password = “2869";
private static Connection con;
private static Statement st;
private static String sqlQry;
private static void CreateConnection(){
try{
Class.forName(“org.apache.derby.jdbc.ClientDriver");
System.out.println("Connection to database…”);
Driver derbyEmbeddedDriver = new EmbeddedDriver();
DriverManager.registerDriver(derbyEmbeddedDriver);
con = DriverManager.getConnection(dbURL,user,password);
System.out.println("Database Connected Successfully!");
}catch(Exception e){
System.out.println(e.getMessage());
}
}
private static void insert(int id , String name, int age){
```

```
try{
sqlQry = "insert into EMPLOYEE values('"+id+"', '"+name+"', '"+age+"')";
st = con.createStatement();
st.executeUpdate(sqlQry);
System.out.println("Data Inserted for id: “+id);
st.close();
}catch(Exception e){
System.out.println(e);
}
}
private static void update(int id , int age){
try{
sqlQry = "update EMPLOYEE set age = '"+age+"'where id = '"+id+"'";
st = con.createStatement();
st.executeUpdate(sqlQry);
System.out.println("Data updated for id: “+id);
st.close();
}catch(Exception e){
System.out.println(e.getMessage());
}
}
private static void update(int id , String name){
try{
sqlQry = "update EMPLOYEE set name = '"+name+"'where id = ‘“+id+"'";
st = con.createStatement();
st.executeUpdate(sqlQry);
System.out.println("Data updated for id: “+id);
st.close();
}catch(Exception e){
System.out.println(e.getMessage());
}
}
private static void delete(int id){
try{
sqlQry = "delete from EMPLOYEE where id = ‘"+id+"'";
st = con.createStatement();
st.executeUpdate(sqlQry);
System.out.println("Data deleted for id: “+id);
st.close();
}catch(Exception e){
System.out.println(e.getMessage());
}}
private static void show(){
try{
sqlQry = "select * from EMPLOYEE”;
st = con.createStatement();
ResultSet rs = st.executeQuery(sqlQry);
System.out.println("ID\t"+"Name\t"+"Age");
while(rs.next()){
int id = rs.getInt(“ID");
String name = rs.getString("Name");
int age = rs.getInt("Age");
System.out.println(id +"\t"+name+"\t"+age);
}
st.close();
}catch(Exception e){
System.out.println(e.getMessage());
}
}
public static void main(String []ar){
CreateConnection();
insert(101,"Shreya",23);
insert(102,"Anil",25);
insert(103,"Prem",30);
insert(104,"Rahul",43);
insert(105,"Ashok",54);
show();
update(103 , 29);
update(103, "Shital");
show();
delete(104);
show();
}
}
```

Output:



```
run:
Connection to database...
Database Connected Successfully!
Data Inserted for id: 101
Data Inserted for id: 102
Data Inserted for id: 103
Data Inserted for id: 104
Data Inserted for id: 105
ID      Name    Age
101     Shreya  23
102     Anil    25
103     Prem    30
104     Rahul   43
105     Ashok   54
Data updated for id: 103
Data updated for id: 103
ID      Name    Age
101     Shreya  23
102     Anil    25
103     Shital  29
104     Rahul   43
105     Ashok   54
Data deleted for id: 104
ID      Name    Age
101     Shreya  23
102     Anil    25
103     Shital  29
105     Ashok   54
BUILD SUCCESSFUL (total time: 1 second)
```

Experiment –2

Objective: Write a program in JSP to provide Login Password Functionality using Database Driver

Equipment/Software Used:

S.no.	Hardware	Software
1.	I7 processor	Os(windows 8)
2.	8 gb ram	Microsoft word
3.	Keyboard	Turbo c/c++
4.	Mouse	
5.	Monitor	
6.	Printer	

Theory-

Java Server Pages (JSP) is a server-side programming technology that enables the creation of dynamic, platform-independent method for building Web-based applications. JSP have access to the entire family of Java APIs, including the JDBC API to access enterprise databases. The web server needs a JSP engine ie. container to process JSP pages. The JSP container is responsible for intercepting requests for JSP pages. This tutorial makes use of Apache which has built-in JSP container to support JSP pages development. A JSP container works with the Web server to provide the runtime environment and other services a JSP needs. It knows how to understand the special elements that are part of JSPs.

Source Code:

home.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<%@ page import="java.sql.*" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Home</title>
</head>
<body>
<%
Connection con= null;
PreparedStatement ps = null;
ResultSet rs = null;

String driverName = "com.mysql.jdbc.Driver";
String url = "jdbc:mysql://localhost:3306/record";
String user = "root";
String password = "root";
String sql = "select usertype from userdetail";

try {
Class.forName(driverName);
con = DriverManager.getConnection(url, user, password);
ps = con.prepareStatement(sql);
rs = ps.executeQuery();
%>
<form method="post" action="login.jsp">
<center><h2 style="color:green">JSP Login Example</h2></center>
<table border="1" align="center">
<tr>
<td>Enter Your Name :</td>
<td><input type="text" name="name"/></td>
</tr>
<tr>
<td>Enter Your Password :</td>
<td><input type="password" name="password"/></td>
</tr>
<tr>
<td>Select UserType</td>
<td><select name="usertype">
<option value="select">select</option>
<%
while(rs.next())
{
String usertype = rs.getString("usertype");
%>
<option value=<%=usertype%>><%=usertype%></option>
<%
}
}
catch(SQLException sqe)
{
out.println("home"+sqe);
}
%>
</select>
</td>
</tr>
<tr>
<td></td>
<td><input type="submit" value="submit"/></td>
</table>
</form>
</body>
</html>
```

login.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<%@ page import="java.sql.*" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Login</title>
</head>
<body>
<%! String userdbName;
String userdbPsw;
String dbUsertype;
%>
<%
Connection con= null;
PreparedStatement ps = null;
ResultSet rs = null;

String driverName = "com.mysql.jdbc.Driver";
String url = "jdbc:mysql://localhost:3306/record";
String user = "root";
String dbpsw = "root";

String sql = "select * from userdetail where name=? and password=? and usertype=?";

