Week-1

AIM: Design the static web pages required for an online book store web site.

1) HOME PAGE

DESCRIPTION:

The static home page must contain three **frames**.

- **Top frame** : Logo and the college name and links to Home page, Login page, Registration page,
- **Left frame** : At least four links for navigation, which will display the catalogue of respective links.

For e.g.: When you click the link **"CSE"** the catalogue for **CSE** Books should be displayed in the Right frame.

- **Right frame**: The pages to the links in the left frame must be loaded here. Initially this page contains description of the web site.

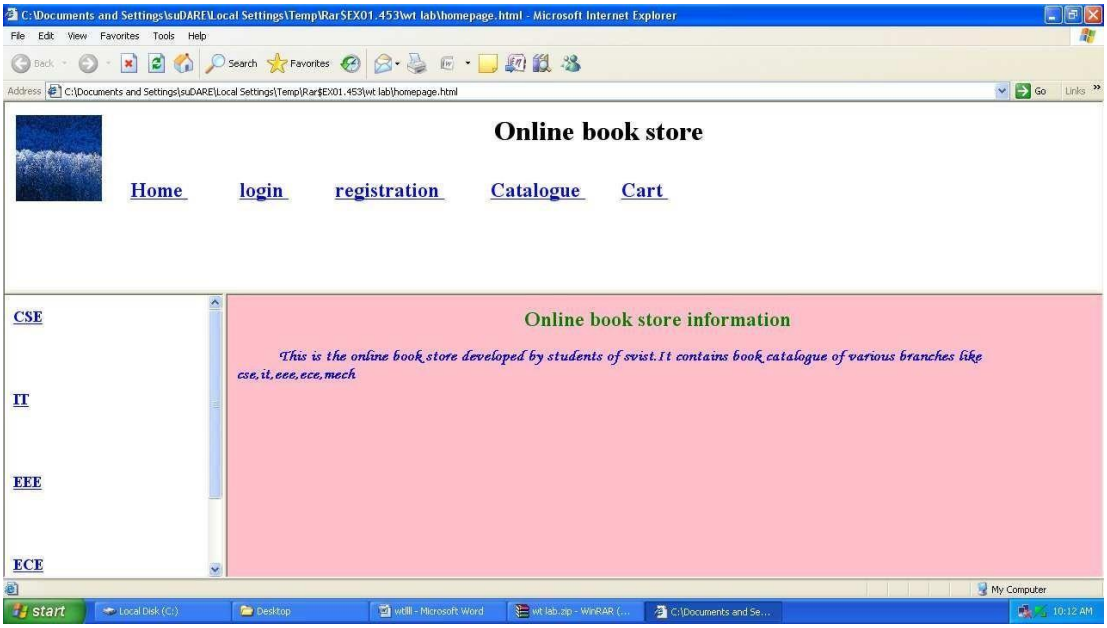
PROGRAM:

Homepage

```
<head>
 <frameset rows="20%,*">
        <frame src="topframe.html"name="f1">
         <frameset cols="20%,*">
        <frame src="leftframe.html"name="f2">
```

```
        <frame src="rightframe.html"name="f3">

          </frameset>

 </frameset>

</head>
```

OUTPUT:

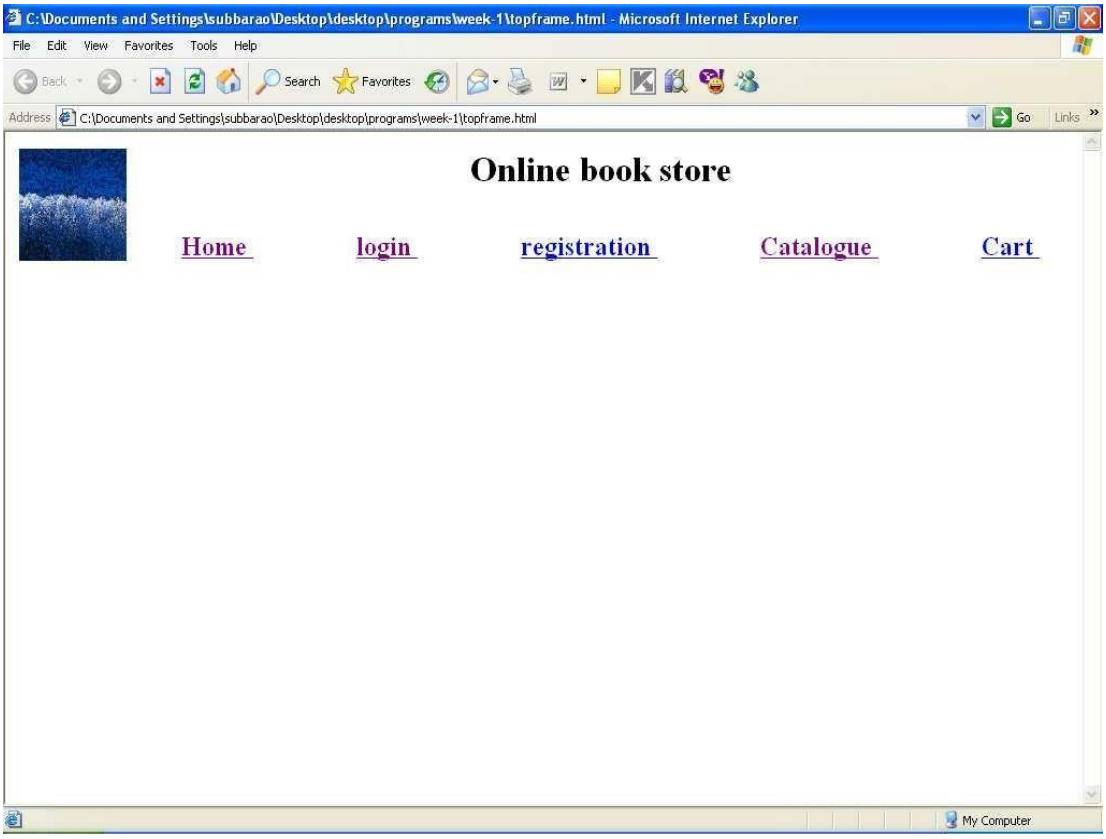


Top frame:

```
<html>
 <body>
  <p>
   <img src="C:\Documents and Settings\All Users\Documents\My Pictures\Sample Pictures\Winter.jpg"
align=left width=100 height=100">
    <h1 align=center>Online book store</h1>
  </p>
  <br>
 <p>
   <h2> nbsp;  
  <a href="homepage.html" target=_parent>
    Home
  </a>
            
  <a href="login.html" target="f3">
    login
  </a>
          
 <a href="registration.html" target="f3">
    registration
  </a>
       nbsp;  
```

```
  <a href="catalogue.html" target="f3">
    Catalogue
  </a>
          
  <a href="cart.html" target="f3">
    Cart
  </a>
 
</h2>
 </p>
</body>
</html>
```

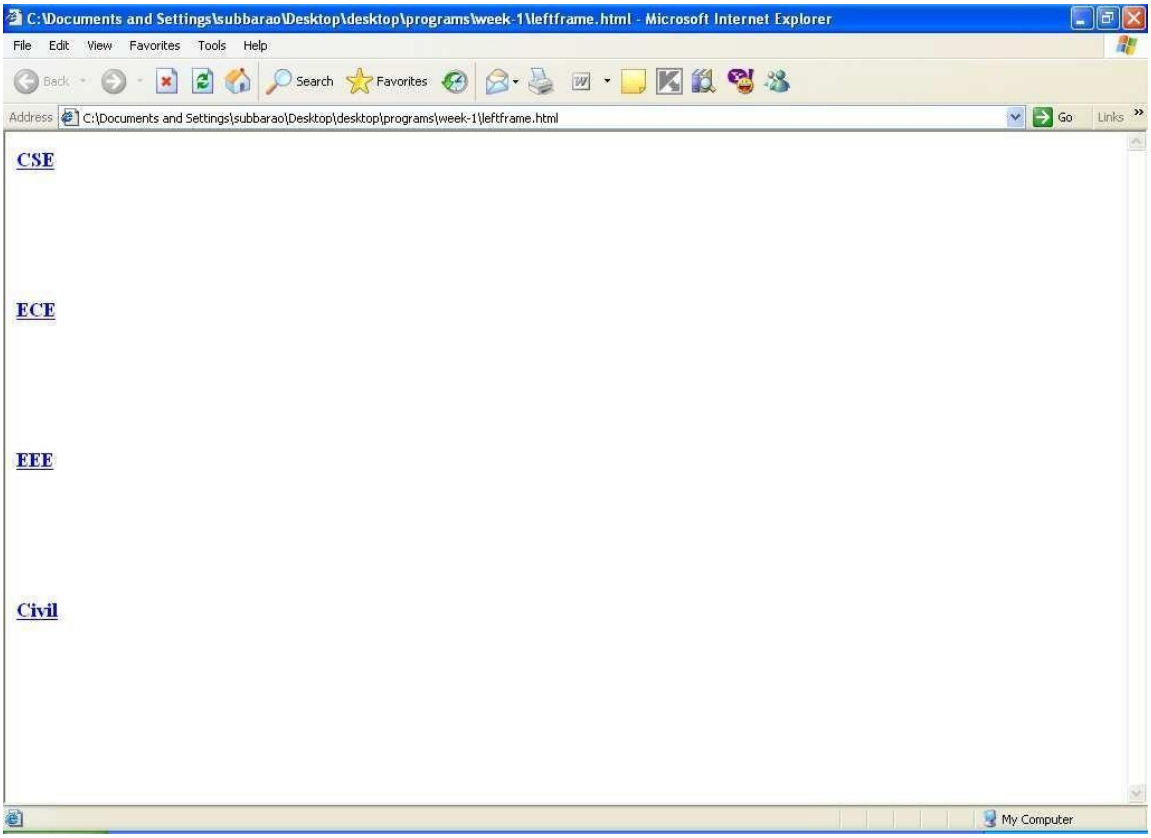
OUTPUT:



Leftframe:

```
<html>

  <body>

  <a href=cse.html target="f3"><h3>CSE</h3> </a><br><br><br><br><br>

  <a href=ece.html target="f3"><h3>ECE</h3></a><br><br><br><br><br>

  <a href=eee.html target="f3"><h3>EEE</h3></a><br><br><br><br><br>

  <a href=civil.html target="f3"><h3>Civil</h3></a>

 </body>

</html>
```

OUTPUT:

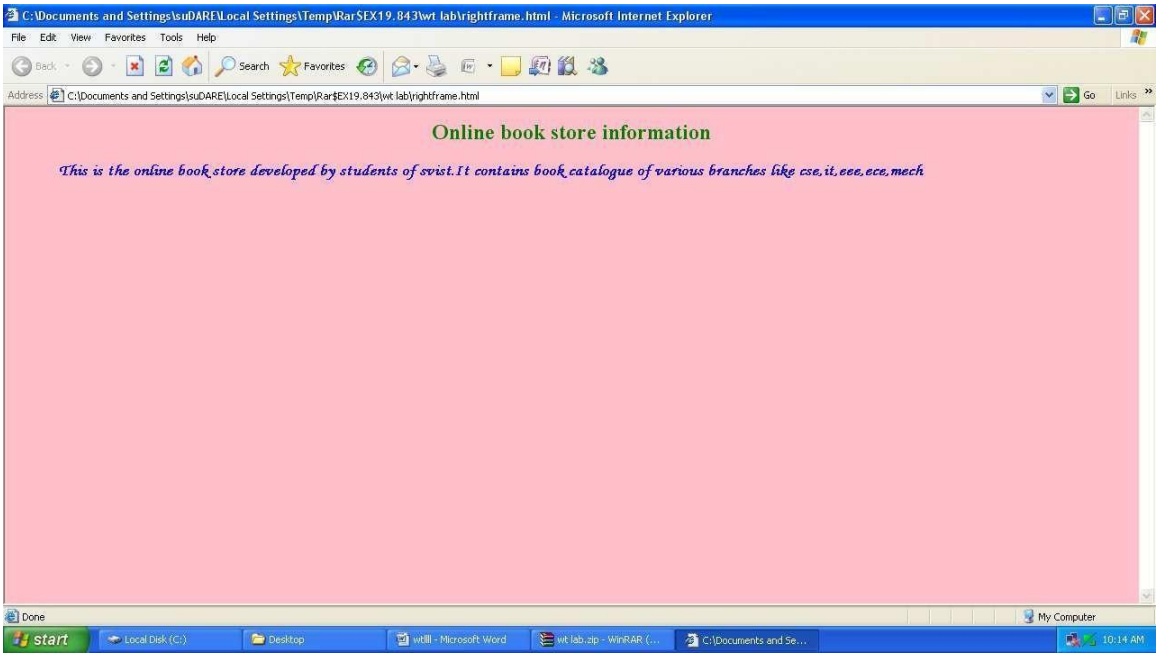


Right frame:

```
<html>

  <body bgcolor="pink">

  <p>
```

```
<h2 align="center"> <font face="times new roman" color="green" >Online book store information </font>
</h2>

<h3>           <font face="monotype corsiva" color=blue> This is the online book store developed by students of SVIST.It contains book catalogue of various branches like cse,ece,eee,civil </font></h3>

 </p>

 </body>

</html>
```

OUTPUT:



2) LOGIN PAGE

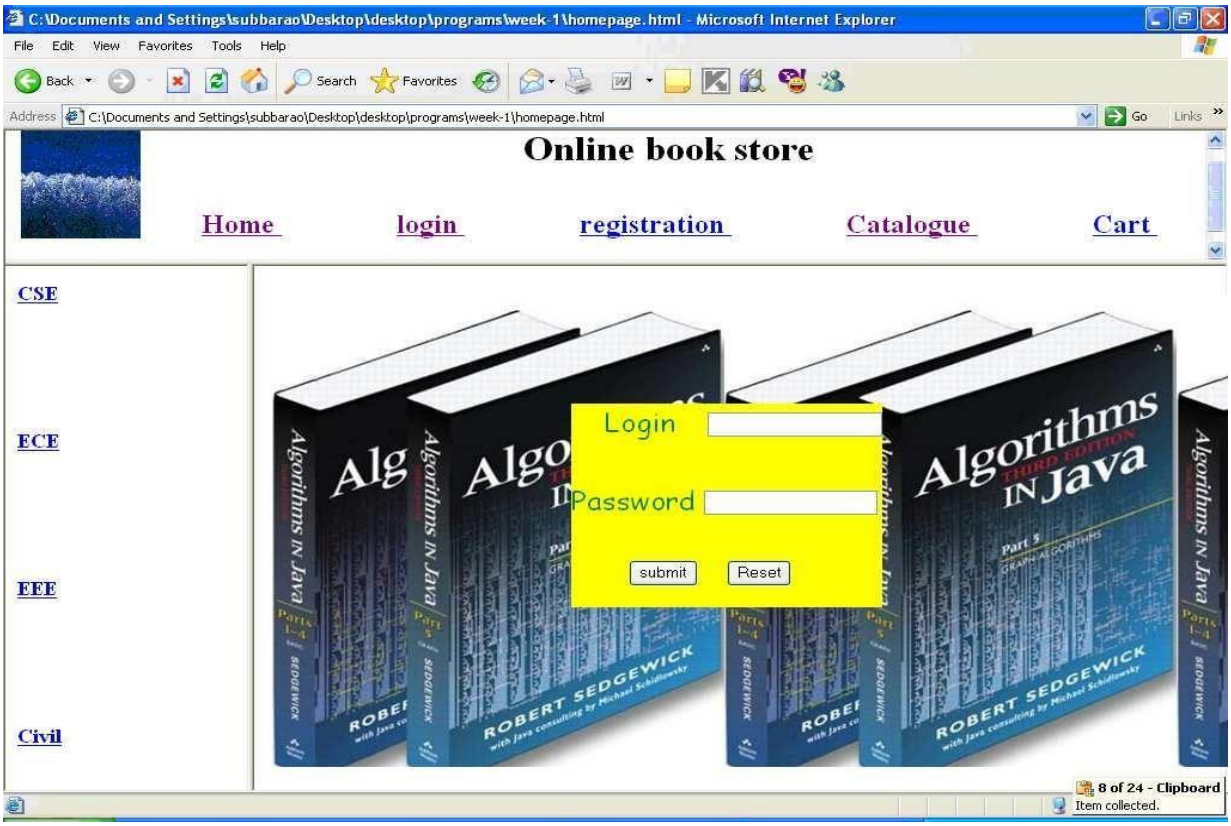
DESCRIPTION:

The login page contains the user name and the password of the user to authenticate.

PROGRAM:

```
<html>
        <head>
          <p style= "background:yellow; top:100px; left:250px; position:absolute; ">
        </head>
        <body background="E:\1.jpg">
         <form action="index.jsp" method="get">
         <label>       
              <font face="comic sans ms" color="green" size="5">Login </font>
         </label>     
           <input type="text" size="20" maxlength="20" value=' '><br><br><br>
         <label>
           <font face="comic sans ms" color="green" size="5">Password </font>          </label>
           <input type="password" size="20" maxlength="20"> <br><br><br>       
    
           <input type="submit" value="submit">     
           <input type="reset">
         </form>
        </body>
</html>
```

OUTPUT:


C:\Documents and Settings\subbarao\Desktop\desktop\programs\week-1\homepage.html - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address C:\Documents and Settings\subbarao\Desktop\desktop\programs\week-1\homepage.html
Go Links
Online book store
Home
login
registration
Catalogue
Cart
CSE
ECE
EEE
Civil
Login
Password
submit
Reset
Algorithms IN Java
ROBERT SEDGEWICK
8 of 24 - Clipboard
Item collected.

3) CATOLOGUE PAGE

DESCRIPTION:

The catalogue page should contain the details of all the books available in the web site in a table.

The details should contain the following:

1. Snap shot of Cover Page.
2. Author Name.
3. Publisher.
4. Price.
5. Add to cart button.

PROGRAM:

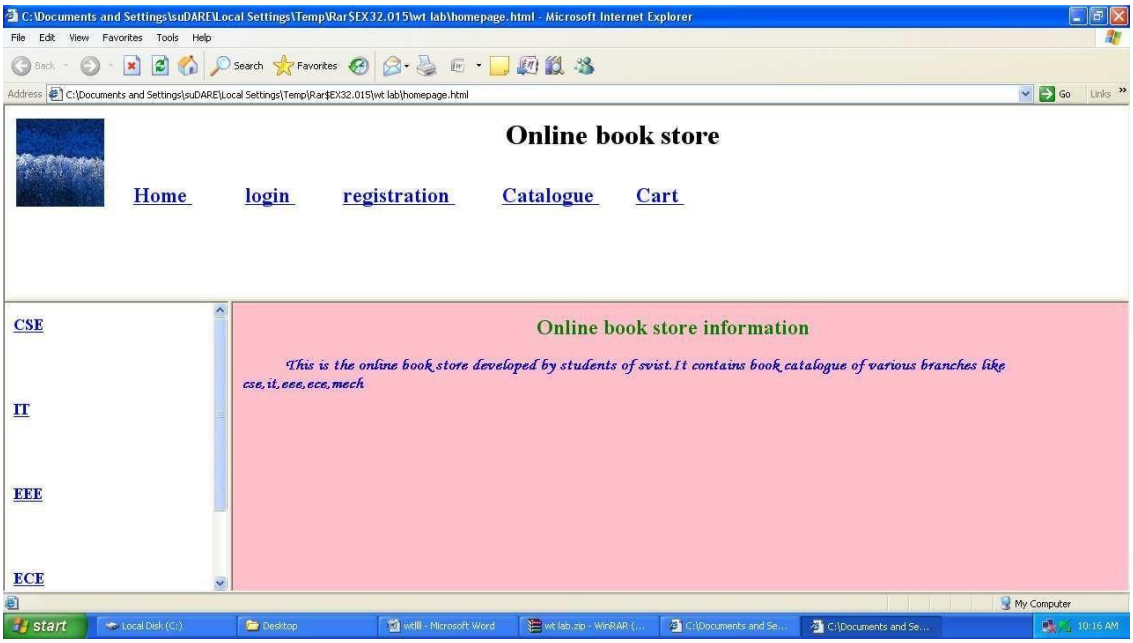
```
<html>
<body>
<center>
<table border=1>
<tr>
<th> Book Preview </th>
<th> Book Details </th>
<th> Price </th>
<th> Payment </th>
</tr>
 <tr>
  <td> <img src="C:\Documents and Settings\All users\My Documents\My Pictures\xml.bmp" width=100
```

```
height=50>
      </img>
    </td>
    <td>
    <pre>
        <font face="comic sans ms" size=4 color="green" >
        book:XML Bible
        Author:winston
        Publisher:Wiesley
        </font>
    </pre>
    </td>
  <td> $40   </td>
  <td>     <a href="cart.html" target="_blank">
      <img src="C:\Documents and Settings\All users\My Documents\My Pictures\cart.bmp" width="150"
  height="100"></img>
      </a>    
   </td>
  </tr>
  <tr>
    <td> <img src="C:\Documents and Settings\All users\My Documents\My Pictures\java.bmp" width=100
  height=50>
        </img>
```

```
  </td>
<td>
  <pre>
      <font face="comic sans ms" size=4 color="green" >
       book:Java 2
       Author:Watson
       Publisher:BPB publications
      </font>
  </pre>
  </td>
<td>  $40  </td>
<td>     <a href="cart.html" target="_blank">
     <img src="C:\Documents and Settings\All users\My Documents\My Pictures\cart.bmp" width="150"
height="100"></img>
     </a>    
 </td>
</tr>
</table> </center>
</body>
</html>
```

OUTPUT:


C:\Documents and Settings\subbarao\Desktop\desktop\programs\week-1\homepage.html - Microsoft Internet Explorer
Online book store
Home
login
registration
Catalogue
Cart
CSE
ECE
EEE
Civil
Book Preview
Book Details
Price
Payment
book:XML Bible
Author:winston
Publisher:Wiesley
$40
Add to cart
book:Java 2
Author:Watson
Publisher:BPB publications
$40
Add to cart
file:///C:/Documents%20and%20Settings/subbarao/Desktop/desktop/programs/week-1/catalogue.html
My Computer


C:\Documents and Settings\suDARE\Local Settings\Temp\Rar$EX32.015\wt lab\homepage.html - Microsoft Internet Explorer
Online book store
Home
login
registration
Catalogue
Cart
CSE
IT
EEE
ECE
Online book store information
This is the online book store developed by students of svist.It contains book catalogue of various branches like cse,it,eee,ece,mech
My Computer

RESULT:

Thus the home page, login page, catalogue page for the online book store are created successfully

Week-2

AIM: Design of the cart page and the registration page required for online book store.

4) CART PAGE

DESCRIPTION:

The cart page contains the details about the books which are added to the cart.

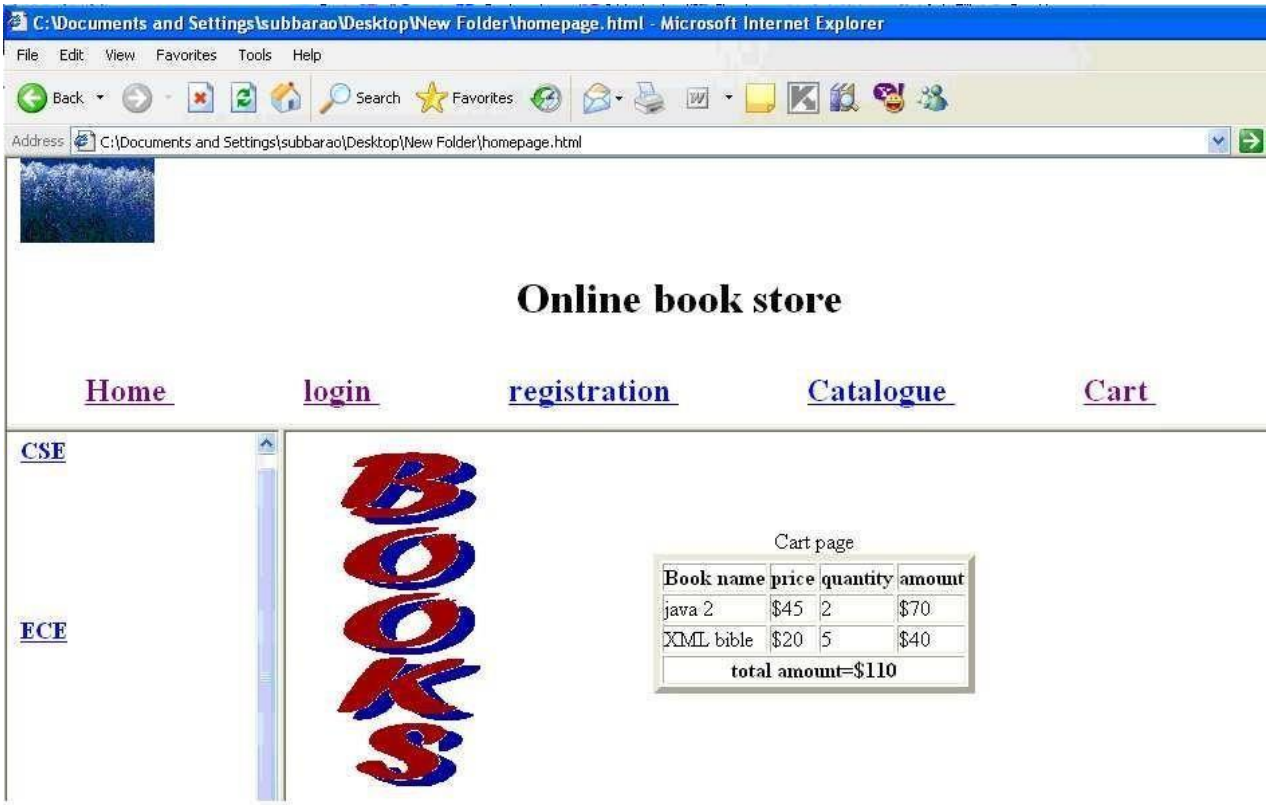
PROGRAM:

```
<html>
<body>
<center><br><br><br>
<img src="E:\aa.jpg">
<table border=1 cellpadding=center>
<thead>
<tr>
<th>Book name</th>
```

```
<th>price</th>

<th>quantity</th>

<th>amount</th>

</tr>

</thead>

<tr>

<td>java 2</td>

<td>$45</td>

<td>2</td>

<td>$70</td>

</tr>

<tr>

<td> XML bible</td>

<td> $20</td>

<td> 5</td>

<td> $40</td>

</tr>

<th colspan=4>total amount=$110>

</th>

</tr>

</table>

</center>
```

```
</body>

</html>
```

OUTPUT:



5) REGISTRATION PAGE

DESCRIPTION:

Create a "*registration form* "with the following fields

1) Name(Textfield)
2) Password (password field)
3) E-mail id (text field)
4) Phone number (text field)
5) Sex (radio button)
6) Date of birth (3 select boxes)
7) Languages known (check boxes – English, Telugu, Hindi, Tamil)
8) Address (text area)

PROGRAM:

```
<html>
<body>
<center>
<img src="E:\aaa.jpg" align="left">
  <form>
<label>name</label>
<input type="text" size="20"><br><br> <br>
<label>password</label>
<input type="password" size="20" maxsize="28"><br> <br> <br>
```

```
<label>email</label>

<input type="text" size="30"><br> <br> <br>

<label>phone no</label>

<input type="text" size="2">

<input type="text" size="6">

<input type="text" size="10"><br> <br> <br>

<label>Sex</label>

<input type="radio" name="sex">m

<input type="radio" name="sex">f <br> <br> <br>

<label> date of birth</label>

<select>

  <option>1</option>

  <option>2</option>

  <option>3</option>

  <option>4</option>

  <option>5</option>

</select>

<select>

    <option>jan</option>

    <option>feb</option>

    <option>mar</option>

    <option>apr</option>
```

```
</select>

<select>

 <option>1980</option>

 <option>1981</option>

 <option>1982</option>

 <option>1983</option>

</select> <br> <br> <br>

<label> Languages Known </label>    

<input type="checkbox"> English     

<input type="checkbox"> Telugu      

<input type="checkbox"> Hindi      

<input type="checkbox"> Tamil <br> <br> <br>

 <label> Address </label>

<textarea rows=5 cols=20 scrolling="yes"> </textarea>

</center>

</body>

</html>
```

OUTPUT:

RESULT:

Thus the registration and cart pages for online book store pages are created successfully

Week-3

AIM: Write *JavaScript* to validate the following fields of the above registration page.

