**EX NO 1a.** **WEB PAGE DESIGN**

**Aim:**

Develop a web page using all basic HTML Tags.

**Procedure:**

Step 1: Create a HTML document.

Step 2: Including the image of the college in the html document using the <IMG>

tag. Step 3: Using <FRAMESET> tag divide the html page as preferred.

Step 4: Create relevant web pages for the college and display the menu in the left side of the window.

Step 5: Make the respective forms to be displayed in the right side of web page when the menu is clicked.

**Source Code:**

**main.htm**
```
<html>
<title>SRI SAIRAM INSTITUTE OF TECHNOLOGY</title>
<frameset cols="35%,30%,35%">
<frame name="1" src="f1.htm">
<frame name="2" src="f2.htm">
<frame name="3" src="f3.htm">
</frameset>
<a href="f1.htm"> IMAGE </a>
</html>
```

**f1.htm**
```
<html>
<title>college</title>
<head>SSIT</head>
<body>
<body bgcolor="white">
<img src="C:\Users\student\Pictures\ssit.jpg" height="500" width="380">
</body>
</html>
```

**f3.htm**
```
<html>
<head><h2><font color="black">HOME</h2></font><body>
<body bgcolor="pink">
<h4><li>Sri Sairam Institue of Technology is one of the top most colleges under Anna
```

```
University</li>
<li>The college aims for providing the students through knowledge and discipline</li>
<li>The vision of our college is <br>TRUTH<br>PEACE<br>LOVE</li>
</h4>
</body>
</head>
</html>
```

**f2.htm**

```
<html>
<head><h1><font color="Black">MENU</h1></font></head>
<body>
<body bgcolor="grey">
<li><a href="courses.htm">COURSES</a></li>
<li><a href="academics.htm">ACADEMICS</a></li>
<li><a href="achievements.htm">ACHIEVEMENTS</a></li>
<li><a href="contact.htm">CONTACT</a></li>
</body>
</html>
```

**courses.htm**

```
<html>
<head><h2><font color="white">COURSES</font><h2>
<title>course</title>
<body>
<body bgcolor="blue">
<h3><li>ECE</li></h3>
<h3><li>EEE</li></h3>
<h3><li>CSE</li></h3>
<h3><li>IT</li></h3>
</body>
</head>
</html>
```

**academics.htm**

```
<html>
<head><h2><font color="white">ACADEMICS</font></h2>
<body>
<body bgcolor="lavender">
<h4>Rules</h4>
<li>Students must attend the classes regularly</li>
<li>Attendance must be above 90% for every individual</li>
<li>Formal shirts and pants are only allowed</li>
<li>Jeans pant and t shirts and half sarees are also not allowed</li>
<h4>Pass percentage</h4>
<table>
```

```
<tr>
<th>Year</th>
<th>Pass Percentage</th>
<th>No. of University Ranks</th>
</tr>
<tr>
<td>2009</td>
<td>97.6</td>
<td>7</td
</tr>
<tr>
<td>2010</td>
<td>96</td>
<td>9</td>
</tr>
<tr>
<td>2011</td>
<td>98.3</td>
<td>12</td>
</tr>
</table>
</head>
</body>
</html>
```

**achievements.htm**

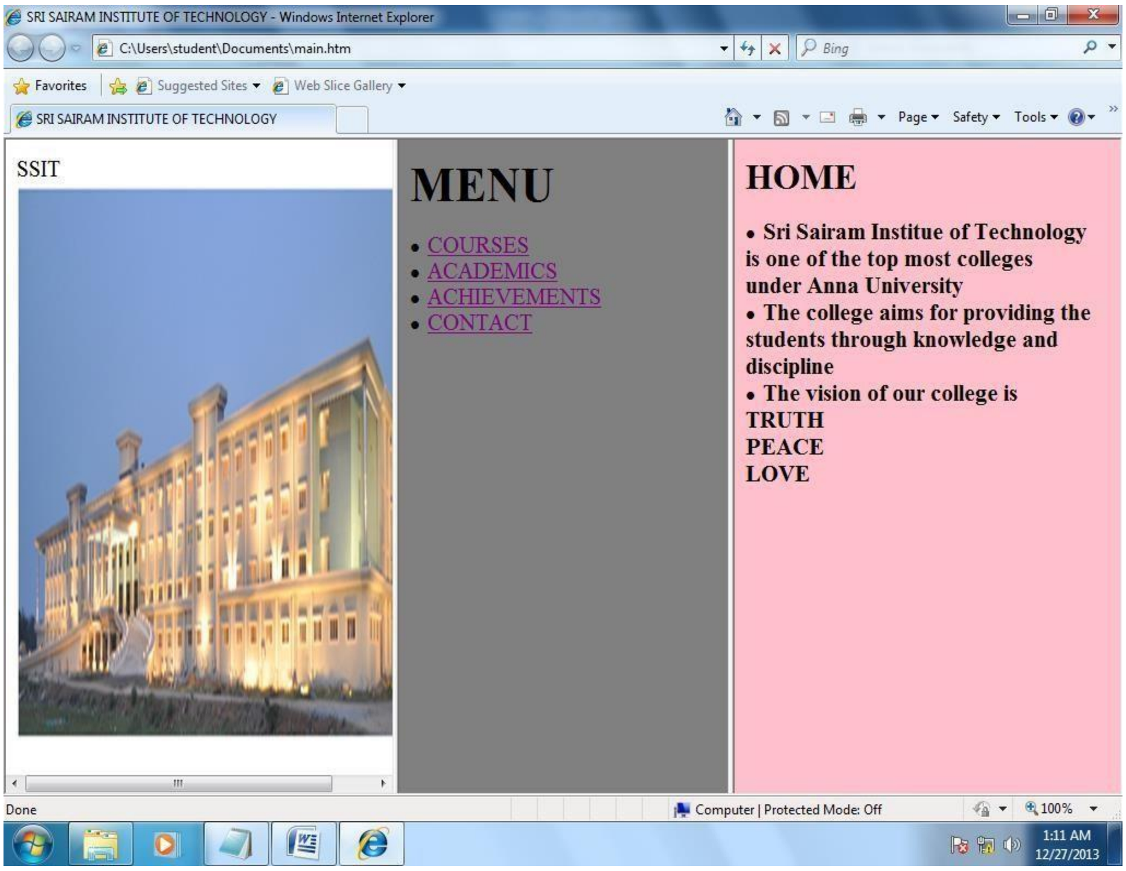
```
<html>
<title>Rank holders</title>
<body>
<body bgcolor="purple">
<head><h2><font color="white">ACHIEVEMENTS OF SSIT</font></h2>
<li>3 universtiy rank holders</li>
<li>Pass Percentage 96% </li>
<li>100% placement for eligible students</li>
<li>Experienced staffs</li>
</head>
</body>
</html>
```

**contact.htm**

```
<html>
<head><h2><font color="white"> CONTACT </font></h2></head>
<body>
<body bgcolor="orange">
<p>Sri Sairam Institute of Technology<br>
  Sai Leo Nagar<br>
  West Tambaram<br>
  Chennai-44.<br>
  E-mail:www.sairamit.edu.in</p>
```

```
<p>Administrative Office:<br>
  "SAI BHAVAN",#31,Madley road,<br>
  T.Nagar,Chennai-600017.<br>
  Tel no:044-42267777.</p>
</body>
</html>
```

**Output:**


SRI SAIRAM INSTITUTE OF TECHNOLOGY - Windows Internet Explorer
C:\Users\student\Documents\main.htm
Bing
Favorites
Suggested Sites
Web Slice Gallery
SRI SAIRAM INSTITUTE OF TECHNOLOGY
Page
Safety
Tools
Done
Computer | Protected Mode: Off
100%
1:11 AM
12/27/2013

SSIT

MENU

- COURSES
- ACADEMICS
- ACHIEVEMENTS
- CONTACT

HOME

- Sri Sairam Institue of Technology is one of the top most colleges under Anna University
- The college aims for providing the students through knowledge and discipline
- The vision of our college is
TRUTH
PEACE
LOVE

**Result:**

## EX NO: 1b WEB PAGE DESIGN TO EMBED IMAGE MAP, HOTSPOTS AND HYPERLINKS

**Aim:**

To create a web page which includes a map and display the related information when a hot spot is clicked in the map.

**Procedure:**

Step 1: Create a html file with map tag

Step 2: Set the source attribute of the img tag to the location of the image and also set the usemap attribute

Step 3:Specify an area with name, shape and href set to the appropriate values

Step 4: Repeat step 3 as many hot spots you want to put in the map

Step 5: Create html files for each and every hot spots the user will select.

**Source Code:**

**link.htm**

```
<html>
<head>
<title>Image Map</title>
</head>
<body><img src="C:\Users\student\Pictures\india.jpg" usemap="#metroid" ismap="ismap" >
<map name="metroid" id="metroid">
<area href='TamilNadu.htm' shape='circle' coords='175,495,30' title='TamilNadu'/>
<area href = "Karnataka.htm" shape = "rect" coords = "100,400,150,450" title = "Karnataka" />
<area href = "AndhraPradesh.htm" shape = "poly" coords = "150, 415,
175,348,265,360,190,420,190,440" title = "Andhra Pradesh" />
<area href = "Kerala.htm" shape = "poly" coords =
"108,455,150,515,115,490,148,495,110,448,155,501" title = "Kerala" />
</map>
</body>
</html>
```

**Tamilnadu.htm**

```
<html>
<head>TAMILNADU</head>
<body>
<p> Tamil nadu's capital is Chennai. People speak Tamil language.It is famous for it culture and
temples</p>
</body>
</html>
```

**Karnataka.htm**

```
<html>
<head>KARNATAKA.htm</head>
<body>
<p> Karnataka's capital is Bangalore.People speak Kannada language..Mysore is the place which is
famous for silk and sandal</p>
</body>
</html>
```

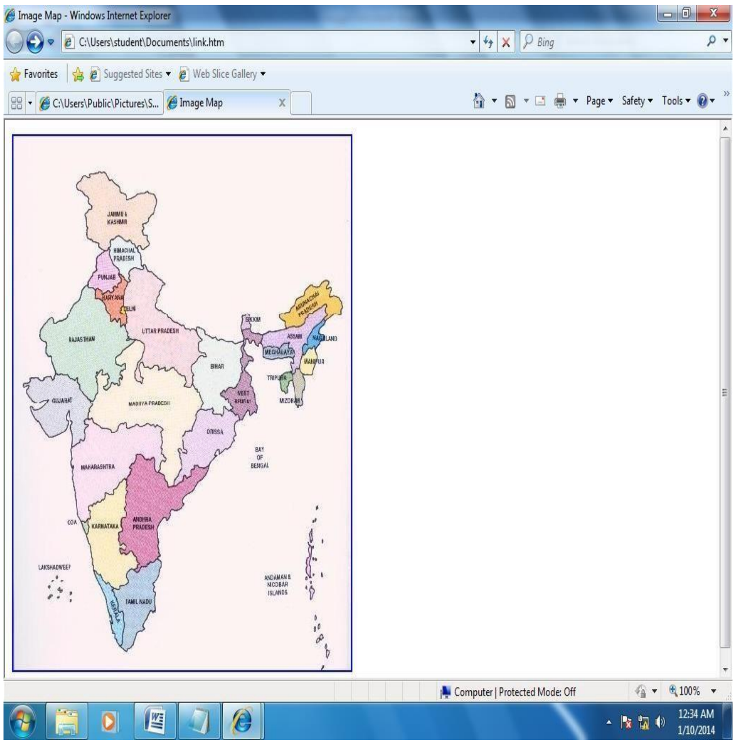
**Andhrapradesh.htm**

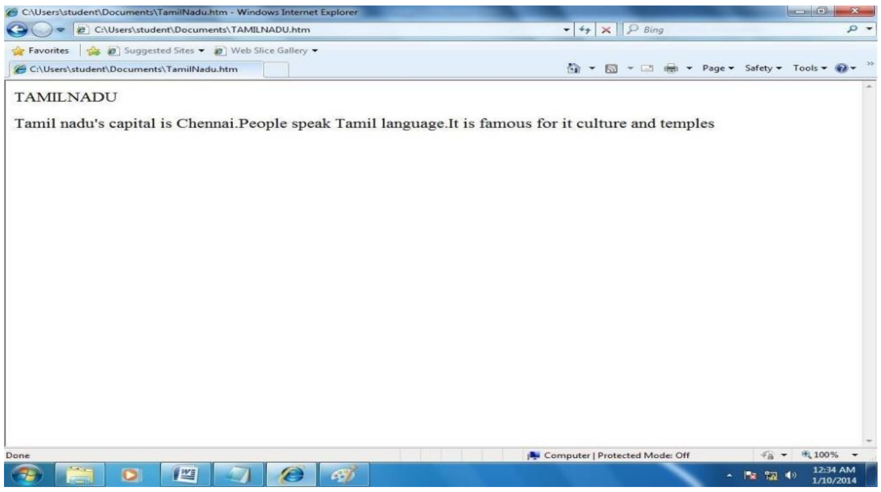
```
<html>
<head>ANDHRAPRADESH</head>
<body>
<p> Andhra Pradesh's capital is Hyderabad. People speak Telugu language.It is famous for rice
varieties.</p>
</body>
</html>
```

**Kerala.htm**

```
<html>
<head>KERALA</head>
<body>
<p> Kerala's capital is Thiruvanandhapuram. People speak the language Malayalam.It is famous for
temples and several medicinal treatments.</p>
</body>
</html>
```

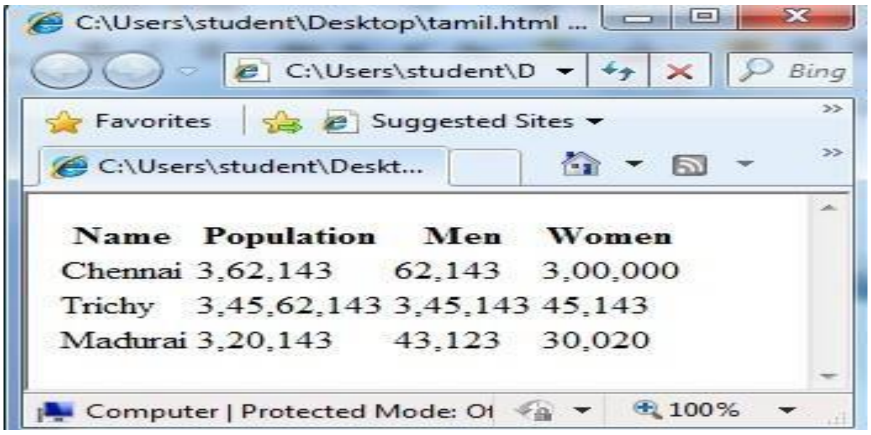
Output:





TAMILNADU

Tamil nadu's capital is Chennai.People speak Tamil language.It is famous for it culture and temples



| Name | Population | Men | Women |
|---|---|---|---|
| Chennai | 3,62,143 | 62,143 | 3,00,000 |
| Trichy | 3,45,62,143 | 3,45,143 | 45,143 |
| Madurai | 3,20,143 | 43,123 | 30,020 |

**Result:**

## EX.NO:2 WEB PAGES DESIGN USING DIFFERENT TYPES OF CASCADING STYLE SHEETS

**AIM:**

To create a web page that displays college information using various style sheets.

