Exp No: 1

INTRODUCTION

A web application or web app is any software that runs in a web browser. It is created in a browser-supported programming language (such as the combination of JavaScript, HTML and CSS) and relies on a web browser to render the application.

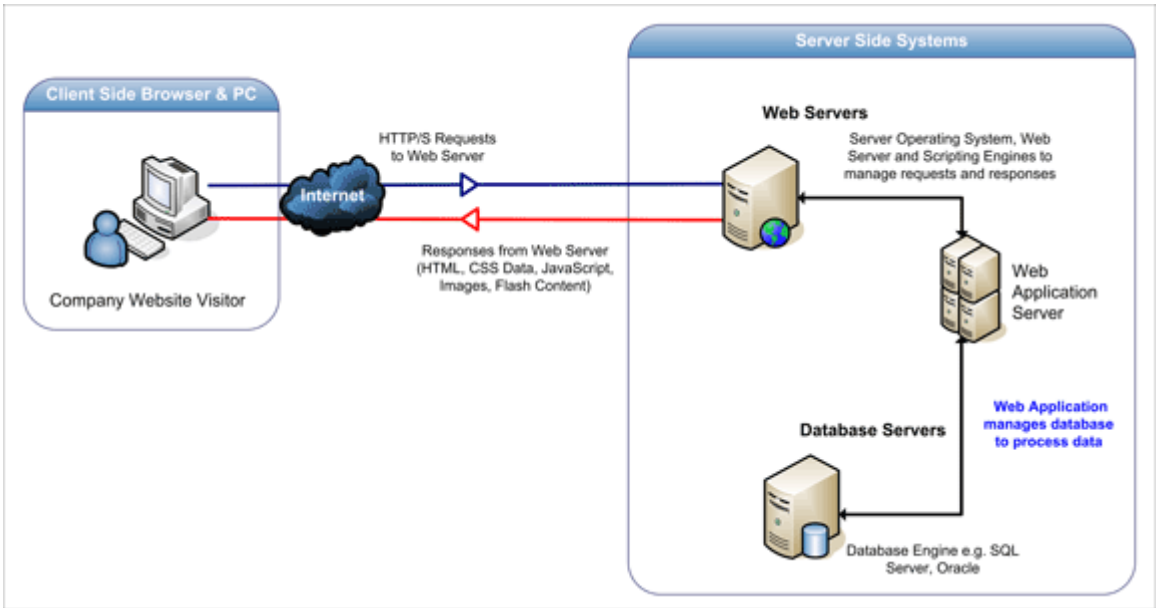
Web applications are by nature distributed applications, meaning that they are programs that run on more than one computer and communicate through a network or server. Specifically, web applications are accessed with a web browser and are popular because of the ease of using the browser as a user client. For the enterprise, the ability to update and maintain web applications without deploying and installing software on potentially thousands of client computers is a key reason for their popularity. Web applications are used for web mail, online retail sales, discussion boards, weblogs, online banking, and more. One web application can be accessed and used by millions of people.

Like desktop applications, web applications are made up of many parts and often contain mini programs, some of which have user interfaces, and some of which do not require a graphical user interface (GUI) at all. In addition, web applications frequently require an additional markup or scripting language, such as HTML, CSS, or JavaScript programming language. Also, many applications use only the Java programming language, which is ideal because of its versatility.

A web application can be as simple as a page that shows the current date and time or as complex as a set of pages on which you can look up and book the most convenient flight, hotels, and car rentals for your next vacation.

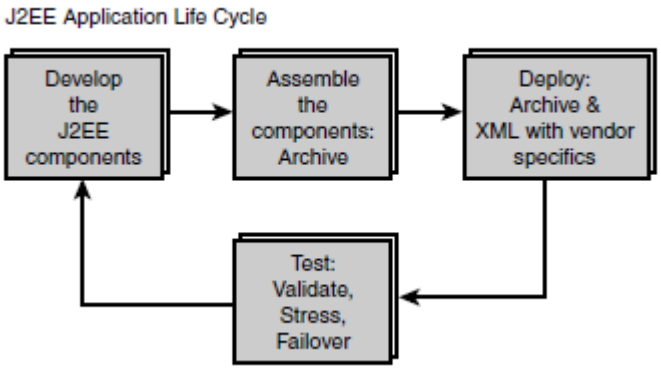
How do web applications work?

The figure below details the three-layered web application model. The first layer is normally a web browser or the user interface; the second layer is the dynamic content generation technology tool such as Java servlets (JSP) or Active Server Pages (ASP), and the third layer is the database containing content (e.g., news) and customer data (e.g., usernames and passwords, social security numbers and credit card details).



WEB APPLICATION LIFE CYCLE

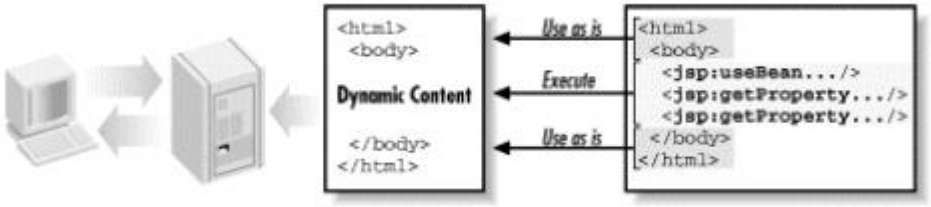
A web application contains web components static resource files such as images and helper classes and libraries. The life cycle of a J2EE web application can be shown in the figure:



A web module can be deployed as an unpacked file structure or can be packaged in a JAR file known as Web Archive (WAR) file. To display a WAR on the application server the file must also contain run-time deployment descriptor. The run-time deployment descriptor is an XML file that contains information such as the content of the web application and mapping of the portable names of application resources to the application server's resources.

JAVA SERVER PAGES (JSP)

In late 1999, Sun Microsystems added a new element to the collection of Enterprise Java tools: Java Server Pages (JSP). Java Server Pages are built on top of Java servlets and are designed to increase the efficiency in which programmers, and even nonprogrammers can create web content. A JSP page contains standard Markup Language elements, such as HTML (Hyper Text Markup Language) tags, just like a regular web page. A JSP page also contains special JSP elements that allow the server to insert dynamic content in the page. JSP elements can be used for a wide variety of purposes, such as retrieving information from a database or registering user preferences. When a user asks for a JSP page, the server executes the JSP elements, merges the results with the static parts of the page, and sends the dynamically composed page back to the browser.



Generating Dynamic Content with JSP Elements

JSP defines a number of standard elements useful for any web application, such as accessing JavaBeans components, passing control between pages, and sharing information between requests, pages, and users. Programmers can also extend the JSP syntax by implementing application-specific elements that perform tasks such as accessing databases and Enterprise JavaBeans, sending email, and generating HTML to present application-specific data. The combination of standard elements and custom elements allows for the creation of powerful web applications. The JSP technology can be used in all kinds of web applications, from the simplest to the most complex. JavaServer Pages is built on top of the Java Servlets Application Programming Interface (API), JSP has access to all of the powerful Enterprise Java APIs, including:

• JDBC (Java Database Connectivity)

• Remote Method Invocation (RMI) and Common Object Request Broker Architecture (CORBA) support

• Java Naming and Directory Interface (JNDI)

• Enterprise JavaBeans (EJB)

• Java Message Service (JMS)

• Java Transaction API (JTA)

This means that we can easily integrate Java Server Pages with the existing Java Enterprise solutions, or take advantage of many aspects of enterprise computing if we are starting from scratch.

JSP Advantages:

• JSP supports both scripting and element-based dynamic content, and allows programmers to develop custom tag libraries to satisfy application-specific needs.

• JSP pages are precompiled for efficient server processing.

• JSP pages can be used in combination with servlets that handle the business logic, the model supported by Java servlet template engines.

• JSP is a specification, not a product. This means vendors can compete with different implementations, leading to better performance and quality.

• JSP is an integral part of Java 2 Java Enterprise Edition, a complete platform for Enterprise class applications.

JSP Elements

Aside from the regular HTML, there are three main types of JSP constructs that embed in a page:

Scripting Elements, Directives and Actions. Scripting elements specify Java code that will become part of the resultant servlet. Directives control the overall structure of the servlet and Action specify existing components that should be used and otherwise control the behavior of the JSP engine.

1.Scripting Elements

Scripting element allow to add small piece of code to a JSP page, such as if statement to generate different HTML condition. It is executed when the page is requested.

Scripting Element	**Example**
Comment	<%-- comment --%>
Directive	<%@ directive %>
Declaration	<%! declarations %>
Scriptlet	<% scriplets %>
Expression	<%= expression %>

2. Directive Elements

The directive elements are used to specify the information about the page itself that remains the same between page requests. The page directive has two attributes that may be used when use scripting elements: language and import: The language attribute specifies the scripting language used in the page. The import makes the possible short class names in the scripting elements.

e.g.:

<%@ page language="java" import="java.util.*" %>

Directive	Description
<%@ page ... %>	defines page dependent properties such as language, session, error Page etc.
<%@ include ... %>	defines file to be included.
<%@ taglib ... %>	declares tag library used in the page

3. Action Elements

Action elements typically perform some action based on information that is required at the exact time the page is required at the exact time the page is requested by the client. An action element can for instance, access parameters sent the request to do a database look up. It can also dynamically generate HTML, such as table filled with information retrieved from an external system.

Action Tag	Description
jsp:forward	forward the request to a new page
jsp:getProperty	retrieve a property from a JavaBean instance.
jsp:include	include the runtime response of a JSP page.
jsp:plugin	Generates client browser-specific construct that makes an OBJECT or EMBED tag for the Java Applets
jsp:element	Defines XML elements dynamically
jsp:attribute	defines dynamically defined XML element's attribute
jsp:param	Adds parameters to the request
jsp:fallback	Supplies alternate text if java applet is unavailable on the client
jsp:body	Used within standard or custom tags to supply the tag body.
jsp:text	Use to write template text in JSP pages and documents.
jsp:setProperty	store data in JavaBeans instance.
jsp:useBean	instantiates a JavaBean

Implicit jsp objects

When use scripting elements in a JSP page, always have access to number of objects that the JSP container makes available. These are called implicit objects. These objects are instances of classes defined by the servlet and JSP specifications.JSP implicit objects are also called pre-defined variables. The implicit objects are:

Implicit Object	Description
request	The **HttpServletRequest** object associated with the request.
response	The **HttpServletRequest** object associated with the response that is sent back to the browser.
out	The **JspWriter** object associated with the output stream of the response.
session	The **HttpSession** object associated with the session for the given user of request.
application	The **ServletContext** object for the web application.
config	The **ServletConfig** object associated with the servlet for current JSP page.
pageContext	The **PageContext** object that encapsulates the environment of a single request for this current JSP page
page	The **page** variable is equivalent to **this** variable of Java programming language.
exception	The **exception** object represents the **Throwable** object that was thrown by some other JSP page.

HTML (Hyper Text Markup Language)

HTML is a format that tells a computer how to display a web page. The documents themselves are plain text files with special "tags" or codes that a web browser uses to interpret and display information on your computer screen.

- HTML stands for Hyper Text Markup Language
- An HTML file is a text file containing small markup tags
- The markup tags tell the Web browser how to display the page
- An HTML file must have an htm or html file extension

HTML Tags

- HTML tags are used to mark-up HTML elements
- HTML tags are surrounded by the two characters < and > The surrounding characters are called angle brackets
- HTML tags normally come in pairs like and .The first tag in a pair is the

start tag, the second tag is the end tag. The text between the start and end tags is the element content.

- HTML tags are not case sensitive, <b> means the same as <B>

Logical vs. Physical Tags

In HTML there are both logical tags and physical tags. Logical tags are designed to describe (to the browser) the enclosed text's meaning. An example of a logical tag is the tag. By placing text in between these tags you are telling the browser that the text has some greater importance. By default all browsers make the text appear bold when in between the and tags. Physical tags on the other hand provide specific instructions on how to display the text they enclose.

Examples of physical tags include:

- <b>: Makes the text bold.
- <big>: Makes the text usually one size bigger than what's around it.
- <i>: Makes text italic.

Physical tags were invented to add style to HTML pages because style sheets were not around, though the original intention of HTML was to not have physical tags. Rather than use physical tags to style your HTML pages, you should use style sheets.

CSS (Cascading Styles Sheets)

CSS, or Cascading Styles Sheets, is a way to style and present HTML. Whereas the HTML is the meaning or content, the style sheet is the presentation of that document. Styles don't smell or taste anything like HTML, they have a format of 'property: value' and most properties can be applied to most HTML tags.

Cascading Style Sheets (CSS) is a style sheet language used for describing the look and formatting of a document written in a markup language. While most often used to change the style of web pages and user interfaces written in HTML and XHTML, the language can be applied to any kind of XML document, including plain XML, SVG and XUL. Along with HTML and JavaScript, CSS is a cornerstone technology used by most websites to create visually engaging web pages, user interfaces for web applications, and user interfaces for many mobile applications.

CSS is designed primarily to enable the separation of document content from document presentation, including elements such as the layout, colors, and fonts. This

separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple HTML pages to share formatting by specifying the relevant CSS in a separate .css file, and reduce complexity and repetition in the structural content, such as semantically insignificant tables that were widely used to format pages before consistent CSS rendering was available in all major browsers. CSS makes it possible to separate presentation instructions from the HTML content in a separate file or style section of the HTML file. For each matching HTML element, it provides a list of formatting instructions. For example, a CSS rule might specify that "all heading 1 elements should be bold," leaving pure semantic HTML markup that asserts "this text is a level 1 heading" without formatting code such as a <bold> tag indicating how such text should be displayed.

This separation of formatting and content makes it possible to present the same markup page in different styles for different rendering methods, such as on-screen, in print, by voice (when read out by a speech-based browser or screen reader) and on Braille-based, tactile devices. It can also be used to display the web page differently depending on the screen size or device on which it is being viewed. While the author of a web page typically links to a CSS file within the markup file, readers can specify a different style sheet, such as a CSS file stored on their own computer, to override the one the author has specified. If the author or the reader did not link the document to a style sheet, the default style of the browser will be applied. Another advantage of CSS is that aesthetic changes to the graphic design of a document (or hundreds of documents) can be applied quickly and easily, by editing a few lines in one file, rather than by a laborious (and thus expensive) process of crawling over every document line by line, changing markup.

WEBSPHERE APPLICATION SERVER

IBM WebSphere Application Server offers options for a faster, more flexible Java application server runtime environment. It offers enhanced reliability and resiliency for building and running applications, including cloud and mobile. It supports environments from single server and midsize configurations to large deployments requiring web tier clustering over multiple application server instances.