1. Name (Name should contains alphabets and the length should not be less than 6 characters).
2. Password (Password should not be less than 6 characters length).
3. E-mail id (should not contain any invalid and must follow the standard pattern name@domain.com)
4. Phone number (Phone number should contain 10 digits only).

DESCRIPTION:

JavaScript is a simple scripting language invented specifically for use in web browsers to make websites more dynamic. On its own, HTML is capable of outputting more-or-less static pages. Once you load them up your view doesn't change much until you click a link to go to a new page. Adding JavaScript to your code allows you to change how the document looks completely, from changing text, to changing colors, to changing the options available in a drop-down list. JavaScript is a *client-side* language.

JavaScripts are integrated into the browsing environment, which means they can get information about the browser and HTML page, and modify this information, thus changing how things are presented on your screen. This access to information gives JavaScript great power to modify the browsing experience. They can also react to *events*, such as when the user clicks their mouse, or points to a certain page element. This is also a very powerful ability.

Regular Expressions:

One of the most common situations that come up is having an HTML form for users to enter data. Normally, we might be interested in the visitor's name, phone number and email address, and so forth. However, even if we are very careful about putting some

hints next to each required field, some visitors are going to get it wrong, either accidentally or for malicious purposes. Here's where regular expressions come in handy. A regular expression is a way of describing a pattern in a piece of text. In fact, it's an easy way of matching a string to a pattern. We could write a simple regular expression and use it to check, quickly, whether or not any given string is a properly formatted user input. This saves us from difficulties and allows us to write clean and tight code.

A regular expression is a JavaScript object. There are multiple ways of creating them. They can be created statically when the script is first parsed or dynamically at run time. A static regular expression is created as follows:

```
regx=/fish|fow1/;
```

Dynamic patterns are created using the keyword to create an instance of the RegExp class:

```
regx=new RegExp("fish|fow1");
```

Functions:

test(string)- Tests a string for pattern matches. This method returns a Boolean that indicates whether or not the specified pattern exists within the searched string. This is the most commonly used method for validation. It updates some of the properties of the parent RegExp object following a successful search.

exec(string)- Executes a search for a pattern within a string. If the pattern is not found, exec() returns a null value. If it finds one or more matches it returns an array of the match results. It also updates some of the properties of the parent RegExp object

PROGRAM:

Valid.js

```
function fun()
{
```

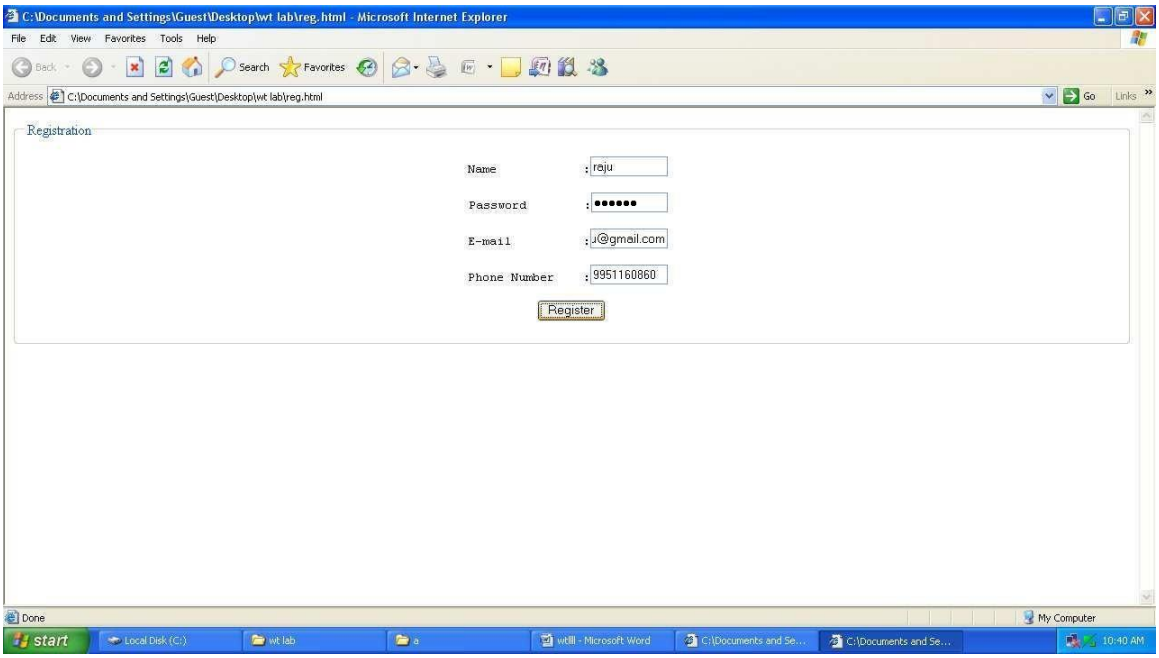
```
var userv=document.forms[0].user.value;

var pwdv=document.forms[0].pwd.value;

var emailv=document.forms[0].email.value;

var phv=document.forms[0].ph.value;

        var userreg=new RegExp("^[a-zA-Z][a-zA-Z0-9]*$");

        var  emailreg=new  RegExp("^[a-zA-Z][a-zA-Z0-9_.]*@[a-zA-Z][a-zA-Z0-9_.]*.[a-zA-Z][a-
zA-Z0-9_.]{2}.[a-zA-Z][a-zA-Z0-9_.]{2}$|^[a-zA-Z][a-zA-Z0-9_.]*@[a-zA-Z][a-zA-Z0-9_.]*.[a-zA-
Z][a-zA-Z0-9_.]{3}$");

        var phreg=new RegExp("^[0-9]{10}$");

var ruser=userreg.exec(userv);

var remail=emailreg.exec(emailv);

var rph=phreg.exec(phv);

if(ruser && remail && rph && (pwdv.length > 6))

        {

        alert("All values are valid");

        return true;

        }

        else

        {

        if(!ruser)  {  alert("username  invalid");document.forms[0].user.focus();}

        if(!remail) { alert("password invalid");document.forms[0].user.focus();}

        if(!rph) { alert("phone number invalid");document.forms[0].ph.focus();}

        if(pwdv.length < 6) { alert("password invalid");document.forms[0].pwd.focus();}

        return false;

        }
```

```
}
```

Register.html

```
<html>
<body>
<center>
<fieldset>
<legend>Registration</legend>
<form action="Database" method="get" onSubmit="return fun()">
<pre>
Name            :<input type="text" name="user" size="10"><br>
Password        :<input type="password" name="pwd" size="10"><br>
E-mail          :<input type="text" name="email" size="10"><br>
Phone Number  :<input type="text" name="ph" size="10"><br>
<input type="submit" value="Register">
</pre>
</form>
</body>
<script src="valid.js"></script>
</html>
```

OUTPUT:



RESULT:

Thus the home page, login page, catalogue page for the online book store are created successfully

Week-4

AIM:

Design a web page using **CSS (C**ascading **S**tyle **S**heets**)** which includes the following:

1) Use different font, styles: In the style definition you define how each selector should work .Then, in the body of your pages, you refer to these selectors to activate the styles.
2) Set a background image for both the page and single elements on the page.
3) Control the repetition of the image with the background-repeat property

DESCRIPTION:

Cascading Style Sheets (**CSS**) is a stylesheet language used to describe the presentation of a document written in a markup language. Its most common application is to style web pages written in HTML and XHTML, but the language can be applied to any kind of XML document.

In CSS, *selectors* are used to declare which elements a style applies to, a kind of match expression. Selectors may apply to all elements of a specific type, or only those elements which match a certain attribute; elements may be matched depending on how they are placed relative to each other in the markup code, or on how they are nested within the document object model

A style sheet consists of a list of *rules*. Each rule or rule-set consists of one or more *selectors* and a *declaration block*. A declaration-block consists of a list of semicolon-separated *declarations* in braces. Each declaration itself consists of a *property*, a colon (*:*), a *value*, then a semi-colon (*;*)

Generally speaking we can say that all the styles will "cascade" into a new "virtual" style sheet by the following rules:

1. External style sheet
2. Internal style sheet (inside the <head> tag)
3. Inline style (inside an HTML element)

An inline style (inside an HTML element) has the highest priority, which means that it will override a style declared inside the <head> tag, in an external style sheet, or in a browser (a default value).

Syntax

The CSS syntax is made up of three parts: a selector, a property and a value:

selector {property: value}

The selector is normally the HTML element/tag you wish to define, the property is the attribute you wish to change, and each property can take a value. The property and value are separated by a colon, and surrounded by curly braces:

body {color: black}

External Style Sheet

An external style sheet is ideal when the style is applied to many pages. With an external style sheet, you can change the look of an entire Web site by changing one file. Each page must link to the style sheet using the <link> tag. The <link> tag goes inside the head section:

```
<head>

<link rel="stylesheet" type="text/css"

href="mystyle.css" />

</head>
```

The browser will read the style definitions from the file mystyle.css, and format the document according to it.

Internal Style Sheet

An internal style sheet should be used when a single document has a unique style. You define internal styles in the head section by using the <style> tag,

```
<head>
<style>
selector {property:value; property:value;…..}
```

```
</style>
</head>
```

Inline Styles

An inline style loses many of the advantages of style sheets by mixing content with presentation. Use this method sparingly, such as when a style is to be applied to a single occurrence of an element.

To use inline styles you use the style attribute in the relevant tag. The style attribute can contain any CSS property.

```
<p style="color: sienna; margin-left: 20px">

This is a paragraph  </p>
```

PROGRAM:

Cas.css:

```
a:link{color:black;}

a:visited{color:pink;}

a:active{color:red;}

a:hover{color:green;}


.right {

        text-align:center;

        text-decoration:underline;

        font-weight:bold;

        color:blue;
```

```
font-family:comic sans ms;
font-size:30; }
.image {
text-align:left;
font-family:"monotype corsiva";
font-weight:10;
}

.image1 {
background-image:url("C:\Documents and Settings\All Users\My Documents\My Pictures\krishna.jpg");
background-attachment:fixed;
background-repeat:no-repeat;
width:150;
height:150; }
table { align:center;border:10;
border-style:ridge;
border-color:yellow;}
```

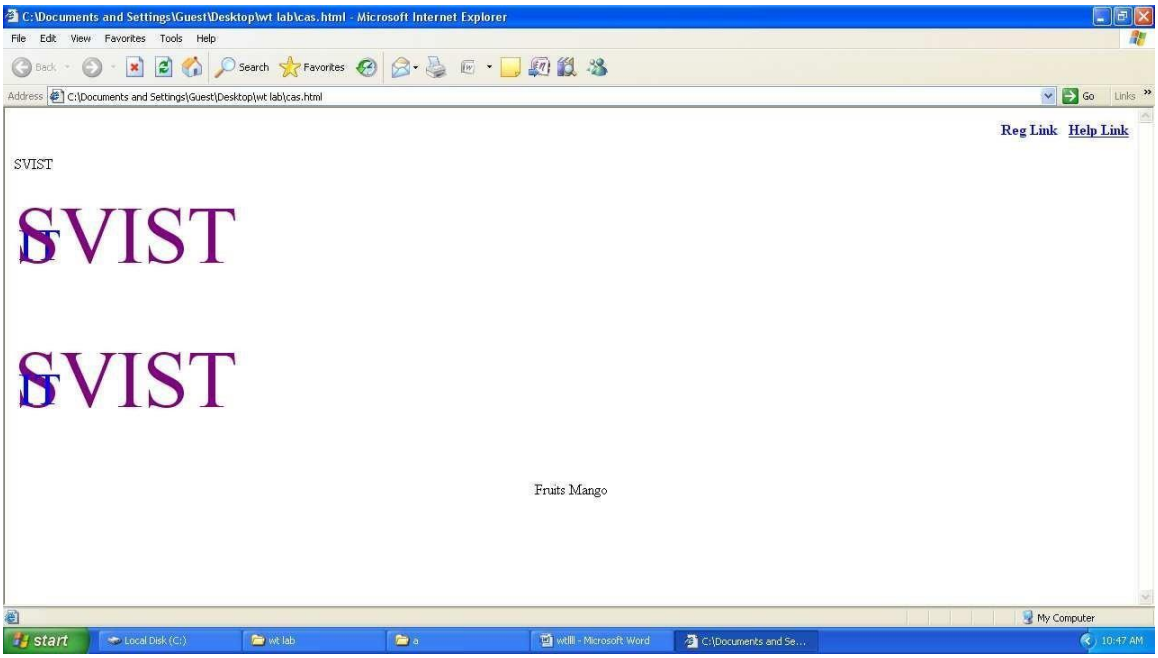
htm.html:

```
<html>
<head>
  <link rel="stylesheet" href="cas.css" type="text/css">
```

```
  <style>
    .xlink{ text-decoration:none;font-weight:bold;cursor:crosshair;}
    .ylink{text-decoration:underline;font-weight:bold;cursor:help;}
  </style>
</head>
<body class="image">


   <p style="text-align:right;">
      <a href="registration.html" class="xlink"> Reg Link</a>  
      <a href="topframe.html" class="ylink"> Help Link</a>
   </p>
  <p class="right">SVIST</p>
      <div style="position:relative;font-size:90px;z-index:5;color:purple;">SVIST</div>
      <div style="position:relative;font-size:50px;z-index:1;top:-70; left:5;color:blue;">CSE</div>
      <div style="position:relative;font-size:90px;z-index:1;color:purple;">SVIST</div>
      <div style="position:relative;font-size:50px;z-index:5;top:-70; left:5;color:blue;">CSE</div>
   <table align="center" class="image1">
      <tr>
         <td> Fruits</td>
         <td> Mango</td>
     </tr>
  </table>
```

```
</body>

</html>
```

OUTPUT:



RESULT: Thus different style of CSS and different type of the properties are applied.

Week-5:

AIM: Write an XML file which will display the Book information.

