WEB TECHNOLOGIES

B.Tech III/IV (CSE) – II SEMESTER

### DEPARTMENT OF COMPUTER SCIENCE & ENGINEERINHG

### SREENIDHI INSTITUTE OF SCIENCE & TECHNOLOGY

Yamnampet, Ghatkesar, Hyderabad – 501 301

Syllabus for B. Tech. IV Year I semester

Computer Science and Engineering

MULTIMEDIA AND WEB TECHNOLOGIES LAB

Objective :

To create a fully functional website with mvc architecture. To Develop an online Book store using we can sell books (Ex amazon .com).

Hardware and Software required :

* 1. A working computer system with either Windows or Linux
  2. A web browser either IE or firefox
  3. Tomcat web server and Apache web server
  4. XML editor like Altova Xml-spy [www.Altova.com/XMLSpy – free ] , Stylusstudio , etc.,
  5. A database either Mysql or Oracle
  6. JVM(Java virtual machine) must be installed on your system
  7. BDK(Bean development kit) must be also be installed

Week-1:

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

1) HOME PAGE:

The static home page must contain three frames.

Top frame : Logo and the college name and links to Home page, Login page, Registration page,

Catalogue page and Cart page (the description of these pages will be given below).

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.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Logo | Web Site Name | | | |
| Home | Login | Registration | Catalogue | Cart |
| CSE  ECE  EEE  CIVIL | Description of the Web Site | | | |

Fig 1.1

2) LOGIN PAGE:

This page looks like below:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Logo | Web Site Name | | | |
| Home | Login | Registration | Catalogue | Cart |
| CSE  ECE  EEE  CIVIL | Login :  Password:  Submit  Reset | | | |

3) CATOLOGUE PAGE:

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.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Logo | Web Site Name | | | | | | |
| Home | Login | | Registration | Catalogue | | | Cart |
| CSE  ECE  EEE  CIVIL |  | Book : XML Bible  Author : Winston  Publication : Wiely | | | $ 40.5 |  | |
|  | Book : AI  Author : S.Russel  Publication : Princeton hall | | | $ 63 |  | |
|  | Book : Java 2  Author : Watson  Publication : BPB publications | | | $ 35.5 |  | |
|  | Book : HTML in 24 hours  Author : Sam Peter  Publication : Sam publication | | | $ 50 |  | |

Note: Week 2 contains the remaining pages and their description.

Week-2:

4) CART PAGE:

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

The cart page should look like this:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Logo | Web Site Name | | | |
| Home | Login | Registration | Catalogue | Cart |
| CSE  ECE  EEE  CIVIL | Book name Price Quantity Amount  Java 2 $35.5 2 $70  XML bible $40.5 1 $40.5  Total amount - $130.5 | | | |

5) REGISTRATION PAGE:

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

1) Name (Text field)   
 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)

WEEK 3:

VALIDATION:

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](mailto:name@domain.com))

4. Phone number (Phone number should contain 10 digits only).

<html>

<head>

<title>Registration</title>

<style>

tr,input{font-family:"Monotype Corsiva";font-size:30;color:brown;text-align:center}

input,option,select{font-size:24;color:blue}

</style>

<script type="text/javascript">

function validate()

{

if(frm.UserName.value == "")

alert("Name field cannot be empty");

if(frm.Password.value == "")

alert("Password field cannot be empty");

usr=frm.UserName.value;

len=usr.length;

if(len < 6 && len >=0)

alert("Name must contain atleast 6 characters");

pwd=frm.Password.value;

len=pwd.length;

if(len < 6 && len >= 0)

alert("Password must contain atleast 6 characters");

for( x in usr)

{ ch=usr.charCodeAt(x);

if(ch < 65 || (ch > 90 && ch < 97) || ch> 122 )

{

alert("invalid UserName");

break;

}

}

var regex\_pwd=/([\_]|[a-z]|[A-Z])([\_][a-z][A-Z][0-9])\*/g;

if((pwd.match(regex\_pwd))==null)

alert("Invalid Password");

if(frm.Password.value != frm.ReTypePassword.value)

alert("Password typed incorrectly!");

e=frm.Email\_id.value;

var regex\_mail=/([\_][a-z][A-Z])([\_][a-z][A-Z][0-9])\*(@)([a-z])\*(.)([a-z])\*/g;

if((e.match(regex\_mail))==null)

alert("Invalid E-mail id");

pnum=frm.PhoneNum.value;

len=pnum.length;

if(len != 10)

alert("Phone number must contain exactly 10 characters");

var regex\_pno=/([0-9])([0-9])\*/g;

if((pnum.match(regex\_pno))==null)

alert("Invalid Phone Number");

if(frm.Address.value == "")

alert("Address field cannot be empty");