WebSphere Application Server helps to:

- **Increase developer productivity** with open standards and broad programming models, including lightweight options for web deployments.
- **Deploy and manage applications and services** without the constraint of time, location or device type.
- **Includes the Liberty profile**, a dynamic web application server profile.
- **Enhance security and control** using integrated management and administrative tools.

IBM DB2

DB2 is a database product from IBM. It is a Relational Database Management System (RDBMS). DB2 is designed to store, analyze and retrieve the data efficiently. DB2 product is extended with the support of Object-Oriented features and non-relational structures with XML. Initially, IBM had developed DB2 product for their specific platform. Since year 1990, it decided to develop a Universal Database (UDB) DB2 Server, which can run on any authoritative operating systems such as Linux, UNIX, and Windows.

Connecting Database with web application

The seven basic steps in connecting to databases

1. Load the JDBC driver.
2. Define the connection URL.
3. Establish the connection.
4. Create a statement object.
5. Execute a query or update.
6. Process the results.
7. Close the connection

Load the Driver

The driver is the piece of software that knows how to talk to the actual database server. To load the driver, all you need to do is to load the appropriate class; a static block in the class itself automatically makes a driver instance and registers it with the JDBC driver manager. To make your code as flexible as possible, it is best to avoid hard-coding the reference to the class name.' Use Class.forName.' This method takes a string representing a fully qualified class name (i.e., one that includes package names) and loads the corresponding class. This call could throw a ClassNotFoundException, so should be inside a try/catch block. Here is an

example:

```
try
{
Class.forName("connect.microsoft.MicrosoftDriver");
Class.forName("oracle.jdbc.driver.OracleDriver");
Class.forName("com.sybase.jdbc.SybDriver");
}
catch(ClassNotFoundException cnfe)
{ System.err.println("Error loading driver: " + cnfe); }
```

Define the Connection URL

Once you have loaded the JDBC driver, you need to specify the location of the database server. URLs referring to databases use the jdbc: protocol and have the server host, port, and database name (or reference) embedded within the URL. The exact format will be defined in the documentation that comes with the particular driver, but here is an examples:

```
String host = "dbhost.yourcompany.com";
String dbName = "someName"; int port = 1234;
String oracleURL = "jdbc:oracle:thin:@" + host + ":" + port + ":" + dbName;
String sybaseURL = "jdbc:sybase:Tds:" + host + ":" + port + ":" + "?SERVICENAME=" + dbName;
```

Establish the Connection

To make the actual network connection, pass the URL, the database username, and the password to the getConnection method of the DriverManager class, as illustrated in the following example. Note that getConnection throws an SQLException, so you need to use a try/catch block.

String username = "jay_debesee"; String password = "secret";

Connection connection = DriverManager.getConnection(IBMdb2URL, username, password);

Create a Statement

A Statement object is used to send queries and commands to the database and is created from the Connection as follows:

Statement statement = connection.createStatement();

Execute a Query

Once you have a Statement object, you can use it to send SQL queries by using the executeQuery method, which returns an object of type ResultSet.

Here is an example:

String query = "SELECT col1, col2, col3 FROM sometable"; ResultSet resultSet = statement.executeQuery(query);

Process the Results

The simplest way to handle the results is to process them one row at a time, using the ResultSet's next method to move through the table a row at a time. Within a row, ResultSet provides various getXxx methods that take a column index or column name as an argument and return the result as a variety of different Java types.

```
while(resultSet.next())
{
System.out.println(results.getString(1) + " " + results.getString(2) + " " + results.getString(3));
}
```

Close the Connection

To close the connection explicitly, you would do: connection.close(); You should postpone this step if you expect to perform additional database operations, since the overhead of opening a connection is usually large.

Viva Questions:

1. What are the advantages of JSP?
2. List some popular web servers.
3. Explain steps in Web Application Life Cycle
4. What do you see as the most critical and current threats effecting Internet accessible websites?
5. What is Cross Site Scripting?

Exp No: 2

INCOME TAX CALCULATION USING SESSION BEAN

Aim:

To calculate income tax using session bean.

Description:

JAVABEANS

JavaBeans are simply regular Java classes designed according to a set of guidelines. By following these guidelines, development tools can figure out how the bean is intended to be used and how it can be linked to other beans. The JavaBeans specification characterizes beans as classes that:

- Support introspection so that a builder tool can analyze how a bean works
- Support customization, so that when using an application builder, a user can customize the appearance and behavior of a bean.
- Support events as a simple communication metaphor that can be used to notify beans of interesting things
- Support properties, both for customization through a tool and for programmatic use
- Support persistence, so that a bean can be customized in an application builder and then have its state saved away and reloaded later.

Enterprise JavaBeans(EJB)

Enterprise JavaBeans technology enables a simplified approach to multitier application development, concealing application complexity and enabling the component developer to focus on business logic. It provides support for interoperability between components includes transaction propagation, naming services, and security services. The EJB specification supports both transient and persistent objects. A transient object is referred to as a session bean and a persistent object is known as an entity bean.

Entity beans represents collections of data such as rows in a relational database and encapsulate operations on the data they represent.

Session beans is an EJB that is created by a client and usually exist only for the duration of a single client-server session. A session bean usually performs operations such as calculations or database access on behalf of the client. Session bean objects can be either stateless or they can maintain a conversational state across methods and transactions. Session

bean can been created as Session.setAttribute(“username”,username);

Algorithm:

Step 1:Create an index page for login that gives user name and password. If it is new user a link to registration page is provided.

Step 2:For each new user allow him\her to register their details in the registration form.

Step3:After registration is done the details of the user information can be stored in the database.

Step 4:.After enter the username and password and check whether the user is an authenticated user. If he\she is a valid user go to step 5,else go to step9.

Step5: For each signed in user create a session, then user is forwarded to the calculation page.

Step 6:The calculation page read the basic income as well as hra,savings,calculate da.And goes to the next step.

Step 7:if savings >100000 then tax=ba+hra+da-100000 and go to step 9

Step 8:else tax=ba+hra+da-savings.

Step 9:Display the tax amount to be paid.

Step 10:Then return back to login page.

Step 11:stop

Sample Code

```
<html>
<body>
<%
String id=session.getAttribute(“username”).toString();
String bs=request.getParameter(“bs”);
String hra=request.getParameter(“hra”);
float b=Float.parseFloat(“bs”);
float h=Float.parseFloat(“hra”);
float total;
float tax;
float diff;
float da;
try
{
Class.forName(“COM.ibm.db2.jdbc.app.DB2Driver”).newInstance();
String dburl=”jdbc:db2:INCOME”;
Connection con=DriverManager.getConnection(dburl,”db2admin”,”db2admin”);
Statement sm=con.createStatement();
ResultSet rs;
rs=sm.executeQuery(select * from new where userid=’”+id+”’);
```

```
boolean nr=rs.next();
da=(80*b)/100;
Total=(b+h+da-sa)*12;
If(total<100000)
{
  Out.println(well"+rs.getString(1)+" "+rs.getString(2)+"");
Out.println("no tax");
}
Else  if(total>100000 && total<=200000)
{
Diff=total-100000;
Tax=.1*diff;
Out.println("welcome"+rs.getString(1)+" +rs.getString(2)+");
Out.println("basic salary is="+b+");
Out.println("hra  is="+h+");
Out.println("da is="+da+");
Out.println("your basic salary is Rs:="+tax+");
}
Else  if(total>200000 && total<=300000)
{
Diff=total-100000;
Tax=.2*diff;
Out.println("welcome"+rs.getString(1)+" +rs.getString(2)+");
Out.println("basic salary is="+b+");
Out.println("hra  is="+h+");
Out.println("da is="+da+");
Out.println("your basic salary is Rs:="+tax+");
}
Else  if(total>300000 && total<=400000)
{
Diff=total-100000;
Tax=.3*diff;
Out.println("welcome"+rs.getString(1)+" +rs.getString(2)+");
Out.println("basic salary is="+b+");
Out.println("hra  is="+h+");
Out.println("da is="+da+");
Out.println("your basic salary is Rs:="+tax+");
}
Else  if(total>400000 )
{
Diff=total-100000;
Tax=.4*diff;
Out.println("welcome"+rs.getString(1)+" +rs.getString(2)+");
Out.println("basic salary is="+b+");
```

```
Out.println("hra  is="+h+");
Out.println("da is="+da+");
Out.println("your basic salary is Rs:="+tax+");
}%></body></html>
```

Design

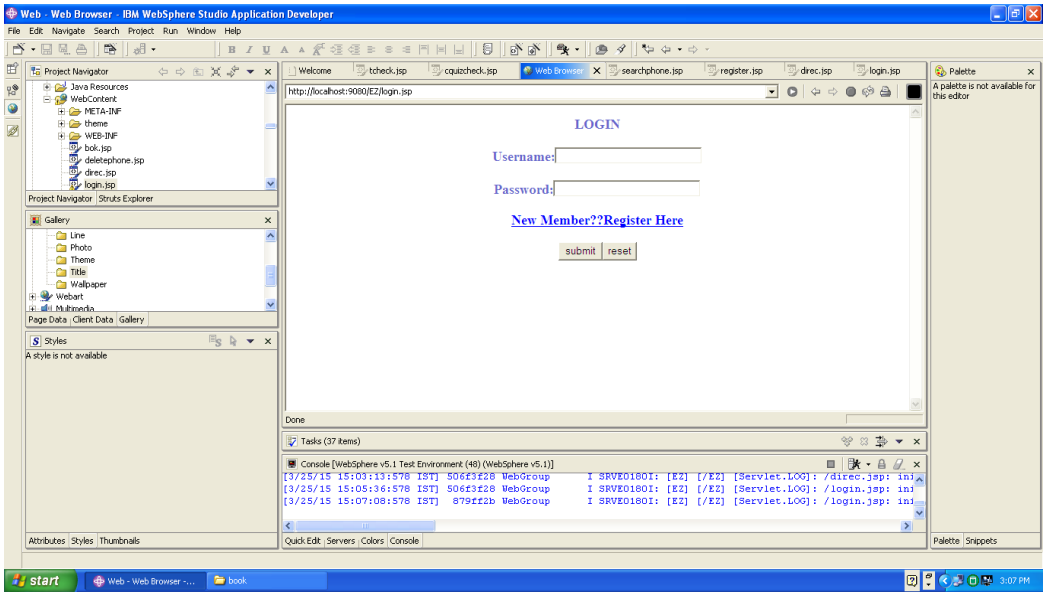


Fig 2.1: Login form

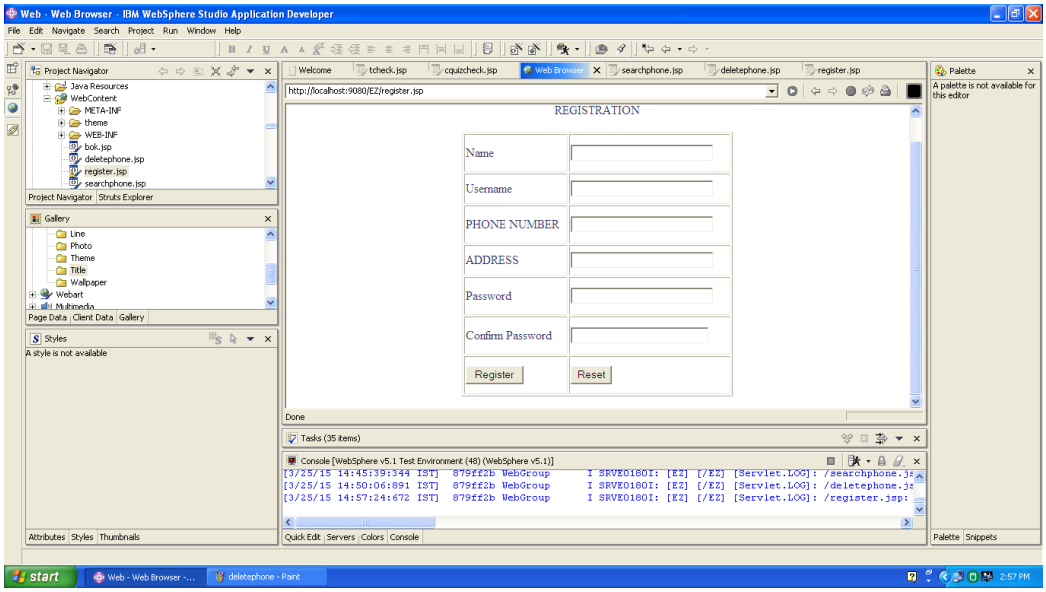


Fig 2.2: Registration form

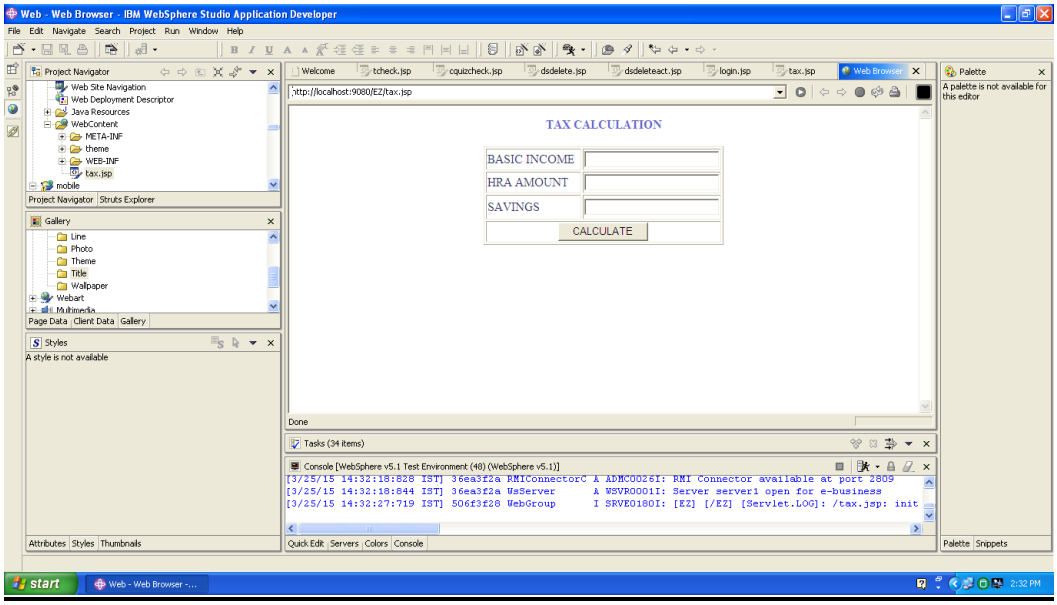


Fig 2.3: Tax calculation Form

Result:

A website for income tax calculation has been created successfully.