**ALGORITHM:**

```
Step 1: Create a web page with frame sets consisting two frames
Step 2: In the first frame include the links
Step 3: In the second frame set display the web page of the link
Step 4: create a external style sheets
Step 5: create an inline and internal style sheets and make it link to the external style
sheets.
```

**Source Code:**

**Inline Css**

```
<html>
<!DOCTYPE html>
<html lang="en">
<head>
   <title>Inline CSS Example</title>
</head>
<body>
   <p style="color: red; font-size: 18px;">This is a paragraph with inline CSS styling.</p>
</body>
</html>
</html>
```

**Internal Css**

```
<!DOCTYPE html>
<html>
<head>
   <title>Internal CSS Example</title>
   <style>
      /* Internal CSS styles */
      .container {
  position: relative;
}

/* Bottom right text */
.text-block {
  position: absolute;
  top: 20px;
  left: 20px;
  background-color: black;
  color: white;
  padding-left: 20px;
  padding-right: 20px;
}
    </style>
</head>
<body>
    <div class="container">
    <img src="C:\Users\Divya Transport\Desktop\istock-664061854-scaled.jpg" alt="internal css
example" width="1500" height="800">
    <div class="text-block">
    <h1>WEB TECHNOLOGY</h1>
    </div>
</body>
</html>
```

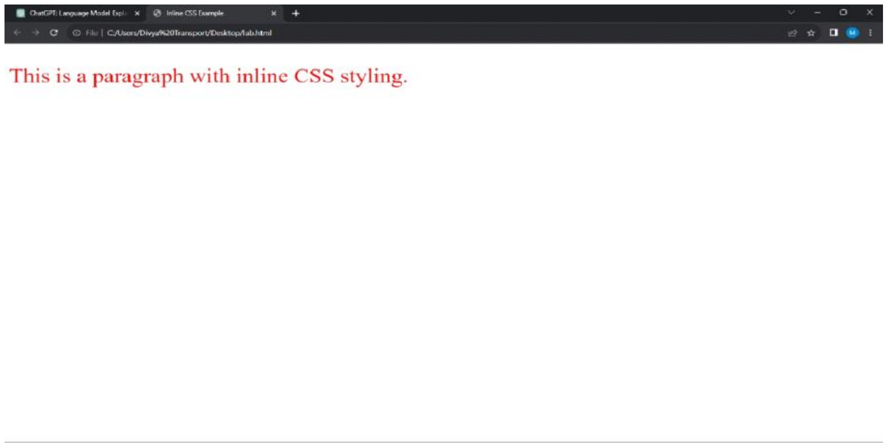
**External Css :**
**external.html**

```
<!DOCTYPE html>
<html >
<head>
    <link rel="stylesheet" href="styles.css"> <!-- Linking to the external CSS file -->
    <title>External CSS Example</title>
</head>
<body>
    <body bgcolor="#B6D0E2">
    <h1>Welcome to my website</h1>
    <p>This is an example of using external CSS to style HTML elements.</p>
    <form>
    Username:<input type="text" size="20">@gmail.com<br>
    Password:<input type="password" name="pwd"><br>
    <input type="submit">
    </form>
</body>
</html>
```

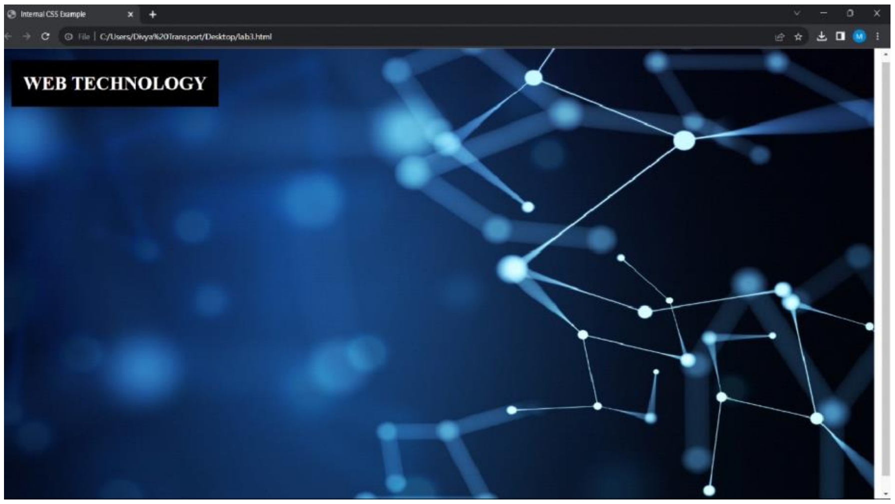
**styles.css**

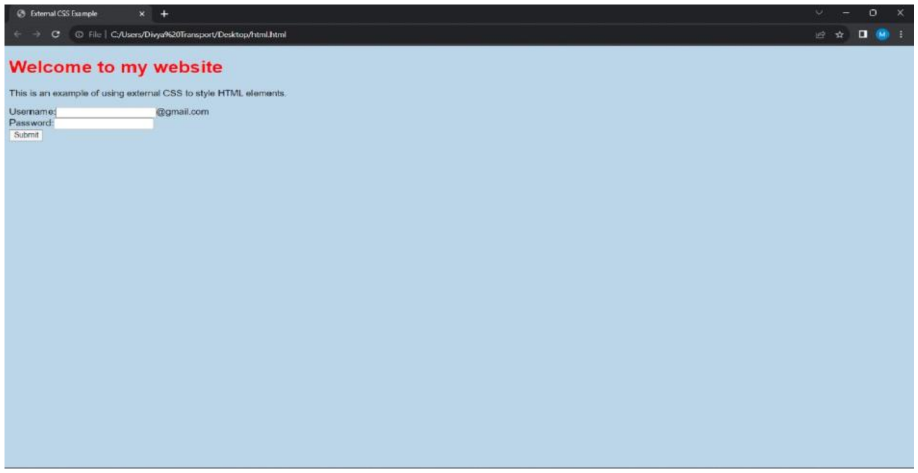
```
/* External CSS styles */
body {
    font-family: Arial, sans-serif;
}

h1 {
    color: red;
}

p {
    font-size: 16px;
```

}


ChatGPT: Language Model Expl
Inline CSS Example
File | C:/Users/Divya%20Transport/Desktop/lab.html

This is a paragraph with inline CSS styling.





Welcome to my website

This is an example of using external CSS to style HTML elements.

Username: @gmail.com
Password:
Submit

.

## Result

## EX NO: 3 CLIENT SIDE SCRIPTS FOR VALIDATING WEB FORM CONTROLS USING DHTML

**Aim:**

To develop an html webpage to validate form using DHTML

**Algorithm:**

Step 1: Create a JavaScript document which has username, password, email id,phone no etc. Step

2: validate user name, it should contain only alphabets.

Step 3: and validate the password, fix length for validation

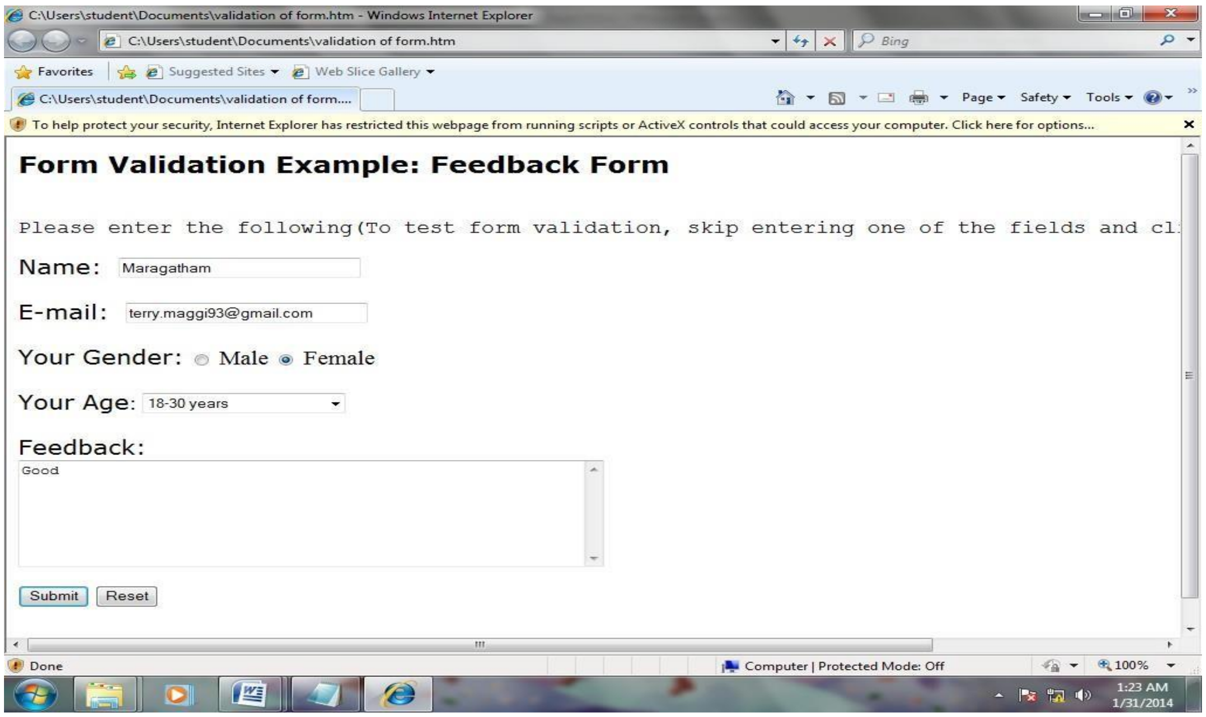
Step 4: Email id must contain the symbol @,check whether the phone number is having 10 digits.

**Source code:**

```
<!doctype html public "-//w3c//dtd html 4.0 transitional//en">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>
<body>
<b><font face="Verdana"><font size=+1>Form Validation Example: Feedback
Form</font></font></b>
<br><script LANGUAGE="JavaScript">
<!--
function ValidateForm(form)
{
ErrorText= "";
if (form.username.value == "")
{
ErrorText= "\nPlease enter your name"
}
if ((form.email.value == "" || form.email.value.indexOf('@', 0) == -1) ||
form.email.value.indexOf('.') == -1)
{
ErrorText+= "\nPlease enter a proper value for your email"
}
if (form.feedback.value == "")
{
ErrorText+= "\nPlease enter your feedback"
}
if (ErrorText!= "")
{
alert("Error :" + ErrorText);
return false;
}
if (ErrorText= "")
{
form.submit()
}
ErrorText= "";
if ( ( form.gender[0].checked == false ) && ( form.gender[1].checked == false ) )
{
alert ( "Please choose your Gender: Male or Female" );
return false;
}
if (ErrorText= "")
{
form.submit()
}
ErrorText= "";
if ( form.age.selectedIndex == 0 )
{
alert ( "Please select your Age." );
return false;
}
if (ErrorText= "")
{
form.submit()
}
alert ( "Thank u" )
}
-->
</script>
<br><form name="feedback" action="mailto:webgimmicks@email.com" method=post>
<dl>
```

```
<dt>
<font face="Verdana">
<pre>
Please enter the following(To test form validation, skip entering one of the fields and click
'Submit' button):
</pre>
</font>
<font face="Verdana">Name: &nbsp;</font><input type="text" value="" name="username"
size=30></dt><br>
<br><dt> <font face="Verdana">E-mail: &nbsp;</font><input type="text" value=""
name="email" size=30></dt><br>
<br><dt><font face="Verdana">Your Gender:</font>
<input type="radio" name="gender" value="Male"> Male
<input type="radio" name="gender" value="Female"> Female </dt><dt> </dt> <br>
<font face="Verdana, Arial, Helvetica, sans-serif">Your Age</font>:
<select name="age"> <option value="">Please Select an Option:</option>
<option value="0-18 years">0-18 years</option>
<option value="18-30 years">18-30 years</option>
<option value="30-45 years">30-45 years</option>
<option value="45-60 years">45-60 years</option>
<option value="60+ years">60+ years</option> </select>
<br>
<br><font face="Verdana">Feedback:</font>
<dt><textarea rows=7 cols=60 name="feedback"></textarea></dt>
</dl>
<input type="button" name="SubmitButton" value="Submit"
onClick="ValidateForm(this.form)">
<input type="reset" value="Reset">
</form>
<pre><font face="Verdana"><font size=+0>&nbsp; &nbsp;</font></font>
</pre>
</form>
</body>
</html>
```

**Output:**


C:\Users\student\Documents\validation of form.htm - Windows Internet Explorer

To help protect your security, Internet Explorer has restricted this webpage from running scripts or ActiveX controls that could access your computer. Click here for options...

**Form Validation Example: Feedback Form**

Please enter the following(To test form validation, skip entering one of the fields and cl

Name: Maragatham

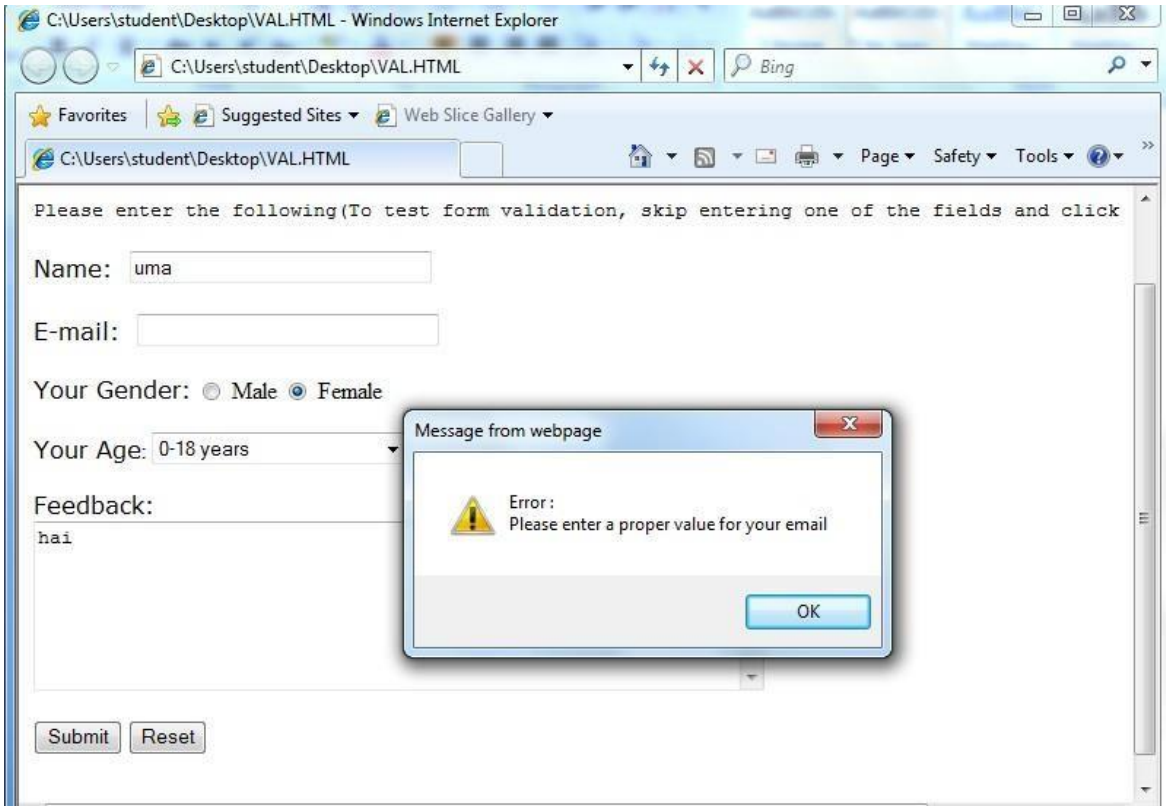
E-mail: terry.maggi93@gmail.com

Your Gender: Male [✓ Female]

Your Age: 18-30 years

Feedback:
Good

Submit Reset



C:\Users\student\Desktop\VAL.HTML - Windows Internet Explorer

```
Please enter the following(To test form validation, skip entering one of the fields and click
```

Name: uma

E-mail:

Your Gender: ○ Male ● Female

Your Age: 0-18 years

Feedback:

```
hai
```

Message from webpage

Error :
Please enter a proper value for your email

**Result**

## EX.NO 4 INSTALLING APACHE TOMCAT WEBSERVER

**Aim:**

To study and practice about installation of Apache Tomcat web server.

**Apache Tomcat HTTP Server**

Apache Tomcat is a Java-capable HTTP server, which could execute special Java programs known as "Java Servlet" and "Java Server Pages (JSP)". Tomcat is an *open-source* project, under the "Apache Software Foundation" (which also provides the most use, open-source, industrial-strength Apache HTTP Server). The mother site for Tomcat is http://tomcat.apache.org. Alternatively, you can find tomcat via the Apache mother site @ http://www.apache.org.

**The various Tomcat releases are:**

Tomcat 3.0 (1999): Reference Implementation (RI) for Servlet 2.2 and JSP 1.1.
Tomcat 4.1 (Sep 2002): RI for Servlet 2.3 and JSP 1.2.
Tomcat 5.0 (Dec 2003): RI for Servlet 2.4 and JSP 2.0.
Tomcat 6.0 (Feb 2007): RI for Servlet 2.5 and JSP 2.1.
Tomcat 7.0 (Jan 2011): RI for Servlet 3.0, JSP 2.2 and EL 2.2.
Tomcat 8.0 (Jun 2014): RI for Servlet 3.1, JSP 2.3, EL 3.0 and WebSocket 1.0. Tomcat 8.5 (June 2016) supports HTTP/2, OpenSSL, TLS virtual hosting and JASPIC 1.1.
Tomcat 9.0 (Jan 2018): RI for Servlet 4.0, JSP 2.3, EL 3.0, WebSocket 1.0, JASPIC 1.1.
Tomcat 10.0 (???):

**Steps to Install Tomcat**

**STEP 0: Create a Directory to Keep all your Works**

Create a directory called "c:\myWebProject" to keep all your work.

**STEP 1: Download and Install Tomcat For Windows**

Goto http://tomcat.apache.org ⇒ Under "Tomcat 9.0.{xx} Released", where {xx} is the latest update number ⇒ Click "Download" ⇒ Under "9.0.{xx}" ⇒ Binary Distributions ⇒ Core ⇒ "zip" (e.g., "apache-tomcat-9.0.{xx}.zip", about 11 MB).
UNZIP (right-click ⇒ Extract All) the downloaded file into your project directory "c:\myWebProject". Tomcat shall be unzipped into directory "c:\myWebProject\apache-tomcat- 9.0.{xx}".