String name = request.getParameter("name");
String password = request.getParameter("password");
String usertype = request.getParameter("usertype");

if((!(name.equals(null) || name.equals("")) && !(password.equals(null) || password.equals(""))) &&
!usertype.equals("select"))
{
try{
Class.forName(driverName);
con = DriverManager.getConnection(url, user, dbpsw);
ps = con.prepareStatement(sql);
ps.setString(1, name);
ps.setString(2, password);
ps.setString(3, usertype);
rs = ps.executeQuery();
if(rs.next())
{
userdbName = rs.getString("name");
userdbPsw = rs.getString("password");
dbUsertype = rs.getString("usertype");
if(name.equals(userdbName) && password.equals(userdbPsw) && usertype.equals(dbUsertype))
{
```

```
session.setAttribute("name",userdbName);
session.setAttribute("usertype", dbUsertype);
response.sendRedirect("welcome.jsp");
}
}
else
response.sendRedirect("error.jsp");
rs.close();
ps.close();
}
catch(SQLException sqe)
{
out.println(sqe);
}
}
else
{
%>
<center><p style="color:red">Error In Login</p></center>
<%
getServletContext().getRequestDispatcher("/home.jsp").include(request, response);
}
%>
</body>
</html>
```

welcome.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Welcome</title>
</head>
<body>
<p>Welcome, <%=session.getAttribute("name")%></p>
<p><a href="logout.jsp">Logout</a>
</body>
</html>
```

error.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Login Error</title>
</head>
```

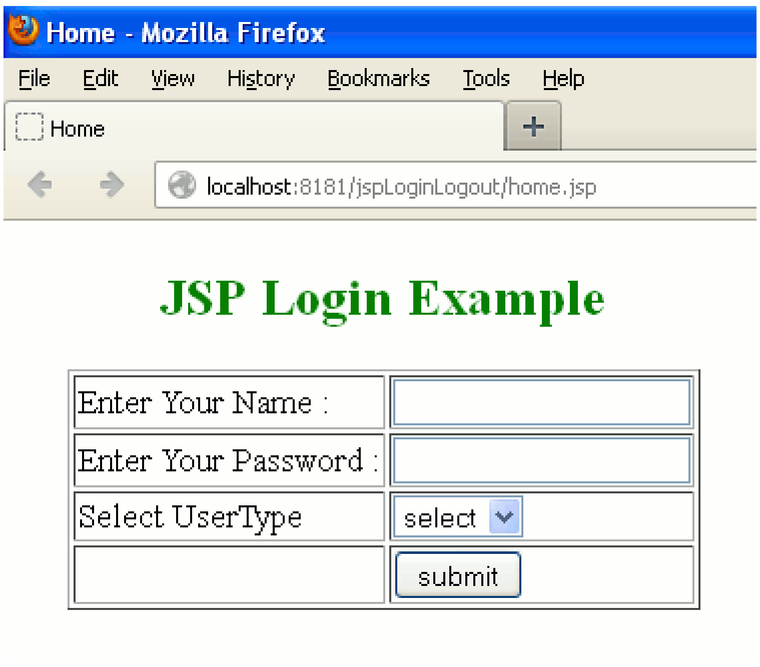
```
<body>
<center><p style="color:red">Sorry, your record is not available.</p></center>
<%
getServletContext().getRequestDispatcher("/home.jsp").include(request, response);
%>
</body>
</html>
```

logout.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Logout</title>
</head>
<body>
<% session.invalidate(); %>
<p>You have been successfully logout</p>
</body>
</html>
```

Output:

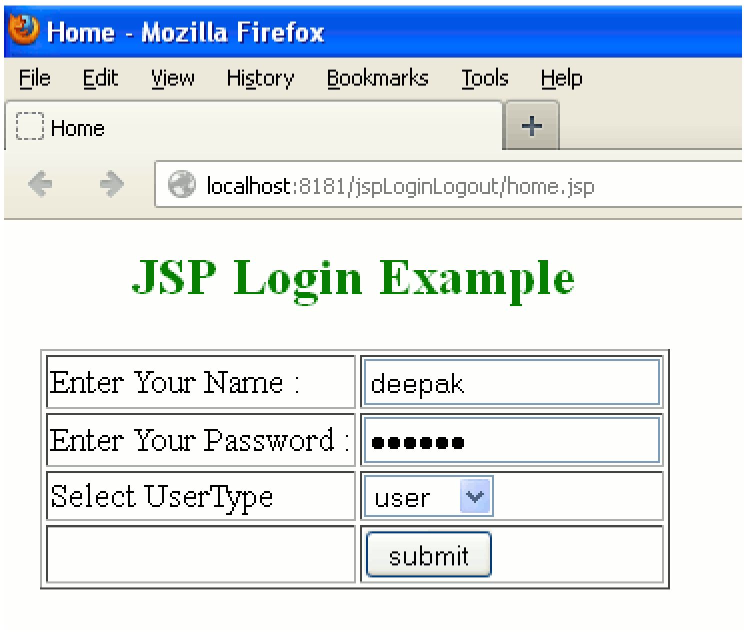
1. Home page will be looked as follows :


Home - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Home
localhost:8181/jspLoginLogout/home.jsp

JSP Login Example

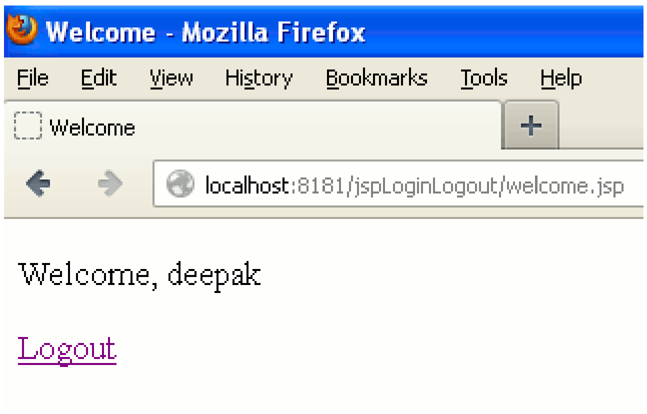
Enter Your Name :	
Enter Your Password :	
Select UserType	select
	submit

2. When you will enter the value and if it matched from the corresponding database table value then the output will be as follows :


Home - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Home
localhost:8181/jspLoginLogout/home.jsp