Viva Questions:

1. What is EJB?
2. How stateful session bean remembers it's client state?
3. When two entity beans are said to be identical? Which method is used to compare identical or not?
4. What is EJB Query Language ?
5. What is the difference between ejbStore() and ejbLoad() ?

University Sample Question

1. EMI calculator for Home Loan, Car Loan & Personal Loan using session bean

Exp No: 3

ONLINE SHOPPING OF BOOKS

Aim:

Write a web application program on on-line shopping of books

Algorithm:

step1: Create Login page

step2: For new user registration has to done and the information should be entered in database.

step3: For each signed-in user create session

step4: User can search books by author, or title or category

step5: User can select a book and buy it by giving the account details and bank name.

step6: The book details must be entered in the database

step7: If user has sufficient balance book is purchased else print insufficient balance.

Step8: When the user is logged out session has to be removed.

Step9: Return to home page

Design:

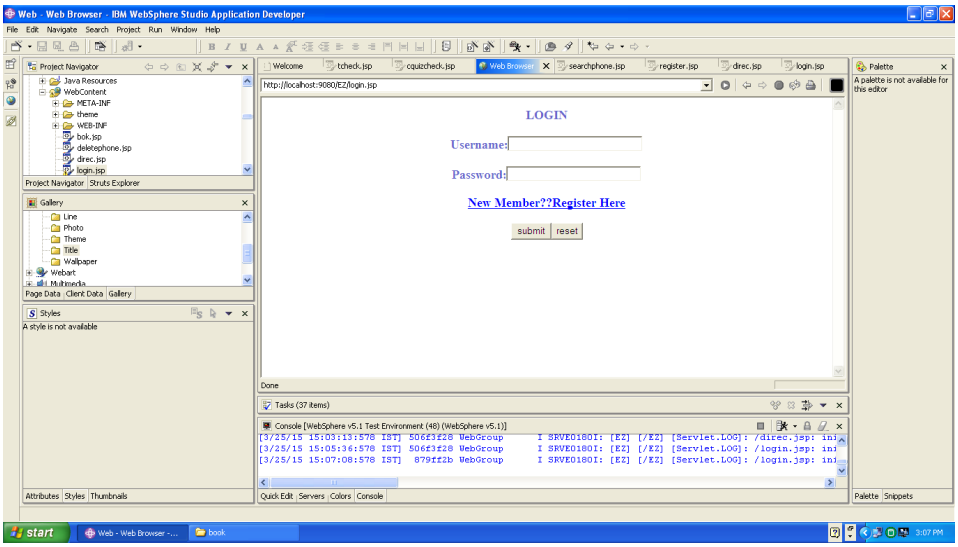


Fig 3.1: Login Form

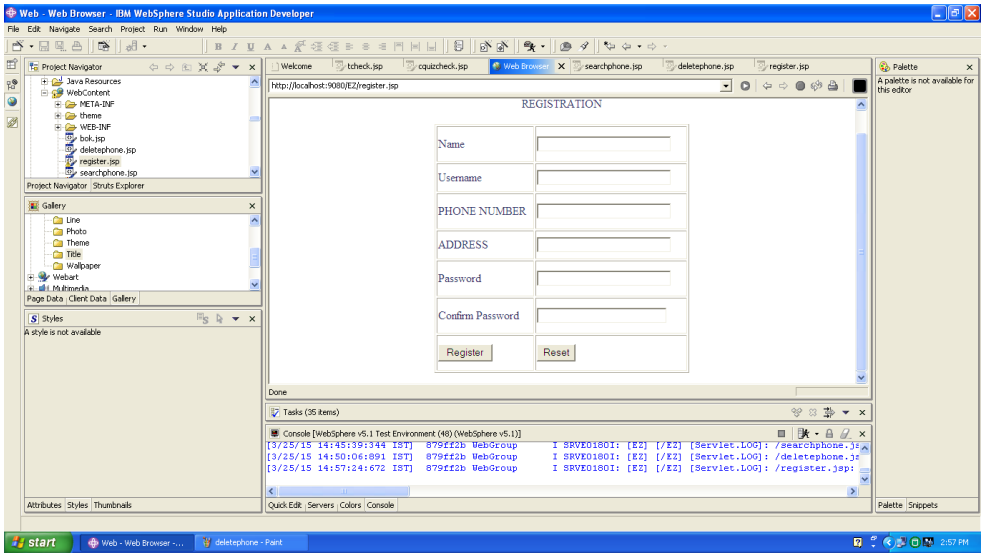


Fig 3.2: Registration Form

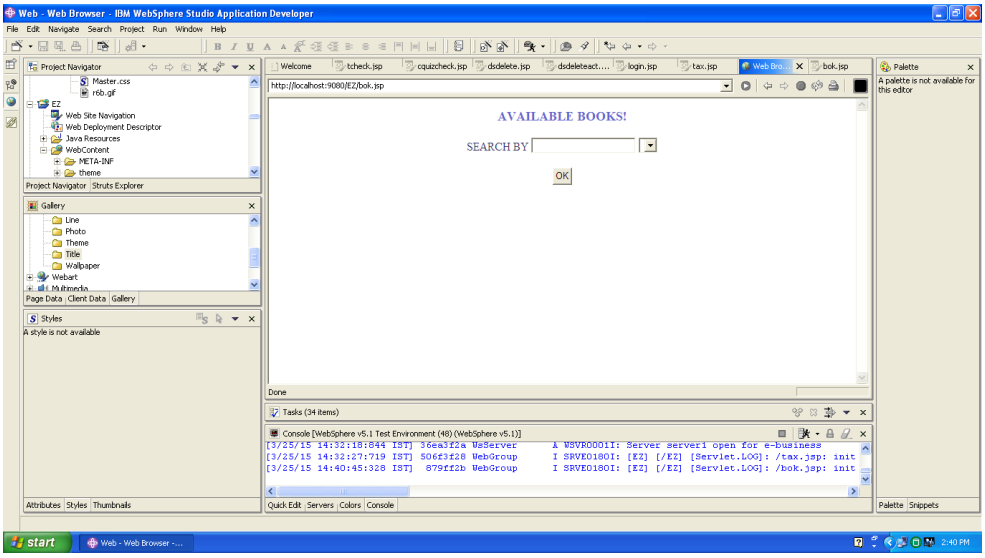


Fig 3.3 Form for searching books.

Result:

A website for implementing online book shopping has been created successfully.

Viva Questions

1. What is the life cycle of Stateful session bean ?
2. What is EJB architecture(components) ?
3. Does Stateful Session bean support instance pooling ?
4. Why does EJB needs two interfaces(Home and Remote Interface) ?
5. Is it possible to invoke multiple Session beans from one Session bean using Reflection .

University Sample Question

1. Web application program for on-line shopping of Mobile phone

Exp No: 4

TELEPHONE DIRECTORY

Aim:

Create a GUI based application which can be used for telephone directory. The telephone directory is stored as database and has one table named telephone directory which stores three different data-phone number, owner, address. The owner name has three parts-first, middle and last. Address has five parts-house number ,address 1:,address2, area name, city name. Application allows search facility using three different ways :

(a) search by telephone no:

(b) search by name(First, middle, last-exact or partial)

(c) search by address (exact or partial)

Algorithm:

Step1: Create Login page where existing user can enter their username and password to login to the telephone directory system. For new user they can register by filling up the registration form and the information entered is updated in the database named directory.

Step2: For each signed in user create session by using Session.setAttribute("username",username)

Step3: Users signed in are provided with the options for searching and deletion.

Step4: User can search for information by entering phone no: or name, which can be first name, middle name or last name, or address.

Step5: Based on the searched criteria details of the users are being displayed.

Step6: User can delete an existing connection by verifying the telephone number of the connection to be deleted.

Step7: The connection details associated with the entered telephone number would be deleted from the directory database by the maintained database connectivity.

Step8: After performing search or deletion the user is logged out and established session can then be removed.

Step9: User is thus returned to the home page.

Design:

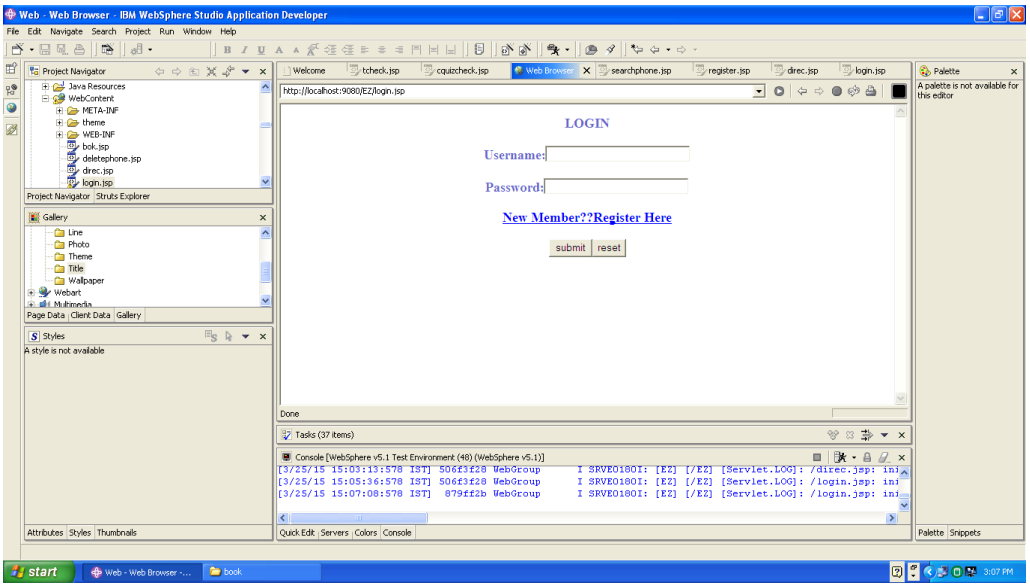


Fig 4.1: Login form

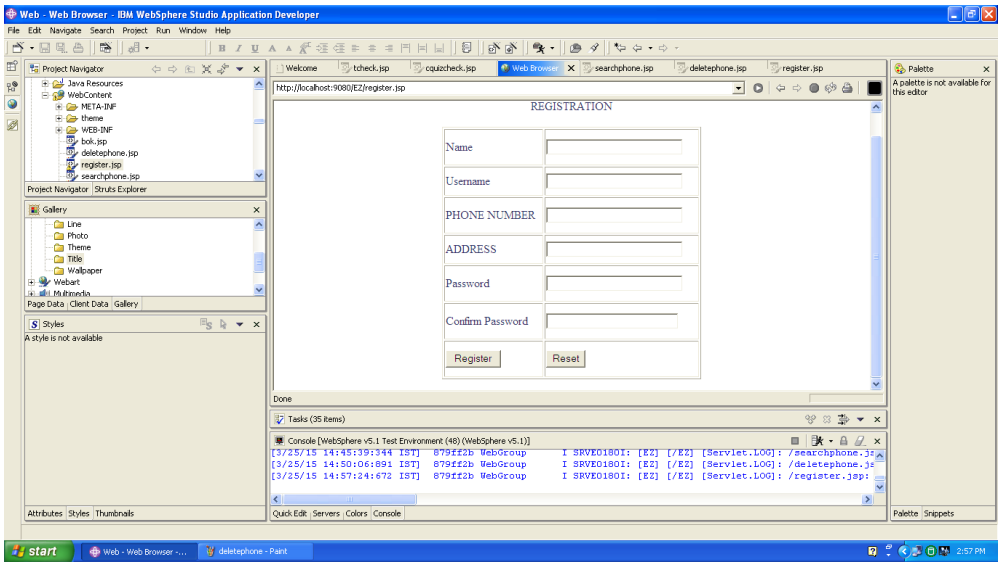


Fig 4.2: Registration Form

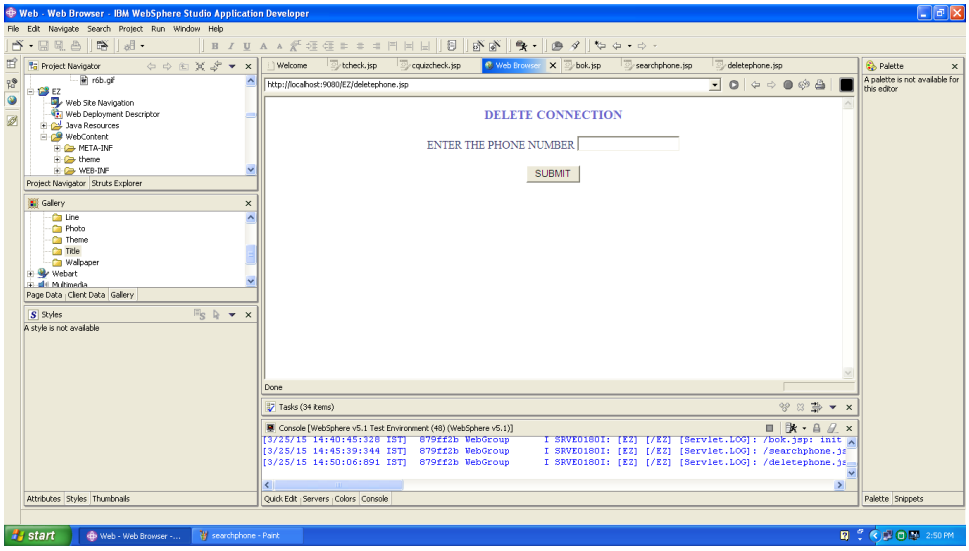


Fig 4.3: Form for deleting connection

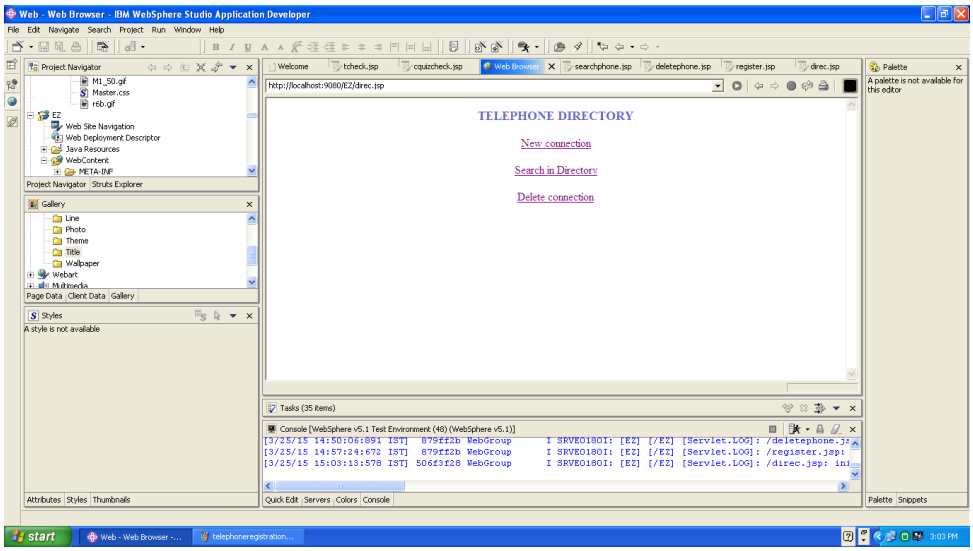


Fig 4.4: Form for options

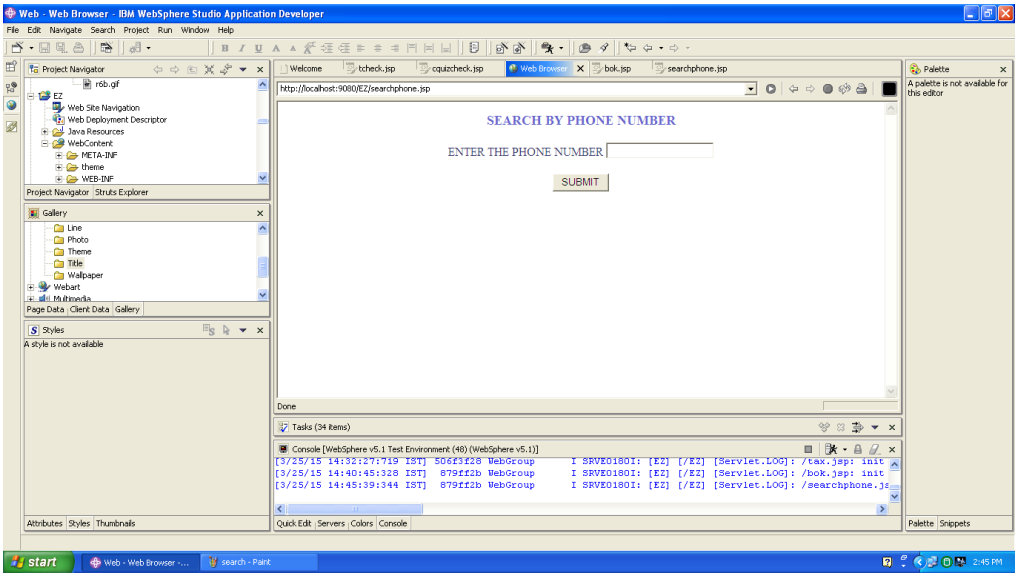


Fig 4.5: Form for search by phone number

Result:

A website for implementing telephone directory has been created successfully.

Viva Questions

1. What is Message Driven Bean?
2. what is difference between servlet and jsp ?
3. What is lazy loading ?
4. What is the difference between find and select methods in EJB ?
5. What is local interface. How values will be passed ?

University Sample Question

1. Web application program for Online Dictionary.

Exp No: 5

EXAM RESULT MANAGEMENT USING SERVLETS

Aim:

Use servlet to create user registration functionality to get registered with exam result section. The registration page take the following information from the user-User id, password, confirm password, full name, semester, roll no, email id, contact no. Registration servlet checks uniqueness of user id among all users and if found unique then only stores registration information in the database. Create a JSP based application which allows a user to edit his registration information. If login is successful the user authentication servlet creates the welcome message for the user in sessions scope and then forward the request to jsp page which handles the edit operation.

Description:

SERVLET

Servlets are server side components that provide a powerful mechanism for developing server side programs. Servlets provide component-based, platform-independent methods for building Web-based applications, without the performance limitations of CGI programs. Unlike proprietary server extension mechanisms (such as the Netscape Server API or Apache modules), servlets are server as well as platform-independent. Using servlets web developers can create fast and efficient server side application which can run on any servlet enabled web server. Servlets run entirely inside the Java Virtual Machine. Since the Servlet runs at server side so it does not checks the browser for compatibility. Servlets can access the entire family of Java APIs, including the JDBC API to access enterprise databases. Servlets can also access a library of HTTP-specific calls; receive all the benefits of the mature java language including portability, performance, reusability, and crash protection. Today servlets are the popular choice for building interactive web applications. Third-party servlet containers are available for Apache Web Server, Microsoft IIS, and others. Servlet containers are usually the components of web and application servers, such as BEA WebLogic Application Server, IBM WebSphere, Sun Java System Web Server, Sun Java System Application Server and others.

Servlets are not designed for specific protocols. It is different thing that they are most commonly used with the HTTP protocols Servlets uses the classes in the java

packages javax.servlet and javax.servlet.http. Servlets provides a way of creating the sophisticated server side extensions in a server as they follow the standard framework and use the highly portable java language.

Servlet Life Cycle

A servlet follows a certain life cycle. The servlet life cycle is managed by the servlet container. The life cycle contains the following steps:

The init() method :

The init method is designed to be called only once. It is called when the servlet is first created, and not called again for each user request. So, it is used for one-time initializations, just as with the init method of applets. The init () method simply creates or loads some data that will be used throughout the life of the servlet.

The init method definition looks like this:

```
public void init() throws ServletException
{
// Initialization code...
}
```

The service() method :

The service() method is the main method to perform the actual task. The servlet container (i.e. web server) calls the service() method to handle requests coming from the client(browsers) and to write the formatted response back to the client. Each time the server receives a request for a servlet, the server spawns a new thread and calls service. The service() method checks the HTTP request type (GET, POST, PUT, DELETE, etc.) and calls doGet, doPost, doPut, doDelete, etc. methods as appropriate.

```
public void service(ServletRequest request, ServletResponse response) throws
ServletException,IOException
{
}
```

The doGet() Method

A GET request results from a normal request for a URL or from an HTML form that has no METHOD specified and it should be handled by doGet() method.

public void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException,IOException
{
// Servlet code
}

The doPost() Method

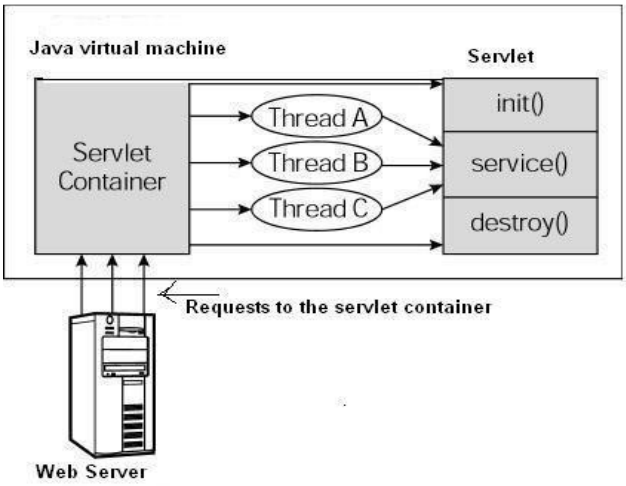
A POST request results from an HTML form that specifically lists POST as the METHOD and it should be handled by doPost() method.

public void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException,IOException
{
// Servlet code
}

The destroy() method :

The destroy() method is called only once at the end of the life cycle of a servlet. This method gives your servlet a chance to close database connections, halt background threads, write cookie lists or hit counts to disk, and perform other such cleanup activities.

public void destroy()
{
// Finalization code... }



Algorithm:

step1: Create Login page by using userid and password.

step2: For each new user allow him/her to register their details.

Step3: Registration page take the following information's full name, semester, roll no, email id, contact no user-User id, password, confirm password. Registration servlet checks uniqueness of userid among all users and if found unique then only stores registration information in the database.

Step3: After the successful Registration go back to the login page.

Step4 :When user enters the userid and password in the login page, It check whether the user is valid or not by checking the corresponding value with the database.

Step5: If he\she is an authenticated user then go to step 6 else go to step12.

Step6: create a session and the authentication servlet creates the welcome message for the user.

Step7: From this page user can select appropriate option he/she want. If option is edit profile then go to step 8.else the option is view result then go to step 10.Otherwise return back to the home page.

Step8: A signed in user can edit his/her registration information which can be entered at the time of registration if needed.

Step9: The new edited information can be updated in the database through update query. Then user can go back to the login page and to revisit the details if they want.

Step10: when signed user select the view result option a servlet request can be generated to retrieve data from the database. Through the servlet response the retrieved data from the database can be displayed to the user.

Step11 : When the user is logged out session has to be removed.

Step12: Return back to the login page.

Design

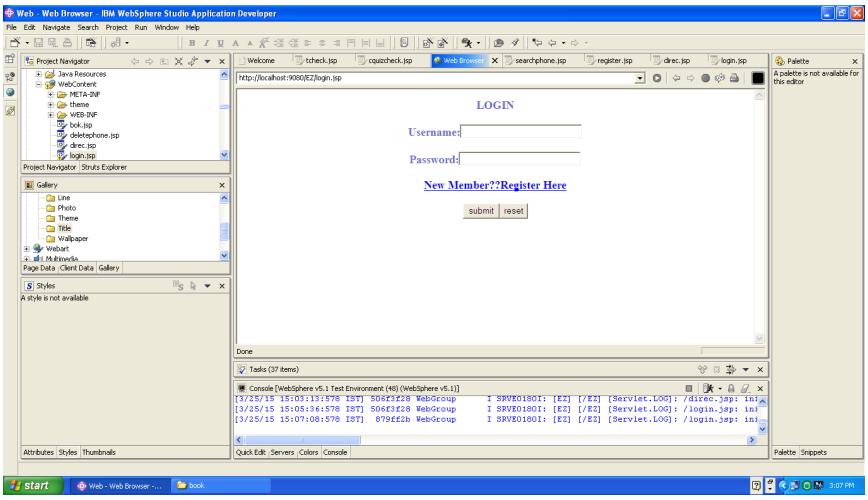


Fig5.1: Form for Login

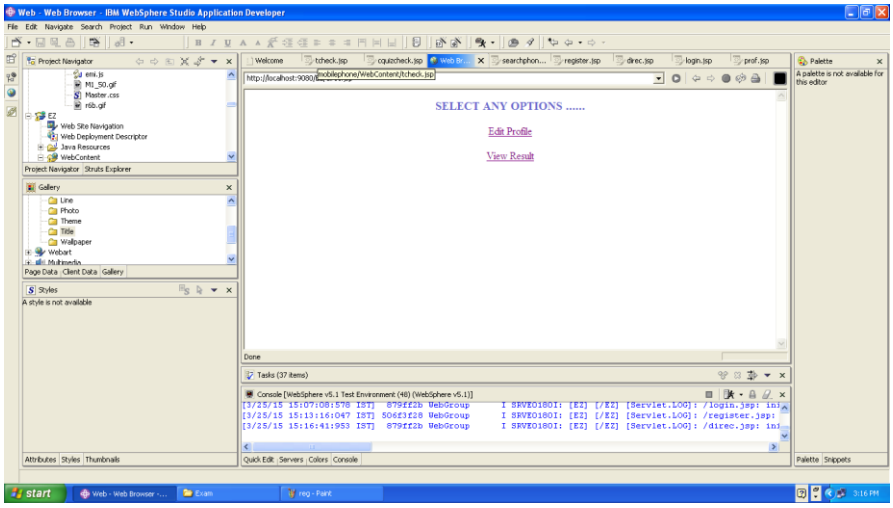


Fig: 5.2 Form for options

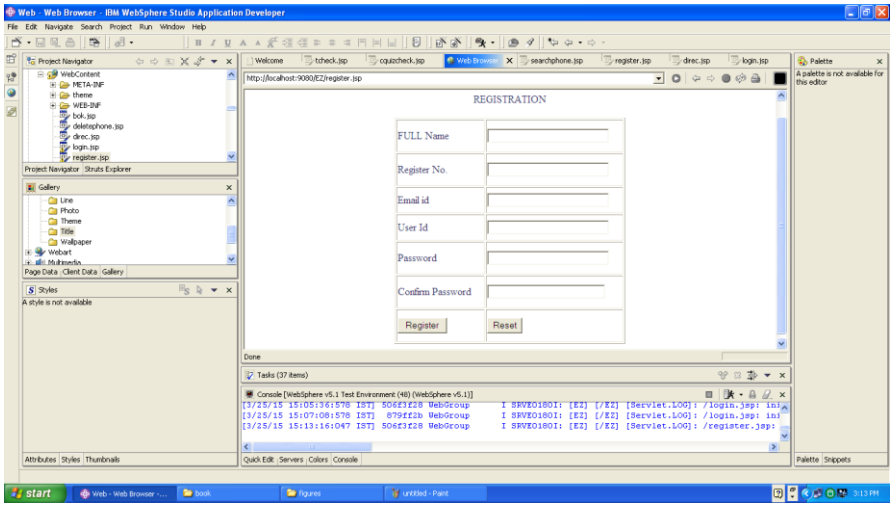


Fig 5.3: Form for registration

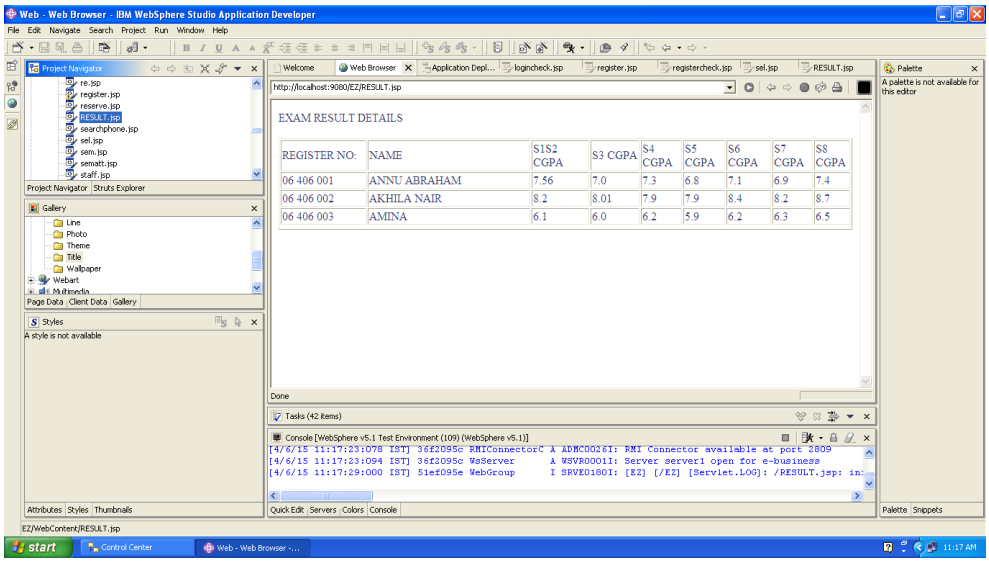


Fig 5.4: form for Exam result display

Result:

A website for implementing exam result management using servlets has been created successfully.