For EASE OF USE, we shall shorten and rename this directory to "c:\myWebProject\tomcat". Take note of Your Tomcat Installed Directory. Hereafter, I shall refer to the Tomcat installed directory as <TOMCAT_HOME>.

**STEP 2: Create an Environment Variable JAVA_HOME (For Windows)**

You need to create an *environment variable* (system variable available to all applications) called "JAVA_HOME", and set it to your JDK installed directory.
STEP 3: Configure the Tomcat Server
The Tomcat configuration files, in XML format, are located in the "conf" sub-directory of your Tomcat installed directory, e.g. "c:\myWebProject\tomcat\conf" (for Windows) or "~/myWebProject/tomcat/conf" (for macOS). The important configuration files are:
server.xml web.xml context.xml
Make a BACKUP of the configuration files before you proceed!!!

**Step 3(a) "conf\server.xml" - Set the TCP Port Number**

Use a programming text editor (e.g., Sublime Text, Atom) to open the configuration file "server.xml".The default TCP port number configured in Tomcat is 8080, you may choose any number between 1024 and 65535, which is not used by existing applications. We shall choose 9999 in this article. (For production server, you should use port 80, which is pre-assigned to HTTP server as the default port number.)
Locate the following lines (around Line 69) that define the HTTP connector, and change port="8080" to port="9999".

```
<!-- A "Connector" represents an endpoint by which requests are received and responses are returned. Documentation at :
Java HTTP Connector: /docs/config/http.html Java AJP Connector: /docs/config/ajp.html APR (HTTP/AJP) Connector: /docs/apr.html
Define a non-SSL HTTP/1.1 Connector on port 8080
-->
<Connector port="9999" protocol="HTTP/1.1" connectionTimeout="20000" redirectPort="8443" />
```

**Step 3(b) "conf\web.xml" - Enable Directory Listing**

Again, use a programming text editor to open the configuration file "web.xml".We shall enable directory listing by changing "listings" from "false" to "true" for the "default" servlet. This is handy for test system, but not for production system for security.Locate the following lines (around Line
122) that define the "default" servlet; and change the "listings" from "false" to "true".

```
<servlet>
<servlet-name>default</servlet-name>
<servlet-class>org.apache.catalina.servlets.DefaultServlet</servlet-class>
<init-param>
<param-name>debug</param-name>
<param-value>0</param-value>
</init-param>
<init-param>
<param-name>listings</param-name>
<param-value>true</param-value>
</init-param>
<load-on-startup>1</load-on-startup>
</servlet>
```

Step 3(c) "conf\context.xml" - Enabling Automatic Reload
We shall add the attribute reloadable="true" to the <Context> element to enable automatic reload after code changes. Again, this is handy for test system but not recommended for production, due to the overhead of detecting changes.
Locate the <Context> start element (around Line 19), and change it to <Context reloadable="true">.

```
<Context reloadable="true">
......
......
</Context>
```

**STEP 4: Start Tomcat Server**
The Tomcat's executable programs and scripts are kept in the "bin" sub-directory of the Tomcat installed directory.
Step 4(a) Start Server

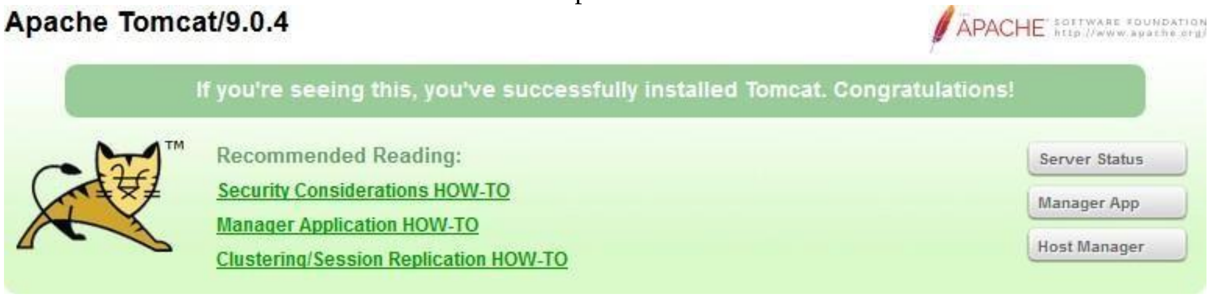
For Windows

I shall assume that Tomcat is installed in "c:\myWebProject\tomcat". Launch a CMD shell and issue:

```
c:          // Change drive
cd \myWebProject\tomcat\bin // Change directory to your Tomcat's binary directory
startup // Run startup.bat to start tomcat server ".
```

Step 4(b) Start a Client to Access the Server

Start a browser (Firefox, Chrome) as an HTTP client. Issue URL"http://localhost:9999" to access the Tomcat server's welcome page. The hostname "localhost" (with IP address of 127.0.0.1) is meant for local loop-back testing within the same machine. For users on the other machines over the net, they have to use the server's IP address or DNS domain name in the form of "http://*serverHostnameOrIPAddress*:9999".


Apache Tomcat/9.0.4
APACHE SOFTWARE FOUNDATION http://www.apache.org/
If you're seeing this, you've successfully installed Tomcat. Congratulations!
Recommended Reading:
Security Considerations HOW-TO
Manager Application HOW-TO
Clustering/Session Replication HOW-TO
Server Status
Manager App
Host Manager

(Optional) Try issuing URL http://localhost:9999/examples to view the servlet and JSP examples. Try running some of the servlet examples.

Step 4(c) Shutdown Server For Windows

You can shutdown the tomcat server by either:

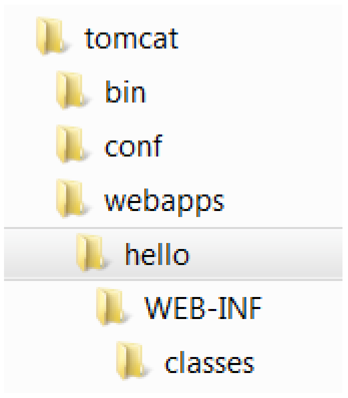
Press Ctrl-C on the Tomcat console; OR

Run "<TOMCAT_HOME>\bin\shutdown.bat" script. Open a new "cmd" and issue:

```
c:          // Change the current drive
cd \myWebProject\tomcat\bin // Change directory to your Tomcat's binary directory
shutdown          // Run shutdown.bat to shut down the server
```

**STEP 5: Develop and Deploy a WebApp**

**Step 5(a) Create the Directory Structure for your WebApp**



Let's call our first webapp "hello". Goto Tomcat's "webapps" sub-directory and create the following directory structure for your webapp "hello" (as illustrated). The directory names are case-sensitive!!

Under Tomcat's "webapps", create your webapp's *root* directory "hello" (i.e., <TOMCAT_HOME>\webapps\hello").

Under "hello", create a sub-directory "WEB-INF" (case sensitive, a "dash" not an underscore) (i.e.,

"<TOMCAT_HOME>\webapps\hello\WEB-INF").
Under "WEB-INF", create a sub-sub-directory "classes" (case sensitive, plural) (i.e., "<TOMCAT_HOME>\webapps\hello\WEB-INF\classes").
You need to keep your web resources (e.g., HTMLs, CSSs, images, scripts, servlets, JSPs) in the proper directories:
"hello": The is called the *context root* (or *document base directory*) of your webapp. You should keep all your HTML files and resources visible to the web users (e.g., HTMLs, CSSs, images, scripts, JSPs) under this *context root*.
"hello/WEB-INF": This directory, although under the context root, is *not visible* to the web users. This is where you keep your application's web descriptor file "web.xml".
"hello/WEB-INF/classes": This is where you keep all the Java classes such as servlet class-files.
You should RE-START your Tomcat server to pick up the hello webapp. Check the Tomcat's console to confirm that "hello" application has been properly deployed:

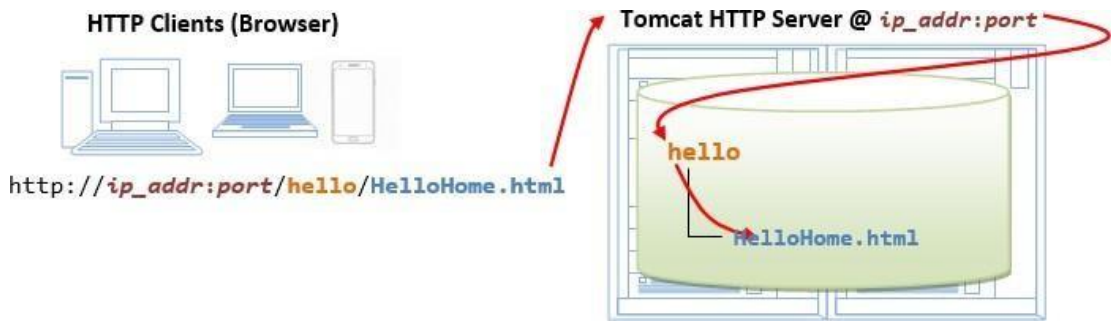
```
......
xxxxx INFO [main] org.apache.catalina.startup.HostConfig.deployDirectory Deploying web application directory [xxx\webapps\hello]
xxxxx INFO [main] org.apache.catalina.startup.HostConfig.deployDirectory Deployment of web application directory [xxx\webapps\hello] has finished in [38] ms
......
```

You can issue the following URL to access the web application "hello": http://localhost:9999/hello
You should see the directory listing of the directory "<TOMCAT_HOME>\webapps\hello", which shall be empty at this point of time. Take note that we have earlier enabled directory listing in "web.xml". Otherwise, you will get an error "404 Not Found".

**Step 5(b) Write a Welcome Page**
Create the following HTML page and save as "HelloHome.html" in your webapp's root directory "hello".

```
<!DOCTYPE html>
<html>
<head><title>My Home Page</title></head>
<body>
<h1>My Name is so and so. This is my HOME.</h1>
</body>
</html>
```

You can browse this page by issuing this URL: http://localhost:9999/hello/HelloHome.html


HTTP Clients (Browser)
http://ip_addr:port/hello/HelloHome.html
Tomcat HTTP Server @ ip_addr:port
hello
HelloHome.html

Alternatively, you can issue an URL to your webapp's root "hello": http://localhost:9999/hello
The server will return the directory listing of your base directory. You can then click on "HelloHome.html".
Rename "HelloHome.html" to "index.html", and issue a directory request again: http://localhost:9999/hello

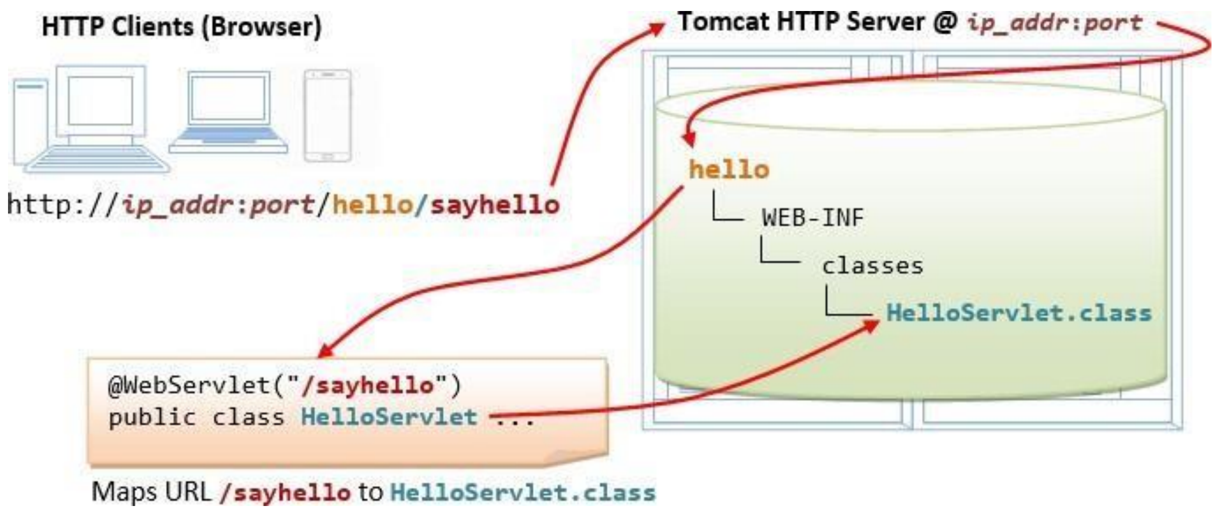
Now, the server will redirect the directory request to "index.html", if the root directory contains an "index.html", instead of serving the directory listing.
You can check out the home page of your peers by issuing:
http://*YourPeerHostnameOrIPAddr*:9999/hello http://*YourPeerHostnameOrIPAddr*:9999/hello/HelloHome.html with a valid "*YourPeerHostnameOrIPAddr*", provided that your peer has started his tomcat server and his firewall does not block your access. You can use command such as "ipconfig" (Windows), "ifconfig" (macOS and Unix) to find your IP address.
(Skip Unless...) The likely errors are "Unable to Connect", "Internet Explorer cannot display the web page", and "404 File Not Found". Read "How to Debug" section.

**STEP 6: Write a "Hello-world" Java Servlet**

A *servlet* is Java program that runs inside a Java-capable HTTP Server, such as Apache Tomcat. A web user invokes a servlet by issuing an appropriate URL from a web browser (HTTP client).
Before you proceed, I shall assume that you are familiar with Java Programming and have installed the followings:
JDK (Read "How to install JDK and Get Started").
A programming text editor, such as Sublime Text or Atom.

**Step 6(a) Write a "Hello-world" Java Servlet**

A Java servlet is a Java program that runs inside a HTTP server. A web user invokes a servlet by issuing a URL from a browser (or HTTP client).
In this example, we are going to write a Java servlet called HelloServlet, which says "Hello, world!". We will configure such that web users can invoke this servlet by issuing
URL http://*ip_addr*:*port*/hello/sayhello from their browser, as illustrated:


HTTP Clients (Browser)
http://ip_addr:port/hello/sayhello
Tomcat HTTP Server @ ip_addr:port
hello
WEB-INF
classes
HelloServlet.class
@WebServlet("/sayhello")
public class HelloServlet ...
Maps URL /sayhello to HelloServlet.class

Write the following source codes called "HelloServlet.java" and save it under your application "classes" directory (i.e., "<TOMCAT_HOME>\webapps\hello\WEB-INF\classes\HelloServlet.java"). This servlet says "Hello", echoes some request information, and prints a random number upon each request.

```
// To save as "<TOMCAT_HOME>\webapps\hello\WEB-INF\classes\HelloServlet.java" import java.io.*;
import javax.servlet.*; import javax.servlet.http.*;
import javax.servlet.annotation.*;


@WebServlet("/sayhello") // Configure the request URL for this servlet (Tomcat 7/Servlet 3.0 upwards)
public class HelloServlet extends HttpServlet {


// The doGet() runs once per HTTP GET request to this servlet. @Override
public void doGet(HttpServletRequest request, HttpServletResponse response) throws IOException,
ServletException {

// Set the response MIME type of the response message
response.setContentType("text/html");
// Allocate a output writer to write the response message into the network socket
PrintWriter out = response.getWriter();


// Write the response message, in an HTML page out.println("<!DOCTYPE html>"); out.println("<html>");
out.println("<head><title>Hello, World</title></head>");
out.println("<body>");
out.println("<h1>Hello, world!</h1>"); // says Hello
// Echo client's request information
out.println("<p>Request URI: " + request.getRequestURI() + "</p>"); out.println("<p>Protocol: " +
request.getProtocol() + "</p>"); out.println("<p>PathInfo: " + request.getPathInfo() + "</p>");
out.println("<p>Remote Address: " + request.getRemoteAddr() + "</p>");
// Generate a random number upon each request
out.println("<p>A Random Number: <strong>" + Math.random() + "</strong></p>");
out.println("</body></html>");
out.close(); // Always close the output writer
}
```

Take note that in Line 7, we configure this HelloServlet to URL "/sayhello" via annotation @WebServlet("/sayhello"), which is applicable to Tomcat 7 onwards. In other words, the full URL shall be http://*ip_addr*:*port*/hello/sayhello to trigger this HelloServlet.

Step 6(b) Compiling the Servlet (DIFFICULT)

We need the Java Servlet API to compile the servlet. Servlet API is NOT part of JDK. Tomcat provides a copy in <TOMCAT_HOME>/lib/servlet-api.jar. We need to include this JAR file in the compilation via the -cp (classpath) option as follows:

(For Windows)

```
// Assume that Tomcat is installed in c:\myWebProject\tomcat
// Change directory to the Java source directory
c:cd \myWebProject\tomcat\webapps\hello\WEB-INF\classes
```

```
// Compile
javac -cp.; c:\myWebProject\tomcat\lib\servlet-api.jar HelloServlet.java
```

**Step 6(c) Invoke the Servlet**

Restart your Tomcat Server (just in case ...).
To invoke this servlet, start a browser, and issue the request URL configured as follows:
http://localhost:9999/hello/sayhello
You shall see the output of the servlet displayed in your web browser.
Refresh the browser, you shall see a new random number upon each refresh. In other word, the doGet() method of the servlet runs once per request.

