JAKARTA SERVER PAGES, JSP LIFE-CYCLE AND ITS PHASES

Prof. Hardik Parmar
Assistant Professor
Information Technology Department
Dharmsinh Desai University, Nadiad

JAKARTA SERVER PAGES

Jakarta Server Pages (**JSP**; formerly **Java Server Pages**) is a collection of technologies that helps software developers create dynamically generated web pages based on HTML, XML, SOAP, or other document types.

Released in 1999 by Sun Microsystems, JSP is similar to PHP and ASP, but uses the Java programming language.

To deploy and run Jakarta Server Pages, a compatible web server with a servlet container, such as Apache Tomcat or Jetty, is required.

JSP pages are HTML pages with embedded code that allows to access data from Java code running on the server.

JAKARTA SERVER PAGES

Jakarta Server Pages (JSP) is a technology used for creating dynamic web pages in Java-based web applications.

It's a part of the Jakarta EE (formerly Java EE) platform, which provides a set of specifications for building enterprise-level, scalable, and reliable Java web applications.

JSP contains an extension of **.jsp**

BENEFITS OF JSP

- Write Once, Run Anywhere properties
- High quality tool support
- Separation of Roles
- Reuse of components and tag libraries
- Separation of dynamic and static content
- Support for actions, expressions, and scripting
- Web access layer for N-tier enterprise application architecture(s)

WHAT IS JSP PAGE?

A JSP page is a text-based document that describes how to process a *request* to create a *response*. The description intermixes template data with dynamic actions and leverages the Java Platform.

JSP technology supports a number of different paradigms for authoring dynamic content. The key features of Jakarta Server Pages are:

- ❑ Standard directives
- ❑ Standard actions
- ❑ Scripting elements
- ❑ Tag Extension mechanism
- ❑ Template content

JSP VS. SERVLET

Relationship between JSP & Servlet

JSP is an interface on top of Servlets.

A JSP program is compiled into a Java servlet before execution.

Why JSP?

- ✓ Easier to write than servlets.
- ✓ Designers can write HTML, programmers can write Java portions

Servlets came first, followed by JSP

JSP VS. SERVLET

Advantages of Servlets

1. Performance
   - Get loaded upon first request
   - Multithreading
     - ✓ Each request runs in its own separate thread
2. Simplicity
   - Run inside controlled server environment
   - No specific client software is needed: Web browser is enough

COMPONENTS AND CONTAINERS

JSP pages and servlet classes are collectively referred to as *web components*. JSP pages are delivered to a *container* that provides the services indicated in the *JSP Component Contract*.

The separation of components from containers allows the reuse of components, with quality-of-service features provided by the container.