It includes the following:

1) Title of the book
2) Author Name
3) ISBN number
4) Publisher name
5) Edition
6) Price

Write a Document Type Definition (DTD) to validate the above XML file.

Display the XML file as follows.

The contents should be displayed in a table. The header of the table should be in color GREY. And the Author names column should be displayed in one color and should be capitalized and in bold. Use your own colors for remaining columns.

Use XML schemas XSL and CSS for the above purpose.

DESCRIPTION:

DTD vs XML Schema

The DTD provides a basic grammar for defining an XML Document in terms of the metadata that comprise the shape of the document. An XML Schema provides this, plus a detailed way to define what the data can and cannot contain. It provides far more control for the developer over what is legal, and it provides an Object Oriented approach, with all the benefits this entails.

Many systems interfaces are already defined as a DTD. They are mature definitions, rich and complex. The effort in re-writing the definition may not be worthwhile.

DTD is also established, and examples of common objects defined in a DTD abound on the Internet -- freely available for re-use. A developer may be able to use these to define a

DTD more quickly than they would be able to accomplish a complete re-development of the core elements as a new schema.

Finally, you must also consider the fact that the XML Schema is an XML document. It has an XML Namespace to refer to, and an XML DTD to define it. This is all overhead. When a parser examines the document, it may have to link this all in, interpret the DTD for the Schema, load the namespace, and validate the schema, etc., all *before* it can parse the actual XML document in question. If you're using XML as a protocol between two systems that are in heavy use, and need a quick response, then this overhead may seriously degrade performance.

- **Write a Document Type Definition (DTD) to validate the XML file.**

PROGRAM:

XML document (bookstore.xml)

```
<bookstore>
    <book>
        <title>web programming</title>
        <author>chrisbates</author>
        <ISBN>123-456-789</ISBN>
        <publisher>wiley</publisher>
        <edition>3</edition>
        <price>350</price>
    </book>
    <book>
        <title>internet worldwideweb</title>
        <author>ditel&amp;ditel</author>
        <ISBN>123-456-781</ISBN>
```

```
            <publisher>person</publisher>
            <edition>3</edition>
            <price>450</price>
        </book>
</bookstore>
```

XML document Validation using DTD

DTD document (bookstore.dtd)

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT bookstore (book+)>
<!ELEMENT book (title,author,ISBN,publisher,edition,price)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT author (#PCDATA)>
<!ELEMENT ISBN (#PCDATA)>
<!ELEMENT publisher (#PCDATA)>
<!ELEMENT edition (#PCDATA)>
<!ELEMENT price (#PCDATA)>
```

Bookstore.xml

```
<!DOCTYPE bookstore SYSTEM "C:\Documents and Settings\Administrator\My Documents\bookstore.dtd">
<bookstore>
        <book>
```

```
        <title>web programming</title>
        <author>chrisbates</author>
        <ISBN>123-456-789</ISBN>
        <publisher>wiley</publisher>
        <edition>3</edition>
        <price>350</price>
    </book>
    <book>
        <title>internet worldwideweb</title>
        <author>ditel&amp;ditel</author>
        <ISBN>123-456-781</ISBN>
        <publisher>person</publisher>
        <edition>3</edition>
        <price>450</price>
    </book>
</bookstore>
```

XML document Validation using DTD

XML Schema (bookstore.xsd)

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema elementFormDefault="qualified" attributeFormDefault="unqualified"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="bookstore">
```

```
                    <xs:complexType>
                    <xs:sequence>
<xs:element name="book" maxOccurs="unbounded">
<xs:complexType>
        <xs:sequence>
<xs:element name="title"        type="xs:string"></xs:element>
<xs:element name="author"       type="xs:string"></xs:element>
<xs:element name="ISBN"         type="xs:string"></xs:element>
<xs:element name="publisher"  type="xs:string"></xs:element>
<xs:element name="edition"      type="xs:int"></xs:element>
<xs:element name="price"        type="xs:decimal"></xs:element>
        </xs:sequence>
</xs:complexType>



</xs:element>


</xs:schema>
```

Bookstore.xml

```
<bookstore xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="C:\Documents and Settings\Administrator\My
Documents\bookstore.xsd">
	<book>
		<title>web programming</title>
		<author>chrisbates</author>
		<ISBN>123-456-789</ISBN>
		<publisher>wiley</publisher>
		<edition>3</edition>
		<price>350</price>
	</book>
	<book>
		<title>internet worldwideweb</title>
		<author>ditel&amp;ditel</author>
		<ISBN>123-456-781</ISBN>
		<publisher>person</publisher>
		<edition>3</edition>
		<price>450</price>
	</book>
</bookstore>
```

- **Display the XML file as follows.**

PROGRAM:

XML:

```
<?xml version="1.0"?>
```

```
<?xml-stylesheet type="text/xsl" href="bookstore.xsl"?>
```

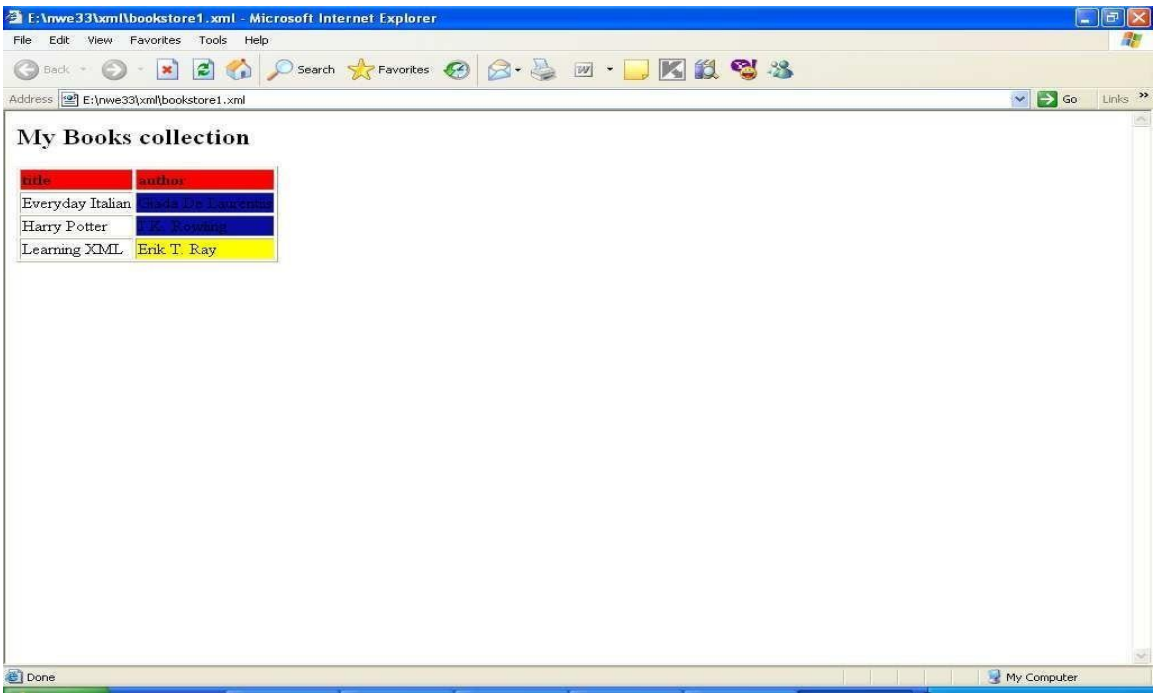
```
<bookstore>
<book>
  <title>Everyday Italian</title>
  <author>Giada De Laurentiis</author>
  <year>2005</year>
  <price>30.00</price>
</book>
<book>
  <title>Harry Potter</title>
  <author>J K. Rowling</author>
  <year>2005</year>
  <price>29.99</price>
</book>
<book>
  <title>Learning XML</title>
  <author>Erik T. Ray</author>
  <year>2003</year>
  <price>39.95</price>
</book>
</bookstore>
```

XSL:

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">

<html>

<body>

<h2> My Books collection</h2>

<table border="1">

<tr bgcolor="red">

<th align="left">title</th>

<th align="left">author</th>

</tr>

<xsl:for-each select="bookstore/book">

<tr>

<td><xsl:value-of select="title"/></td>

<xsl:choose>

<xsl:when test="price &gt; 30">

<td bgcolor="yellow"><xsl:value-of select="author"/></td>

</xsl:when>

<xsl:when test="price &gt; 10">

<td bgcolor="magneta"><xsl:value-of select="author"/></td>

</xsl:when>

<xsl:otherwise>

<td><xsl:value-of select="author"/></td>
```

```
</xsl:otherwise>

</xsl:choose>

</tr>

</xsl:for-each>

</table>

</body>

</html>

</xsl:template>

</xsl:stylesheet>
```

OUTPUT:



RESULT: Thus the XML stylesheets are successfully used to display the content in a table format.

Week-7:

AIM: Install TOMCAT web server and APACHE.

While installation assign port number 8080 to APACHE. Make sure that these ports are available i.e., no other process is using this port.

DESCRIPTION:

- **Set the JAVA_HOME Variable**

You must set the `JAVA_HOME` environment variable to tell Tomcat where to find Java. Failing to properly set this variable prevents Tomcat from handling JSP pages. This variable should list the base JDK installation directory, not the bin subdirectory.

On Windows XP, you could also go to the Start menu, select Control Panel, choose System, click on the Advanced tab, press the Environment Variables button at the bottom, and enter the `JAVA_HOME` variable and value directly as:

Name: JAVA_HOME

Value: C:\jdk

- **Set the CLASSPATH**

Since servlets and JSP are not part of the Java 2 platform, standard edition, you have to identify the servlet classes to the compiler. The server already knows about the servlet classes, but the compiler (i.e., `javac`) you use for development probably doesn't. So, if you don't set your `CLASSPATH`, attempts to compile servlets, tag libraries, or other classes that use the servlet and JSP APIs will fail with error messages about unknown classes.

Name: JAVA_HOME

Value: *install_dir/common/lib/servlet-api.jar*

- **Turn on Servlet Reloading**

The next step is to tell Tomcat to check the modification dates of the class files of requested servlets and reload ones that have changed since they were loaded into the server's memory. This slightly degrades performance in deployment situations, so is turned off by default. However, if you fail to turn it on for your development server, you'll have to restart the server every time you recompile a servlet that has already been loaded into the server's memory.

To turn on servlet reloading, edit *install_dir/conf/server.xml* and add a `DefaultContext` subelement to the main `Host` element and supply `true` for the `reloadable` attribute. For example, in Tomcat 5.0.27, search for this entry:

```
<Host name="localhost" debug="0" appBase="webapps"  ...>
```

and then insert the following immediately below it:

```
<DefaultContext reloadable="true"/>
```

Be sure to make a backup copy of *server.xml* before making the above change.

- **Enable the Invoker Servlet**

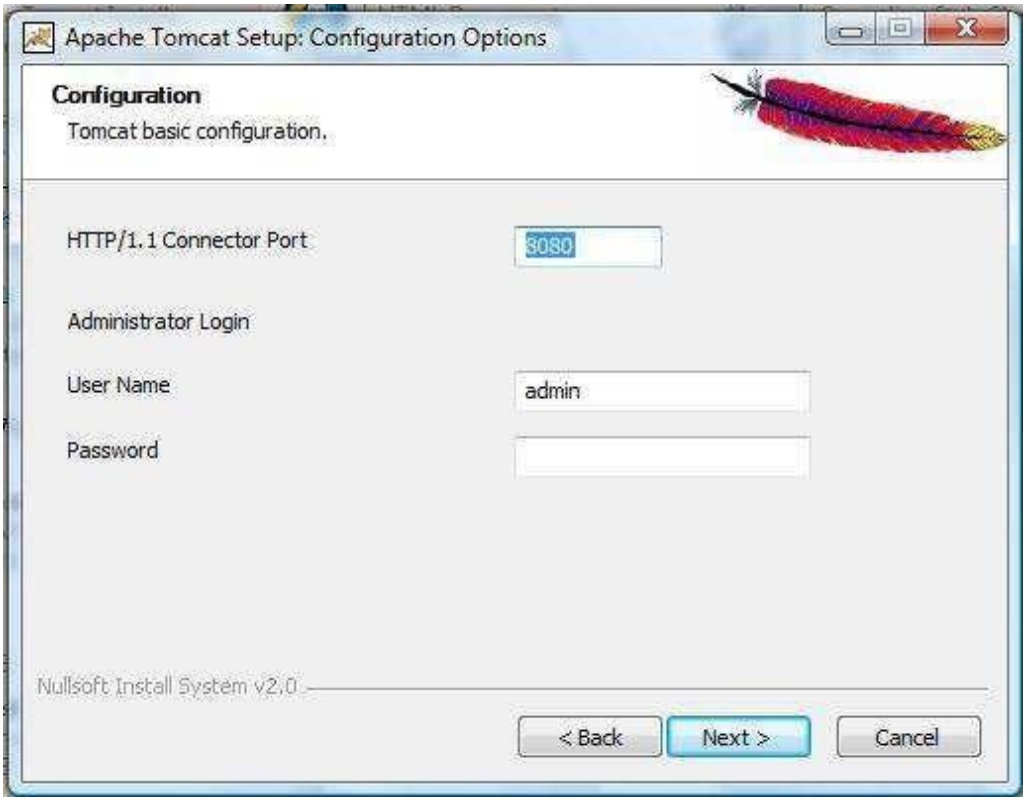
The invoker servlet lets you run servlets without first making changes to your Web application's deployment descriptor. Instead, you just drop your servlet into *WEB-INF/classes* and use the URL *http://host/servlet/ServletName*. The invoker servlet is extremely convenient when you are learning and even when you are doing your initial development.

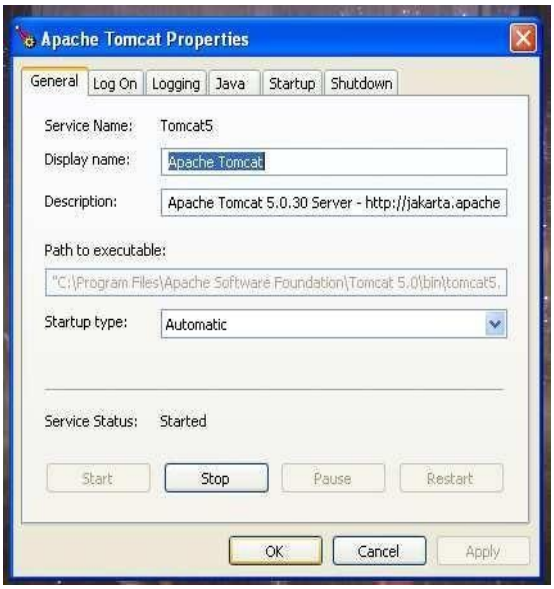
To enable the invoker servlet, uncomment the following `servlet` and `servlet-mapping` elements in *install_dir/conf/web.xml*. Finally, remember to make a backup copy of the original version of this file before you make the changes.