**Result**

## EX.NO.5a INVOKING SERVLET FROM HTML

**Aim:**

Design a HTML page to invoke servlet.

**Procedure:**

Step1: Open Net Beans IDE, in that select new project->web application.

Step2-: Accept all the default options and click finish.

Step3: Right click on the project name and open new html file and named it newhtml.

Step4: In the newhtml file add the tags to get the details of the user.

Step5: Right click on the project name and open new servlet file and design the servlet page.

Step6: call the servlet in the html file using the following tag.

```
<form method="get" action="NewServlet">
```

Step6: Call that newhtml file in the default jsp page index.jsp with the help of form tag with get or post method.

**index.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 method ="get" action="newhtml.html">

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

**Newhtml.html**

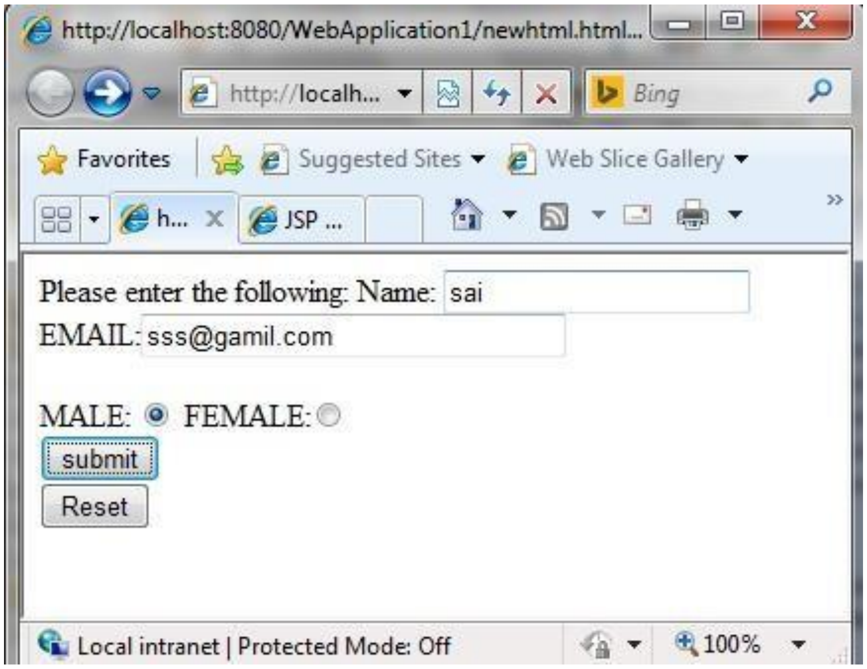
```
<html>
   <head>
      <title></title>
```

```
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    </head>
    <body>
        <form method="get" action="NewServlet">
          Please enter the following:
          Name: <input type="text" name="t1">
          <br>
           EMAIL:<input type="text" value="" name="email" size=30>
           </br>
           <br>
      MALE: <input type="radio" name="gender" value="Male">

          FEMALE:<input type="radio" name="gender" value="Female">
          <br> <input type="submit" value="submit"></br>
         <input type="reset" value="Reset">


        </form>

    </body>
</html>
```

**NewServlet.java**

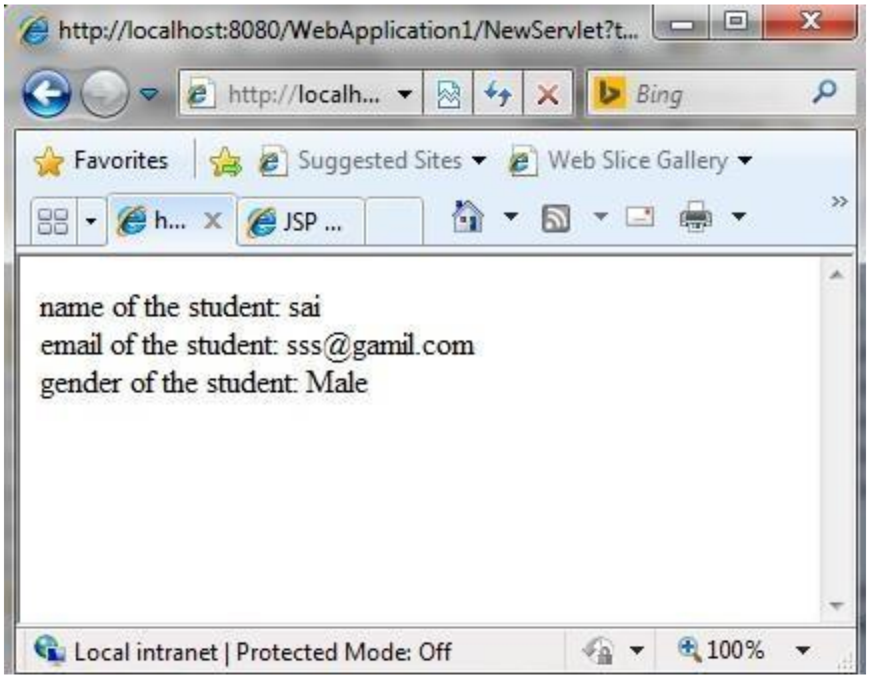
```
package p;

import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;


@WebServlet(name = "NewServlet", urlPatterns = {"/NewServlet"})
public class NewServlet extends HttpServlet {

    protected void processRequest(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        try {

            String name=request.getParameter("t1");
            String gender=request.getParameter("gender");


            String email=request.getParameter("email");
            out.println("name of the student:");
            out.println(name+"<br/>");
            out.println("email of the student:");

            out.println(email+"<br/>");
            out.println("gender of the student:");
            out.println(gender+"<br/>");


        } finally {
            out.close();
        }
    }
```

**OUTPUT:-**


http://localhost:8080/WebApplication1/newhtml.html...
http://localh...
Bing
Favorites
Suggested Sites
Web Slice Gallery
h...
JSP ...
Local intranet | Protected Mode: Off
100%

Please enter the following: Name: sai
EMAIL: sss@gamil.com

MALE: [✓] FEMALE:
submit
Reset


http://localhost:8080/WebApplication1/NewServlet?t...
http://localh...
Bing
Favorites
Suggested Sites
Web Slice Gallery
h...
JSP ...
Local intranet | Protected Mode: Off
100%

```
name of the student: sai
email of the student: sss@gamil.com
gender of the student: Male
```

**Result:**

## EX.NO.5b SESSION TRACKING USING SERVLET

**Aim:**

Servlet Program to get Session Details.

**Procedure:**

Step1: Open Net Beans IDE, in that select new project->web application.

Step2-: Accept all the default options and click finish.

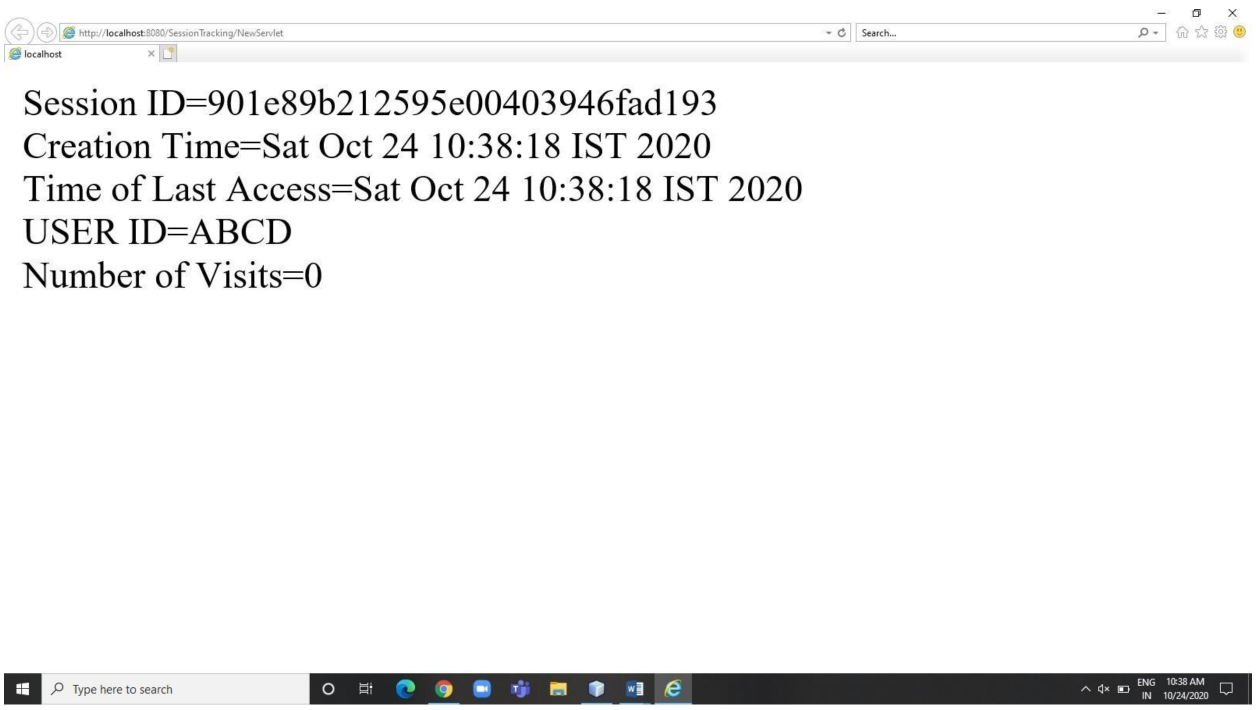
Step3: Right click on the project name and open new servlet file and design the servlet page.

**Program:**

```
import java.io.IOException;
import java.io.PrintWriter;
import static java.lang.Character.UnicodeBlock.of;
import java.util.Date;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

/**
 *
 * @author HAI
 */
@WebServlet(urlPatterns = {"/NewServlet"})
public class NewServlet extends HttpServlet {

    /**
 * Processes requests for both HTTP <code>GET</code> and <code>POST</code>
 * methods.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
@Override
public void doGet(HttpServletRequest request, HttpServletResponse response)
  throws ServletException, IOException {

  // Create a session object if it is already not created.
  HttpSession session = request.getSession(true);

  // Get session creation time.
  Date createTime = new Date(session.getCreationTime());

  // Get last access time of this web page.
  Date lastAccessTime = new Date(session.getLastAccessedTime());

  String title = "Welcome Back to my website";
  Integer visitCount;
    visitCount = 0;
  String visitCountKey = "visitCount";
  String userIDKey = "userID";
  String userID = "ABCD";
```

```
      // Check if this is new comer on your web page.
      if (session.isNew()) {
         title = "Welcome to my website";
         session.setAttribute(userIDKey, userID);
      } else {
         visitCount = (Integer)session.getAttribute(visitCountKey);
         visitCount = visitCount + 1;
         userID = (String)session.getAttribute(userIDKey);
      }
      session.setAttribute(visitCountKey,  visitCount);


      // Set response content type
      response.setContentType("text/html");
      PrintWriter out = response.getWriter();
  out.println("Session ID="+session.getId() + "</br>"+"Creation Time="+createTime +
"</br>"+"Time of Last Access="+lastAccessTime + "</br>"+ "USER ID="+userID +"</br>"+
"Number of Visits="+visitCount);
   }
}
```

**Output**


http://localhost:8080/SessionTracking/NewServlet
Search...
localhost
Type here to search
ENG IN 10:38 AM 10/24/2020

```
Session ID=901e89b212595e00403946fad193
Creation Time=Sat Oct 24 10:38:18 IST 2020
Time of Last Access=Sat Oct 24 10:38:18 IST 2020
USER ID=ABCD
Number of Visits=0
```

**Result**

**EX.NO.6a** **ONLINE EXAMINATION USING JSP AND DATABASES**

**Aim:**

Design a jsp page to conduct a online examination.

**Algorithm:**

Step1: Open Net Beans IDE, in that select new project->web application.

Step2: open a default jsp page add the necessary tags to design a online exam page .

Step3:Right click on the project name and select New Servlet,and compare the answers here.

Step4:Call the servlet in the jsp using form tag.

<form method="get" action="NewServlet">

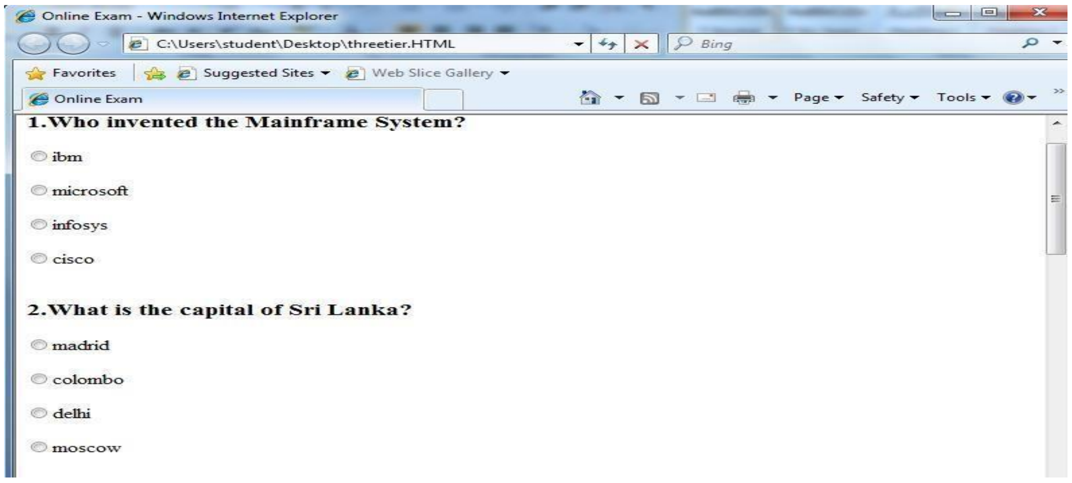
```
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>Online Exam</title>
    </head>
    <body>
        <form method="get" action="three">
            method="get">
            <h3>1.Who invented the Mainframe System?</h3>
            <input type="radio" name="s1" value="IBM">ibm</input><br></br>
             <input type="radio" name="s1" value="MICROSOFT">microsoft</input><br></br>
              <input type="radio" name="s1" value="INFOSYS">infosys</input><br></br>
               <input type="radio" name="s1" value="CISCO">cisco</input><br></br>
               <h3>2.What is the capital of Sri Lanka?</h3>
                 <input type="radio" name="s2" value="MADRID">madrid</input><br></br>
        <input type="radio" name="s2" value="COLOMBO">colombo</input><br></br>
         <input type="radio" name="s2" value="DELHI">delhi</input><br></br>
         <input type="radio" name="s2" value="MOSCOW">moscow</input><br></br>
        <h3>3.What is the National Game of INDIA?</h3>
         <input type="radio" name="s3" value="HOCKEY">hockey</input><br></br>
         <input type="radio" name="s3" value="CRICKET">cricket</input><br></br>
<input type="radio" name="s3" value="FOOTBALL">football</input><br></br>
<input type="radio" name="s3" value="VOLLEY BALL">volleyball</input><br></br>
<h3>4.What is the NATO name of SUKHOI jets?</h3>
<input type="radio" name="s4" value="FLANKERS">flankers</input><br></br>
<input type="radio" name="s4" value="FOXBOAT">foxboat</input><br></br>
      <input type="radio" name="s4" value="FISHBOAT">fishboat</input><br></br>
      <input type="radio" name="s4" value="FOXROT">foxrot</input><br></br>
      <h3>5.What wasthe first name of Graphical browser?</h3>
      <input type="radio" name="s5" value="NETSCAPE">netscape</input><br></br>
      <input type="radio" name="s5" value="IE">ie</input><br></br>
      <input type="radio" name="s5" value="MOZILLA">mozilla</input><br></br>
      <input type="radio" name="s5" value="OPERA">opera</input><br></br>
      <input type="submit" value="submit"></input></form>

   </body>
</html>
```

Servlet code:

```
package tier;
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;


@WebServlet(name = "three", urlPatterns = {"/three"})
public class three extends HttpServlet {


   protected void doGet(HttpServletRequest request, HttpServletResponse response)
         throws ServletException, IOException {
      response.setContentType("text/html;charset=UTF-8");
      PrintWriter out = response.getWriter();
      int score=0;

      try {
          String str=request.getParameter("s1");
          if(str.equals("IBM"))
             score++;
          str=request.getParameter("s2");
```

```
            if(str.equals("COLOMBO"))
                score++;
            str=request.getParameter("s3");
            if(str.equals("HOCKEY"))
                score++;
            str=request.getParameter("s4");
            if(str.equals("FLANKERS"))
                score++;
            str=request.getParameter("s5");
            if(str.equals("MOZILLA"))
                score++;

            out.println("<html>");
            out.println("<head>");
            out.println("<title>Three tier</title>");
            out.println("</head>");
            out.println("<body> <p> THE TOTAL SCORE IS:</p>");
            out.println(Integer.toString(score));
            out.println("</body>");
            out.println("</html>");

        } finally {
            out.close();
        }
    }
}
```