}

</script>

</head>

<body bgcolor=yellow>

<br>

<br>

<form name="frm">

<table border=1 align=center>

<tr>

<td>Name: </td>

<td>

<input type="text" name="UserName" />

</td>

</tr>

<tr>

<td> Password: </td>

<td>

<input type="password" name="Password" />

</td>

</tr>

<tr>

<td> Re-Type Password: </td>

<td>

<input type="password" name="ReTypePassword" />

</td>

</tr>

<tr>

<td>E-mail id: </td>

<td>

<input type="text" name="Email\_id" />

</td>

</tr>

<tr>

<td> Phone number: </td>

<td>

<input type="text" name="PhoneNum" />

</td>

</tr>

<tr>

<td>Sex:</td>

<td align=left>

<input type="radio" name="sex" value="male"/> Male

<br>

<input type="radio" name="sex" value="female"/> Female

</td>

</tr>

<tr>

<td> Date Of Birth: </td>

<td>

<select name="Day">

<option value="1">1</option>

<option value="2">2</option>

-----------------------------

-----------------------------

<option value="30">30</option>

<option value="31">31</option>

</select>

<select name="Month">

<option value="Jan">Jan</option>

<option value="Feb">Feb</option>

-----------------------------

-----------------------------

<option value="Dec">Dec</option>

</select>

<select name="Year">

<option value="1989">1989</option>

<option value="1990">1990</option>

-----------------------------

-----------------------------

</select>

</td>

</tr>

<td>Languages Known: </td>

<td align=left>

<input type="checkbox" name="lang" value="English"/>English

<br>

<input type="checkbox" name="lang" value="Telugu"/>Telugu

<br>

<input type="checkbox" name="lang" value="Hindi"/>Hindi

</td>

</tr>

<tr>

<td> Address: </td>

<td>

<textarea name="Address"></textarea>

</td>

</tr>

<tr>

<td>

<input type="submit" value="Submit" onclick="validate()" />

</td>

<td>

<input type="reset" value="Reset" />

</td>

</tr>

</table>

</form>

</body>

</html>

Week-4:

Design a web page using CSS (Cascading Style Sheets) which includes the following:

1) Use different font, styles:

In the style definition you define how each selector should work (font, color etc.).  
 Then, in the body of your pages, you refer to these selectors to activate the styles.  
  
For example:

|  |  |  |
| --- | --- | --- |
|  | <HTML> <HEAD> <style type="text/css"> B.headline {color:red; font-size:22px; font-family:arial; text-decoration:underline} </style>  </HEAD>  <BODY> <b>This is normal bold</b><br> Selector {cursor:value}  For example:  <html> <head> <style type="text/css"> .xlink {cursor:crosshair} .hlink{cursor:help} </style> </head>  <body> <b> <a href="mypage.htm" class="xlink">CROSS LINK</a> <br> <a href="mypage.htm" class="hlink">HELP LINK</a> </b> </body> </html>  <b class="headline">This is headline style bold</b> </BODY>  </HTML> |  |
|  | | |

2) Set a background image for both the page and single elements on the page.

You can define the background image for the page like this:

|  |  |  |
| --- | --- | --- |
|  | BODY {background-image:url(myimage.gif);} |  |
|  | | |

3) Control the repetition of the image with the background-repeat property.

As background-repeat: repeat  
 Tiles the image until the entire page is filled, just like an ordinary background image in plain HTML.