Viva Questions

1. Explain Servlet life cycle?
2. Why do we need constructor in servlet if we use the init ()?
3. How servlet is loaded?
4. What is the difference between GenericServlet and HttpServlet?
5. What is the difference between the doGet () and doPost ()?

University Sample Question

1. Student Fee management using Servlet.

Exp No: 6

COOKIES

Aim:

Write a JSP program to store current date, time in a cookie to display the last visited date, time in the web page upon reopening of the same page.

Description:

COOKIES

Cookies are text files stored on the client computer and they are kept for various information tracking purpose. JSP transparently supports HTTP cookies using underlying servlet technology. A web server can assign a unique session ID as a cookie to each web client and for subsequent requests from the client they can be recognized using the received cookie. Although cookies cannot carry viruses, and cannot install malware on the host computer, tracking cookies and especially third-party tracking cookies are commonly used as ways to compile long-term records of individuals' browsing histories—a potential privacy concern that prompted European and US law makers to take action in 2011.Cookies can also store passwords and forms a user has previously entered, such as a credit card number or an address. When a user accesses a website with a cookie function for the first time, a cookie is sent from server to the browser and stored with the browser in the local computer. Later when that user goes back to the same website, the website will recognize the user because of the stored cookie with the user's information.

Session cookie

A user's session cookie(also known as an in-memory cookie or transient cookie) for a website exists in temporary memory only while the user is reading and navigating the website. When an expiry date or validity interval is not set at cookie creation time, a session cookie is created.

Persistent cookie

A persistent cookie will outlast user sessions. If a persistent cookie has its Max-Age set to 1 year (for example), then, during that year, the initial value set in that cookie would be sent back to the server every time the user visited the server.

Secure cookie

A secure cookie has the secure attribute enabled and is only used via HTTPS, ensuring that the cookie is always encrypted when transmitting from client to server. This makes the cookie less likely to be exposed to cookie theft via eavesdropping.

HttpOnly cookie

The HttpOnly attribute is supported by most modern browsers.[19][20] On a supported browser, an HttpOnly cookie will be used only when transmitting HTTP (or HTTPS) requests, thus restricting access from other, non-HTTP APIs (such as JavaScript). This restriction mitigates but does not eliminate the threat of session cookie theft via cross-site scripting (XSS).

Third-party cookie

First-party cookies are cookies that belong to the same domain that is shown in the browser's address bar (or that belong to the sub domain of the domain in the address bar). Third-party cookies are cookies that belong to domains different from the one shown in the address bar. Web pages can feature content from third-party domains (such as banner adverts), which opens up the potential for tracking the user's browsing history.

Supercookie

A "supercookie" is a cookie with an origin of a Top-Level Domain (such as .com) or a Public Suffix (such as.co.uk). It is important that supercookies are blocked by browsers, due to the security holes they introduce. If unblocked, an attacker in control of a malicious website could set a supercookie and potentially disrupt or impersonate legitimate user requests to another website that shares the same Top-Level Domain or Public Suffix as the malicious website.

Zombie cookie

Some cookies are automatically recreated after a user has deleted them; these are called zombie cookies. This is accomplished by a script storing the content of the cookie in some other locations, such as the local storage available to Flash content, HTML5 storages and other client side mechanisms, and then recreating the cookie from backup stores when the cookie's absence is detected.

Algorithm:

Step1: Create Login page

Step2: For each new user allow him/her to register. Registration checks uniqueness of userid among all users and if found unique then only stores registration information in the database.

Step3: For each signed in user create a session. If login is successful the user visited time and date is stored in a cookie.

Step4: When the user has logged out session has to be removed.

Step5: When the user next login the site has to display the user's previously visited date and time.

Step6: Return to home page

Design

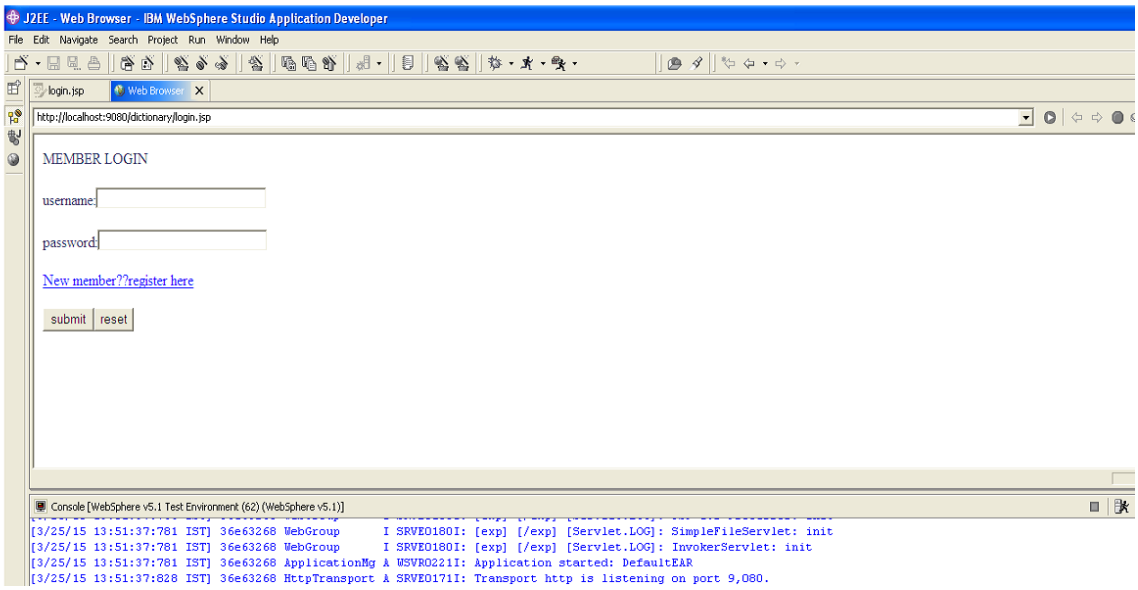


Fig 6.1: Login Form

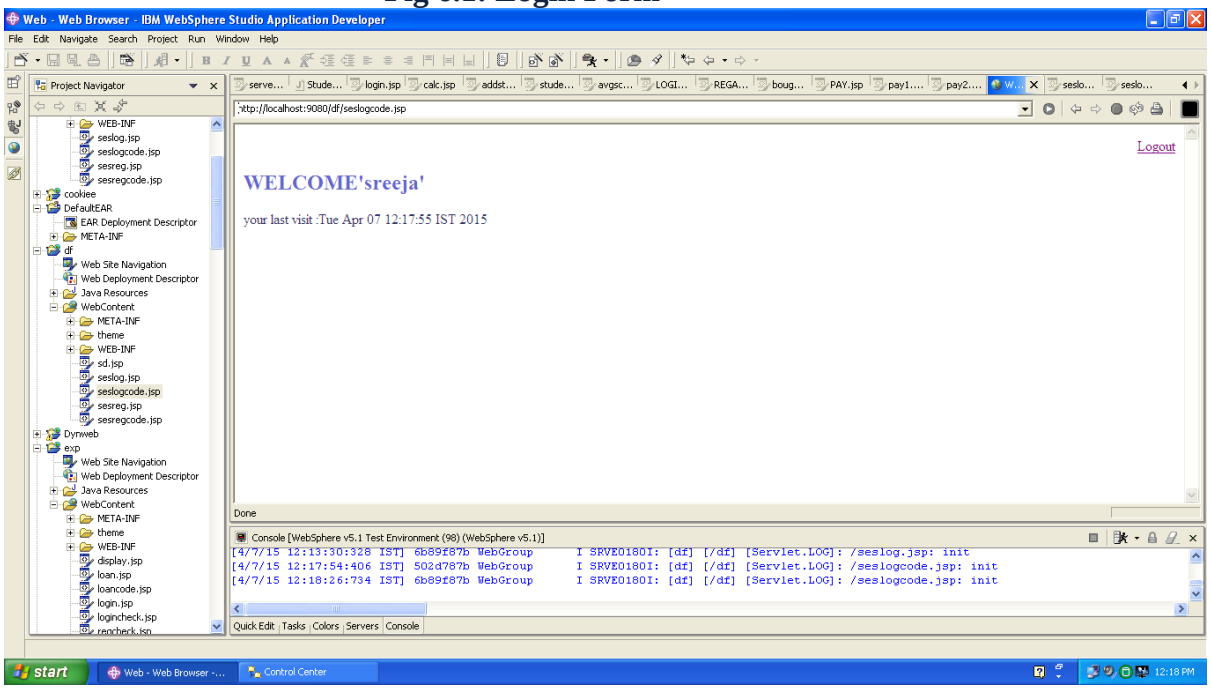


Fig 6.2 Form For displaying last visiting time of the user

Example:

Program to set the cookies for first and last name and to store the last visited date.

```
// Create cookies for first and last names.
  Cookie firstName = new Cookie("first_name",request.getParameter("first_name"));
  Cookie lastName = new Cookie("last_name",request.getParameter("last_name"));
 // Set expiry date after 24 Hrs for both the cookies.
firstName.setMaxAge(60*60*24);
lastName.setMaxAge(60*60*24);
// Add both the cookies in the response header.
response.addCookie( firstName );
response.addCookie( lastName );
request.getParameter("first_name")
request.getParameter("last_name")
//To store the last displaying time of last visit of user on the page using cookie
Cookie all[]=request.getCookies();
Cookie ck=null;
   if(all!=null)
for(int i=0;i<all.length;i++)
```

```
 String name=all[i].getName();
 if(name.equals("date"));
ck=all[i];
  break;
 if(ck==null)
ck=new Cookie("date",newjava.util.Date().toString());
 else
out.println("Your last visit : "+ck.getValue());
ck.setValue(new java.util.Date().toString());
response.addCookie(ck);
```

Result:

A website for implementing cookies has been created successfully.

Viva Questions

1. What is the difference between session and cookie?
2. How to create a cookie object?
3. How to set cookies in PHP?
4. List few types of cookie?
5. Compare session cookie and persistent cookie.

Exp No: 7

AIRLINE RESERVATION SYSTEM

Aim:

Create a portal for the airline reservation \billing system which allows

(a) Cancellation of reservation can be done hrs before departure.

(b) Cancellation done before

(i) 8 days of journey will lost only 5% of amount

(ii) 5 days of journey will lost only 10% of amount

(iii) 1 day up to 8 hrs will loss 15 % of amount

Algorithm:

Step 1: Start

Step 2:Create home page of an airline booking service having provisions for getting flight details, login and registration.

Step 3: Create a database to store details of passengers.

Step 4: On login a session is created.

Step 5: Now customer can book or cancel airline tickets and there is provisions for checking their reservations.

Step 6: Stop

Flight details

Step 1: Start

Step 2: Create a database to store flight details.

Step 3: Details of flight can get by entering travel route, date, time and airport.

Step 4: Stop

My Reservation

Step 1: Start

Step 2: Get the reservation details of passenger from database.

Step 3: Display it as table.

Step 4: Stop

Flight Reservation

Step 1: Start

Step 2: Create databases to store details of reservation and bank.

Step 2: Enter the flight details and travel date.

Step 3: Enter the no: of tickets needed and class.

Step 4.Then travel fare is displayed, amt.

Step 5: To reserve click reserve button and then enter the bank details.

Step 6: Issue ticket and reservation no: if account has enough balance.

Step 7: Stop

Flight Cancelation

Step 1: Start

Step 2: Enter the reservation code and reason for cancelling.

Step 3: Retrieve all information from database about given reservation code.

Step 4: Find difference between departure date and cancellation date, d.

Step 5: If d > 8 days then reduce 5% of actual fare.

Amt=amt-(0.05*amt)

Step 6: Else if d<8 & d>5, then reduce 10% of actual fare.

Amt=amt-(0.10*amt)

Step 7: Else if d<5 & d>1 then reduce 12% of actual fare.

Amt=amt-(0.12*amt)

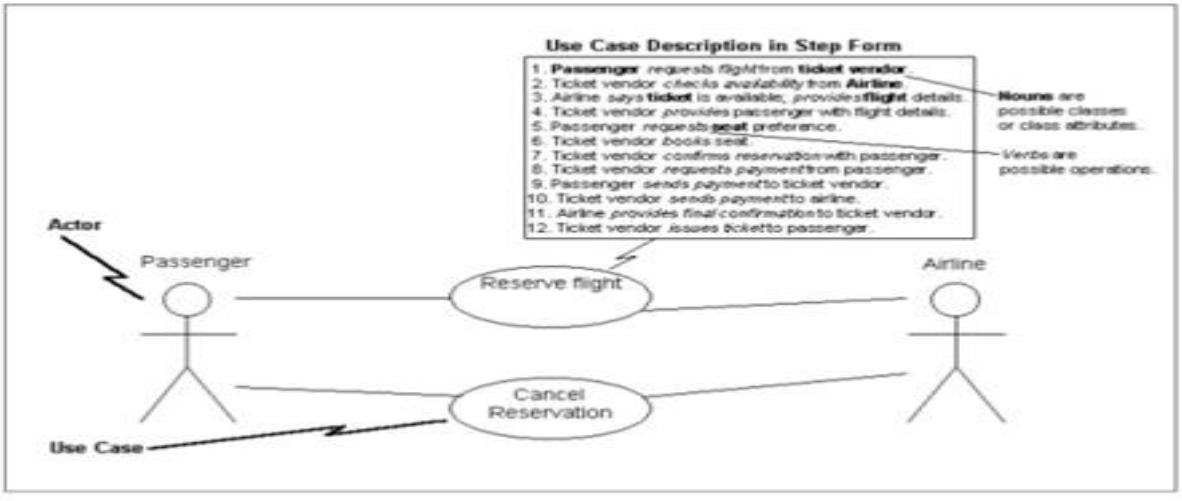
Step 8: Else if d<1 & d>8 hour then reduce 15% of actual fare.

Amt=amt-(0.15*amt)

Step 9: Else print not refundable.