JSP Login Example

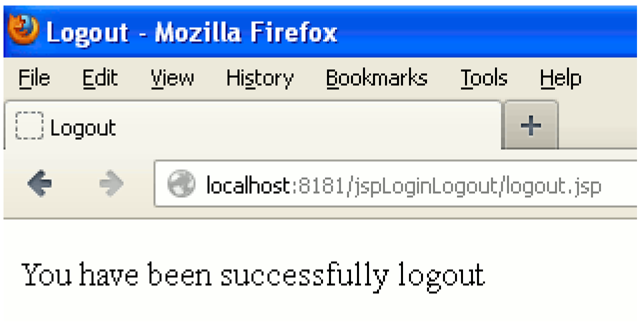
Enter Your Name :	deepak
Enter Your Password :	••••••
Select UserType	user
	submit


Welcome - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Welcome
localhost:8181/jspLoginLogout/welcome.jsp

Welcome, deepak

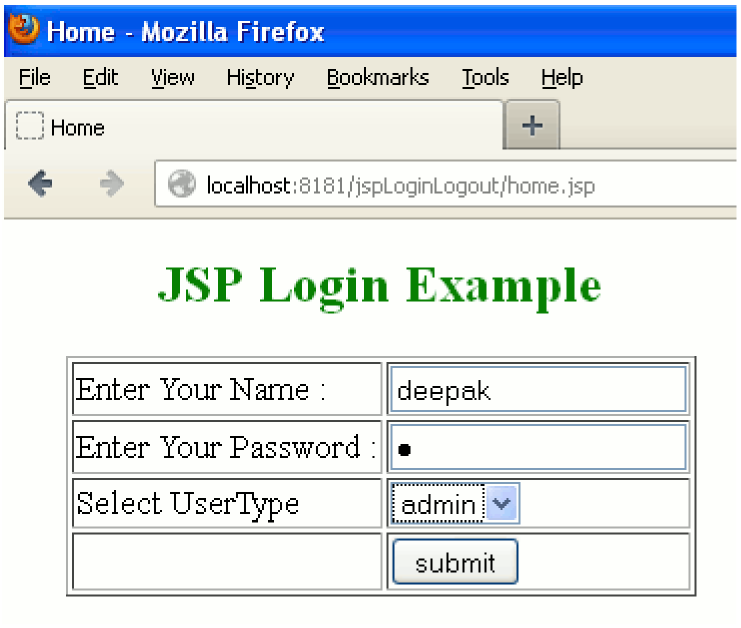
Logout

3. When you will click on the link for logout then the output will be as follows :


Logout - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Logout
localhost:8181/jspLoginLogout/logout.jsp

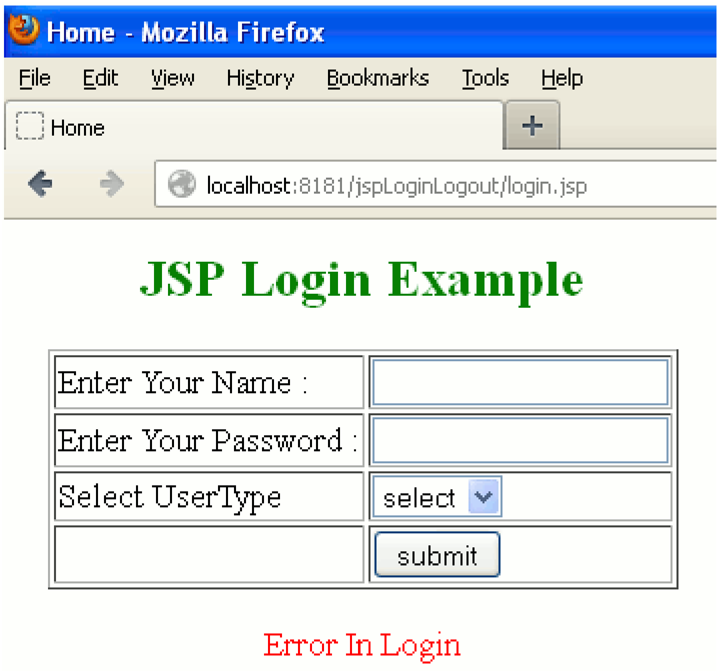
You have been successfully logout

4. But, when you will incorrect value in the respective fields then the output will be as follows :


Home - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Home
localhost:8181/jspLoginLogout/home.jsp

JSP Login Example

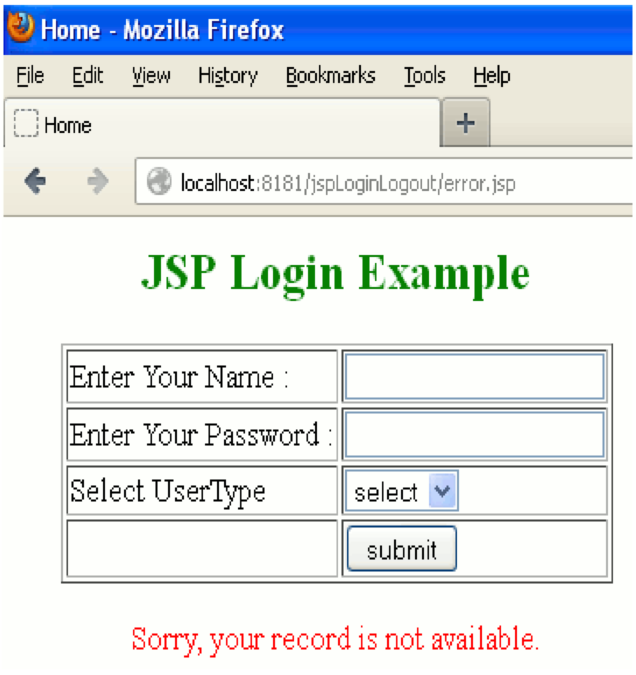
Enter Your Name :	deepak
Enter Your Password :	●
Select UserType	admin
	submit


Home - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Home
localhost:8181/jspLoginLogout/login.jsp