4) Define styles for links as

A:link  
 A:visited  
 A:active  
 A:hover

Example:

<style type="text/css">  
 A:link {text-decoration: none}  
 A:visited {text-decoration: none}  
 A:active {text-decoration: none}  
 A:hover {text-decoration: underline; color: red;}  
 </style>

5) Work with layers:

For example:

LAYER 1 ON TOP:  
 <div style="position:relative; font-size:50px; z-index:2;">LAYER 1</div>  
 <div style="position:relative; top:-50; left:5; color:red; font-size:80px; z- index:1">LAYER 2</div>  
  
 LAYER 2 ON TOP:  
 <div style="position:relative; font-size:50px; z-index:3;">LAYER 1</div>  
 <div style="position:relative; top:-50; left:5; color:red; font-size:80px; z- index:4">LAYER 2</div>

6) Add a customized cursor:

Selector {cursor:value}  
 For example:

|  |  |  |
| --- | --- | --- |
|  | <html> <head> <style type="text/css"> .xlink {cursor:crosshair} .hlink{cursor:help} </style> </head>  <body> <b> <a href="mypage.htm" class="xlink">CROSS LINK</a> <br> <a href="mypage.htm" class="hlink">HELP LINK</a> </b> </body> </html> |  |
|  | | |

Week-5:

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

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.

Note: Give at least for 4 books. It should be valid syntactically.

Hint: You can use some xml editors like XML-spy

**Displaying using CSS**

books.xml

<?xml version="1.0" ?>

<?xml-stylesheet type="text/css" href="Books.css" ?>

<bookstore>

<book category="cooking">

<title lang="en">Everyday Italian</title>

<author id="a">Giada De Laurentiis</author>

<isbn>111</isbn>

<publisher>TATA Mac Graw Hill </publisher>

<edition>edition1.0</edition>

<price>300.00</price>

</book>

<book category="children">

<title lang="en">Harry Potter</title>

<author>J K. Rowling</author>

<isbn>112</isbn>

<publisher>TATA Mac Graw Hill </publisher>

<edition>edition1.0</edition>

<price>400.00</price>

</book>

<book category="web">

<title lang="en">XQuery Kick Start</title>

<author>James McGovern</author>

<isbn>112</isbn>

<publisher>TATA Mac Graw Hill </publisher>

<edition>edition3.0</edition>

<price>500.00</price>

</book>

</bookstore>

**Books.css**

title,author,isbn,publisher,edition,price,{

color: red;

font-size:13pt;

border: thin cyan solid;

width=16%;

}

**Displaying using XSL**

**Books.xml**

<?xml version="1.0" ?>

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

<bookstore>

<book category="cooking">

<title lang="en">Everyday Italian</title>

<author id="a">Giada De Laurentiis</author>

<isbn>111</isbn>

<publisher>TATA Mac Graw Hill </publisher>

<edition>edition1.0</edition>

<price>300.00</price>

</book>

<book category="children">

<title lang="en">Harry Potter</title>

<author>J K. Rowling</author>

<isbn>112</isbn>

<publisher>TATA Mac Graw Hill </publisher>

<edition>edition1.0</edition>

<price>400.00</price>

</book>

<book category="web">

<title lang="en">XQuery Kick Start</title>

<author>James McGovern</author>

<isbn>112</isbn>

<publisher>TATA Mac Graw Hill </publisher>

<edition>edition3.0</edition>

<price>500.00</price>

</book>

<book category="web" >

<title lang="en">Learning XML</title>

<author>Erik T. Ray</author>

<isbn>114</isbn>

<publisher>TATA Mac Graw Hill </publisher>

<edition>edition1.0</edition>

<price>400.00</price>

</book>

<book category="web" >

<title lang="en">Learning XML</title>

<author>Erik T. Ray</author>

<isbn>114</isbn>

<publisher>TATA Mac Graw Hill </publisher>

<edition>edition1.0</edition>

<price>400.00</price>

</book>

<book category="web" >

<title lang="en">Learning XML</title>

<author>Erik T. Ray</author>

<isbn>114</isbn>

<publisher>TATA Mac Graw Hill </publisher>

<edition>edition1.0</edition>

<price>400.00</price>

</book>

</bookstore>

**Books.xsl**

<?xsl version="1.0"?>

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

<BODY>

<H1>Book Order</H1>

<TABLE border="1" cellpadding="6">

<TR style="color:olive">

<TH>Title</TH>

<TH>Author</TH>

<TH>ISBN</TH>

<TH>Publisher</TH>

<TH>Edition</TH>

<TH>Price</TH>

</TR>

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

<TR>

<TD style="color:blue"><xsl:value-of select="title"/> </TD>

<TD style="color:red;font-weight:bold"><xsl:value-of select="author"/></TD>

<TD style="color:cyan"><xsl:value-of select="isbn"/></TD>

<TD style="color:magenta"><xsl:value-of select="publisher"/></TD>

<TD style="color:orange"><xsl:value-of select="edition"/></TD>

<TD style="color:grey"><xsl:value-of select="price"/></TD>

</TR>

</xsl:for-each>

</TABLE>

</BODY>

</HTML>

Week-6:

1. Install TOMCAT web server and APACHE.

While installation assign port number 4040 to TOMCAT and 8080 to APACHE. Make sure that these ports are available i.e., no other process is using this port.