Step 10: Stop

Use Case Diagram



Design

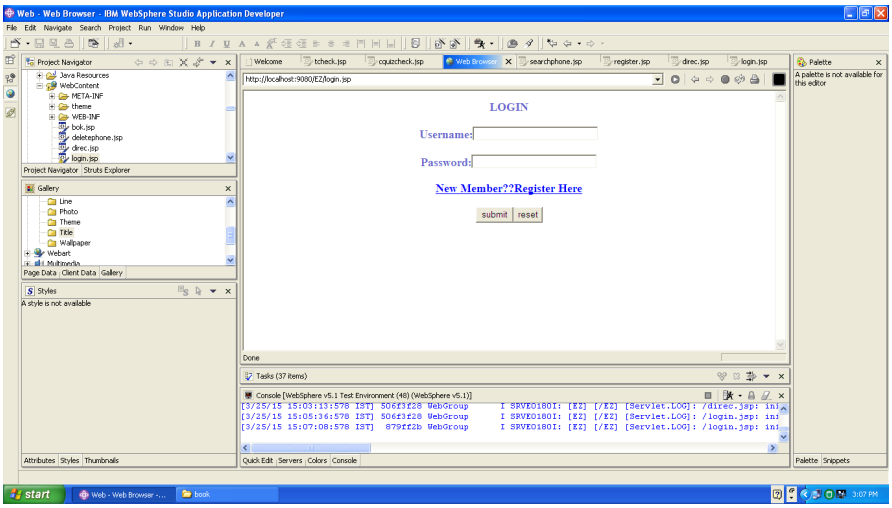


Fig 7.1: Form for Login

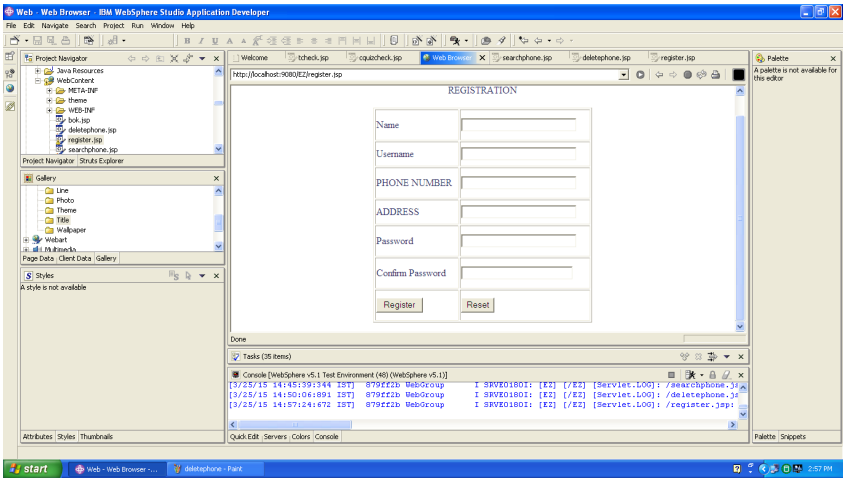


Fig7.2: Form for registration

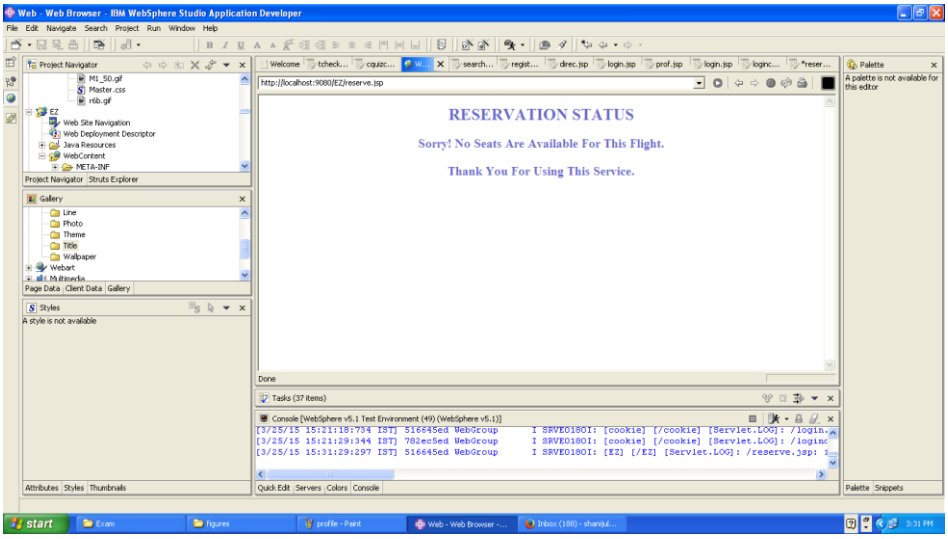


Fig 7.3: Form for reservation status

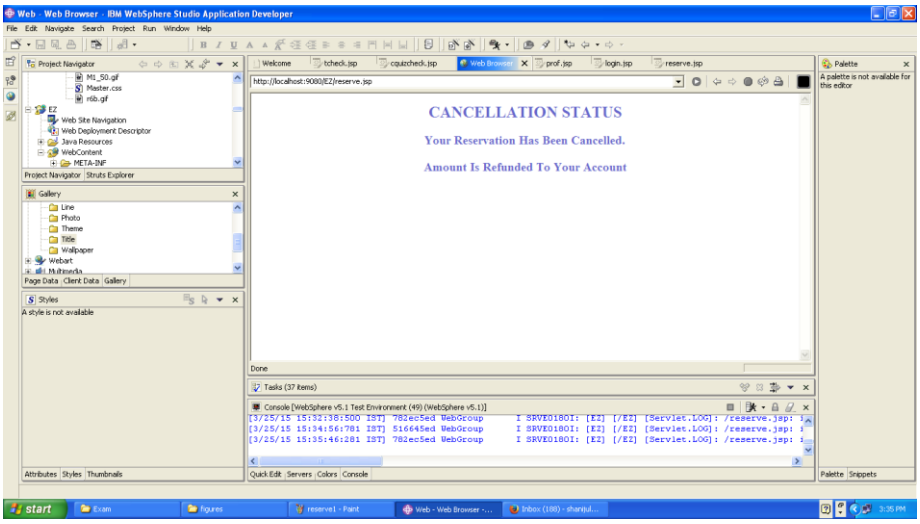


Fig7.4 Form for Cancellation status

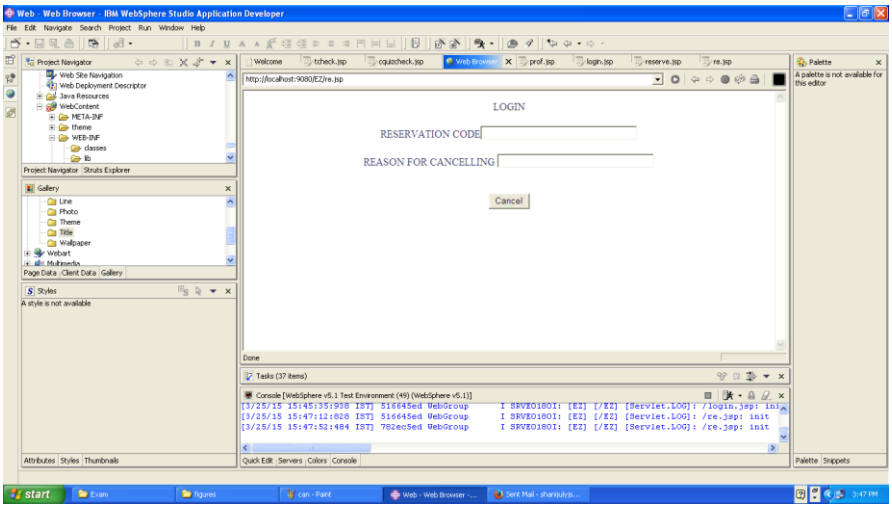


Fig 7.5 Form for cancellation

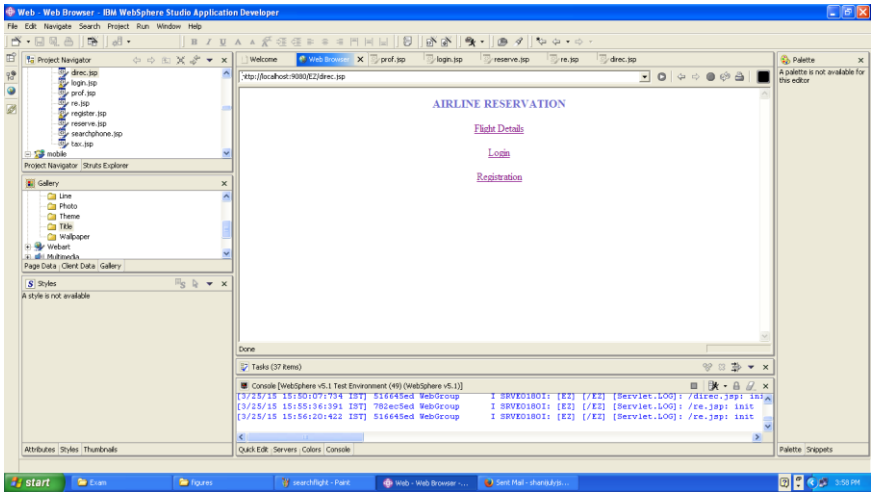


Fig7.6 Form for options

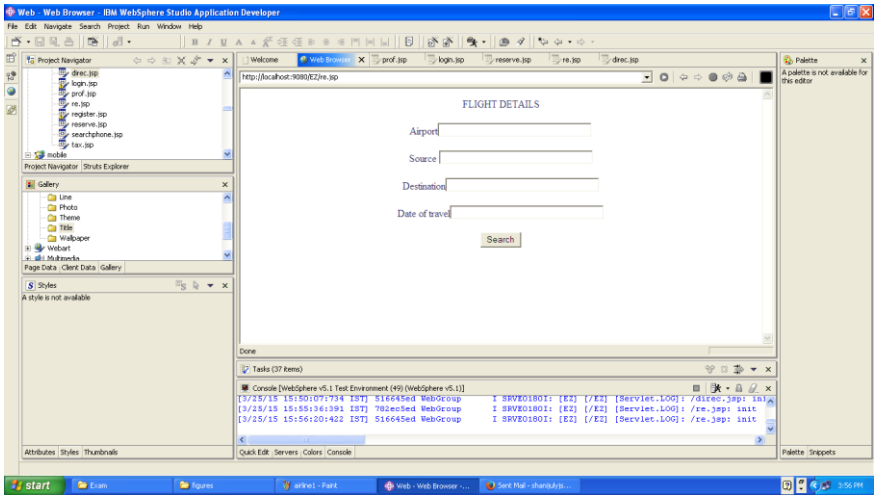


Fig7.7 Form for reservation

Result:

A website for implementing airline reservation has been created successfully.

Viva Questions

1. What is Xquery?
2. What are XML indexes?
3. What are secondary XML indexes?
4. What is the OPENXML statement in SQL Server?
5. What is XMLA?

University Sample Questions

1. Implement Rent-a-Car reservation System
2. Implement Railway reservation system

Exp No: 8

ONLINE LIBRARY MANAGEMENT

Aim:

Create a web application for library management which allows

1. To provide new memberships and maintain the details of members.
2. To update the database.
3. To maintain the borrowing list, returning list, and fine list.

Algorithm:

Step1: Create a login page for a user and admin to login and if new user he can register by clicking on the register here link.

Step2: When a registered user logs in through username and password ,it is checked in the database for existence. If found then the user can perform the following operations

i. Borrow books

ii .Return books

Step3:If a new user is registering he is given four tickets and a unique id.

Step4:If an admin logs in he can add new books in the library and can delete the books.

Algorithm For Borrow Books:

Step1:After a valid user logs in he/she can search the books available in the library for borrowing Step2: A user can select the required books.

Step3:Before lending the book the borrowing list of the user is verified to see whether the user has the tickets free for taking the book.

Step4:If the tickets are available the user can take the book , or else can take only the books according to the tickets available to them.

Step5:After taking books, the remaining number of tickets has to be recorded for that user in the database.

Step6:When a book is being borrowed to a user the current date and due date has to be recorded in the database.

Algorithm For Receiving Books:

Step1:When the user has to return a book the due date and current date is checked. If the due date is not expired then book is returned with no fine.

Step2:If the due date has expired then the user has to pay fine.

Step3:Calculate the days passed after due date as diff=currentdate-duedate and the amount to be paid as fine, fineamt=diff*5.

Step4:The calculated amount has to be collected from the user.

Step5:Record the fine details in the database.