JSP Login Example

Enter Your Name :	
Enter Your Password :	
Select UserType	select
	submit

Error In Login

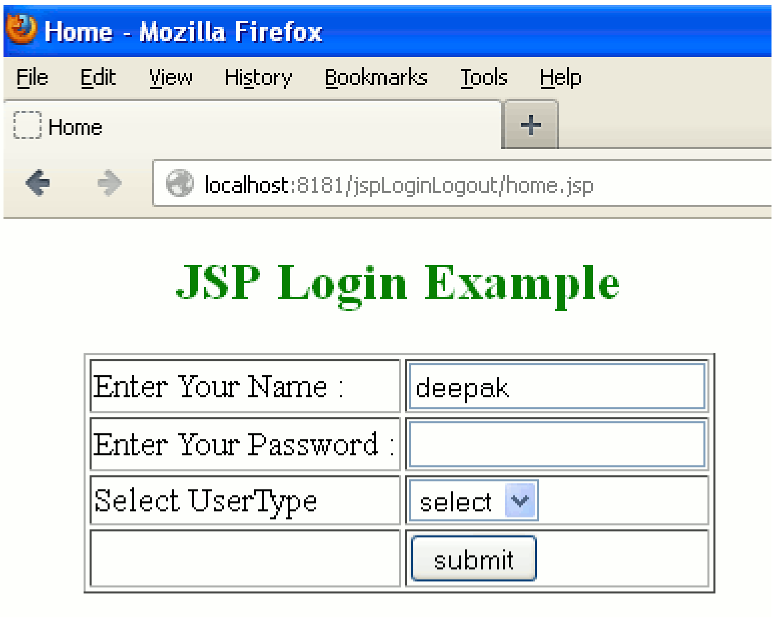

Home - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Home
localhost:8181/jspLoginLogout/error.jsp

JSP Login Example

Enter Your Name :	
Enter Your Password :	
Select UserType	select
	submit

Sorry, your record is not available.

5. If you left the any field empty or if you don't select the usertype then the output will be as follows :


Home - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Home
localhost:8181/jspLoginLogout/home.jsp

JSP Login Example

Enter Your Name :	deepak
Enter Your Password :	
Select UserType	select
	submit

Experiment – 3

Objective: Write a program using servlet to write persistent and non-persistent cookies on client side.

Equipment/Software Used:

S.no.	Hardware	Software
1.	I7 processor	Os(windows 8)
2.	8 gb ram	Microsoft word
3.	Keyboard	Turbo c/c++
4.	Mouse	
5.	Monitor	
6.	Printer	

Theory-

Servlets are the Java platform technology of choice for extending and enhancing Web servers. Servlets provide a component-based, platform-independent method for building Web-based applications, without the performance limitations of CGI programs. And unlike proprietary server extension mechanisms (such as the Netscape Server API or Apache modules), servlets are server- and platform-independent. This leaves you free to select a "best of breed" strategy for your servers, platforms, and tools. Servlets have access to the entire family of Java APIs, including the JDBC API to access enterprise databases. Servlets can also access a library of HTTP-specific calls and receive all the benefits of the mature Java language, including portability, performance, reusability, and crash protection. Today servlets are a popular choice for building interactive Web applications. Third-party servlet containers are available for Apache Web Server, Microsoft IIS, and others. Servlet containers are usually a component of Web and application servers, such as BEA WebLogic Application Server, IBM WebSphere, Sun Java System Web Server, Sun Java System Application Server, and others.

Source Code:

```
<form action="servlet1" method="post">
Name:<input type="text" name="userName"/><br/>
<input type="submit" value="go"/>
</form>
FirstServlet.java
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;


public class FirstServlet extends HttpServlet {

 public void doPost(HttpServletRequest request, HttpServletResponse response){
  try{

  response.setContentType("text/html");
  PrintWriter out = response.getWriter();
```

```
    String n=request.getParameter("userName");
    out.print("Welcome "+n);

    Cookie ck=new Cookie("uname",n);//creating cookie object
    response.addCookie(ck);//adding cookie in the response

    //creating submit button
    out.print("<form action='servlet2'>");
    out.print("<input type='submit' value='go'>");
    out.print("</form>");

    out.close();

        }catch(Exception e){System.out.println(e);}
  }
}
```

SecondServlet.java

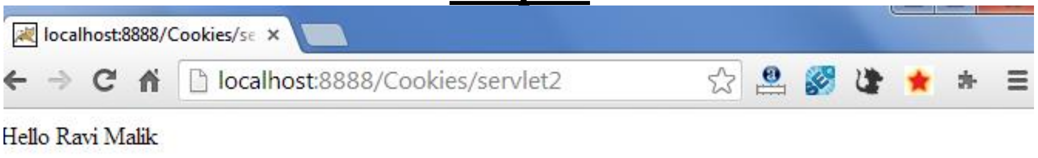
```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class SecondServlet extends HttpServlet {

public void doPost(HttpServletRequest request, HttpServletResponse response){
    try{

    response.setContentType("text/html");
    PrintWriter out = response.getWriter();

    Cookie ck[]=request.getCookies();
    out.print("Hello "+ck[0].getValue());

    out.close();

         }catch(Exception e){System.out.println(e);}
    } }
```

web.xml

```
<web-app>

<servlet>
<servlet-name>s1</servlet-name>
<servlet-class>FirstServlet</servlet-class>
</servlet>
  <servlet-mapping>
<servlet-name>s1</servlet-name>
<url-pattern>/servlet1</url-pattern>
</servlet-mapping>
<servlet>
<servlet-name>s2</servlet-name>
```

```
<servlet-class>SecondServlet</servlet-class>
</servlet>
<servlet-mapping>
<servlet-name>s2</servlet-name>
<url-pattern>/servlet2</url-pattern>
</servlet-mapping>
</web-app>
```

Output:



```
Hello Ravi Malik
```

Experiment – 4

Objective: Write a program to create a custom tag in JSP that gives Forward and Include Actions.

Equipment/Software Used:

S.no.	Hardware	Software
1.	I7 processor	Os(windows 8)
2.	8 gb ram	Microsoft word
3.	Keyboard	Turbo c/c++
4.	Mouse	
5.	Monitor	
6.	Printer	

Theory-

The jsp:forward action tag is used to forward the request to another resource it may be jsp, html or another resource.

The jsp:include action tag is used to include the content of another resource it may be jsp, html or servlet. The jsp include action tag includes the resource at request time so it is better for dynamic pages because there might be changes in future. The jsp:include tag can be used to include static as well as dynamic pages.