**Output:**


Online Exam - Windows Internet Explorer
C:\Users\student\Desktop\threetier.HTML
Bing
Favorites
Suggested Sites
Web Slice Gallery
Online Exam
Page
Safety
Tools

**1.Who invented the Mainframe System?**

◯ ibm

◯ microsoft

◯ infosys

◯ cisco

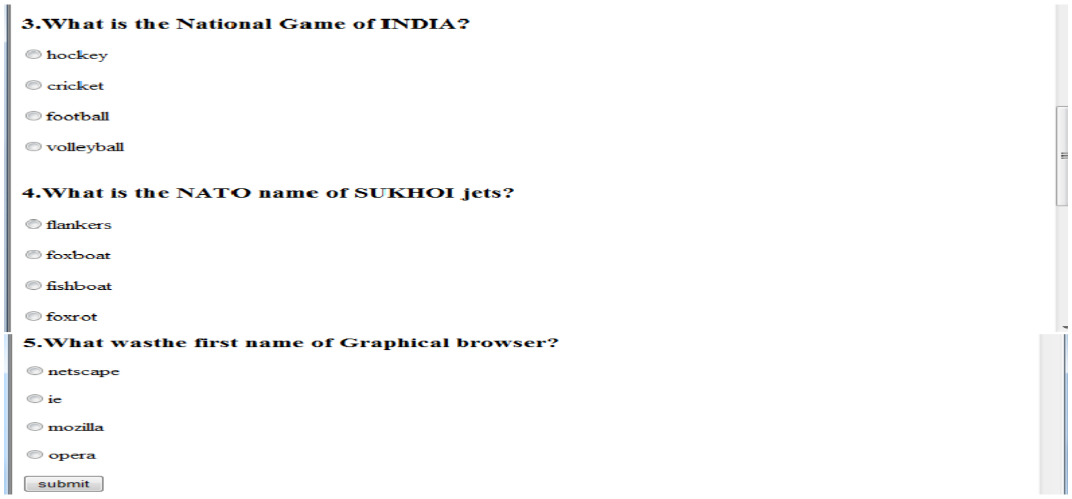
**2.What is the capital of Sri Lanka?**

◯ madrid

◯ colombo

◯ delhi

◯ moscow



**3.What is the National Game of INDIA?**

◯ hockey

◯ cricket

◯ football

◯ volleyball

**4.What is the NATO name of SUKHOI jets?**

◯ flankers

◯ foxboat

◯ fishboat

◯ foxrot

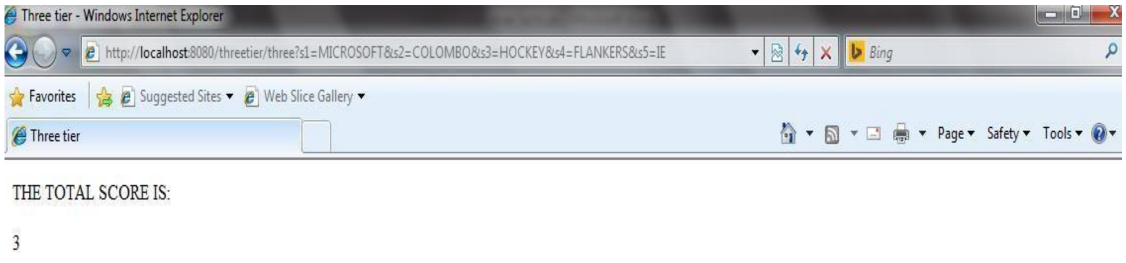
**5.What wasthe first name of Graphical browser?**

◯ netscape

◯ ie

◯ mozilla

◯ opera


Three tier - Windows Internet Explorer
http://localhost:8080/threetier/three?s1=MICROSOFT&s2=COLOMBO&s3=HOCKEY&s4=FLANKERS&s5=IE
Bing
Favorites
Suggested Sites
Web Slice Gallery
Three tier
Page
Safety
Tools

THE TOTAL SCORE IS:

3

**Result**

**EX.NO:6b DISPLAYING STUDENT MARK LIST**

Aim:

Develop a three tier application using JSP and JDBC.

**Algorithm:**

Step1: Open Net Beans IDE, in that select new project->web application.

Step2: open a default jsp page add the necessary tags to design a student information system .

Step3: Create a table in derby database by clicking services tab->click Java DB->Right Click on the Java DB-→Create Database→Give Database Name, user name and password.

Step4: Right click and connect the newly created Database connectivity.

Step5: Right click on the DB name and create new table then add data to the table.

Source code:

Index.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
      </head>
   <body>
      <h5>STUDENT INFORMATION SYSTEM</h5>
       <form action="newjsp.jsp" method="get">
         <input type="text" name="t1"></input>
         <input type="submit" value="submit"></input>
      </form>
      </body>
</html>
```

**Newjsp.jsp**

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@page import="java.sql.*;" %>
<!DOCTYPE html>
<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
      <title>Esp Page</title>
   </head>
```

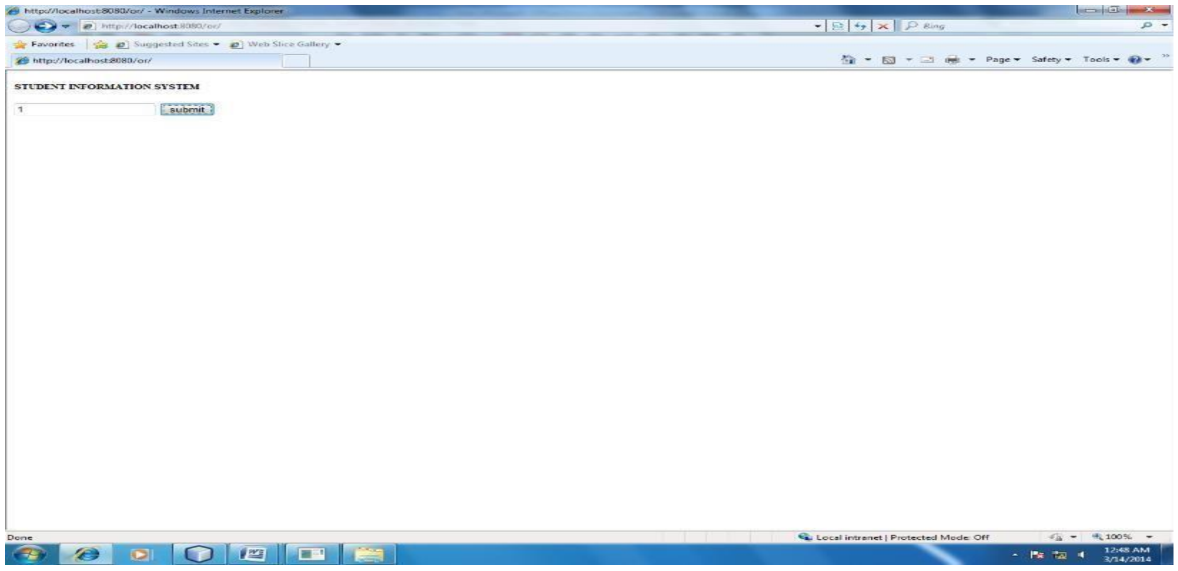
```
<body>
   <%!String name;%>
   <%!    int id,d,m;%>
      <table border="2px" align="center">
           <tr>
             <th> rollno</th>
             <th> mark1 </th>
             <th> mark2 </th>
         </tr>
   <%
          try
   {
               id=Integer.parseInt(request.getParameter("t1"));
               //String name=request.getParameter("t1");
               out.println(id);
               Class.forName("org.apache.derby.jdbc.ClientDriver");
               Connection
               con=DriverManager.getConnection("jdbc:derby://l
               ocalhost:1527/SDB","sairam","sairam");out.println
               ("connected");
                Statement stmt=con.createStatement();
                 out.println("hi");
                ResultSet rs= stmt.executeQuery("select * from student.m3 where rollno="+id);
                out.println("hi");
                while(rs.next())
                {
                 id=rs.getInt("rollno");
                 d=rs.getInt("mark1");
                 m=rs.getInt("mark2");
         %>
        <tr>
                    <td><%=id%></td>
                    <td><%=d%></td>
                    <td><%=m%></td>

               </tr>

               <%
                }
     }
        catch(Exception e)
        {}

%>
```

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

Student Table

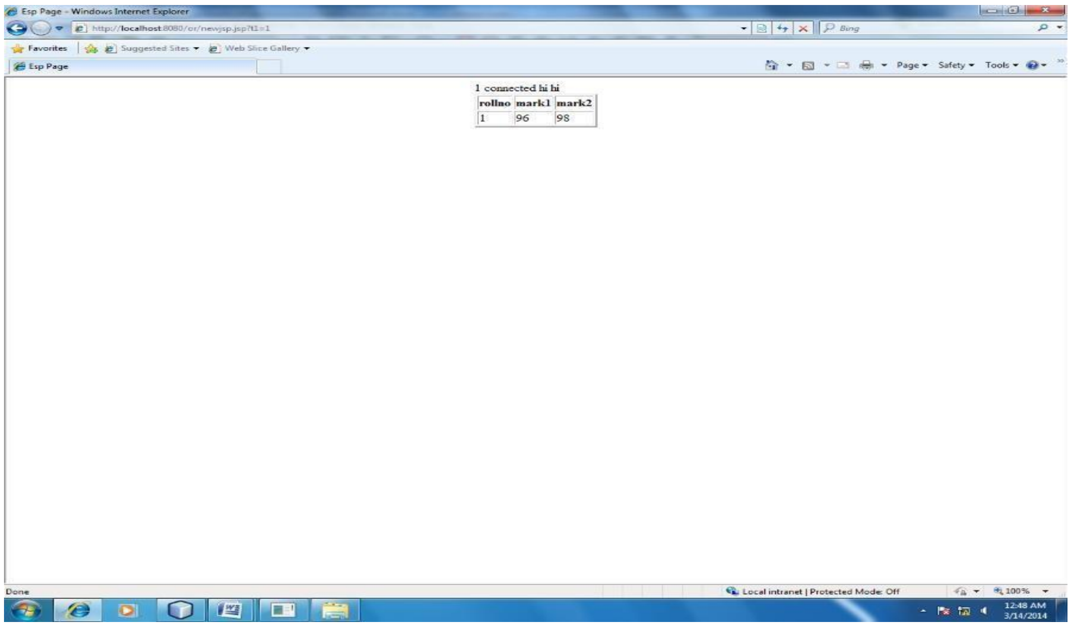
```
    ROLLNO      MARK1       MARK2
---------------------------
         1         96         98
         2         66         88
         3         75         83
```

Output:


http://localhost:8080/or/ - Windows Internet Explorer
http://localhost:8080/or/
Favorites
Suggested Sites
Web Slice Gallery
Page
Safety
Tools
submit
Done
Local intranet | Protected Mode: Off
100%
12:48 AM
3/14/2014

STUDENT INFORMATION SYSTEM

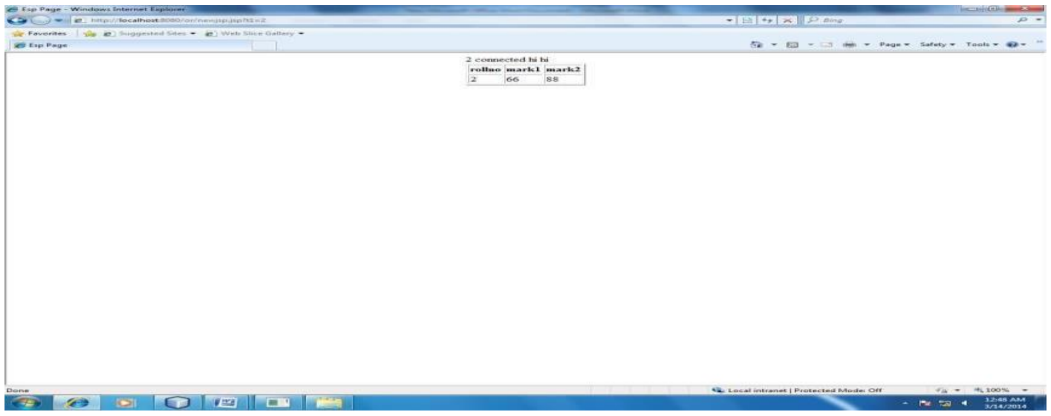
1



http://localhost:8080/or/newjsp.jsp?t1=1

```
1 connected hi hi
```

| rollno | mark1 | mark2 |
|---|---|---|
| 1 | 96 | 98 |



http://localhost:8080/or/newjsp.jsp?t1=2

```
2 connected hi hi
```

| rollno | mark1 | mark2 |
|---|---|---|
| 2 | 66 | 88 |

**Result**

**EX.NO.7** **PROGRAM USING XML WITH XSL**

**Aim:**

Extract the data from XML using XSLT and print the content as HTML Table

**Procedure:**

Step1: Develop a xml file.
Step2: Type the xml tags in notepad and save with extension .xml.
Step3: develop a XSL page with extension .xsl.
Step4: include the xsl in xml using href attribute.
Step5:

**XML:-**

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<?xml-stylesheet type="text/xsl" href="myxs.xsl"?>
<BIKEMODEL>
<CD>
<COMPANY>Honda</COMPANY>
<MODEL>Activa</MODEL>
<COLOR>Red</COLOR>
<YEAR>2003</YEAR>
<PRICE>54k</PRICE>
<ENGINE>Two Stroke</ENGINE>
<MILEAGE>35Kmph</MILEAGE>
</CD>
<CD>
<COMPANY>Bajaj</COMPANY>
<MODEL>Platina</MODEL>
<COLOR>Silver</COLOR>
<YEAR>2008</YEAR>
<PRICE>45k</PRICE>
<ENGINE>Two Stroke</ENGINE>
<MILEAGE>102Kmph</MILEAGE>
</CD>
<CD>
<COMPANY>TVS</COMPANY>
<MODEL>TVS-50</MODEL>
<COLOR>Blue</COLOR>
<YEAR>1993</YEAR>
<PRICE>35k</PRICE>
```

```
<ENGINE>Two Stroke</ENGINE>
<MILEAGE>34Kmph</MILEAGE>
</CD>
<CD>
<COMPANY>Yamaha</COMPANY>
<MODEL>yamaha</MODEL>
<COLOR>Black</COLOR>
<YEAR>1999</YEAR>
<PRICE>39k</PRICE>
<ENGINE>Two Stroke</ENGINE>
<MILEAGE>78Kmph</MILEAGE>
</CD>
<CD>
<COMPANY>Scooty</COMPANY>
<MODEL>Pep+</MODEL>
<COLOR>violet</COLOR>
<YEAR>2005</YEAR>
<PRICE>28k</PRICE>
<ENGINE>Two Stroke</ENGINE>
<MILEAGE>67Kmph</MILEAGE>
</CD>
</BIKEMODEL>
```

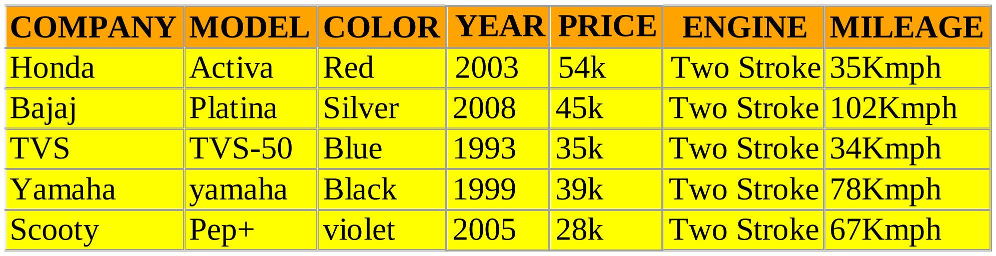
**XSL:-**

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
<html>
<body>
<h2> Motor Bike</h2>
<table border="1">
<tr bgcolor="orange">
<th>COMPANY</th>
<th>MODEL</th>
<th>COLOR</th>
<th>YEAR</th>
<th>PRICE</th>
<th>ENGINE</th>
<th>MILEAGE</th>
</tr>
<xsl:for-each select="BIKEMODEL/CD">
<tr bgcolor="yellow">
<td><xsl:value-of select="COMPANY"/></td>
<td><xsl:value-of select="MODEL"/></td>
<td><xsl:value-of select="COLOR"/></td>
```

```
<td><xsl:value-of select="YEAR"/></td>
<td><xsl:value-of select="PRICE"/></td>
<td><xsl:value-of select="ENGINE"/></td>
<td><xsl:value-of select="MILEAGE"/></td>
</tr>
</xsl:for-each>
</table>
</body>
</html>
</xsl:template>
</xsl:stylesheet>
```

**Output:**

**Motor Bike**



| COMPANY | MODEL | COLOR | YEAR | PRICE | ENGINE | MILEAGE |
|---|---|---|---|---|---|---|
| Honda | Activa | Red | 2003 | 54k | Two Stroke | 35Kmph |
| Bajaj | Platina | Silver | 2008 | 45k | Two Stroke | 102Kmph |
| TVS | TVS-50 | Blue | 1993 | 35k | Two Stroke | 34Kmph |
| Yamaha | yamaha | Black | 1999 | 39k | Two Stroke | 78Kmph |
| Scooty | Pep+ | violet | 2005 | 28k | Two Stroke | 67Kmph |

**Result**

## EX.NO.8 PROGRAM USING DOM/SAX PARSERS TO PARSE A XML DOCUMENT

### Aim:

To write a program that parses an XML document using DOM and SAX parsers.