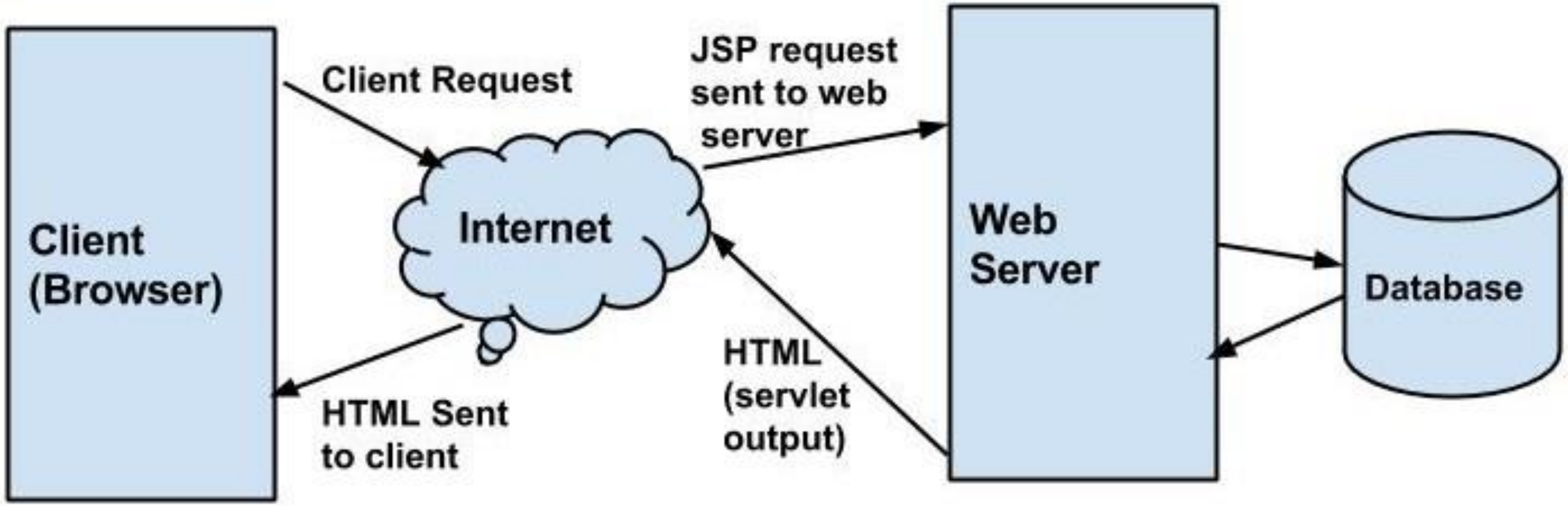
JSP ARCHITECTURE

Jakarta Server Pages are part of a 3-tier architecture.

A server(generally referred to as application or web server) supports the Jakarta Server Pages.

This server will act as a mediator between the client browser and a database.