Source Code:

Forward action

Index.jsp

```
<html>
<body>
<jsp:param name="name" value="hello" />
 <jsp:include page="print.jsp"></jsp:include>
</body>
</html>
```

Print.jsp

```
<html>
<body>
<%= request.getParameter("name") %>
      <% out.print("Today is "+java.util.Calendar.getInstance().getTime() );          %>
   </body>
</html>
```

INCLUDE ACTION

Index.jsp

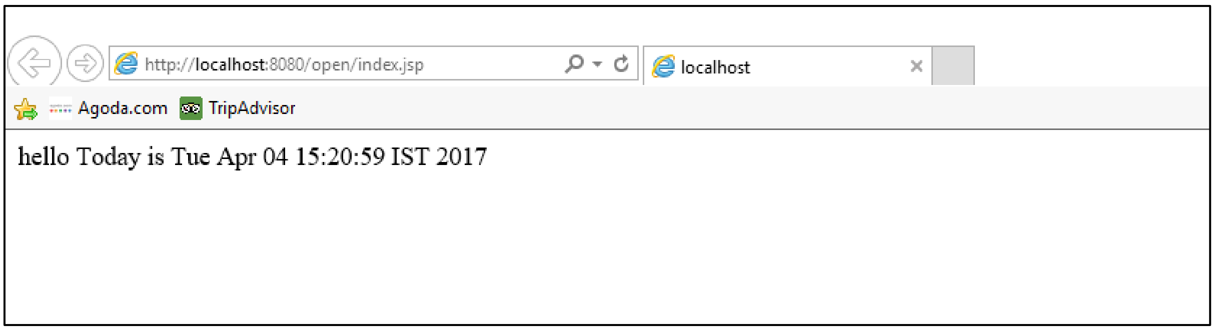
```
<html>
<body>
 <h2>This  is page 1 </h2>
 <jsp:include  page="print.jsp"></jsp:include>
<h2>End  of page 1</h2>
</body>
</html>
```

Print.jsp

```
<html>
<body>
<%= request.getParameter("name")  %>
     <% out.print("Today  is  "+java.util.Calendar.getInstance().getTime()  );          %>
   </body>
</html>
```

Output:

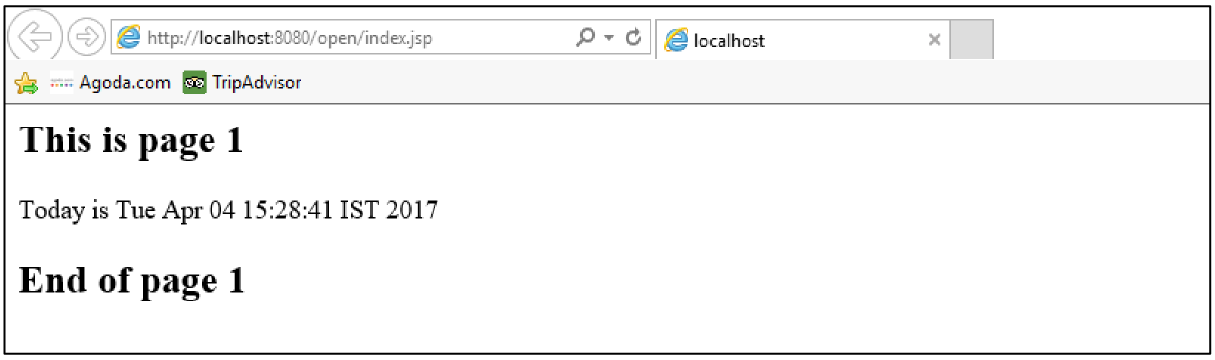
Forward action



http://localhost:8080/open/index.jsp

```
hello Today is Tue Apr 04 15:20:59 IST 2017
```

INCLUDE ACTION



http://localhost:8080/open/index.jsp

```
This is page 1

Today is Tue Apr 04 15:28:41 IST 2017

End of page 1
```

Experiment – 5

Objective: Write a program to print server side information using JSP as Client IP Address,URL , Context Info, hit count.

Equipment/Software Used:

S.no.	Hardware	Software
1.	I7 processor	Os(windows 8)
2.	8 gb ram	Microsoft word
3.	Keyboard	Turbo c/c++
4.	Mouse	
5.	Monitor	
6.	Printer	

Theory-

JavaServer Pages (JSP) is a server-side programming technology that enables the creation of dynamic, platform-independent method for building Web-based applications. JSP have access to the entire family of Java APIs, including the JDBC API to access enterprise databases. The web server needs a JSP engine ie. container to process JSP pages. The JSP container is responsible for intercepting requests for JSP pages. This tutorial makes use of Apache which has built-in JSP container to support JSP pages development. A JSP container works with the Web server to provide the runtime environment and other services a JSP needs. It knows how to understand the special elements that are part of JSPs.

Source Code:

```
<%@ page import="java.io.*,java.util.*" %>

<html>

<head>

<title>Applcation object in JSP</title>

</head>

<body>

<%

    Integer hitsCount =

      (Integer)application.getAttribute("hitCounter");

    if( hitsCount ==null || hitsCount == 0 ){

       /* First visit */

       out.println("Welcome to my website!");
```

```
         hitsCount = 1;

      }else

         out.println("Welcome back to my website!");

         hitsCount += 1;

      }

      application.setAttribute("hitCounter", hitsCount);

%>

<center>

<p>Total number of visits: <%= hitsCount%></p>

</center>

</body>

</html>
```

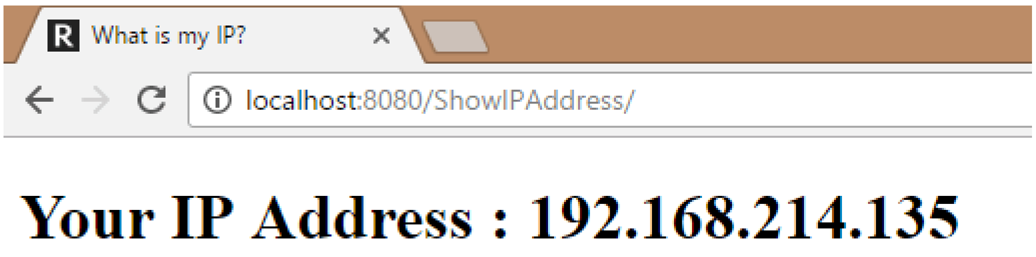
Output

```
Welcome back to my website!


Total number of visits: 12
```

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

    pageEncoding="ISO-8859-1"%>

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

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=ISO-
8859-1">

<title>What is my IP?</title>

</head>

<body>
```

```
        <%! String ipAddress; %>
        <%ipAddress = request.getRemoteAddr();%>