```
<servlet>
  <servlet-name>invoker</servlet-name>
  <servlet-class>
    org.apache.catalina.servlets.InvokerServlet
  </servlet-class>
  ...
</servlet>
...
<servlet-mapping>
```

```
    <servlet-name>invoker</servlet-name>
    <url-pattern>/servlet/*</url-pattern>
  </servlet-mapping>
```

OUTPUT :



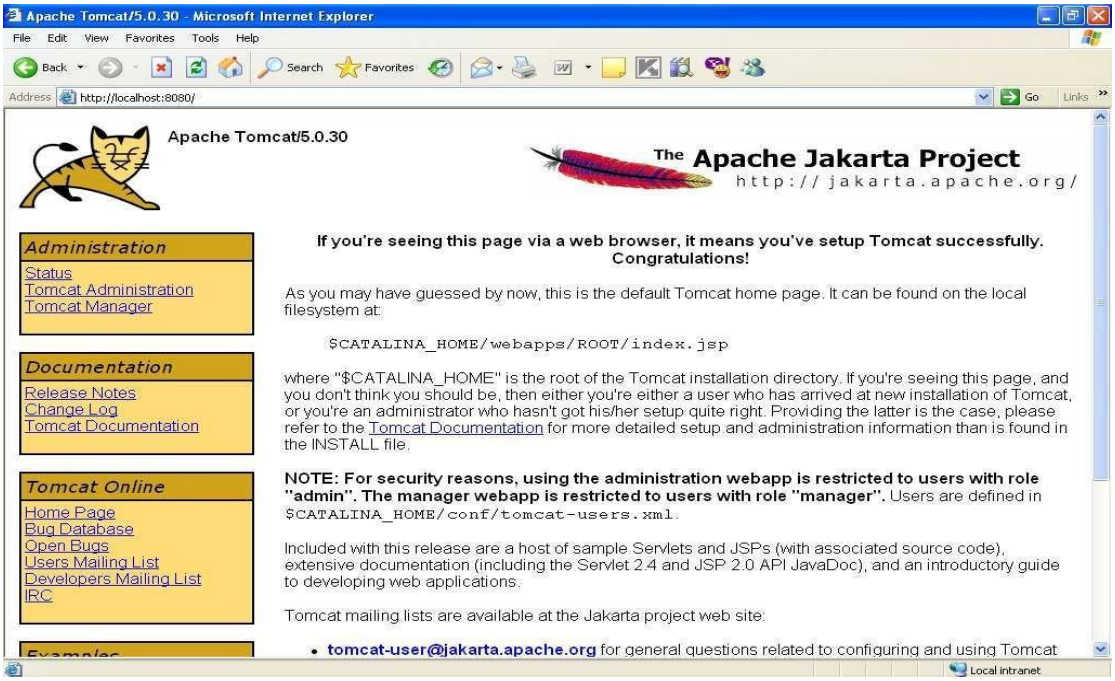


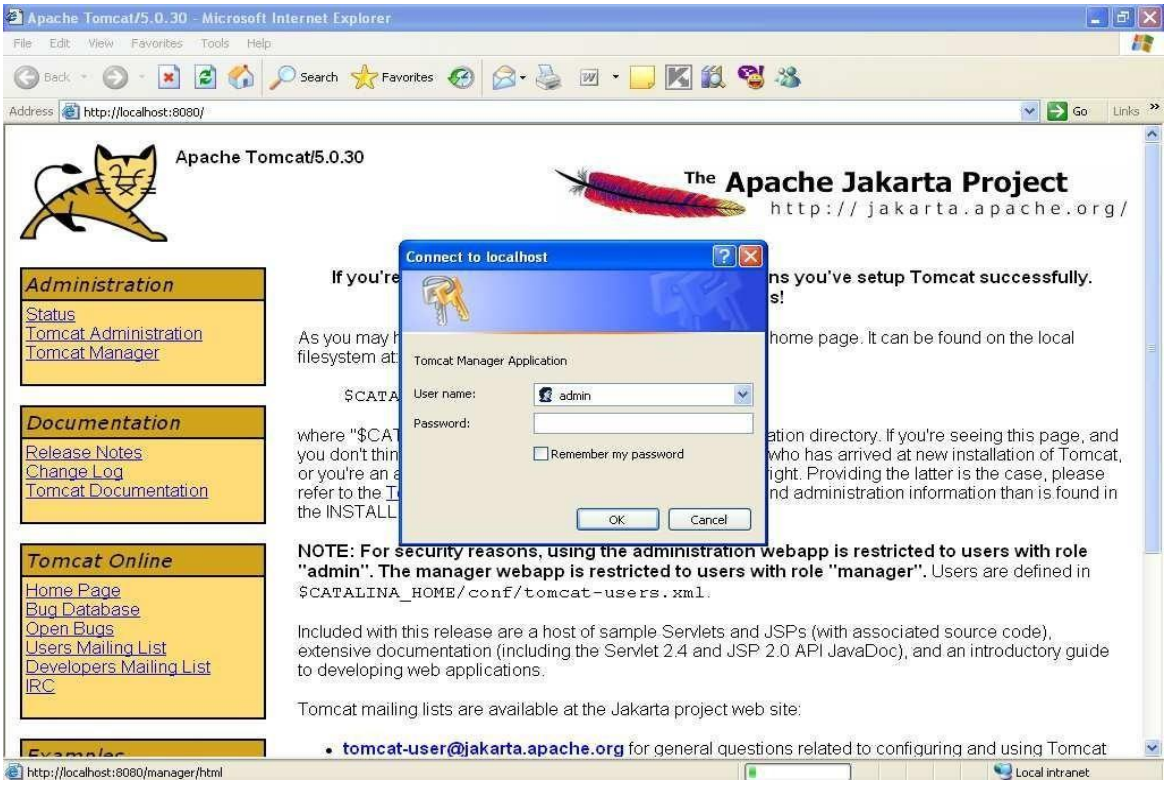
RESULT: Thus TOMCAT web server was installed successfully.

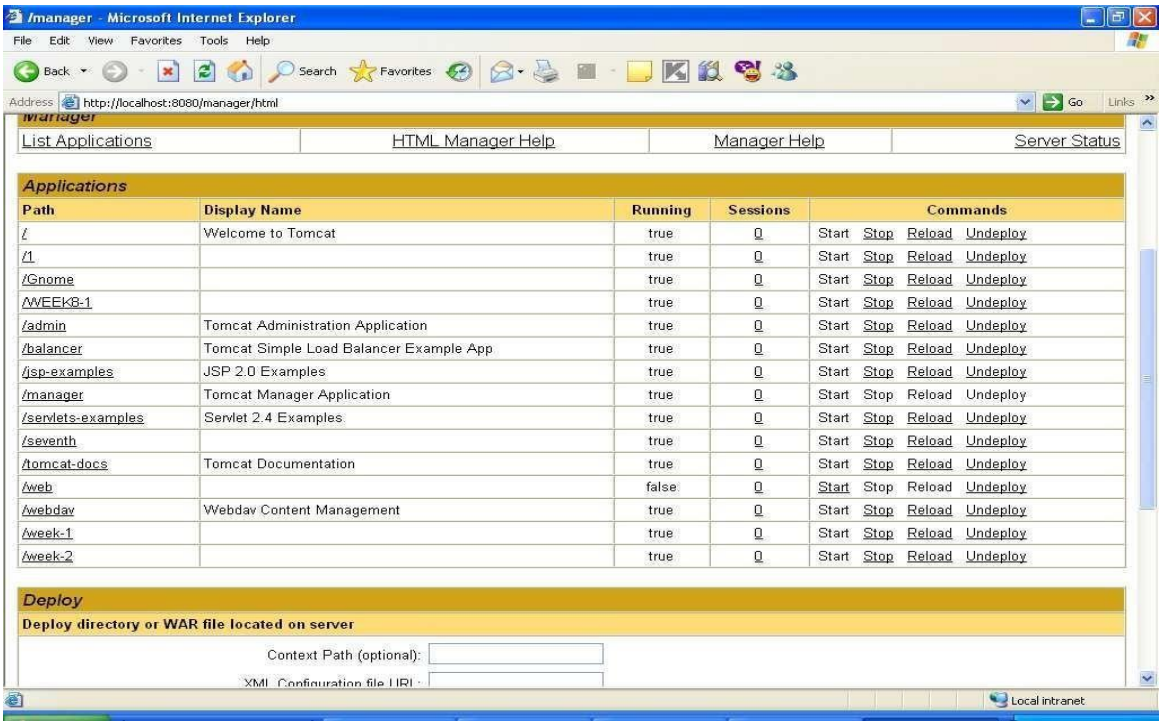
Week-7:

AIM: Access the developed static web pages for books web site, using these servers by putting the web pages developed in week-1 and week-2 in the document root.

OUTPUT







RESULT:

Thus week-1 and week-2 pages are accessed using the TOMCAT web server successfully.

Week-8:

Read the user id and passwords entered in the Login form (week1) and authenticate with the values (user id and passwords) available in the cookies.

If he is a valid user (i.e., user-name and password match) you should welcome him by name (user-name) else you should display "You are not an authenticated user ".

Use init-parameters to do this. Store the user-names and passwords in the webinf.xml and access them in the servlet by using the getInitParameters() method.

home.html:

```
<html>
<head>
  <title>Authentication</title>
</head>
<body>
<form action="ex1">
<label>Username  </label>
<input type="text"size="20" name="user"><br><br>
password<input type="text" size="20" name="pwd"><br><br>
<input  type="submit"  value="submit">
</form>
</body>
</html>
```

Example1.java

```
import javax.servlet.*;
import java.io.*;
public  class  Example1  extends  GenericServlet
{
   private String user1,pwd1,user2,pwd2,user3,pwd3,user4,pwd4,user5,pwd5;
   public void init(ServletConfig sc)
   {
     user1=sc.getInitParameter("username1");
```

```
    pwd1=sc.getInitParameter("password1");

    user2=sc.getInitParameter("username2");
    pwd2=sc.getInitParameter("password2");

            user3=sc.getInitParameter("username3");
    pwd3=sc.getInitParameter("password3");

            user4=sc.getInitParameter("username4");
    pwd4=sc.getInitParameter("password4");
  }
  Public void service(ServletRequest req,ServletResponse res)throws
ServletException,IOException
  {
     res.setContentType("text/html");
     PrintWriter out=res.getWriter();
     user5=req.getParameter("user");
     pwd5=req.getParameter("pwd");

if((user5.equals(user1)&&pwd5.equals(pwd1))||(user5.equals(user2)&&pwd5.equals(pwd2))||(user5
.equals(user3)&&pwd5.equals(pwd3))||(user5.equals(user4)&&pwd5.equals(pwd4)))
       out.println("<p> welcome to"+user5.toUpperCase());
     else
        out.println("You are not authorized user");
  }
}
```

web.xml:

```
<web-app>
<servlet>
<servlet-name>Example</servlet-name>
```

VNR VJIET

Name of the laboratory: _____

Name of the Experiment: _____

_____

Experiment No: 7 Date: _____

User Authentication: Assume four users user1, user2, user3 and user4 having the passwords pwd1, pwd2, pwd3 and pwd4 respectively. Write a servlet for doing the following.

I. Create a Cookie and add these four user id's and passwords to this Cookie.

Html Login form:

```
<html>
<head><title>Input Form</title></head>
<body>
<center>
<h3>Login Page</h3>
<form name="login" method="post" action="http://localhost:8080/Myapp/ck">
<table>
<tr>
<td>Username:</td>
<td><input type="text" name="username"></td>
</tr>
<tr>
<td>Password:</td>
<td><input type="password" name="password"></td>
</tr>
</table>
<input type="submit" value="Enter">
</form>
</center>
</body>
</html>
```

Cookie.java:

```
import java.io.*;
```

```
import java.util.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class cookie1 extends HttpServlet
{
public void init(ServletConfig config) throws ServletException
{
super.init(config);
}
public void doPost(HttpServletRequest req, HttpServletResponse res) throws
IOException,ServletException
{
res.setContentType("text/html");
PrintWriter out = res.getWriter();
Cookie mycookie = new Cookie("null","null");
Enumeration keys;
String key,value;
keys=req.getParameterNames();
while(keys.hasMoreElements())
{
key = (String)keys.nextElement();
value = req.getParameter(key);
mycookie = new Cookie(value,key);
res.addCookie(mycookie);
}
out.println("\nThe cookie is added)");}}
```

Web.xml:

```
<web-app>
	<servlet>
		<servlet-name>servlet1</servlet-name>
		<servlet-class>cookie1</servlet-class>
	</servlet>
	<servlet-mapping>
		<servlet-name>servlet1</servlet-name>
		<url-pattern>/ck</url-pattern>
	</servlet-mapping>
</web-app>
```

II. Read the user id and passwords entered in the Login form (week1) and authenticate with the values (user id and passwords) available in the cookies. If he is a valid user (i.e., user-name and password match) you should welcome him by name(user-name) else you should display “You are not an authenticated user “.

Use init-parameters to do this. Store the user-names and passwords in the web.xml and access them in the servlet by using the getInitParameters() method.

Web.xml:

```
<web-app>
        <servlet-mapping>
                <servlet-name>servlet1</servlet-name>
                <url-pattern>/ck</url-pattern>
        </servlet-mapping>
        <servlet>
                <servlet-name>getc</servlet-name>
                <servlet-class>getcookie</servlet-class>
        </servlet>
        <servlet-mapping>
                <servlet-name>getc</servlet-name>
                <url-pattern>/gc</url-pattern>
        </servlet-mapping>
<servlet>
        <servlet-name>param</servlet-name>
        <servlet-class>initparam</servlet-class>
        <init-param>
                <param-name>username1</param-name>
                <param-value>user1</param-value>
        </init-param>
        <init-param>
                <param-name>password1</param-name>
                <param-value>pwd1</param-value>
        </init-param>
        <init-param>
                <param-name>username2</param-name>
```