1. Access the above 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.

Access the pages by using the urls : <http://localhost:4040/rama/books.html> (for tomcat)

<http://localhost:8080/books.html> (for Apache)

Week-7:

User Authentication :

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

1. Create a Cookie and add these four user id’s and passwords to this Cookie.

2. 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.

Login.html

<html>

<head>

<title>LoginPage</title>

</head>

<body bgcolor=yellow>

<br>

<br>

<br>

<form name="frm" action="login" method="get">

UserName: <input type="text" value="" name="un" /><br>

Password: <input type="password" name="pwd" /><br>

<input type="submit" value="Submit"/>

<input type="reset" value="Reset" />

</form>

</body>

</html>

LoginCookie.java

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

public class LoginCookie extends HttpServlet

{

public void doGet(HttpServletRequest req,HttpServletResponse res) throws IOException,ServletException

{

res.setContentType("text/html");

ServletConfig con=getServletConfig();

String u1=req.getParameter("un");

String p1=req.getParameter("pwd");

String u[]=new String[8];

for( int i=1;i<4;i++)

{

u[i]= con.getInitParameter("user"+(i+1));

p[i+4]= con.getInitParameter("pwd"+(i+1));

}

Cookie c[]=new Cookie[8];

int i,j;

for(i=0;i<8;i++)

{

c[i]=new Cookie("us"+i,u[i]);

res.addCookie(c[i]);

}

PrintWriter out=res.getWriter();

Cookie c[]=req.getCookies();

out.println(c[0].getValue());

for(i=0;i<4;i++) {

if(u1.equals( c[i].getValue() ) && p1.equals(cs[i+4].getValue() ) )

{

out.println(" welcome ...."+c[i].getName() );

break;

}

}

if( i==4 )

out.println(" Invalid user" );

}

}

web.xml

<web-app>

<servlet>

<servlet-name>logincookie</servlet-name>

<servlet-class>LoginCookie</servlet-class>

<init-param>

<param-name>user1</param-name>

<param-value>1</param-value>

</init-param>

<init-param>

<param-name>user2</param-name>

<param-value>2</param-value>

</init-param>

<init-param>

<param-name>user3</param-name>

<param-value>3</param-value>

</init-param>

<init-param>

<param-name>user4</param-name>

<param-value>4</param-value>

</init-param>

<init-param>

<param-name>pwd1</param-name>

<param-value>a</param-value>

</init-param>

<init-param>

<param-name>pwd2</param-name>

<param-value>b</param-value>

</init-param>

<init-param>

<param-name>pwd3</param-name>

<param-value>c</param-value>

</init-param>

<init-param>

<param-name>pwd4</param-name>

<param-value>d</param-value>

</init-param>

</servlet>

<servlet-mapping>

<servlet-name>logincookie</servlet-name>

<url-pattern>/login</url-pattern>

</servlet-mapping>

</web-app>

Week 8

Install a database (Mysql or Oracle).

Create a table which should contain at least the following fields: name, password, email-id, phone number (these should hold the data from the registration form).

Practice 'JDBC' connectivity.

1. Write a java program/servlet/JSP to connect to that database and extract data from the tables and display them. Experiment with various SQL queries.
2. Insert the details of the users who register with the web site, whenever a new user clicks the submit button in the registration page (week2).

a)

**Extract.jsp**

<%@ page language="java" contentType="text/html" %>

<%@ page import="java.sql.\*" %>

<html>

<head><title>employee details </title></head>

<body>

<%

Class.forName("oracle.jdbc.driver.OracleDriver")

Connection con=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","employee","employee");

ResultSet rs=st.executeQuery("select \*from student ");