**ALGORITHM:**

Step1: Create a .xml file that has to be parsed.

**Using DOM parser**

Step1:Get a document builder using document builder factory and parse the xml file to create a DOM object

Step2:Get a list of employee elements from the DOM For each employee element get the id, name, age and type.

Step 3:Create an employee value object and add it to the list.

Step4:At the end iterate through the list and print the employees to verify we parsed it right.

3. **Using SAX Parser**

Step1: Create a Sax parser and parse the xml

Step2: In the event handler create the employee object Step3:

Print out the data

**Source code:**
**Dom.java**

```
import java.io.*;
import javax.xml.parsers.*;
import org.w3c.dom.*; import
org.xml.sax.*; public class dom
{
public static void main(String []arg)
{
try
{
System.out.println("enter the name of xmldoc");
BufferedReader input=new BufferedReader(new InputStreamReader(System.in)); String
fname=input.readLine();
File fp=new File(fname);
if(fp.exists())
{
try
{
DocumentBuilderFactory fobj=DocumentBuilderFactory.newInstance();
DocumentBuilder builder=fobj.newDocumentBuilder();
InputSource ipsrc=new InputSource(fname);
Document doc=builder.parse(ipsrc);
System.out.println(fname+"is wellformed");
}
catch(Exception e)
{
System.out.println(fname+"is not allowed");
}}}
catch(Exception e)
{}
}
}
```

**XML Document**

**XMLDOM**

```
<student>
<p_info>
<name>Maragatham</name>
<dept>it</dept>
</p_info>
</student>
```

**OUTPUT:**

```
enter the name of xmldoc
XMLDOM.xml XMLDOM.xmlis
wellformed
BUILD SUCCESSFUL (total time: 7 seconds)
```

**SAX Parsing**

**Sax.java**

```
import java.io.*; import
org.xml.sax.*;
import org.xml.sax.helpers.*; public
class sax
{
static public void main(String []arg)
{
```

```
try
{
System.out.println("Enter the name of xml document");
BufferedReader input=new BufferedReader(new InputStreamReader(System.in)); String
fname=input.readLine();
File fp=new File(fname);
if(fp.exists())
{
try
{
XMLReader reader=XMLReaderFactory.createXMLReader();
reader.parse(fname);
System.out.println(fname+"is well formed");
}
catch(Exception e)
{
System.out.println(fname+"is not well formed");
}}}
catch(Exception e)
{
}
}}
```

**OUTPUT:**

```
Enter the name of xml document
XMLDOM.xml XMLDOM.xmlis
well formed
BUILD SUCCESSFUL (total time: 10 seconds)
```

**Result**

## EX.NO.9 PROGRAM USING AJAX

**AIM:**

To write a html program using AJAX.

**ALGORITHM:**

Step1: Start the program.
Step2: Create the function named load XMLDOC().
Step3: In that print the message as “AJAX changed the text here”.
Step4: By using the button, onclick function.
Step5: Call the defined function.
Step6: Stop the program.

**Newhtml.htm**

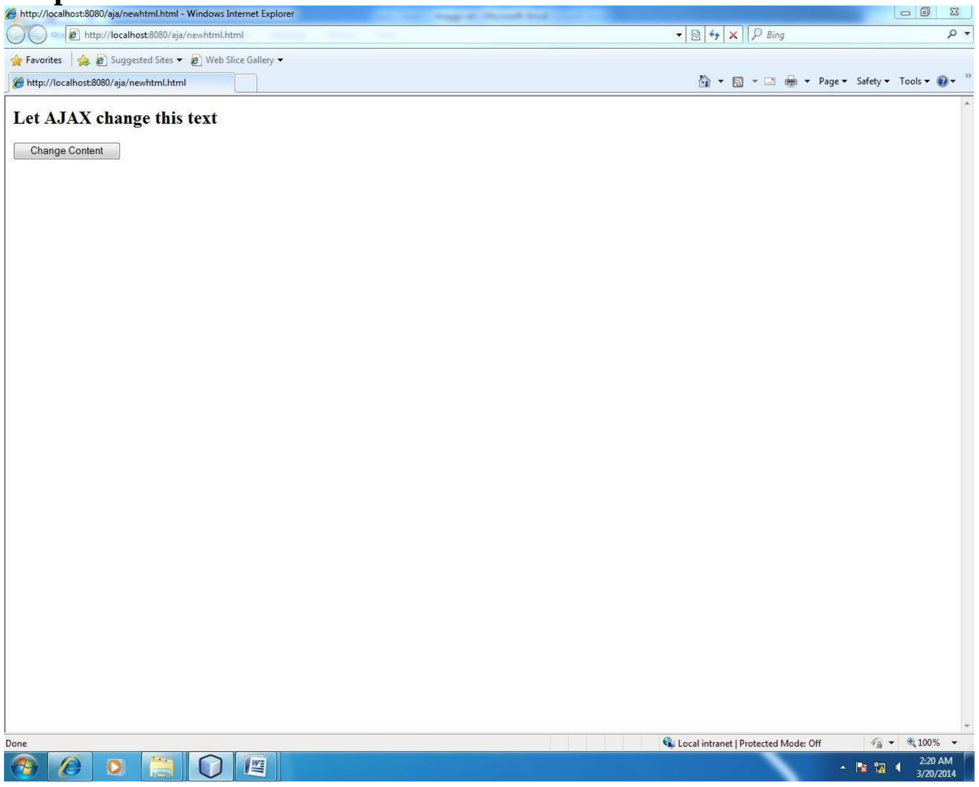
```
<html>
<head>
<script type="text/javascript">
function loadXMLDoc()
{
if (window.XMLHttpRequest)
{// code for IE7+, Firefox, Chrome, Opera, Safari xmlhttp=new
XMLHttpRequest();
}
else
{// code for IE6, IE5
xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
}
xmlhttp.onreadystatechange=function()
{
if (xmlhttp.readyState==4 && xmlhttp.status==200)
{
document.getElementById("myDiv").innerHTML=xmlhttp.responseText;
}
}
xmlhttp.open("GET","newhtml1.html",true);
xmlhttp.send();
}
</script>
</head>
```

```
<body>
<div id="myDiv"><h2>Let AJAX change this text</h2></div>
<button type="button" onclick="loadXMLDoc()">Change Content</button>
</body>
</html>
```

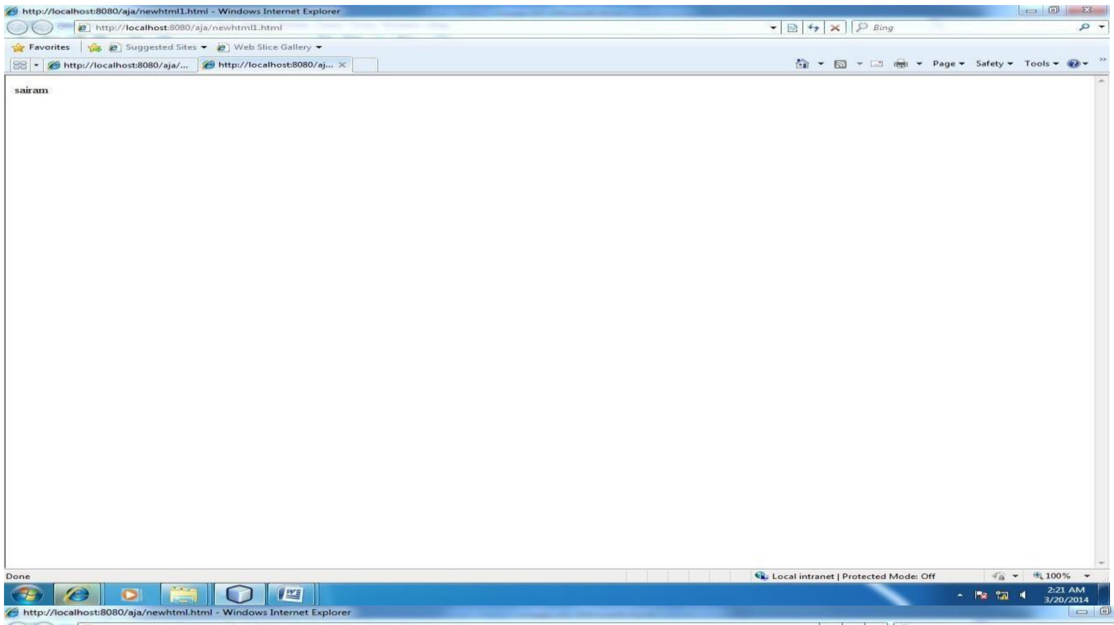
**Newhtml1.htm**

```
<html>
    <head>
        <title></title>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    </head>
    <body>
        <form method="get">
            sairam
        </form>
    </body>
</html>
```

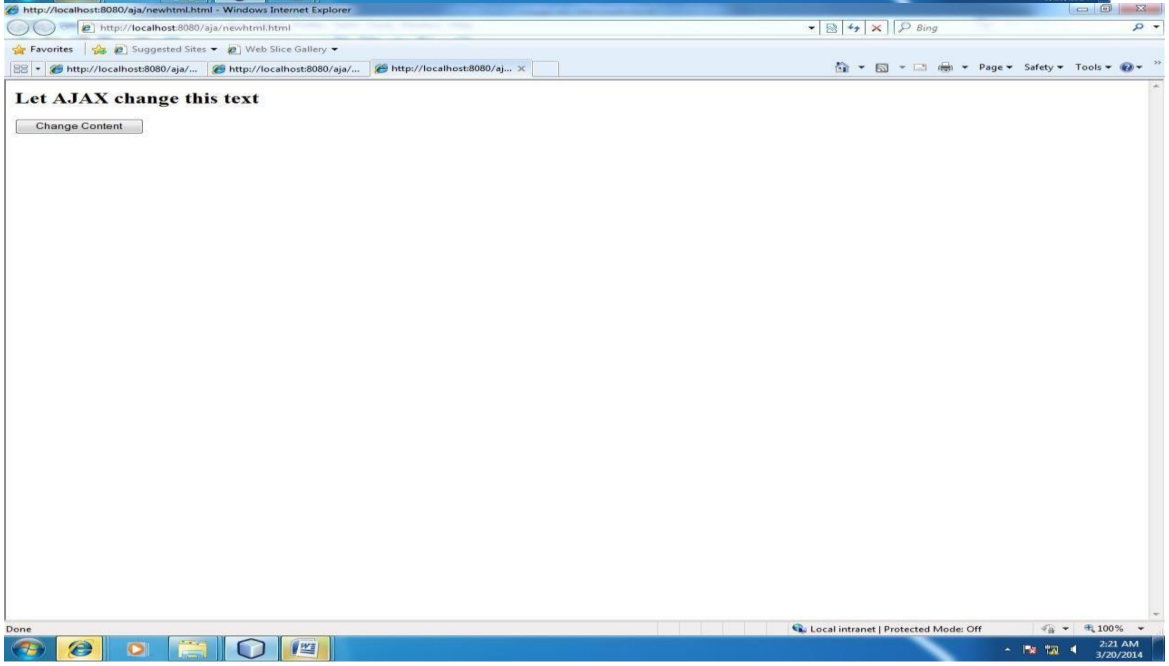
**Output:**


http://localhost:8080/aja/newhtml.html - Windows Internet Explorer
Favorites
Suggested Sites
Web Slice Gallery
Page
Safety
Tools
Change Content
Done
Local intranet | Protected Mode: Off
100%
2:20 AM
3/20/2014

Let AJAX change this text


http://localhost:8080/aja/newhtml1.html - Windows Internet Explorer

sairam


http://localhost:8080/aja/newhtml.html - Windows Internet Explorer

Let AJAX change this text

**Result:**

## EX No. 10.a WEB SERVICE CREATION FOR CALCULATOR APPLICATION

**Date:**

**AIM:**

To develop Calculator  web service.

**ALGORITHM:**

**Step 1:** Go to NetBeans IDE-> File -> New Project -> JavaWeb -> Web Application -> Next and name the projectas "Calculator" -> Next -> Finish.

**Step 2:** Create new web service

a. Right Click the created project name -> New -> Web Service -> Give name for the web service and package -> Finish.

b. A new folder named "web services" will be created under the project folder. Inside that, new web services will be created.

**Step 3:** Creating function for web service

Under the web services folder, right click the web service created -> Add operation.
*Give the appropriate name for the operation
*Select the return type of the function
*Adding parameters:
Click Add -> Give name of the parameter and their data type.
Click OK.

**Step 4:** Write the required operation inside the function and also see to the return type matches.

Write the required coding and save the file.
* Sample Code
@WebMethod(operationName="getBalance")
Public int getBalance(@WebParam(name= "ino") int ino)

**Step 5:** Right Click on the webservice -> Add Operation -> choose the operation name and return type.

**Step 6:** a. Right Click the project -> Deploy

c. In the web service folder, right click on the web service -> Test web service.

**Step 7:** Get into NetBeans IDE, go to File -> New Project -> Java -> Web Application

Give the name of the project as "clientcalc".

**Step 8:** Create a Home Page to receive sequence.

**Step 9:** Create component necessary to invoke the service from the client

a. Right click "clientcalc" -> new -> WebServiceClient

b. Click Browse and select the required service.

**Step 10:** Build and Run the Web Application.

Source code:

**Calcservice.java**

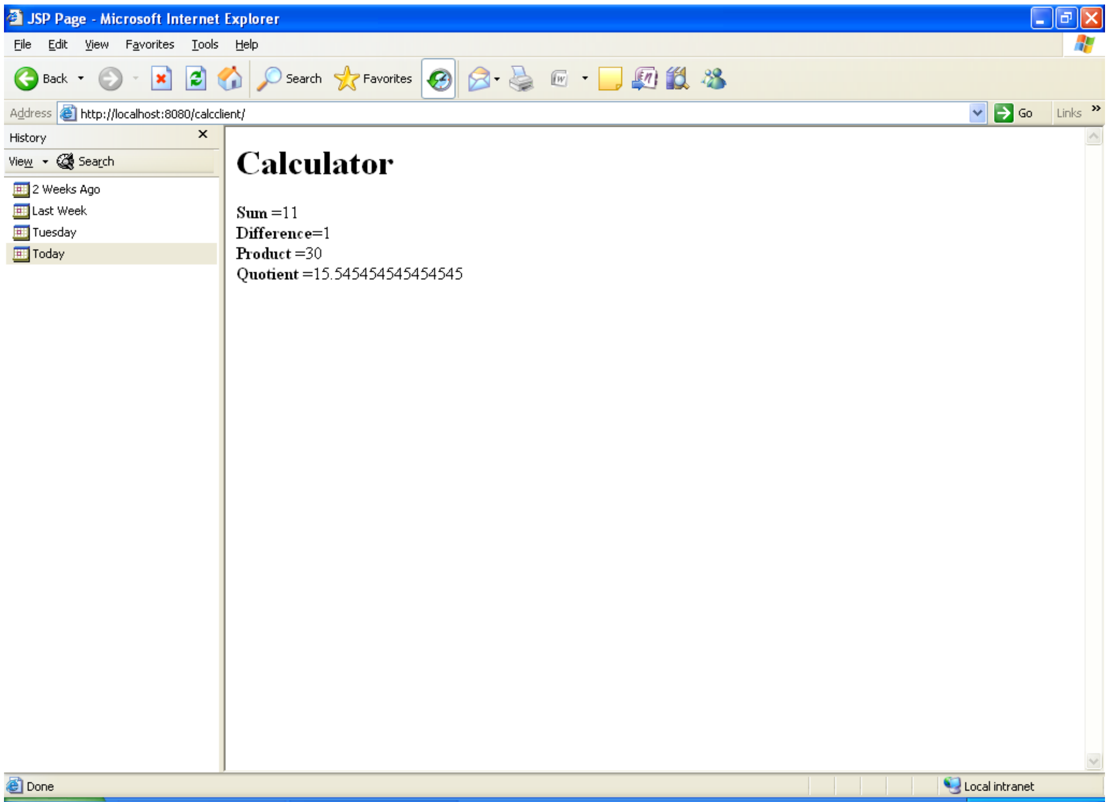
```
package calpack;
import javax.jws.WebService;
import javax.jws.WebMethod;
import javax.jws.WebParam;
import javax.ejb.Stateless;
/**
 *
 * @author student
 */
@WebService(serviceName = "Calcservice")
@Stateless()
public class Calcservice {

    /** This is a sample web service operation */
     /**
     * Web service operation
     */
    @WebMethod(operationName = "add")
    public int add(@WebParam(name = "a") int a, @WebParam(name = "b") int b) {
        //TODO write your implementation code here:
        return a+b;
    }
    /**
     * Web service operation
     */
    @WebMethod(operationName = "sub")
    public int sub(@WebParam(name = "a") int a, @WebParam(name = "b") int b) {
        //TODO write your implementation code here:
        return a-b;
    }
    /**
     * Web service operation
     */
    @WebMethod(operationName = "mult")
    public int mult(@WebParam(name = "a") int a, @WebParam(name = "b") int b) {
        //TODO write your implementation code here:
        return a*b;
    }
    /**
     * Web service operation
     */
    @WebMethod(operationName = "divide")
    public double divide(@WebParam(name = "a") double a, @WebParam(name = "b") double b) {
        //TODO write your implementation code here:
        return a/b;
    }
}
```