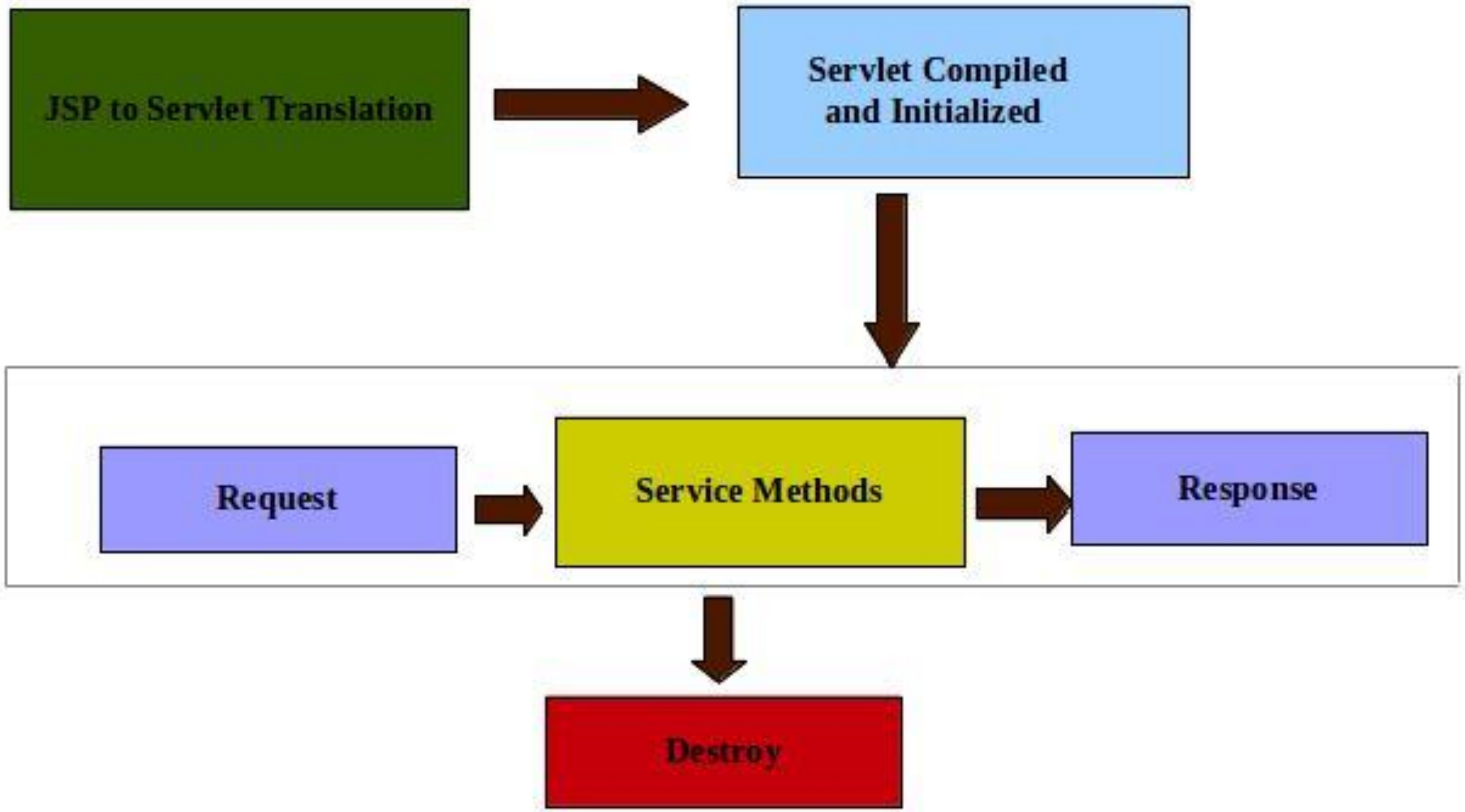
JSP ARCHITECTURE



JSP ARCHITECTURE FLOW

1. The user goes to a JSP page and makes the request via internet in user's web browser.
2. The JSP request is sent to the Web Server.
3. Web server accepts the requested **.jsp** file and passes the JSP file to the JSP Servlet Engine.
4. If the JSP file has been called the first time then the JSP file is parsed otherwise servlet is instantiated. The next step is to generate a servlet from the JSP file. The generated servlet output is sent via the Internet form web server to users web browser.
5. Now in last step, HTML results are displayed on the users web browser.