```
                <param-value>user2</param-value>
        </init-param>
        <init-param>
                <param-name>password2</param-name>
                <param-value>pwd2</param-value>
        </init-param>
        <init-param>
                <param-name>username3</param-name>
                <param-value>user3</param-value>
        </init-param>
        <init-param>
                <param-name>password3</param-name>
                <param-value>pwd3</param-value>
        </init-param>
        <init-param>
                <param-name>username4</param-name>
                <param-value>user4</param-value>
        </init-param>
        <init-param>
                <param-name>password4</param-name>
                <param-value>pwd4</param-value>
        </init-param>
</servlet>
<servlet-mapping>
        <servlet-name>param</servlet-name>
        <url-pattern>/initparam</url-pattern>
</servlet-mapping>
<servlet>
```

```
        <servlet-name>DBConnection</servlet-name>
        <servlet-class>DBConnection</servlet-class>
</servlet>
<servlet-mapping>
        <servlet-name>DBConnection</servlet-name>
        <url-pattern>/DBConnection</url-pattern>
</servlet-mapping>
<servlet>
        <servlet-name>Insertion</servlet-name>
        <servlet-class>Insertion</servlet-class>
</servlet>
<servlet-mapping>
        <servlet-name>Insertion</servlet-name>
        <url-pattern>/Insertion</url-pattern>
</servlet-mapping>
</web-app>
```

Initparam.java:

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class initparam extends HttpServlet{
public void doGet(HttpServletRequest req, HttpServletResponse res) throws
IOException,ServletException{
res.setContentType("text/html");
PrintWriter out = res.getWriter();
String username1=getServletConfig().getInitParameter("username1");
String  password1=getServletConfig().getInitParameter("password1");
String username2=getServletConfig().getInitParameter("username2");
String  password2=getServletConfig().getInitParameter("password2");
String username3=getServletConfig().getInitParameter("username3");

().getInitParameter("password3"); String
```

```
username4=getServletConfig().getInitParameter("username4"); String
password4=getServletConfig().getInitParameter("password4"); String un=
req.getParameter("username"); String p= req.getParameter("password");
if(((username1.equals(un))&&(password1.equals(p)))||
((username2.equals(un))&&(password2.equals(p))||
((username3.equals(un))&&(password3.equals(p)))||
((username4.equals(un))&&(password4.equals(p))))) {
out.println("<b>WelCome to "+un+"kavya(20071A0533)
</b>");
}
else{
out.println("<b>Unathorised User @ user(20071A05XX)
</b>");
}}}
```

Login2.html:

```
<html>
<head>
<title>Input Form</title>
</head>
<body>
<center>
<h3>Login Page</h3>
<form name="login" method="get" action="http://localhost:8080/Myapp/initparam">
<table>
<tr>
```

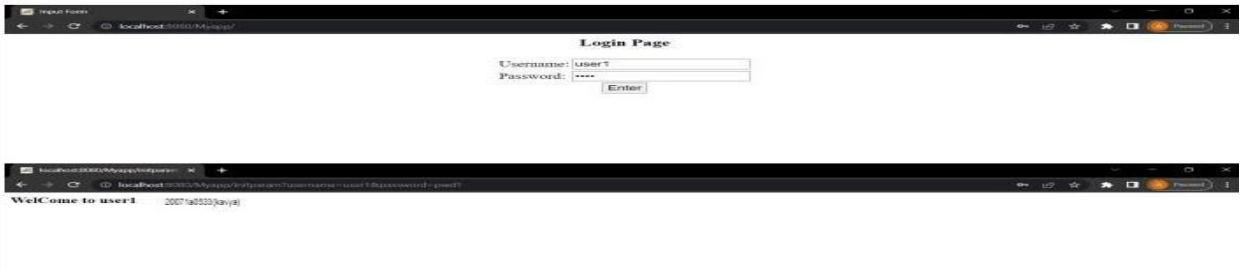
```
<td>Username:</td>
<td><input type="text" name="username"></td>
</tr>
<tr>
<td>Password:</td>
<td><input type="password" name="password"></td>
</tr>
</table>
<input type="submit" value="Enter">
</form>
</center>
</body>
</html>
```

Getcookie.java:

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class getcookie extends HttpServlet{
public void doGet(HttpServletRequest req, HttpServletResponse res) throws
IOException,ServletException{
res.setContentType("text/html");
PrintWriter out = res.getWriter();
Cookie[] mycookie =req.getCookies();
String[] name = new String[10];
String[] value = new String[10];
int i=0;
int n=mycookie.length;
out.println("The number of cookies:"+n);
for(i=0;i<n;i++){
name[i]=mycookie[i].getName();
if(name[i].equals("pwd1")){
```

```
if(name[i+1].equals("user1")){
out.println("WelCome"+name[i+1]);}
else{
out.println("Unauthorised User");}}
else if(name[i].equals("pwd2")){
if(name[i+1].equals("user2")){
out.println("WelCome"+name[i+1]);}
else{
out.println("Unauthorised User");}}
else if(name[i].equals("pwd3")){
if(name[i+1].equals("user3")){
out.println("WelCome"+name[i+1]);}
else{
out.println("Unauthorised User");}}
else if(name[i].equals("pwd4")){
if(name[i+1].equals("user4")){
out.println("WelCome"+name[i+1]);}
else{
out.println("Unauthorised User");}}}
catch(Exception e){
out.println("Invalid username/password");}}}}
```

Output:



VNR VJIET

Name of the laboratory:

Name of the Experiment:

Experiment No: 8 Date:

WEEK 8

CREATE TABLE Students (student-id, student-name, course, branch, year, studentemailid);
Faculty(faculty_id,facultyname,studentid,studentname,course,year,facultyemailid)

I. Write a servlet program to retrieve the names of the student who study a particular course.

DBConnection.java:

```
import java.io.*;
import java.util.*;
import javax.sql.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

public class DBConnection extends HttpServlet {
public void service(HttpServletRequest request,
HttpServletResponse response)
throws IOException, ServletException{
response.setContentType("text/html");
String s1= request.getParameter("course");
PrintWriter out = response.getWriter();
out.println("<html>");
out.println("<body>");
// connecting to database
```

```
ResultSet rs = null;
try {
Class.forName("com.mysql.jdbc.Driver");
con =DriverManager.getConnection("jdbc:mysql://localhost:3306/wt_db","root","root");
stmt = con.createStatement();
rs = stmt.executeQuery("SELECT * FROM students");
// displaying records
out.println("<center><h1>Students having course "+s1+"</h1>");
out.print("<table><tr><th>Student Name</th></tr>");
while(rs.next()){
String s2=rs.getObject(3).toString();
if(s1.equals(s2)){
out.print("<tr><td>");
out.print(rs.getObject(2).toString());
out.print("</td></tr>");}
out.print("<br>");}

} catch (SQLException e) {
throw new ServletException("Servlet Could not display records.", e);
} catch (ClassNotFoundException e) {
throw new ServletException("JDBC Driver not found.", e);
} finally {
try {
if(rs != null) {
rs.close();
rs = null;}
if(stmt != null) {
stmt.close();
stmt = null;}
if(con != null) {
con.close();

con = null;}
```

```
} catch (SQLException e) {}}
out.close();}}
```

Html file:

```
<html>
<body>
        <center>
        <form name="Form1" action="http://localhost:8080/Myapp/DBConnection">
        <B>Enter the course of student to selcect</B>
        <input type=textbox name="course" size=25 value="">
        <input type=submit value="Submit">
        </form>
</body>
</html>
```

Web.xml:

```
<servlet>
        <servlet-name>DBConnection</servlet-name>
        <servlet-class>DBConnection</servlet-class>
</servlet>
<servlet-mapping>
        <servlet-name>DBConnection</servlet-name>
        <url-pattern>/DBConnection</url-pattern>
</servlet-mapping>
```

Output:

Entering course name

Write a servlet program to retrieve the names of the student who study a particular course and year.

DBConnection2.java:

```
import java.io.*;

import java.util.*;

import javax.sql.*;

import javax.servlet.*;

import javax.servlet.http.*;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

public class DBConnection2 extends HttpServlet {

  public void service(HttpServletRequest request,HttpServletResponse response)throws IOException,
ServletException{

  response.setContentType("text/html");

  String s1= request.getParameter("course");
```

```
String s2= request.getParameter("year");
PrintWriter out = response.getWriter();
out.println("<html>");
out.println("<body>");
   // connecting to database
Connection con = null;
Statement stmt = null;
ResultSet rs = null;
try {
Class.forName("com.mysql.jdbc.Driver");
con =DriverManager.getConnection("jdbc:mysql://localhost:3306/wt_db","root","root");
stmt = con.createStatement();
rs = stmt.executeQuery("SELECT * FROM students");
// displaying records
out.println("<center><h1>Students having course "+s1+" and studying in the year "+s2+" are :</h1>");
out.print("<table><tr><th>Student Name</th></tr>");
while(rs.next()){
        String s3=rs.getObject(3).toString();
        String s4=rs.getObject(5).toString();
        if(s1.equals(s3)  &&  s2.equals(s4)){
        out.print("<tr><td>");
   out.print(rs.getObject(2).toString());
        out.print("</td></tr>");}
        out.print("<br>");}

} catch (SQLException e) {
throw new ServletException("Servlet Could not display records.", e);
} catch (ClassNotFoundException e) {
```

```
throw new ServletException("JDBC Driver not found.", e);
} finally {
try {
if(rs != null) {
rs.close();
rs = null;}
if(stmt != null) {
stmt.close();
stmt = null;}
if(con != null) {
con.close();
con = null;}
} catch (SQLException e) {}}
out.close();}}
```

Html file:

```
<html>
<body>
        <center>
        <form name="Form1" action="http://localhost:8080/Myapp/DBConnection2">
        <B>Enter the course of student to select</B>
        <input type=textbox name="course" size=25 value=""><br><br>
        <B>Enter the year of student to select</B>
        <input type=textbox name="year" size=25 value=""><br><br>
        <input type=submit value="Submit">
        </form>
</body>
</html>
```

Web.xml:

```
<servlet>
        <servlet-name>DBConnection2</servlet-name>
```

```
        <servlet-class>DBConnection2</servlet-class>
</servlet>
<servlet-mapping>
        <servlet-name>DBConnection2</servlet-name>
        <url-pattern>/DBConnection2</url-pattern>
</servlet-mapping>
```

Output:

Entering course and year details

Students with course wt and year 3

II. Write a servlet program to retrieve the names of the student,course,branch,year who study under a particular faculty.(Note the Input is provided from the HTML form).

DBConnection3.java:

```
import java.io.*;
```

```
import java.util.*;
import javax.sql.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

public class DBConnection3 extends HttpServlet {
  public void service(HttpServletRequest request,
  HttpServletResponse response)
  throws IOException, ServletException{
response.setContentType("text/html");
String s1= request.getParameter("faculty");
PrintWriter out = response.getWriter();
out.println("<html>");
out.println("<body>");
      // connecting to database
  Connection con = null;
  Statement stmt = null;
  ResultSet rs = null;
  try {
  Class.forName("com.mysql.jdbc.Driver");
 con =DriverManager.getConnection("jdbc:mysql://localhost:3306/wt_db","root","root");
  stmt = con.createStatement();
  rs = stmt.executeQuery("SELECT studentname, course, branch, year " +
                        "FROM students " +
                        "WHERE studentid IN " +
                        "(SELECT studentid FROM faculty WHERE facultyname = '" + s1 + "')")
```

```
  // displaying records
out.println("<center><h1>Students studying under the faculty "+s1+" are: </h1>");
out.print("<table border='1'><tr><th>Student
Name</th><th>Course</th><th>Branch</th><th>Year</th></tr>");
  while(rs.next()){
out.print("<tr><td>");
out.print(rs.getObject(1).toString());
out.print("</td><td>");
out.print(rs.getObject(2).toString());
out.print("</td><td>");
out.print(rs.getObject(3).toString());
out.print("</td><td>");
out.print(rs.getObject(4).toString());
out.print("</td></tr>");
   out.print("<br>");}
  } catch (SQLException e) {
 throw new ServletException("Servlet Could not display records.", e);
  } catch (ClassNotFoundException e) {
  throw new ServletException("JDBC Driver not found.", e);
  } finally {
  try {
  if(rs != null) {
  rs.close();
  rs = null;}
  if(stmt != null) {
  stmt.close();
  stmt = null;}
```

```
 if(con != null) {
 con.close();
 con = null;}
 } catch (SQLException e) {}}
 out.close(); }}
```

HTML file:

```
<html>
<body>
        <center>
        <form name="Form1" action="http://localhost:8080/Myapp/DBConnection3">
        <B>Enter the name of the faculty</B>
        <input type=textbox name="faculty" size=25 value="">
        <input type=submit value="Submit">
        </form>
</body>
</html>
```

Web.xml:

```
<servlet>
        <servlet-name>DBConnection3</servlet-name>
        <servlet-class>DBConnection3</servlet-class>
</servlet>
<servlet-mapping>
        <servlet-name>DBConnection3</servlet-name>
        <url-pattern>/DBConnection3</url-pattern>
</servlet-mapping>
```

VNR VJIET

Name of the laboratory: _____

Name of the Experiment: _____

_____

Experiment No:___9_____Date: _____

WEEK 9

Write a servlet program which does the following job:

Insert the details of the 3 or 4 users who register with the web site by using registration form.

Registration.html:

```
<!DOCTYPE html>
<html>
<head>
<link rel="stylesheet" href="registration.css" />
</head>
<body>
	<h1 align="center">Registration</h2>
	<form name="myform" action="http://localhost:8080/Myapp/Insertion1" method="post"
onSubmit="return validation()">
		<label for="name">Name : </label>
		<input type="text" id="name" name="name" minlength="6" required><br><br>
		<label for="pass">Password : </label>
		<input type="password" id="pass" name="password" minlength="6" required><br><br>
		<label for="email">Email : </label>
		<input type="text" id="email" name="email" required><br><br>
		<label for="phoneno">Phone No : </label>
		<input type="text" id="phoneno" name="phoneno" required><br><br>
		<label>Sex : </label><br>
		<input type="radio" id="male" name="sex" value="male">
		<label for="male">Male</label><br>
		<input type="radio" id="female" name="sex" value="female">
		<label for="female">Female</label><br>
		<input type="radio" id="others" name="sex" value="others">
```