%>

<table border="1" bgcolor="cyan">

<tr><th>Name</th><th>Password</th><th>E-mail</th><th>Phone No </th></tr>

<% while(rs.next()) { %>

<tr><td><%= rs.getString(1) %> </td><td><%= rs.getString(2) %><td><%= rs.getString(3) %> </td><td><%= rs.getString(4) %> </td> </tr>

<% } %>

</table>

</body>

</html>

**Note:** 1) Assume all fields are of type String including phone number

1. Practice various queries such as select \*from student where name LIKE ‘S’);

b)

Registration.html

<html>

<head><title>Registration </title></head>

<body>

<form action=”Registration.jsp">

UserName:<input type="text" value="" name="un" /><br/>

Password:<input type="password" value="" name="pwd" /><br/>

Email Id:<input type="text" value="" name="em" /><br/>

Phone No:<input type="text" value="" name="pn" /><br/>

<input type="submit" value="submit" />

<input type="reset" value="reset" />

</form>

</body>

</html>

Registration.jsp

<%@ page language="java" contentType="text/html" %>

<%@ page import="java.sql.\*" %>

<html>

<head><title>employee details </title></head>

<body>

<%

Class.forName("oracle.jdbc.driver.OracleDriver");

Connection con=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","employee","employee");

Statement st=con.createStatement();

String uname=request.getParameter("un");

String pword=request.getParameter("pwd");

String email=request.getParameter("em");

String phone=request.getParameter("pn");

int i=st.executeUpdate("insert into student values('"+uname+"','"+pword+"','"+email+"','"+phone+"')");

if(i>0){ out.println("insertion success");}else{out.println("insertion failed");}

%>

</body>

</html>

Week-9:

Write a JSP which does the following job:

Insert the details of the 3 or 4 users who register with the web site (week9) by using registration

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

from the database ( similar to week8 instead of cookies).

Registration.html

<html>

<head><title>Registration </title></head>

<body>

<form action=”Registration.jsp">

UserName:<input type="text" value="" name="un" /><br/>

Password:<input type="password" value="" name="pwd" /><br/>

Email Id:<input type="text" value="" name="em" /><br/>

Phone No:<input type="text" value="" name="pn" /><br/>

<input type="submit" value="submit" />

<input type="reset" value="reset" />

</form>

</body>

</html>

Registration.jsp

<%@ page language="java" contentType="text/html" %>

<%@ page import="java.sql.\*" %>

<html>

<head><title>employee details </title></head>

<body>

<%

Class.forName("oracle.jdbc.driver.OracleDriver");

Connectioncon=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","employee","employee");

Statement st=con.createStatement();

String uname=request.getParameter("un");

String pword=request.getParameter("pwd");

String email=request.getParameter("em");

String phone=request.getParameter("pn");

int i=st.executeUpdate("insert into student values('"+uname+"','"+pword+"','"+email+"','"+phone+"')");

if(i>0){ out.println("insertion success");}else{out.println("insertion failed");}

%>

</body>

</html>

Login.html

<html>

<head><title>login form </title></head>

<body>

<form action="Login.jsp">

UserName:<input type="text" value="" name="un" /><br/>

Password:<input type="password" value="" name="pwd" /><br/>

<input type="submit" value="submit" />

<input type="reset" value="reset" />

</form>

</body>

</html>

**Login.jsp**

<%@ page language="java" contentType="text/html" %>

<%@ page import="java.sql.\*" %>

<html>

<head><title>employee details </title></head>

<body>

<%

Class.forName("oracle.jdbc.driver.OracleDriver")

Connection con=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","employee","employee");

Statement st=con.createStatement();

String uname=request.getParameter("un");

String pword=request.getParameter("pwd");

ResultSet rs=st.executeQuery("select \*from student where username='"+uname+"'and password='"+pword+"'");

if(rs.next())

{ out.println("Valid user"); }

else

{ out.println(" Invalid user "); }

%>

</body>

</html>

Week-10:

Create tables in the database which contain the details of items (books in our case like Book name , Price, Quantity, Amount )) of each category. Modify your catalogue page (week 2)in such a way that you should connect to the database and extract data from the tables and display them in the catalogue page using JDBC.