JSP LIFE CYCLE



JSP to Servlet Translation

Servlet Compiled and Initialized

Request

Service Methods

Response

Destroy

JSP LIFE CYCLE

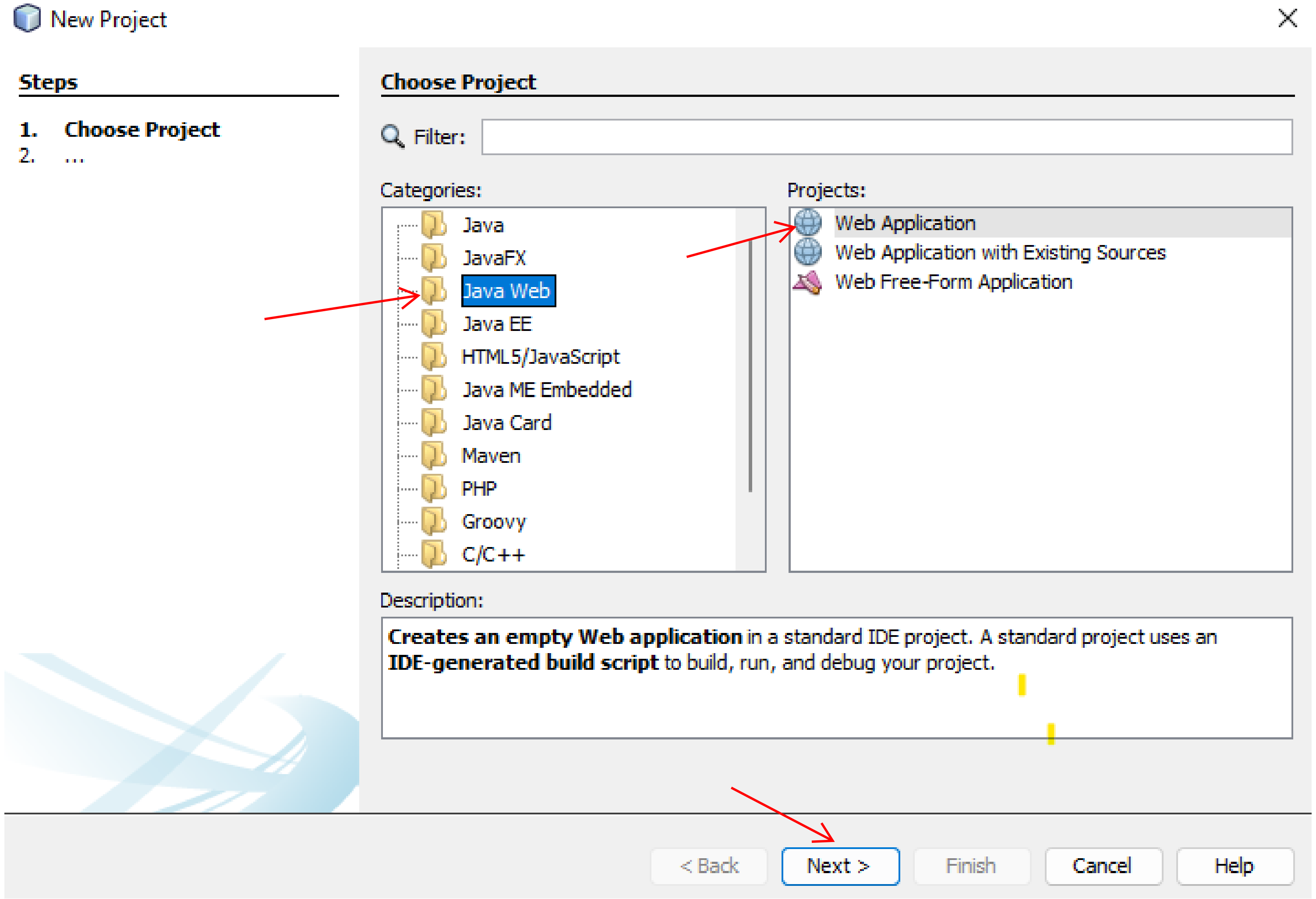
PHASE	DESCRIPTION
Translation	JSP container parses the JSP pages. It then translate the JSP pages to generate corresponding servlet source code. If JSP file name is hello.jsp, usually it is named as hello_jsp.java by the container.
Page Compilation	If the translation is successful, the generated java file is then compiled by the container.
Class Loading	Once JSP is compiled as servlet class, its lifecycle is similar to servlet. The compiled class is then loaded into the memory
Instance Creation	Once JSP class is loaded into memory, its object is instantiated by the container.

JSP LIFE CYCLE

This life-cycle starts after Compilation Step

PHASE	DESCRIPTION
Call jspInit() or Initialization	During this phase the JSP class is initialized transformed from a normal class to servlet. Once initialization is over, ServletConfig and ServletContext objects become accessible to JSP class. Method jspInit() is called only once in JSP lifecycle. It initializes config params.
Call _jspService or Request Processing	This method is called for each client request.
Call jspDestroy or Destroy	This is the last phase and this method is called when the container decides to unload JSP from memory.

WRITING A JSP



Steps

1. **Choose Project**
2. ...