**INDEX.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>
        Calculator
</h1>
<%
      try
                  {
          calpack.Calcservice_Service service=new calpack.Calcservice_Service();
          calpack.Calcservice port=service.getCalcservicePort();
          int a=0;
          int b=0;
          int result1=port.add(5, 6);
          int result2=port.mult(5, 6);
          int result3=port.sub(6, 5);
          double result4=port.divide(85.5, 5.5);

          out.println("<b>Sum</b>          ="+result1);
          out.println("<br><b>Difference</b>="+result3);
          out.println("<br><b>Product</b>   ="+result2);
          out.println("<br><b>Quotient</b>  ="+result4);
              }
      catch(Exception e)
                  {
              }
      %>

</body>
</html>
```

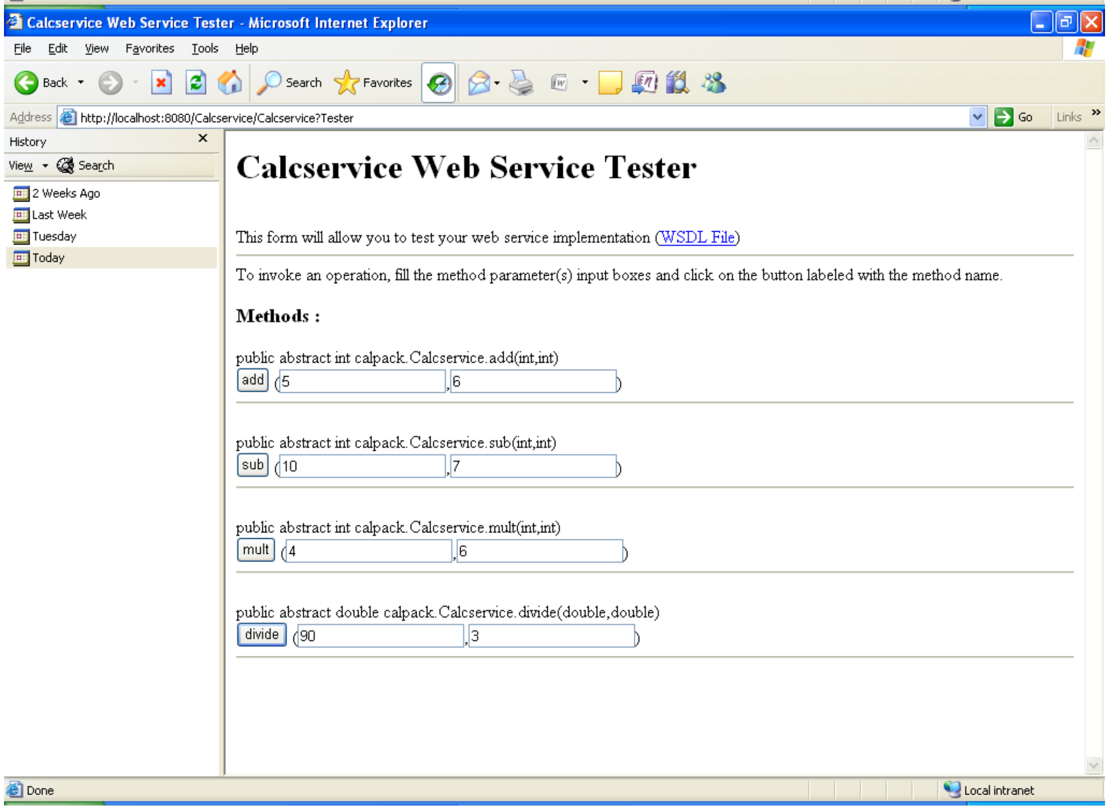
**OUTPUT:**



JSP Page - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://localhost:8080/calcclient/
History
2 Weeks Ago
Last Week
Tuesday
Today
Done
Local intranet

Calculator

**Sum** =11
**Difference**=1
**Product** =30
**Quotient** =15.545454545454545



Calcservice Web Service Tester - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://localhost:8080/Calcservice/Calcservice?Tester
History
2 Weeks Ago
Last Week
Tuesday
Today
Done
Local intranet

Calcservice Web Service Tester

This form will allow you to test your web service implementation (WSDL File)

To invoke an operation, fill the method parameter(s) input boxes and click on the button labeled with the method name.

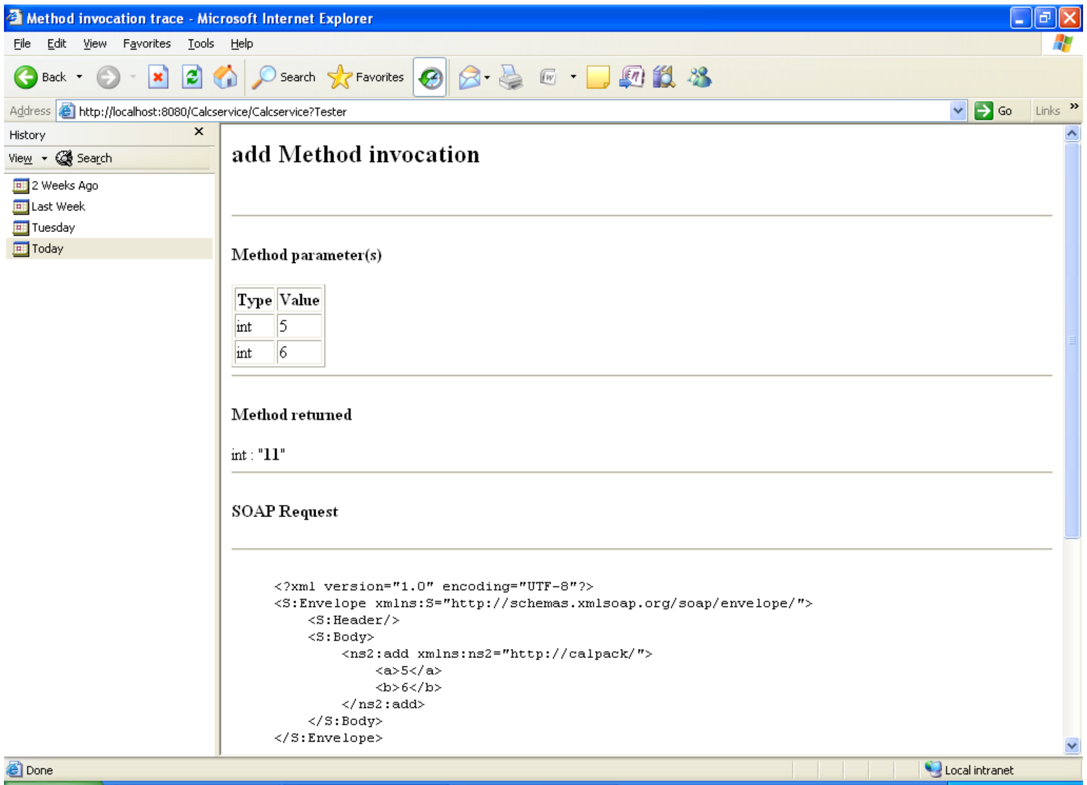
**Methods :**

public abstract int calpack.Calcservice.add(int,int)
add (5 ,6 )

public abstract int calpack.Calcservice.sub(int,int)
sub (10 ,7 )

public abstract int calpack.Calcservice.mult(int,int)
mult (4 ,6 )

public abstract double calpack.Calcservice.divide(double,double)
divide (90 ,3 )



## add Method invocation

**Method parameter(s)**

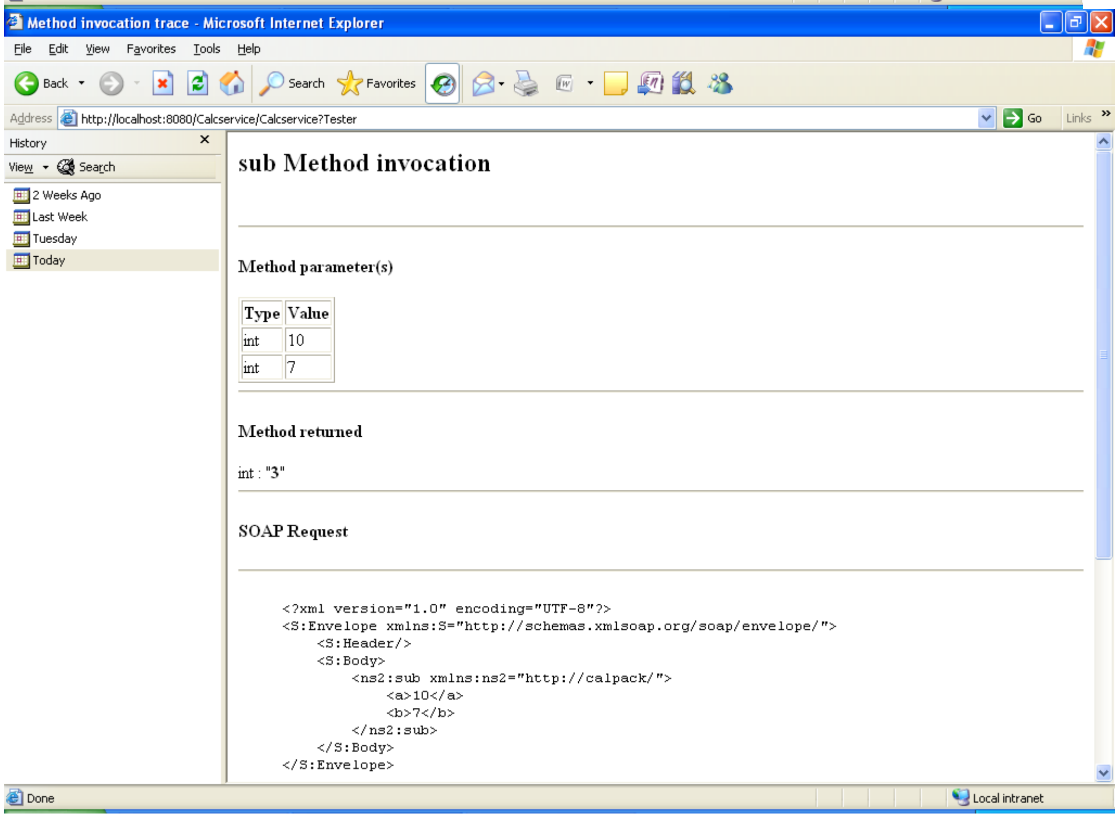
| Type | Value |
|---|---|
| int | 5 |
| int | 6 |

**Method returned**

int : "11"

**SOAP Request**

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
    <S:Header/>
    <S:Body>
        <ns2:add xmlns:ns2="http://calpack/">
            <a>5</a>
            <b>6</b>
        </ns2:add>
    </S:Body>
</S:Envelope>
```



## sub Method invocation

**Method parameter(s)**

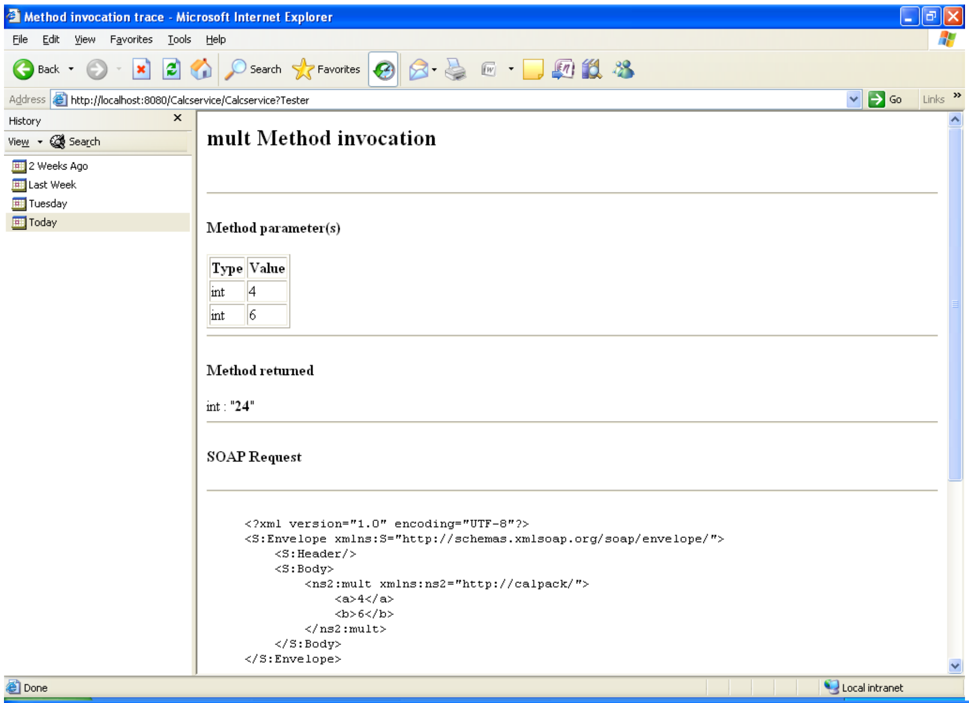
| Type | Value |
|---|---|
| int | 10 |
| int | 7 |

**Method returned**

int : "3"

**SOAP Request**

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
    <S:Header/>
    <S:Body>
        <ns2:sub xmlns:ns2="http://calpack/">
            <a>10</a>
            <b>7</b>
        </ns2:sub>
    </S:Body>
</S:Envelope>
```



Method invocation trace - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://localhost:8080/Calcservice/Calcservice?Tester
Go Links
History
View Search
2 Weeks Ago
Last Week
Tuesday
Today
Done
Local intranet

mult Method invocation

Method parameter(s)

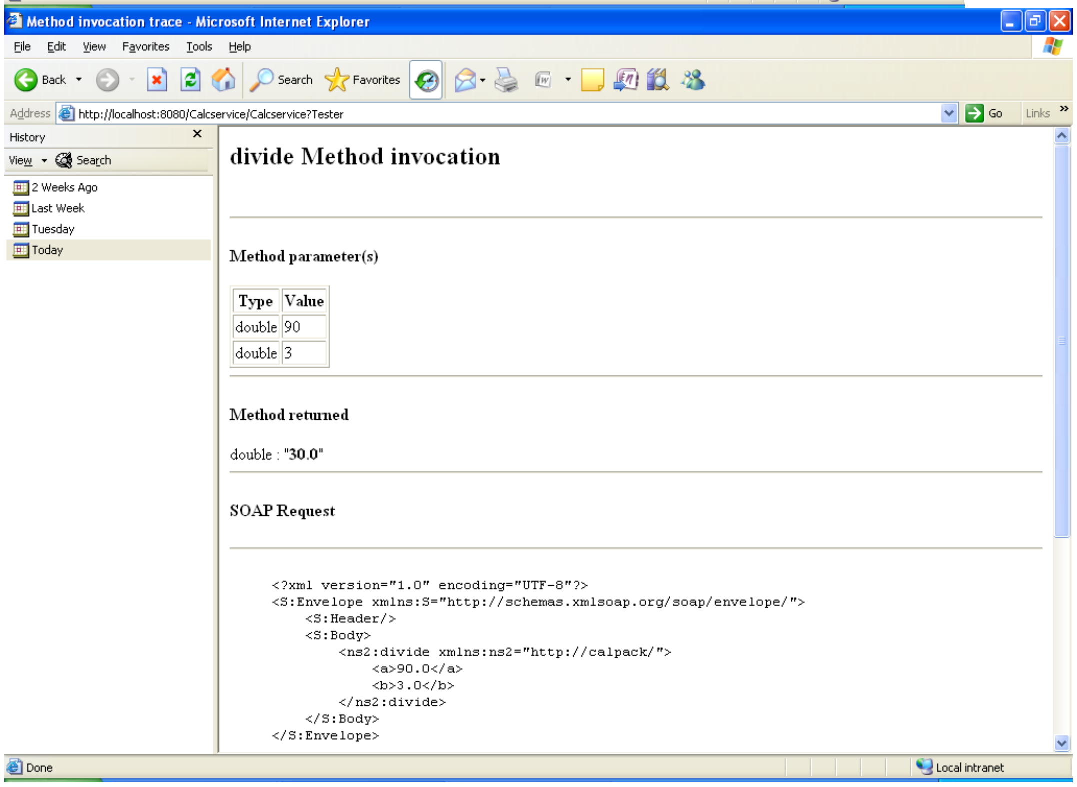
| Type | Value |
|---|---|
| int | 4 |
| int | 6 |

Method returned

int : "24"

SOAP Request

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
    <S:Header/>
    <S:Body>
        <ns2:mult xmlns:ns2="http://calpack/">
            <a>4</a>
            <b>6</b>
        </ns2:mult>
    </S:Body>
</S:Envelope>
```