**Catalogue.jsp**

<%@ page language="java" contentType="text/html" %>

<%@ page import="java.sql.\*" %>

<html>

<head><title>employee details </title></head>

<body>

<%

Class.forName("oracle.jdbc.driver.OracleDriver");

Connection con=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","employee","employee");

Statement st=con.createStatement();

ResultSet rs=st.executeQuery("select \* from book");

%>

<table border="1" bgcolor="cyan">

<tr><th>title</th><th>author</th><th>price</th></tr>

<% while(rs.next()) { %>

<tr><td><%= rs.getString(1) %> </td><td><%= rs.getString(2) %><td><%= rs.getString(3) %> </td> </tr>

<% } %>

</table>

</body>

</html>

Week-11:

HTTP is a stateless protocol. Session is required to maintain the state.

The user may add some items to cart from the catalog page. He can check the cart page for the selected items. He may visit the catalogue again and select some more items. Here our interest is the selected items should be added to the old cart rather than a new cart. Multiple users can do the same thing at a time (i.e., from different systems in the LAN using the ip-address instead of local host). This can be achieved through the use of sessions. Every user will have his own session which will be created after his successful login to the website. When the user logs out his session should get invalidated (by using the method session. invalidate ().

Modify your catalogue and cart JSP pages to achieve the above mentioned functionality using sessions.

Week-12:

Developing and deployment of web based application using Struts 1.X

**Login.html**

<html>

<body bgcolor="lightgreen" >

<font size="16" color="red"><b><center>

<form method="get" action="hello.do">

Enter Name<input type="text" name="uname"/><br>

Enter password <input type="password" name="upwd"/><br>

<input type="submit" value="send"/>

</form>

</center></b></font>

</body>

</html>

**Failure.jsp**

<html>

<body bgcolor="reg" >

<font size="22" color="pink">

<center><b>Login Failure

</center></b></font>

</body>

</html>

Success.jsp

<html>

<body bgcolor="reg" >

<font size="22" color="pink">

<center><b>Login Success

</center></b></font>

</body>

</html>

**Web.xml**

<web-app>

<welcome-file-list>

<welcome-file>login.html</welcome-file>

</welcome-file-list>

<servlet>

<load-on-startup>1</load-on-startup>

<init-param>

<param-name>config</param-name>

<param-value>/WEB-INF/struts-config.xml

</param-value>

</init-param>

<servlet-name>as</servlet-name>

<servlet-class>org.apache.struts.action.ActionServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>as</servlet-name>

<url-pattern>\*.do</url-pattern>

</servlet-mapping>

</web-app>

Struts-config.xml

<!DOCTYPE struts-config PUBLIC

"-//Apache Software Foundation//DTD Struts Configuration 1.3//EN"

"http://struts.apache.org/dtds/struts-config\_1\_3.dtd">

<struts-config>

<form-beans>

<form-bean name="hai" type="com.dss.LoginActionForm"/>

</form-beans>

<action-mappings>

<action name="hai" type="com.dss.LoginAction" path="/hello">

<forward name="success" path="/success.jsp"/>

<forward name="failure" path="/failure.jsp"/>

</action>

</action-mappings>

</struts-config>

**LogingActionForm.java**

package com.dss;

import org.apache.struts.action.\*;

public class LoginActionForm extends ActionForm

{

private String uname;

private String upwd;

public void setUname(String uname)

{

this.uname=uname;

}

public String getUname()

{

return uname;

}

public void setUpwd(String upwd)

{

this.upwd=upwd;

}

public String getUpwd()

{

return upwd;

}

}

**LoginAction.java**

package com.dss;

import org.apache.struts.action.\*;

import javax.servlet.http.\*;

public class LoginAction extends Action

{

public ActionForward execute(ActionMapping mapping,ActionForm form,HttpServletRequest req,HttpServletResponse res) throws Exception

{

LoginActionForm laf=(LoginActionForm)form;

String uname=laf.getUname();

String upwd=laf.getUpwd();

if((uname.equals("sai")) && (upwd.equals("sai")))

{

return mapping.findForward("success");

}

else

{

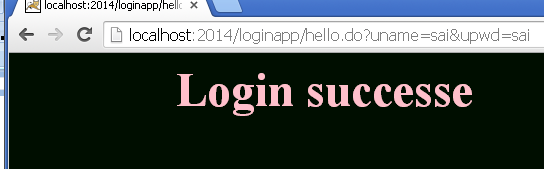
return mapping.findForward("failure");

}

}

}





If wrong