        <h1>Your IP Address : <%=ipAddress %></h1>


</body>
</html>
```

Output:



What is my IP?

localhost:8080/ShowIPAddress/

Your IP Address : 192.168.214.135

Experiment – 6

Objective: Write a program to implement Stateless Session Beans.

Equipment/Software Used:

S.no.	Hardware	Software
1.	I7 processor	Os(windows 8)
2.	8 gb ram	Microsoft word
3.	Keyboard	Turbo c/c++
4.	Mouse	
5.	Monitor	
6.	Printer	

Theory-

Stateless Session bean *is a business object that represents business logic* only. It doesn't have state (data).

In other words, *conversational state* between multiple method calls is not maintained by the container in case of stateless session bean.The stateless bean objects are pooled by the EJB container to service the request on demand.It can be accessed by one client at a time. In case of concurrent access, EJB container routes each request to different instance

EJB is an acronym for enterprise java bean. It is a specification provided by Sun Microsystems to develop secured, robust and scalable distributed applications.To get information about distributed applications, visit RMI Tutorial first.To run EJB application, you need an application server (EJB Container) such as Jboss, Glassfish, Weblogic, Websphere etc. It performs: life cycle management, security, transaction management, and object pooling. EJB application is deployed on the server, so it is called server side component also.

Source Code:

```
import javax.ejb.Remote;
@Remote
public interface AdderImplRemote {
int add(int a,int b);
}
```

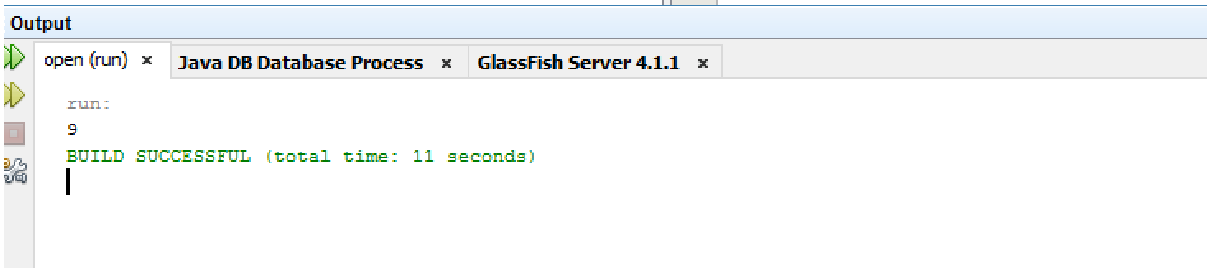
AdderImpl.java

```
import javax.ejb.Stateless;
@Stateless(mappedName="st1")
public class AdderImpl implements AdderImplRemote {
public int add(int a,int b){
return a+b;
}
}
```

AdderImpl.java

```
package com.javatpoint;
import javax.naming.Context;
import javax.naming.InitialContext;
public class Test {
public static void main(String[] args)throws Exception {
Context context=new InitialContext();
AdderImplRemote remote=(AdderImplRemote)context.lookup("st1");
System.out.println(remote.add(4,5));
}
}
```

Output:



```
run:
9
BUILD SUCCESSFUL (total time: 11 seconds)
```

Experiment – 7

Objective: Write a program to implement Struts.

Equipment/Software Used:

S.no.	Hardware	Software
1.	I7 processor	Os(windows 8)
2.	8 gb ram	Microsoft word
3.	Keyboard	Turbo c/c++
4.	Mouse	
5.	Monitor	
6.	Printer	

Theory-

Apache Struts 1 is an open-source web application framework for developing Java EE web applications. It uses and extends the Java Servlet API to encourage developers to adopt a model–view–controller (MVC) architecture. It was originally created by Craig McClanahan and donated to the Apache Foundation in May, 2000. Formerly located under the Apache Jakarta Project and known as Jakarta Struts, it became a top-level Apache project in 2005.

Source Code:

//input.jsp

```
<%@ taglib prefix="s" uri="/struts-tags" %>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Struts2 beginner example application</title>
</head>
<body>
    <center>
        <h2>Calculate sum of two numbers</h2>
        <s:form action="calculateSumAction" method="post">
            <s:textfield name="x" size="10" label="Enter X" />
            <s:textfield name="y" size="10" label="Enter Y" />
            <s:submit value="Calculate" />
        </s:form>
    </center>
</body>
</html>
```

//SumAction.java

```
public class SumAction{
    private int x;
```

```java
    private int y;
    private int sum;

    /**
     * The action method
     * @return name of view
     */
    public String calculate() {
        sum = x + y;
        return SUCCESS;
    }

    // setters and getters for x, y, and sum:

    public int getX() {
        return x;
    }

    public void setX(int x) {
        this.x = x;
    }

    public int getY() {
        return y;
    }

    public void setY(int y) {
        this.y = y;
    }

    public int getSum() {
        return sum;
    }

    public void setSum(int sum) {
        this.sum = sum;
    }
}
```

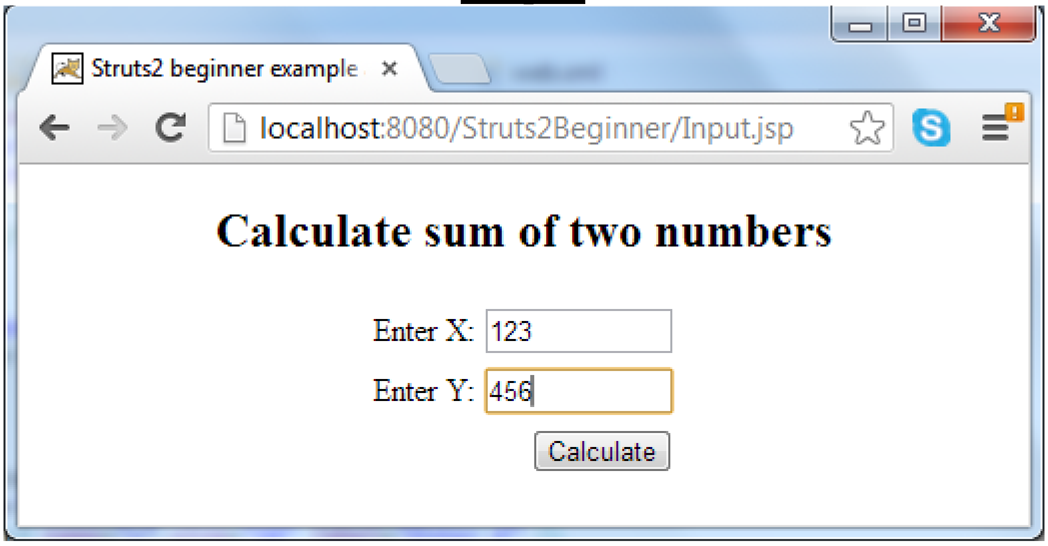
//Result.jsp