Design

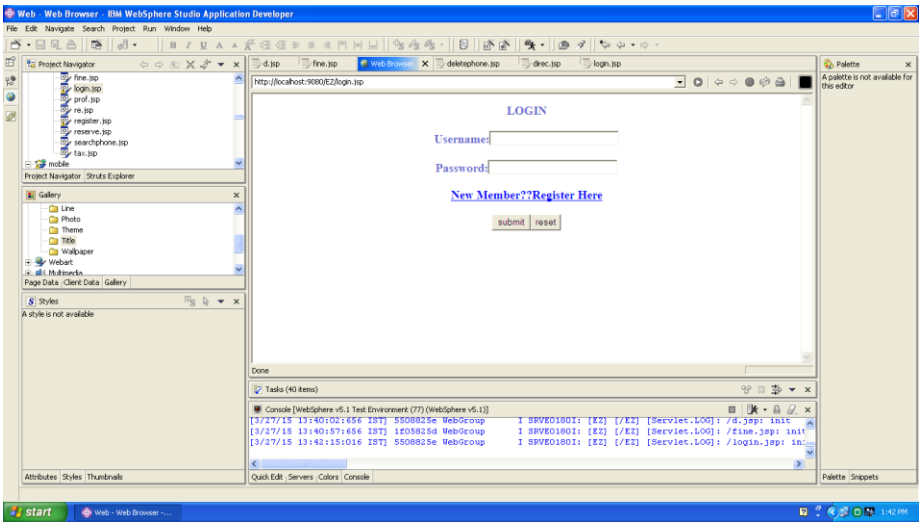


Fig 8.1: Form for Login

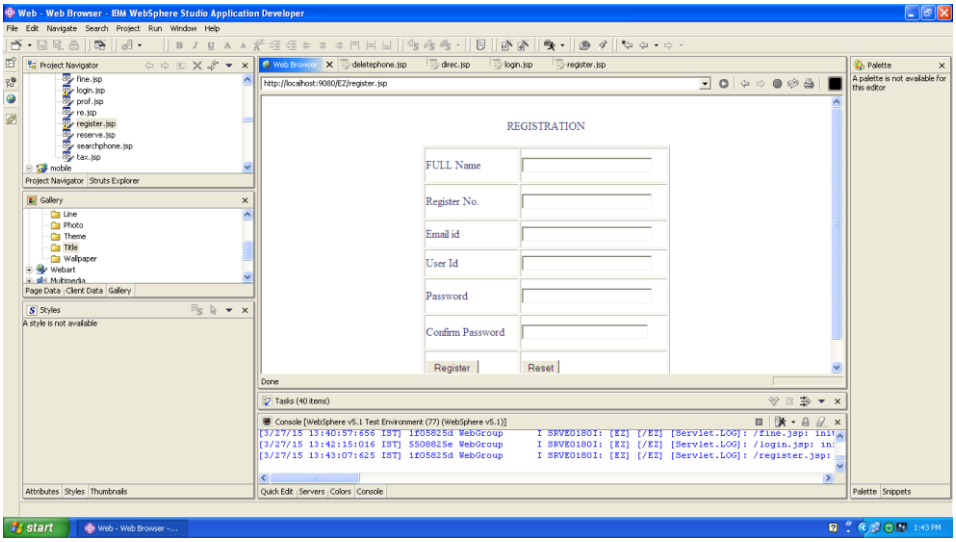


Fig 8.2: Form for registration

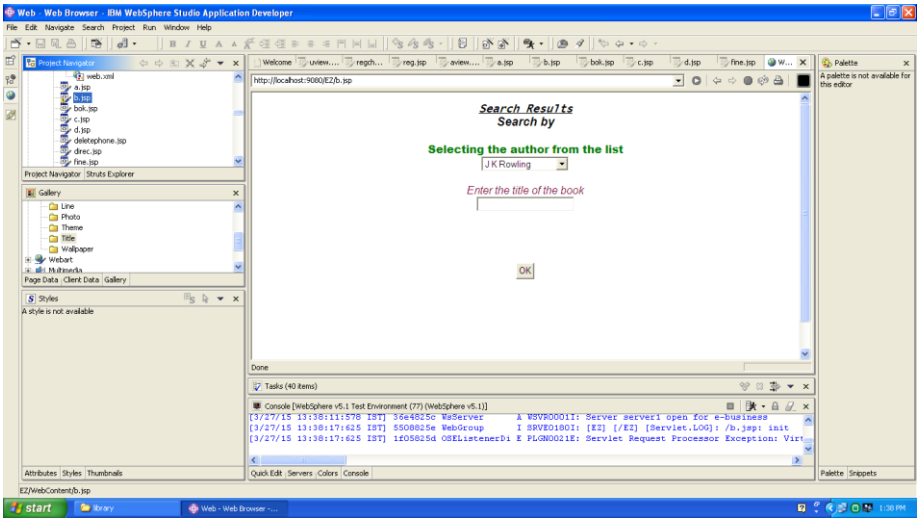


Fig 8.3: Form for searching books

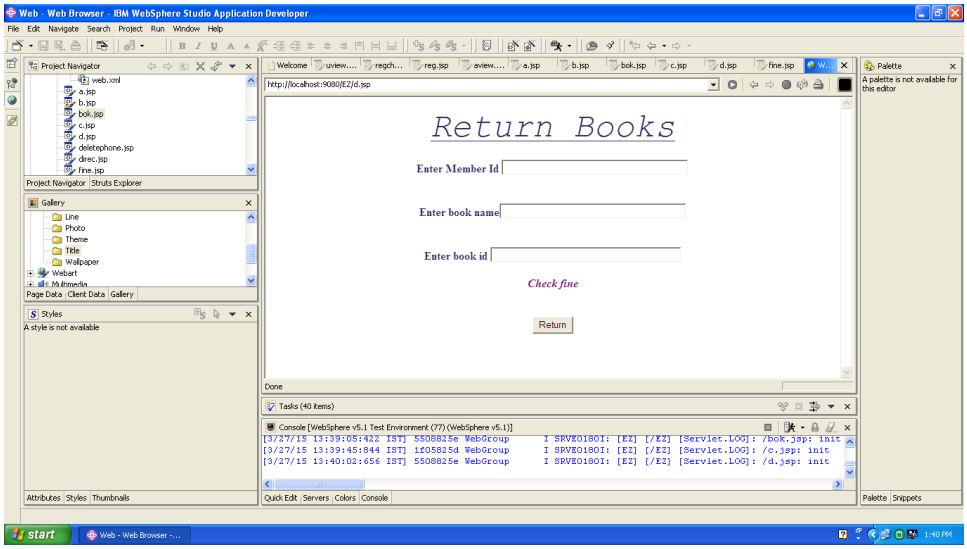


Fig 8.4: Form for book return

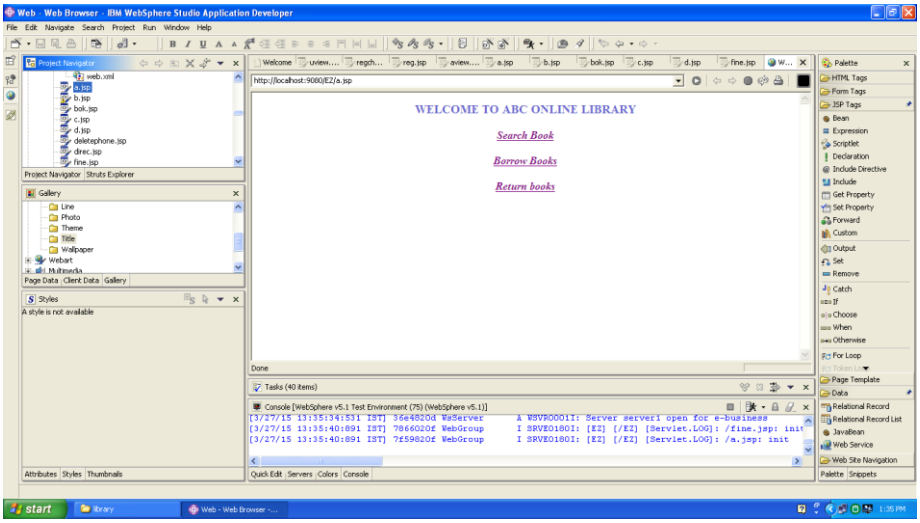


Fig 8.5: Form for options

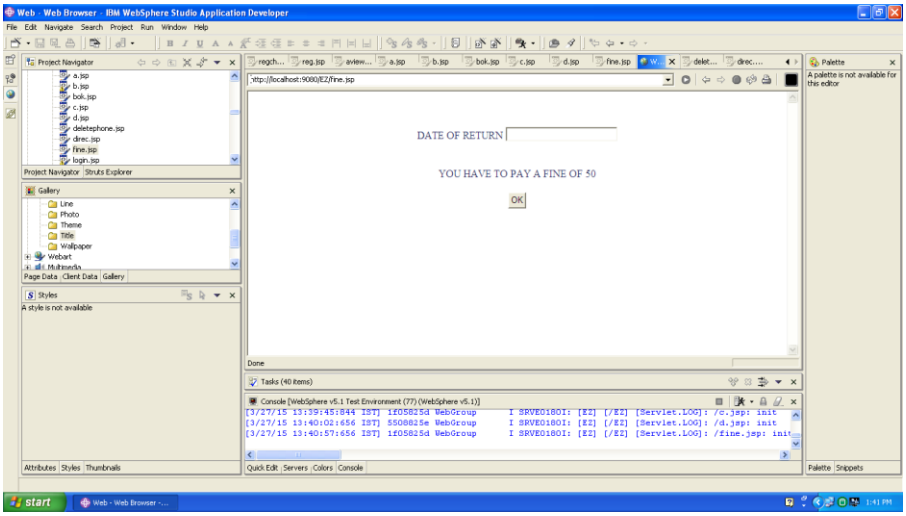


Fig8.6: Form for fine

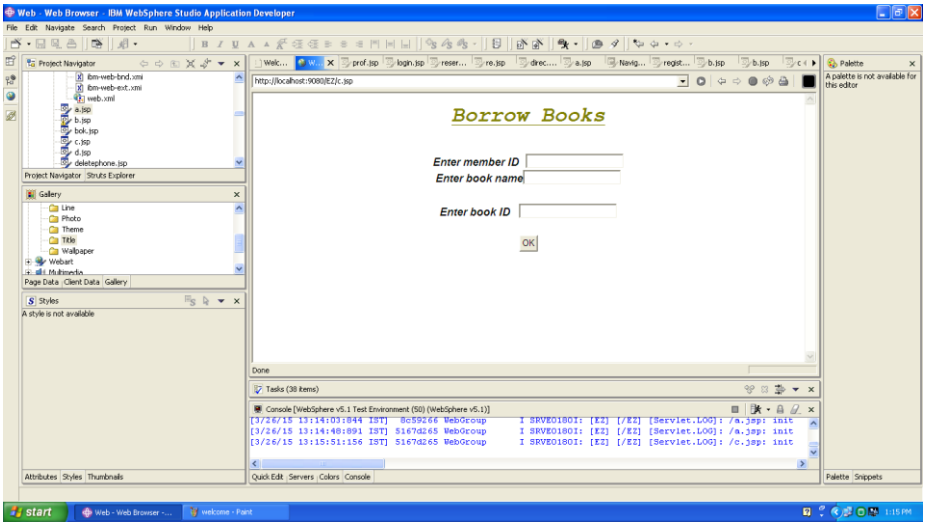


Fig 8.7 Form for borrowing books

Result:

A website for implementing online library has been created successfully.

Viva Questions

1. What is Xquery?
2. What are XML indexes?
3. What are secondary XML indexes?
4. What is the OPENXML statement in SQL Server?
5. What is XMLA?

University Sample Question

1. Implement a Tour Management System

Exp No: 9

E-VOTING

Aim:

To create a portal for voting your favorite star singer, super dancer. The portal must record the IP address, time of voting, visitor count and denial of acceptance after the deadline of voting time. The number of votes got for each contestant should be displayed on a bar chart and overall performance on a pie chart.

Description:

IP ADDRESS

For addresses the designers of TCP/IP can choose a scheme analogous to physical network addressing in which each host on the internet is assigned a 32-bit integer address called its internet protocol address or IP address.

Returns the Internet Protocol (IP) address of the client or last proxy that sent the request. For HTTP servlets, same as the value of the CGI variable REMOTE_ADDR.

To get the current system date and time

```
<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
      <title>JSP Page</title>
      <%@ page import="java.util.*"  %>
   </head>
   <body>
      <h2>The date is::</h2>
      <% Date date=new Date();
      out.println(date); %>
   </body>
</html>
```

Algorithm:

Step 1: Create home page which contain buttons for logging in as User or Admin.

Step 2: If it is a new user, he can register by clicking the link available on the home page.

Step 3: After login user can enter into Star Singer and Super Dancer and can vote for their favorite contestant from the list. Along with this the portal must record the IP address, time of voting and count the visitor.

Step 4: If a user is logged late his/her vote has to be denied.

Step 5: When logged in as admin, he could see the number of visitors and the votes received by each contestant

Step 6: On clicking view details, it gives the details of people who have voted for that person, their ip address, date and time at which the vote was cast.

Step 7: With the number of visitors and number of votes for each contestant ,the pie chart and bar chart of performance is shown

Step 8: Provide links to return to home page

Design

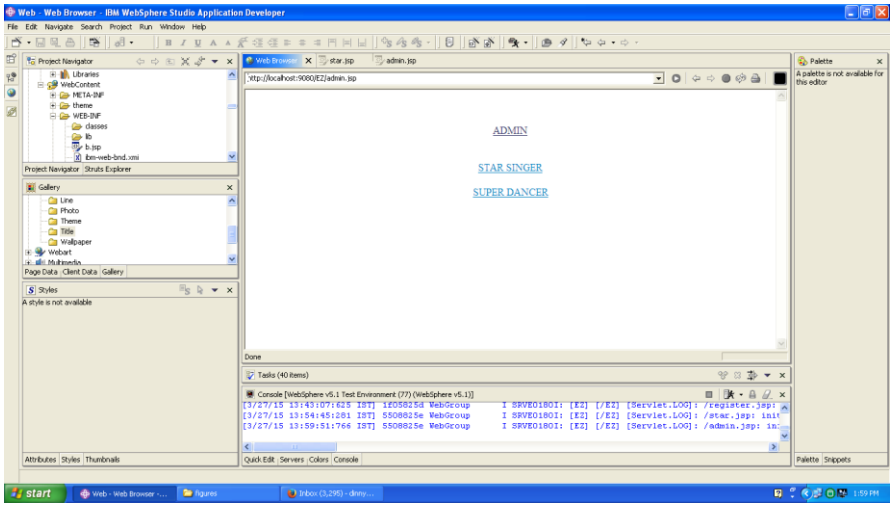


Fig 9.1 Form for options

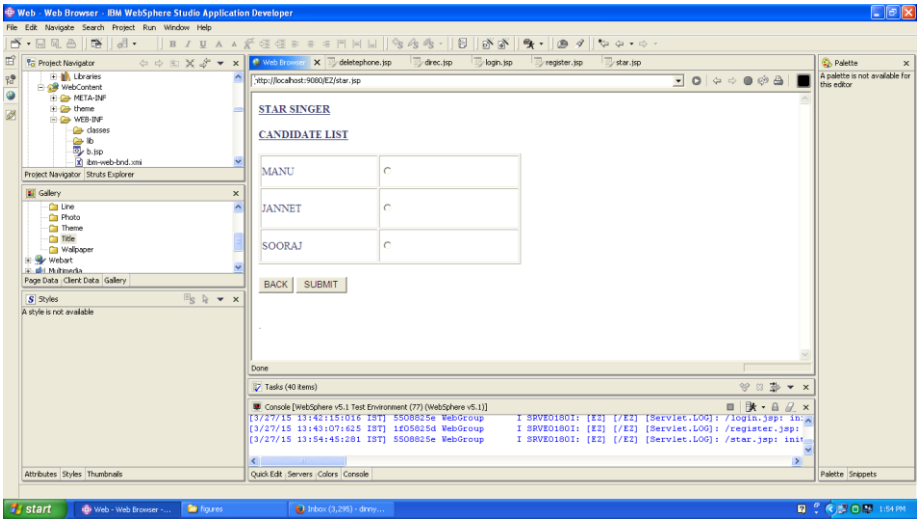


Fig 9.2 :Form for voting

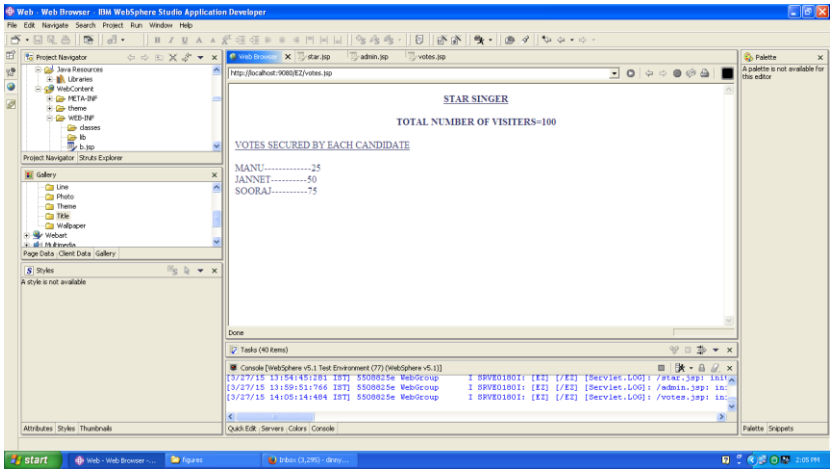


Fig 9.3 Form for viewing total votes

Result:

A website for implementing e-voting has been created successfully.