Method invocation trace - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://localhost:8080/Calcservice/Calcservice?Tester
Go Links
History
View Search
2 Weeks Ago
Last Week
Tuesday
Today
Done
Local intranet

divide Method invocation

Method parameter(s)

| Type | Value |
|---|---|
| double | 90 |
| double | 3 |

Method returned

double : "30.0"

SOAP Request

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
    <S:Header/>
    <S:Body>
        <ns2:divide xmlns:ns2="http://calpack/">
            <a>90.0</a>
            <b>3.0</b>
        </ns2:divide>
    </S:Body>
</S:Envelope>
```

**Result:**

## EX No. 10.b AIRLINE RESERVATION WEB SERVICE

**Date:**

**Aim:**

To create a web service for implementing airline ticketing web service using oracle.

**Algorithm**

1. Code the implementation class.
2. Compile the implementation class.
3. Use wsgen to generate the artifacts required to deploy the service.
4. Package the files into a WAR file.
5. Deploy the WAR file. The web service artifacts (which are used to communicate with clients) are generated by the Application Server during deployment.
6. Code the client class.
7. Use wsimport to generate and compile the web service artifacts needed to connect to the service.
8. Compile the client class.
9. Run the client.

**DATABASE FOR AIRLINE RESERVATION**

| Name | Null? | Type |
|---|---|---|
| NUM | | NUMBER(10) |
| NAME | | VARCHAR2(15) |
| START_FROM | | VARCHAR2(15) |
| DEST | | VARCHAR2(15) |

| NUM | NAME | START_FROM | DEST |
|---|---|---|---|
| 1001 | british airways | pune | chennai |
| 2001 | air india | mumbai | pune |
| 3001 | indian airlines | chennai | pune |
| 2002 | airindia | chennai | mumbai |
| 1002 | british airways | chennai | pune |

**Source code:**

## AIRLINE RESERVATION WEB SERVICE

```
package airinfo;
import javax.jws.WebMethod;
import javax.jws.WebParam;
import javax.jws.WebService;
import java.sql.*;


@WebService( name="Test", serviceName = "TestDB" )
public class TestDB {

    /**
     * Web service operation
     */
    @WebMethod(operationName = "getInfo")
    public String getInfo(@WebParam(name = "src")
    String src, @WebParam(name = "dest")
    String dest) {
        String str="";
    try
    {
Class.forName("org.apache.derby.jdbc.ClientDriver");
Connection con=DriverManager.getConnection("jdbc:derby://localhost:1527/SDB","sairam","sairam");
    Statement s=con.createStatement();
    if(src.equals("mumbai"))
    {
      if(dest.equals("pune"))
        s.executeQuery("SELECT * FROM student. airways_table WHERE Start_from='mumbai'
AND Dest='pune'");
      else if(dest.equals("chennai"))
        s.executeQuery("SELECT * FROM student.airways_table WHERE Start_from='mumbai'
AND Dest='chennai'");
    }
    else if(src.equals("pune"))
    {
      if(dest.equals("mumbai"))
        s.executeQuery("SELECT * FROM student.airways_table WHERE Start_from='pune'
AND Dest='mumbai'");
      else if(dest.equals("chennai"))
        s.executeQuery("SELECT * FROM student.airways_table WHERE Start_from='pune'
AND Dest='chennai'");
    }
    else if(src.equals("chennai"))
    {
      if(dest.equals("pune"))
        s.executeQuery("SELECT * FROM student.airways_table WHERE Start_from='chennai'
AND Dest='pune'");
      else if(dest.equals("Mumbai"))
        s.executeQuery("SELECT * FROM student.airways_table WHERE Start_from='chennai'
AND Dest='mumbai'");
    }
    ResultSet rs=s.getResultSet();
    str+="<table border=1>";
    while(rs.next())
    {
      str+="<tr><b>";
      str+="<td>";
      str+=rs.getString("name");
      str+="</td>";
      str+="</tr></b>";
    }
    str+="</table>";

  }catch(ClassNotFoundException ex)
  {
     System.out.println(ex);
  }
  catch(SQLException ex)
  {
     System.out.println(ex);
  }
  return str;
    }

}
```

## AIRLINE RESERVATION CLIENT

index.jsp

```
<html>
 <head>
  <title>AIRLINE RESERVATION SYSTEM</title>
 </head>
 <body bgcolor="khaki">
  <center>
  <h4>Travel Agent should fill up following information to get information about the
Airline</h4>
   <form name="form1" method=GET action ="reserve.jsp">
   <table>
   <tr>
     <td><b>Source city: </b></td>
     <td><input type="text" name="Src_name" size="20" value=""></td>
   </tr>
   <tr>
    <td><b>Destination city: </b></td>
    <td><input type="text" name="Dest_name" size="20" value=""></td>
   </tr>
   <tr>
     <td><input type="submit" value="Submit"></td>
   </tr>
   </table>
    </form>
   </center>
  </body>
</html>
```

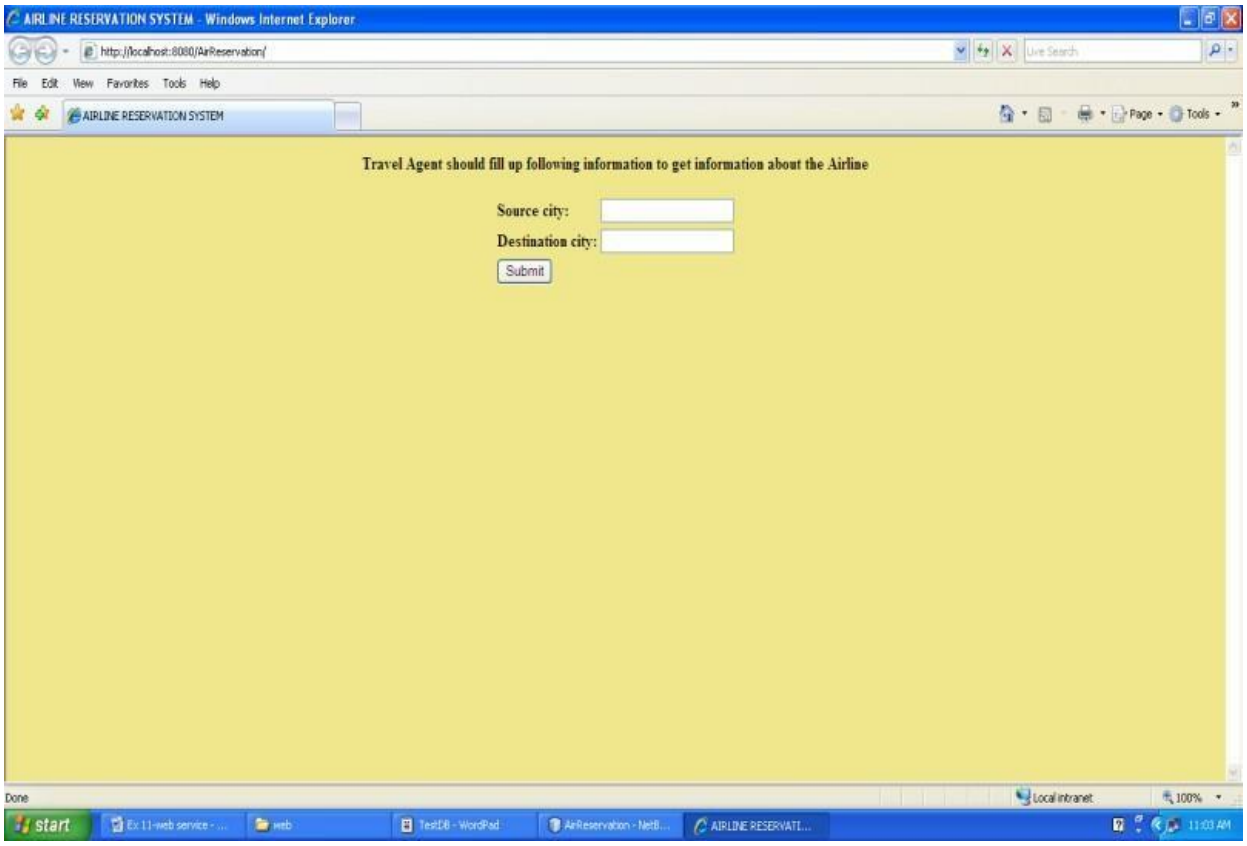
reserve.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>

Airline Seat Confirmation
<%
String a1=request.getParameter("Src_name");
String a2=request.getParameter("Dest_name");
%>
<%-- start web service invocation --%>
```

```
<%
try {
air.TestDB_Service service =new air.TestDB_Service();
air. TestDB port = service.get TestDB Port();
String result = port.get(a1, a2);
%>
<%
} catch (Exception ex) {
ex.printStackTrace();
}
%>

<%-- end web service invocation --%>
```

**Output:**

**AIRLINE RESERVATION**


AIRLINE RESERVATION SYSTEM - Windows Internet Explorer
http://localhost:8080/AirReservation/
File Edit View Favorites Tools Help
Submit
Done
Local intranet
100%
start

Travel Agent should fill up following information to get information about the Airline

Source city:

Destination city:

Airline Seat Confirmation

| |
|---|
| **indian airlines** |
| **british airways** |

**Result**

## Ex No. 11  DESIGN A LOGIN PAGE USING ANGULAR FRAMEWORK

**Aim**

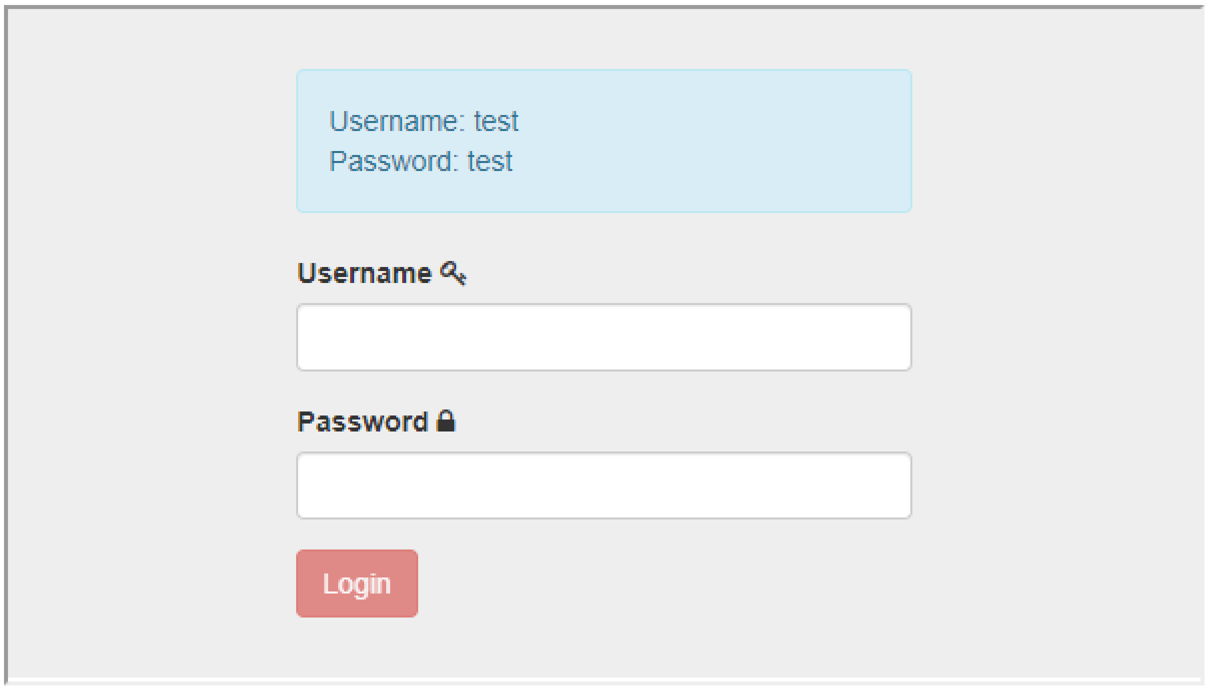
To design a login page using Angular Framework**.**

**Source code**

```
<div class="alert alert-info">

   Username: test<br />

   Password: test

</div>

<div ng-show="error" class="alert alert-danger">{{error}}</div>

<form name="form" ng-submit="login()" role="form">

   <div class="form-group">

      <label for="username">Username</label>

      <i class="fa fa-key"></i>

      <input type="text" name="username" id="username" class="form-control" ng-
model="username" required />

      <span ng-show="form.username.$dirty && form.username.$error.required" class="help-
block">Username is required</span>

   </div>

   <div class="form-group">

      <label for="password">Password</label>

      <i class="fa fa-lock"></i>

      <input type="password" name="password" id="password" class="form-control" ng-
model="password" required />

      <span ng-show="form.password.$dirty && form.password.$error.required" class="help-
block">Password is required</span>

   </div>

   <div class="form-actions">

      <button type="submit" ng-disabled="form.$invalid || dataLoading" class="btn btn-
danger">Login</button>

      <img ng-if="dataLoading" src="”>
```

```
    </div>
</form>
```

**Output:**



Username: test
Password: test

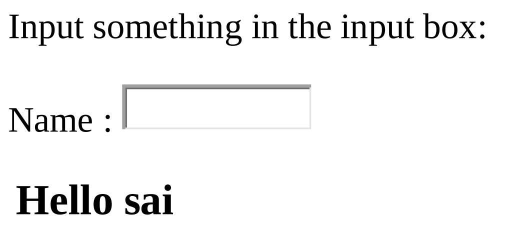
Result

## Basic Angular JS Programs

**1.Program to Greet the user**

```
<!DOCTYPE html>
<html>
<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular.min.js"></script>
<body>
<div ng-app="">
<p>Input something in the input box:</p>
<p>Name : <input type="text" ng-model="name" placeholder="Enter name here"></p>
<h1>Hello {{name}}</h1>
</div>
</body>
</html>
```

**Output:**

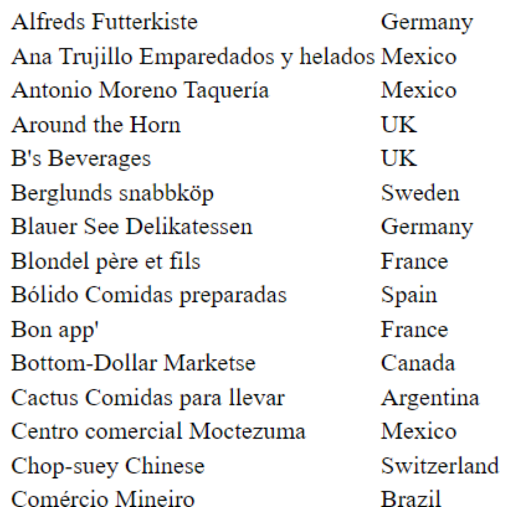


```
Input something in the input box:

Name :

Hello sai
```

**2.List the items using Table**

```
<!DOCTYPE html>
<html>
<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular.min.js"></script>
<body>
<div ng-app="myApp" ng-controller="customersCtrl">
<table>
  <tr ng-repeat="x in names">
    <td>{{ x.Name }}</td>
    <td>{{ x.Country }}</td>
  </tr>
</table>
</div>
<script>
var app = angular.module('myApp', []);
app.controller('customersCtrl', function($scope, $http) {
    $http.get("customers.php")
    .then(function (response) {$scope.names = response.data.records;});
});
</script>
</body>
</html>
```

**Output**



```
Alfreds Futterkiste                 Germany
Ana Trujillo Emparedados y helados  Mexico
Antonio Moreno Taquería             Mexico
Around the Horn                     UK
B's Beverages                       UK
Berglunds snabbköp                  Sweden
Blauer See Delikatessen             Germany
Blondel père et fils                France
Bólido Comidas preparadas           Spain
Bon app'                            France
Bottom-Dollar Marketse              Canada
Cactus Comidas para llevar          Argentina
Centro comercial Moctezuma          Mexico
Chop-suey Chinese                   Switzerland
Comércio Mineiro                    Brazil
```

**3.Displaying Array elements as list items**

```
<!DOCTYPE html>
<html>
<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular.min.js"></script>
<body>
<script>
var app = angular.module("myShoppingList", []);
app.controller("myCtrl", function($scope) {
    $scope.products = ["Milk", "Bread", "Cheese"];
});
</script>
<div ng-app="myShoppingList" ng-controller="myCtrl">
  <ul>
    <li ng-repeat="x in products">{{x}}</li>
  </ul>
</div>
<p>So far we have made an HTML list based on the items of an array.</p>
</body>
</html>
```

**Output**

- Milk
- Bread
- Cheese