```jsp
<%@ taglib prefix="s" uri="/struts-tags" %>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Sum Result</title>
</head>
<body>
    Sum of <s:property value="x"/>
    and <s:property value="y"/>
    is:
    <s:property value="sum"/>
</body>
</html>
```

//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>
    <package name="Struts2Beginner" extends="struts-default">
        <action name="calculateSumAction"
class="net.codejava.struts.SumAction"
            method="calculate">
            <result name="success">/Result.jsp</result>
            <result name="input">/Input.jsp</result>
        </action>
    </package>
</struts>
```

Output:



Calculate sum of two numbers

Enter X: 123

Enter Y: 456

Experiment-8

Objective: Write a program to implement Entity Bean.

Equipment/Software Used:

S.no.	Hardware	Software
1.	I7 processor	Os(windows 8)
2.	8 gb ram	Microsoft word
3.	Keyboard	Turbo c/c++
4.	Mouse	
5.	Monitor	
6.	Printer	

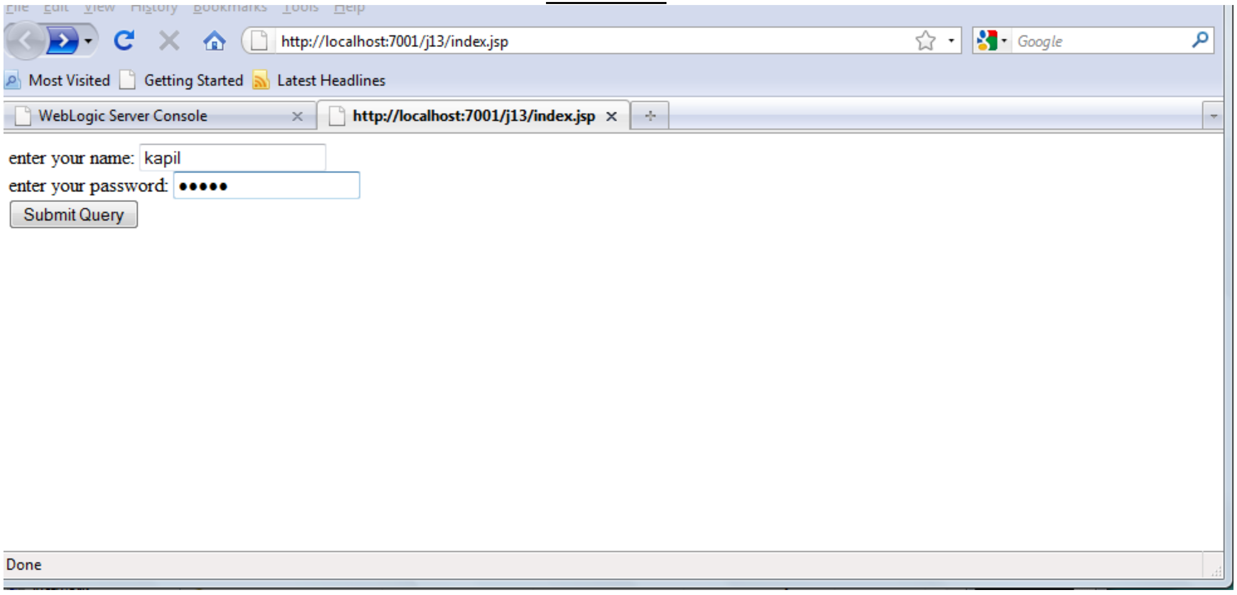
Source Code:

MyBean.java

```
package my;
public class MyBean
{
        private String name;
        private int pass;
        public void setName(String name)
        {
                this.name=name;
        }
        public void setPass(int pass)
        {
                this.pass=pass;
        }
        public String getName()
        {
                return name;
        }
        public int getPass()
        {
                return pass;
        }

        public String validate()
        {
        try
        {
                if(name.equals("kapil"))
                        return "valid";
        }
        catch(Exception e){}
        return "invalid";
        }

        public int add(int x,int y)
        {
                return x+y;
        }
}
```
Bean.jsp
```
<%
        String name=request.getParameter("name");
        String pass=request.getParameter("pass");
%>

<jsp:useBean id="t1" class="my.MyBean"/>
<jsp:setProperty name="t1" property="name"  param="name"/>
<jsp:setProperty name="t1" property="pass" param="pass"/>

<%
        String s=t1.validate();
        int r=t1.add(10,20);
        out.println(s+r);
%>

<jsp:getProperty name="t1" property="name"/>
<jsp:getProperty name="t1" property="pass"/>
```