```
<label for="others">Others</label><br><br>
<label for="dob">Date of Birth:</label>
<select name="day" id="day">
  <option value="">Day</option>
  <option value="01">1</option>
  <option value="02">2</option>
  <option value="03">3</option>
  <option value="21">21</option>
  <option value="28">28</option>
  <option value="29">29</option>
  <option value="30">30</option>
  <option value="31">31</option>
</select>
<select name="month" id="month">
  <option value="">Month</option>
  <option value="01">January</option>
  <option value="02">February</option>
  <option value="03">March</option>
  <option value="04">April</option>
  <option value="05">May</option>
  <option value="06">June</option>
</select>
<select name="year" id="year">
  <option value="">Year</option>
  <option value="2000">2000</option>
  <option value="2001">2001</option>
  <option value="2002">2002</option>
  <option value="2003">2003</option>
</select><br><br>
<label for="languages">Languages known : </label><br>
<input type="checkbox" id="lan1" name="lan1" value="English">
```

```
        <label for="lan1">English</label><br>
        <input type="checkbox" id="lan2" name="lan2" value="Hindi">
        <label for="lan2">Hindi</label><br>
        <input type="checkbox" id="lan3" name="lan3" value="Telugu">
        <label for="lan3">Telugu</label><br>
        <input type="checkbox" id="lan4" name="lan4" value="Tamil">
        <label for="lan4">Tamil</label><br><br>
        <label for="address">Address : </label><br>
        <textarea id="address" name="address" rows="4" cols="50">
        </textarea><br><br>
        <input type="submit" value="Submit">   
        <input type="submit" value="Reset">
</form>
<script type="text/javascript">
        function validation(){
                let uname=document.myform.name.value;
                let unamereq=/^[A-Za-z]+$/;
                if(uname.match(unamereq)){
                        let email=document.myform.email.value;
                        let emailreq=/^\w+([\.-]?\w+)*@\w+([\.-]?\w+)*(\.\w{2,3})+$/;
                        if(email.match(emailreq)){
                                let pno=document.myform.phoneno.value;
                                let pnoreq=/^\d{10}$/;
                                if(pno.match(pnoreq)){
                                        return true;}
                                else{ window.alert("ONLY Numbers");return false;}}
                        else{ window.alert("ONLY Email pattern");return false;}}
                else{ window.alert("ONLY alphabets");return false;}}
```

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

Insertion1.java:

```
import java.io.*;
import java.sql.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class Insertion1 extends HttpServlet{
  public void init(ServletConfig config) throws ServletException{
  super.init(config);}
  public void doPost(HttpServletRequest req, HttpServletResponse res) throws ServletException,IOException{
  ResultSet rs;
  res.setContentType("text/html");
  PrintWriter out = res.getWriter();
  //get the variables entered in the form
  String name = req.getParameter("name");
  String password = req.getParameter("password");
  String email = req.getParameter("email");
  String phoneno = req.getParameter("phoneno");
  String sex = req.getParameter("sex");
  String dob = req.getParameter("day")+req.getParameter("month")+req.getParameter("year");
  String languages="";
  if(req.getParameter("lan1")!=null)
        languages += req.getParameter("lan1");
  if(req.getParameter("lan2")!=null)
        languages = languages + "," + req.getParameter("lan2");
  if(req.getParameter("lan3")!=null)
        languages = languages + "," + req.getParameter("lan3");
  if(req.getParameter("lan4")!=null)
        languages = languages + "," + req.getParameter("lan4");
  String address = req.getParameter("address");
```

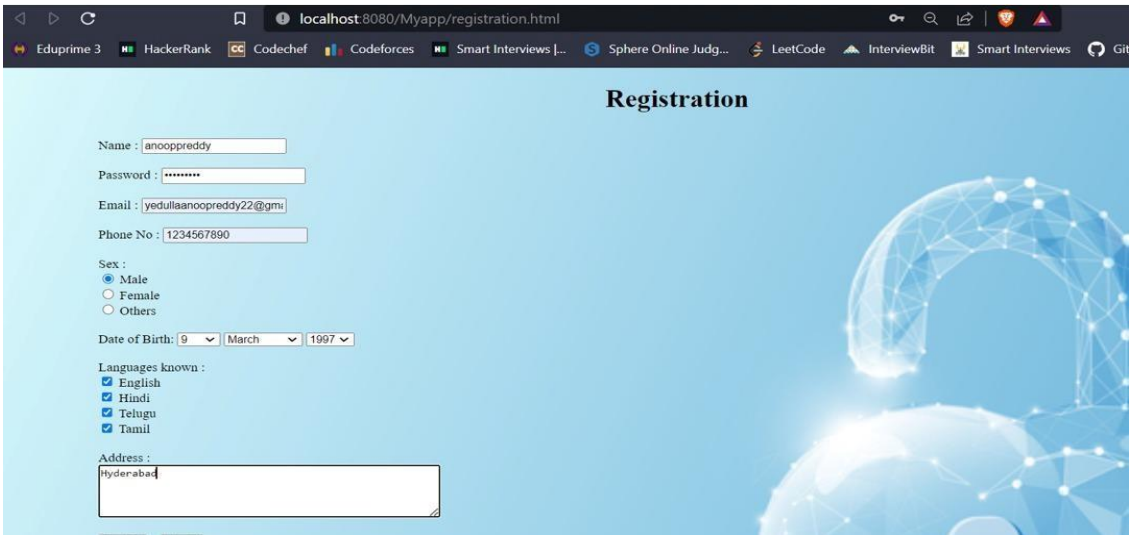
```
Connection conn=null;
 try {
 // Load the database driver
Class.forName("com.mysql.jdbc.Driver");
 // Get a Connection to the database
conn= DriverManager.getConnection("jdbc:mysql://localhost:3306/wt_db", "root", "root");
 //Add the data into the database
 String sql = "insert into users values (?,?,?,?,?,?,?,?)";
 PreparedStatement pst = conn.prepareStatement(sql);
 pst.setString(1, name);
 pst.setString(2, password);
 pst.setString(3, email);
 pst.setString(4, phoneno);
 pst.setString(5, sex);
 pst.setString(6, dob);
 pst.setString(7, languages);
 pst.setString(8, address);
 int numRowsChanged = pst.executeUpdate();
 // show that the new account has been created
 out.println(" Hello : ");
 out.println(" '"+name+"'");
 res.sendRedirect("login.html");
 pst.close();}
 catch(ClassNotFoundException e){
 out.println("Couldn't load database driver: "
 + e.getMessage());}
 catch(SQLException e){
 out.println("SQLException caught: "
```

```
+ e.getMessage());}
catch (Exception e){
out.println(e);}
finally {
// Always close the database connection.
try {
if (conn != null) conn.close();}
catch (SQLException ignored){
out.println(ignored);}}}}
```

web.xml:

```
<servlet>
<servlet-name>Insertion1</servlet-name>
<servlet-class>Insertion1</servlet-class>
</servlet>
<servlet-mapping>
<servlet-name>Insertion1</servlet-name>
<url-pattern>/Insertion1</url-pattern>
</servlet-mapping>
```

Output:



Registering a user

Authenticate the user when he submits the login form using the user name and password from the database.

Login.html:

```
<!DOCTYPE html>

<html>

<head>

        <link rel="stylesheet" href="login.css" />

</head>

<body>

<h1 class="heading">Login</h1>

<img src="https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcT3Mclb0NdAfReSwkqWDtxIh2Oc4vEyPMYzeg&usqp=CAU" />

<form class="loginform" align="center" name="myform" action="http://localhost:8080/Myapp/AuthLogin" method="get" onSubmit="return validation()">

  <label for="fname">Username:</label><br>

  <input type="text" id="fname" name="fname" required minlength="6"><br>
```

```
  <label for="pass">Password:</label><br>
  <input type="password" id="pass" name="pass" required minlength="6"><br><br>
  <input type="submit" value="Submit">
  <input type="submit" value="Reset">
</form>


<script type="text/javascript">
function validation(){
                        let uname=document.myform.fname.value;
                        let unamereq=/^[A-Za-z]+$/;
                        if(uname.match(unamereq)){
                                return true;
                        }
                        else{ window.alert("ONLY alphabets");return false;}


                }
</script>
</body>
</html>
```

AuthLogin.java:

```
import java.io.*;
import java.util.*;
import javax.sql.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
public class AuthLogin extends HttpServlet{
```

```
public void doGet(HttpServletRequest req, HttpServletResponse res) throws IOException,ServletException{
  res.setContentType("text/html");
  PrintWriter out = res.getWriter();
  //get the variables entered in the form
  String fname = req.getParameter("fname");
  String pass = req.getParameter("pass");
  Connection conn=null;
  Statement stmt = null;
  try {
  // Load the database driver
  Class.forName("com.mysql.jdbc.Driver");
  // Get a Connection to the database
  conn= DriverManager.getConnection("jdbc:mysql://localhost:3306/wt_db", "root", "root");
  stmt = conn.createStatement();
  String sql = "SELECT * FROM users WHERE name='" + fname + "' AND password='" + pass + "'";
     ResultSet rs = stmt.executeQuery(sql);
     if (rs.next()) {
       // User is authenticated, redirect to home page
            out.println(" Hello : ");
            out.println(" '"+fname+"'");
            res.sendRedirect("home.html");
     } else {
       // User is not authenticated, display error message
       out.println("Invalid username or password");}}
  catch(ClassNotFoundException e){
  out.println("Couldn't load database driver: "
  + e.getMessage());}
  catch(SQLException e){
```

```
  out.println("SQLException caught: "
  + e.getMessage());}
  catch (Exception e){
  out.println(e);}
  finally {
  // Always close the database connection.
  try {
  if (conn != null) conn.close();}
catch (SQLException ignored){
out.println(ignored);}}}}
```

Web.xml:

```
<servlet>
        <servlet-name>AuthLogin</servlet-name>
        <servlet-class>AuthLogin</servlet-class>
</servlet>
<servlet-mapping>
        <servlet-name>AuthLogin</servlet-name>
        <url-pattern>/AuthLogin</url-pattern>
</servlet-mapping>
```

Output:

Authenticating user

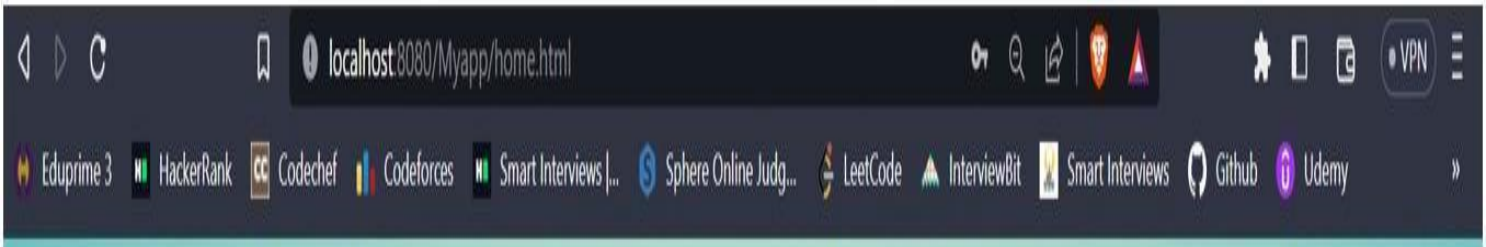
VNR VJIET

Name of the laboratory:

_____

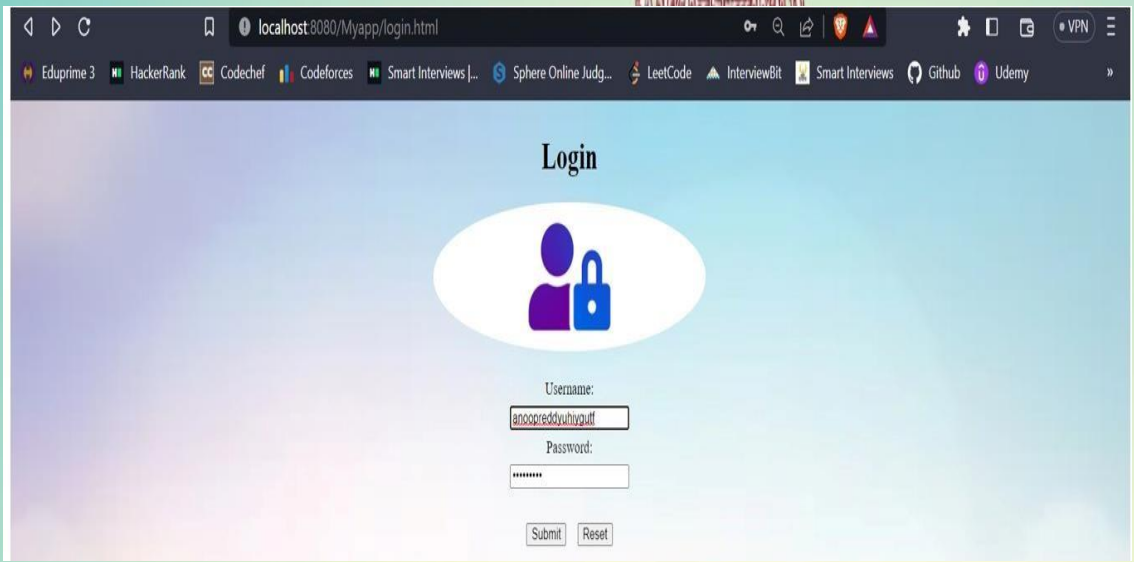
Name of the Experiment: _____

Experiment No:_____Date: _____



VNR VJIET

Redirecting a valid user to Home page



Authenticating a user

Vallurupalli Nageswara Rao Vignana Jyothi Institute of Engineering and Technology (VNRVJIET) is a private engineering college in Hyderabad, India recognized by All India Council for Technical Education(AICTE) and affiliated to the Jawaharlal Nehru Technological University, Hyderabad. Undergraduate programs- CE, EEE, ME, ECE, CSE, EIE and IT in the institute are accredited by the National Board of Accreditation (NBA) New Delhi, since 2008. The institute has Autonomous Status till 2028-2029 A.Y. granted by UGC.

Copyright @ kavya-20071a0533

VNR VJIET

Name of the laboratory: _____

Name of the Experiment: _____

_____

Experiment No: 10 Date: _____

WEEK 10

Create a table with attributes (emp_id, emp_name, job_name, joining_date, salary, department)

I. Write a JSP program to display the names of the employee, empid, salary and department joined after the given input joining date(The input joining date must be given from the HTML form)

Html file:

```
<html>
<head>
        <title>Employee Details</title>
</head>
<body>
        <center><h1>Employee Details</h1>
        <form method="post" action="http://localhost:8080/Myapp/week101.jsp">
                <label>Joining Date (YYYY-MM-DD): </label>
                <input type="date" name="joining_date"><br><br>
                <input type="submit" value="Submit"
        </form>
</body>
</html>
```

Jsp file:

```
<html>
<head>
  <title>Users list</title>
</head>
<body>
  <%@ page import = "java.sql.*" %>
  <%
        Class.forName("com.mysql.jdbc.Driver");
```

```
    Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/wt_db","root","root");
        String joiningDate = request.getParameter("joining_date");
    String query = "SELECT emp_name, emp_id, salary, department FROM employees WHERE joining_date
= ?";
        PreparedStatement ps = conn.prepareStatement(query);
        ps.setString(1, joiningDate);
        ResultSet rs = ps.executeQuery();
        if(rs.next()) {
%>
        <center><table border=1 cellpadding=5>
                <tr>
                        <th>Name</th>
                        <th>Employee ID</th>
                        <th>Salary</th>
                        <th>Department</th>
                </tr>
                <%
                        do {
                                String empName = rs.getString("emp_name");
                                int empId = rs.getInt("emp_id");
                                int salary = rs.getInt("salary");
                                String department = rs.getString("department");
                %>
                        <tr>
                                <td><%= empName %></td>
                                <td><%= empId %></td>
                                <td><%= salary %></td>
                                <td><%= department %></td>
                        </tr>
                <%
                        } while(rs.next());
                %>
```

```
                </table>
        <%
                } else {
                        out.println("No results found");}
                rs.close();
                ps.close();
                conn.close();
        %><h3>Copyright @ Kavya-20071A0533</h3></center>
</body>
</html>
```

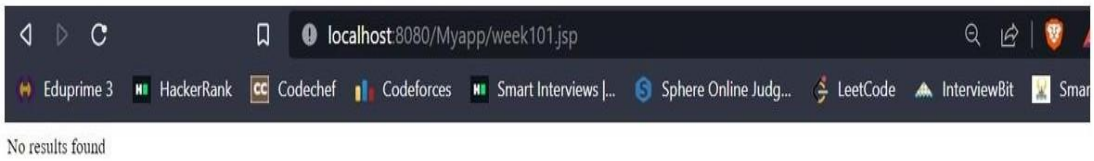
Output:

Checking for Employees joined on that date

Name	Employee ID	Salary	Department
ravi	1	20000	cse
Kiran	2	30000	ece

Employee details

Checking for Employees joined on that date



Employees List

II. Write a JSP program to display the names of the employee with empid, salary, department and the number of days worked from their joining date to till date.

<u>Html file:</u>

```
<html>
<head>
        <title>Employee Details</title>
</head>
<body>
        <center><h1>Employee Details</h1>
        <form method="post" action="http://localhost:8080/Myapp/week101.jsp">
                <label>Joining Date (YYYY-MM-DD): </label>
                <input type="date" name="joining_date"><br><br>
```

```
                <input type="submit" value="Submit">
        </form>
</body>
</html>
```

Jsp file:

```
<html>
<head>
  <title>Users list</title>
</head>
<body>
  <%@ page import = "java.sql.*,java.time.LocalDate, java.time.temporal.ChronoUnit" %>
  <%
        Class.forName("com.mysql.jdbc.Driver");
    Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/wt_db","root","root");
        String joiningDate = request.getParameter("joining_date");
    String query = "SELECT emp_name, emp_id, salary, department, joining_date FROM employees WHERE
joining_date = ?";
        PreparedStatement ps = conn.prepareStatement(query);
        ps.setString(1, joiningDate);
        ResultSet rs = ps.executeQuery();
        if(rs.next()) {
%>
        <center><h1>Employee Details</h1>
                <table border=1 cellpadding=5>
                <tr>
                        <th>Name</th>
                        <th>Employee ID</th>
```

```
                <th>Salary</th>
                <th>Department</th>
                <th>Days Worked</th>
        </tr>
        <%
                do {
                        String empName = rs.getString("emp_name");
                        int empId = rs.getInt("emp_id");
                        int salary = rs.getInt("salary");
                        String department = rs.getString("department");
        %>
                <tr>
                        <td><%= empName %></td>
                        <td><%= empId %></td>
                        <td><%= salary %></td>
                        <td><%= department %></td>
                        <td>
                        <% LocalDate joining_date = rs.getDate("joining_date").toLocalDate();
                                LocalDate today = LocalDate.now();
                                long days_worked = ChronoUnit.DAYS.between(joining_date, today);
                          out.println(days_worked);%></td></tr><%} while(rs.next());
        %></table><%} else {
                        out.println("No results found");}
                rs.close();
                ps.close();
                conn.close();
        %>
</body>
</html>
```

Output:

Employee Details

Name	Employee ID	Salary	Department	Days Worked
ravi	1	20000	cse	862
Kiran	2	30000	ece	862

Employee details

VNR VJIET

Name of the laboratory: _____

Name of the Experiment: _____

_____

Experiment No:____11_____Date: _____

WEEK 11

Create a table book (book_id, booktitle, author, price)

I. Write a JSP program to insert the data to the mysql database from the HTML form.

Html file:

```
<html>
<body>
<center>
<form name="Form1" method="get" action="http://localhost:8080/Myapp/week111.jsp">
<h1>Registration form</h1>
Book ID: <input type="text" name="bookId"><br><br>
Book Title:<input type="text" name="bookname"><br><br>
Author:   <input type="text" name="author"><br><br>
Price: <input type="text" name="price"><br><br>
<input type=submit value="submit">
</form><br><br>
</center>
</body>
</html>
```

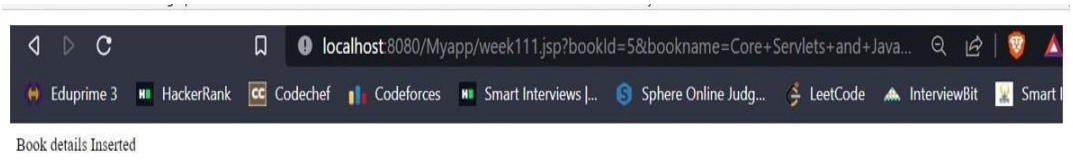
Jsp file:

```
<%@ page import = "java.sql.*" %>
<%
  String bid = request.getParameter("bookId");
  String bname = request.getParameter("bookname");
  String author =request.getParameter("author");
  String price = request.getParameter("price");
  Class.forName("com.mysql.jdbc.Driver");
  Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/wt_db","root","root");
```

```
String sql = "insert into book values (?,?,?,?)";
PreparedStatement pst = conn.prepareStatement(sql);
pst.setString(1, bid);
pst.setString(2, bname);
pst.setString(3,  author);
pst.setString(4, price);
int numRows= pst.executeUpdate();
if(numRows==0){out.print("Unable to Insert");}
else{out.print("Book details Inserted");}
%>
<%
 pst.close();
 conn.close();
%>
```

Output:

Entering Book Details



Book details stored into database

II. Write a JSP program to retrieve the data from the table book given the name of the author. If author is not found, your program must display no author is found.

Html file:

```
<html>
<head>
        <title>Book Details</title>
</head>
<body>
        <center><h1>Book Details</h1>
        <form method="post" action="http://localhost:8080/Myapp/week112.jsp">
                <label>Author Name: </label>
                <input type="text" name="author"><br><br>
                <input type="submit" value="Submit">
        </form>
</body>
</html>
```

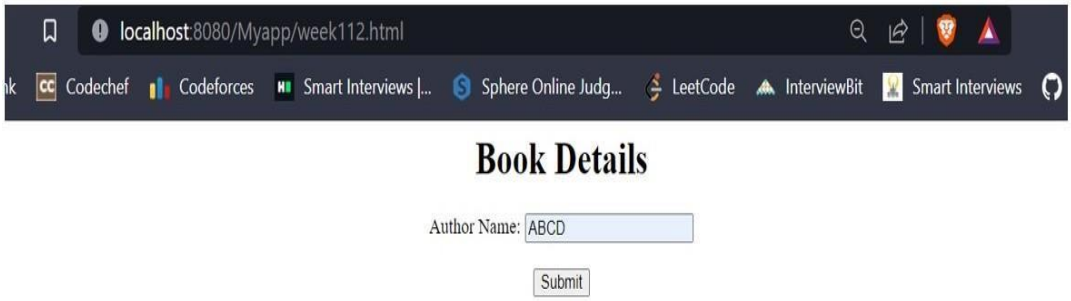
Jsp file:

```
<html>
<head>
  <title>Books list</title>
</head>
<body>


  <%@ page import = "java.sql.*,java.time.LocalDate, java.time.temporal.ChronoUnit" %>
  <%
        Class.forName("com.mysql.jdbc.Driver");
     Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/wt_db","root","root");
```

```
        String author = request.getParameter("author");
    String query = "SELECT book_id, booktitle, author, price FROM book WHERE author = ?";
        PreparedStatement ps = conn.prepareStatement(query);
        ps.setString(1, author);
        ResultSet rs = ps.executeQuery();
        if(rs.next()) {
%>
        <center><h1>Book Details of Author <%=author%> are: </h1>
                <table border=1 cellpadding=5>
                <tr>
                        <th>Book ID</th>
                        <th>Name of the Book</th>
                        <th>Price</th>
                </tr>
                <%
                        do {
                                String bookName = rs.getString("booktitle");
                                int bookId = rs.getInt("book_id");
                                int price = rs.getInt("price");
                %>
                        <tr>
                                <td><%= bookId %></td>
                                <td><%= bookName %></td>
                                <td><%= price %></td>
                        </tr>
                <%
                        } while(rs.next());
                %>
                        </table>
                <%
                        } else {
```

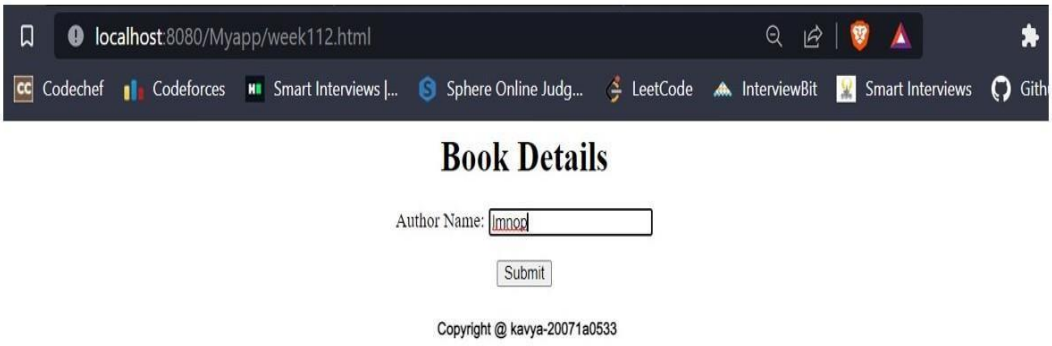
```
                    out.println("No author is found");
                }
                rs.close();
                ps.close();
                conn.close();
            %><h3>Copyright @ Kavya-20071A0533</h3></center>
</body>
</html>
```

Output:

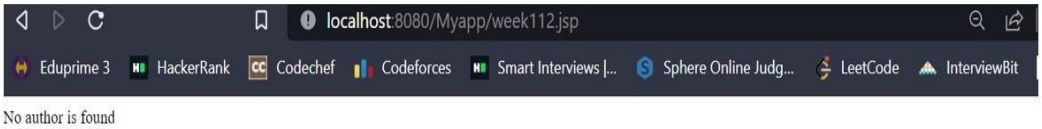


Entering Author name

Book Details



Entering Author name



No author found

III. Write a JSP program to retrieve different authors of the same book given the title of the book in the HTML form.

Html file:

```
<html>
<head>
        <title>Book Details</title>
</head>
<body>
        <center><h1>Book Details</h1>
        <form method="post" action="http://localhost:8080/Myapp/week113.jsp">
                <label>Name of the Book: </label>
                <input type="text" name="book"><br><br>
                <input type="submit" value="Submit">
        </form>
</body>
</html>
```

Jsp file:

```
<html>
```

```
<head>
  <title>Books list</title>
</head>
<body>

  <%@ page import = "java.sql.*,java.time.LocalDate, java.time.temporal.ChronoUnit" %>
  <%
        Class.forName("com.mysql.jdbc.Driver");
    Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/wt_db","root","root");
        String book = request.getParameter("book");
    String query = "SELECT book_id, booktitle, author, price FROM book WHERE booktitle = ?";
        PreparedStatement ps = conn.prepareStatement(query);
        ps.setString(1, book);
        ResultSet rs = ps.executeQuery();
        if(rs.next()) {
%>
        <center><h1>Details of the Book <%=book%> are: </h1>
                <table border=1 cellpadding=5>
                <tr>
                        <th>Book ID</th>
                        <th>Author of the Book</th>
                        <th>Price</th>
                </tr>
                <%
                        do {
                                String author = rs.getString("author");
                                int bookId = rs.getInt("book_id");
                                int price = rs.getInt("price");
```

```
%>
        <tr>
            <td><%= bookId %></td>
            <td><%= author %></td>
            <td><%= price %></td>
        </tr>
<%
        } while(rs.next());
%>
        </table>
<%
        } else {
            out.println("No book is found");
        }
        rs.close();
        ps.close();
        conn.close();
%>
</body>
</html>
```

Output:

Entering book name