Viva Questions

1. What is transact-SQL? Describe its types?
2. How to return IP address of the client or last proxy that sent the request.
3. What is XPath ?
4. Define data, entity, domain and referential integrity.
5. What is XPath ?

University Sample Question

1. Portal for Dynamic Online Quiz

Exp No: 10

ATTENDANCE MONITORING SYSTEM

Aim:

To develop an attendance monitoring system where the staff member have provision for entering the attendance of a particular class, students can view their individual attendance for a specified day/period and the staff advisor can view the details of the whole class for a day/period and must generate the consolidated report.

Algorithm:

Step 1 : Start

Step 2 : Every user must have a user Id and a password to access the details.

Step 3 : Only valid user is allowed to get the attendance.

Step 4 :Attendance is calculated daily at the end and updated.

Step 5 : The staff advisor will generate the consolidated report of the attendance.

Step 6 : Stop

Algorithm For Staff Member

Step 1 : Start

Step 2 : Staff member must have valid user Id and password to login.

Step 3 :If a wrong Id or password is given again it asks for valid Id and password.

Step 4 : Valid staff member can view desired accounts.

Step 5 : He have the permission to enter attendance of particular class.

Step6 : He can add/alter a student's attendance before submission.

Step 7 : Stop

Algorithm For Students

Step 1 : Start

Step 2 : Student member must have valid user Id and password to login.

Step 3 :If a wrong Id or password is given again it asks for valid Id and password.

Step 4 : A student can view his details.

Step 5 : He can view the reports generated by the staff advisor about his(students) attendance status or specified day/period.

Step 6 : Stop

Algorithm For Staff Advisor

Step 1 : Start

Step 2 : Staff advisor must have valid user Id and password to login.

Step 3 :If a wrong Id or password is given again it asks for valid Id and password.

Step 4 : He can view the details of the students of the whole class for a day/period .

Step 5 : He can generate report for the students about their attendance percentage using the formula hours engaged/total hours.

Step 6 : Stop

Design

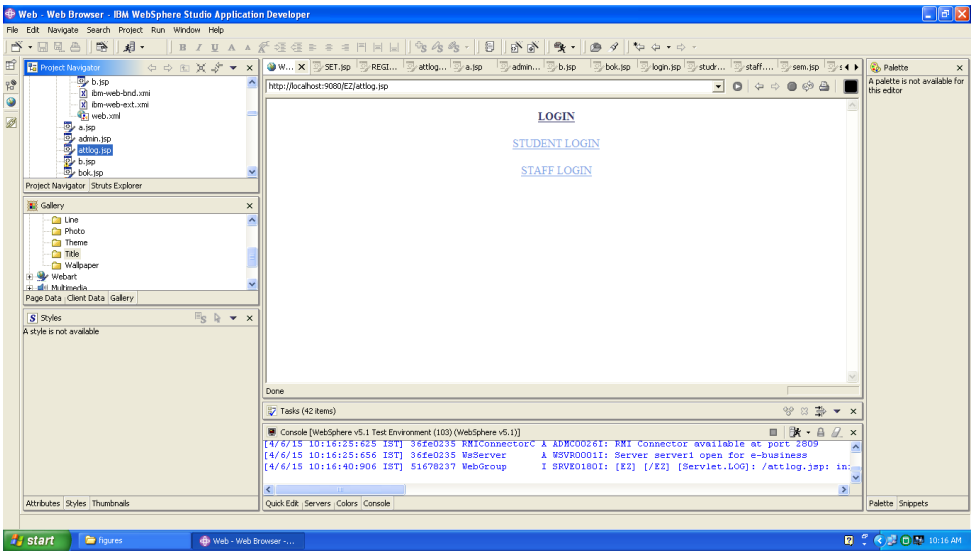


Fig 10.1 Form for Login option

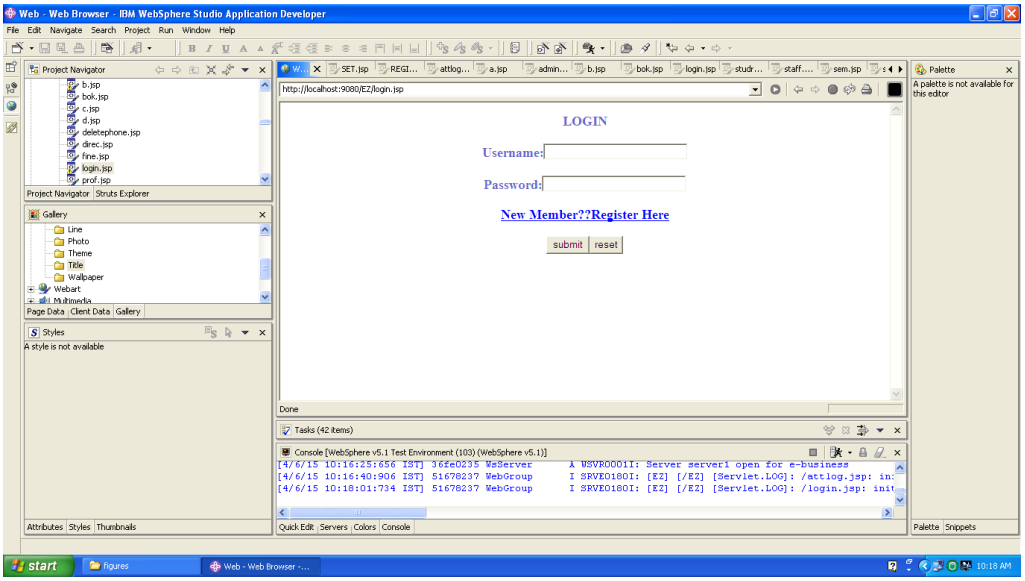


Fig 10.2 Form for Login

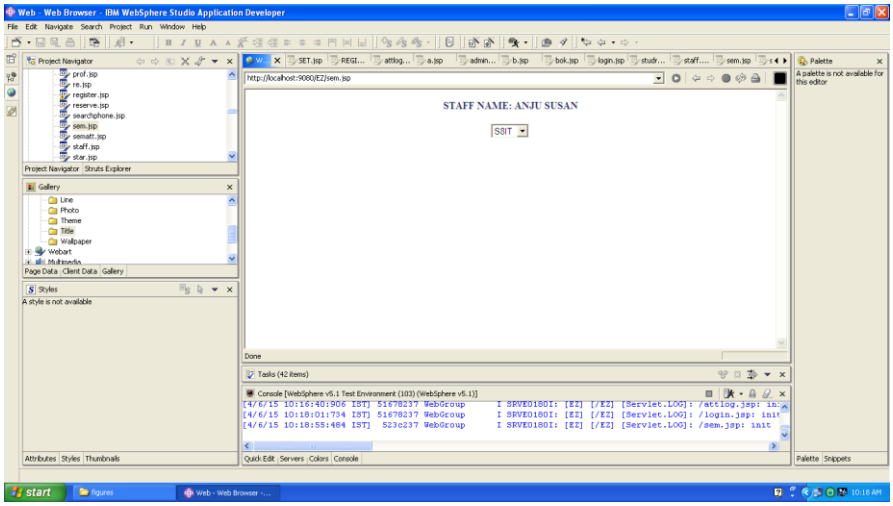


Fig 10.3 Form for selecting semester

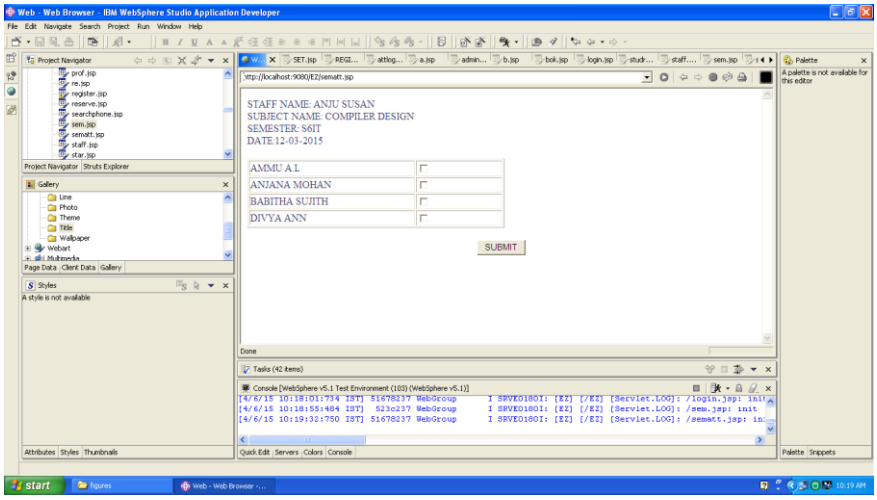


Fig 10.4 Form for marking attendance

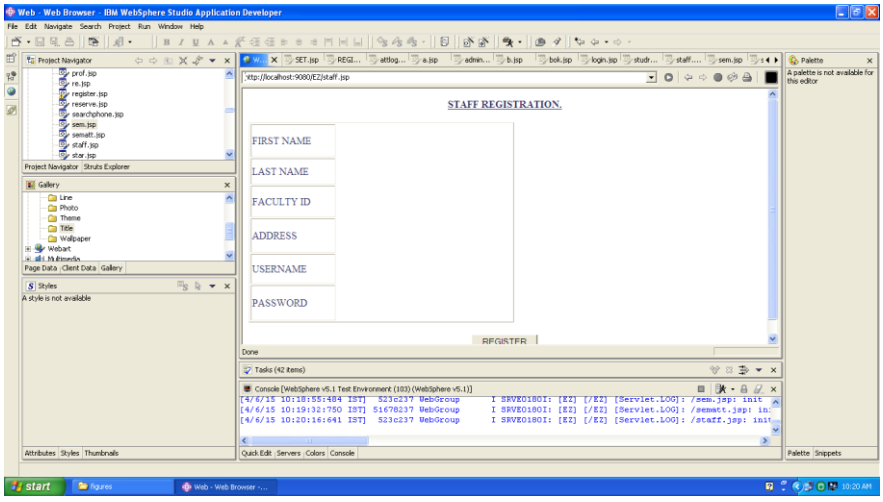


Fig 10.5 Form for staff registration

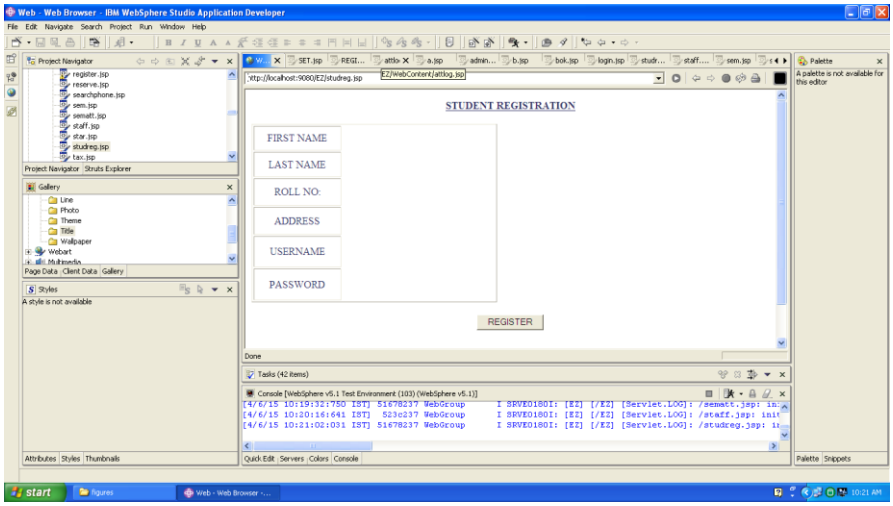


Fig 10.6 Form for student registration

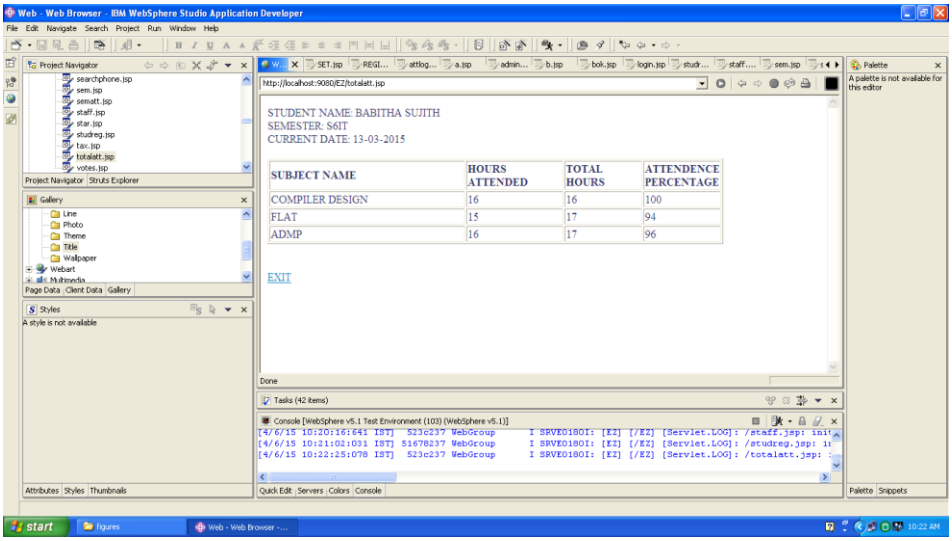


Fig 10.7 Form for viewing total attendance

Result:

A website for implementing attendance monitoring system has been created successfully.

Viva Questions

1. Difference between DTD and XML Schema
2. What is meaning of well formed XML ?
3. What is a CDATA section in XML?
4. What is XML namespace? Why it's important?

University Sample Question

1. Feedback Management System