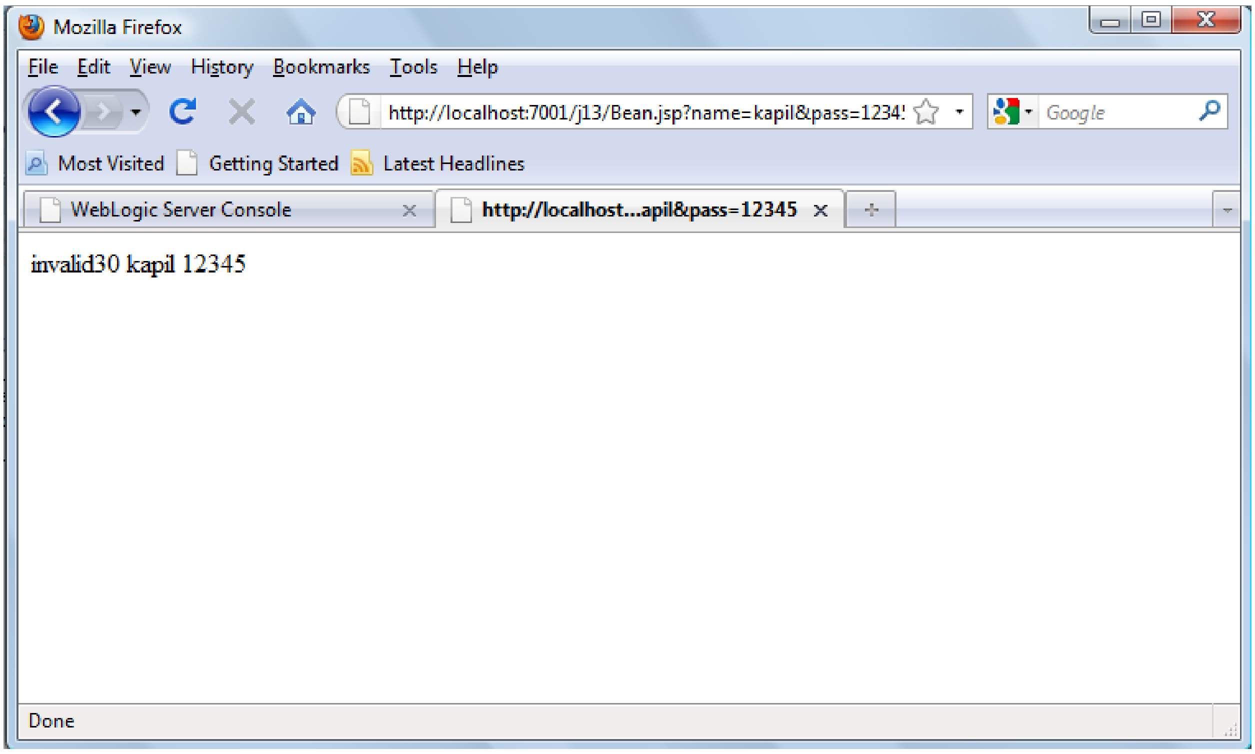
Index.jsp
```
<html>
<body>
<form action="Bean.jsp">
enter your name:
<input type="text" name="name">
<br>
enter your password:
<input type="password" name="pass"/>
<br>
<input type="submit"/>
</body>
</html>
```
Web.xml
```
<web-app>
</web-app>
```

OUTPUT:



http://localhost:7001/j13/index.jsp

enter your name: kapil
enter your password: •••••



http://localhost:7001/j13/Bean.jsp?name=kapil&pass=1234!

```
invalid30 kapil 12345
```

Experiment – 9

Objective: Write a program to develop an application of android.

Equipment/Software Used:

S.no.	Hardware	Software
1.	I7 processor	Os(windows 8)
2.	8 gb ram	Microsoft word
3.	Keyboard	Turbo c/c++
4.	Mouse	
5.	Monitor	
6.	Printer	

Theory-

Android Studio is nothing but an integrated development environment (IDE) for development of Android platform. Software used for development of android applications. For this application nothing but Android Studio2.1.1 version is used. Moreover to install Android Studio , JDK ie. Java Development Kit is required. We create a simple android application for adding two numbers. It's a simple beginner's level application and the understanding of this code will help in the implementation of other features of a basic calculator

Source Code:

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   package="com.example.me_dell.myfirstapplication">
   <application
      android:allowBackup="true"
      android:icon="@mipmap/ic_launcher"
      android:label="@string/app_name"
      android:supportsRtl="true"
      android:theme="@style/AppTheme">
      <activity android:name=".MainActivity">
         <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
         </intent-filter>
      </activity>
   </application>
</manifest>
```

MainActivity.java

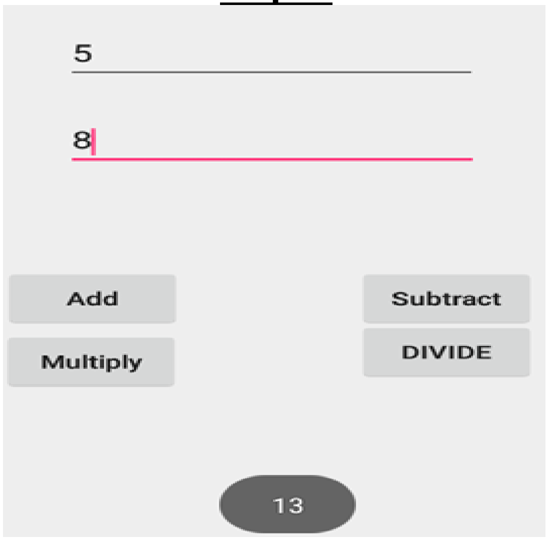
```
public class MainActivity extends AppCompatActivity {
   EditText firstNumber;
   EditText secondNumber;
   TextView Result;
```

```
    Button btnAdd;
    double num1, num2, sum;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        firstNumber = (EditText) findViewById(R.id.num1);
        secondNumber = (EditText) findViewById(R.id.num2);
        Result = (TextView) findViewById(R.id.Result);
        btnAdd = (Button) findViewById(R.id.AddButton);
        btnAdd.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {
                num1 = Double.parseDouble(firstNumber.getText().toString());
                num2 = Double.parseDouble(secondNumber.getText().toString());
                sum = num1 + num2;
                Result.setText("Result: "+Double.toString(sum));
            }});}}
```

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.example.me_dell.myfirstapplication.MainActivity">
    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentStart="true"
        android:layout_alignParentTop="true"
        android:layout_marginTop="40dp"
        android:text="Enter 1st number:" />
    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="51dp"
        android:text="Enter 2nd number:"
        android:layout_below="@+id/textView"
        android:layout_alignParentStart="true" />
    <EditText
        android:id="@+id/num1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignBaseline="@+id/textView"
        android:layout_alignBottom="@+id/textView"
        android:layout_alignParentEnd="true"
        android:ems="10"
        android:inputType="numberDecimal" />
    <EditText
        android:id="@+id/num2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:ems="10"
        android:inputType="numberDecimal"
        android:layout_alignBaseline="@+id/textView2"
        android:layout_alignBottom="@+id/textView2"
        android:layout_alignParentEnd="true" />
    <Button
        android:id="@+id/AddButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentStart="true"
        android:layout_below="@+id/num2"
        android:layout_marginStart="90dp"
        android:layout_marginTop="28dp"
        android:text="Add" />
    <TextView
        android:id="@+id/Result"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/AddButton"
        android:layout_marginTop="57dp"
        android:layout_toEndOf="@+id/textView2"
        android:textSize="30sp"
        android:textStyle="bold" />
</RelativeLayout>
```

Ouput:


5
8
Add
Subtract
Multiply
DIVIDE
13