Choose Project

Filter:

Categories:

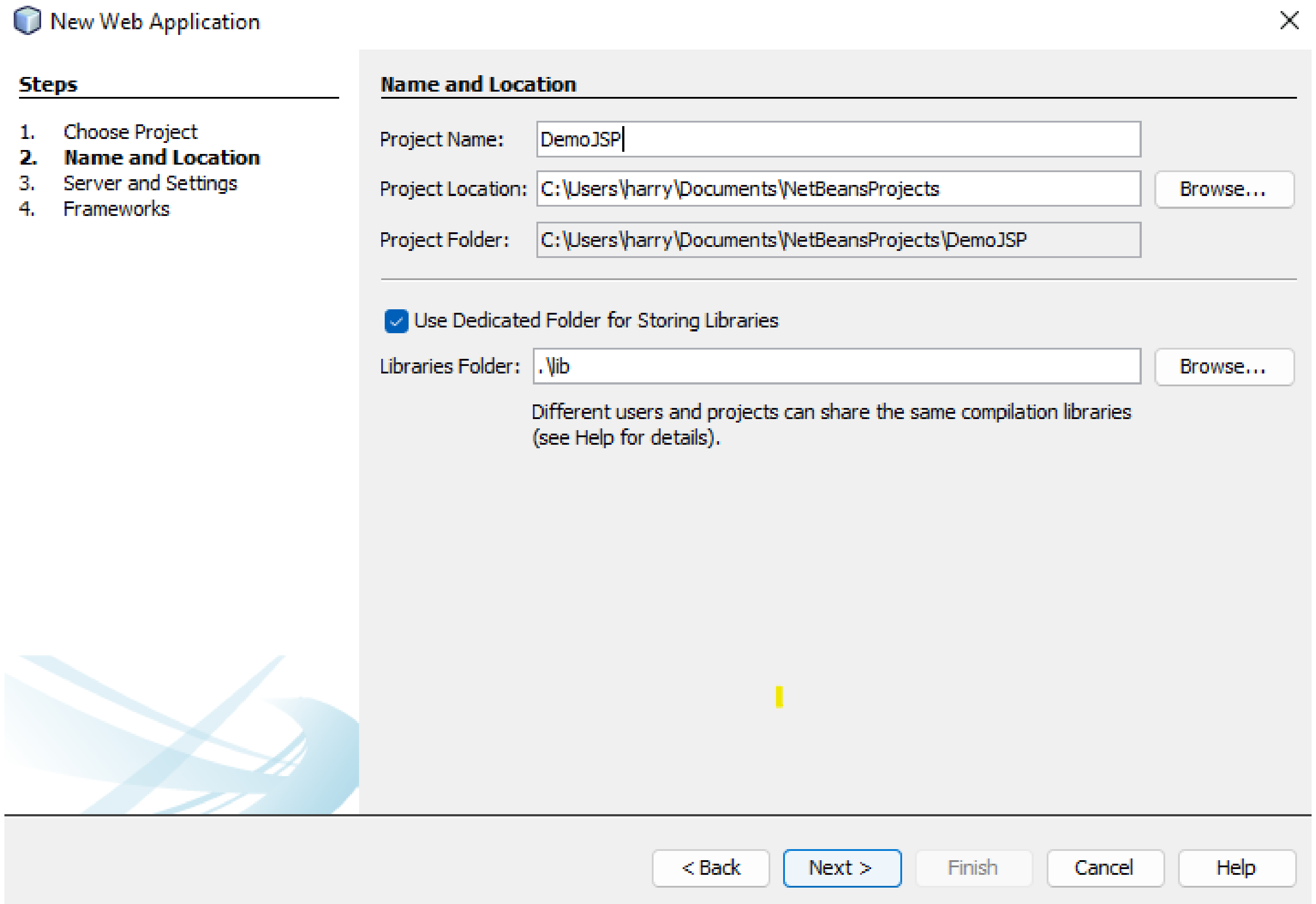
Projects:

Web Application
Web Application with Existing Sources
Web Free-Form Application

Description:

Creates an empty Web application in a standard IDE project. A standard project uses an **IDE-generated build script** to build, run, and debug your project.

WRITING A JSP



New Web Application

Steps

1. Choose Project
2. **Name and Location**
3. Server and Settings
4. Frameworks

Name and Location

Project Name: DemoJSP

Project Location: C:\Users\harry\Documents\NetBeansProjects

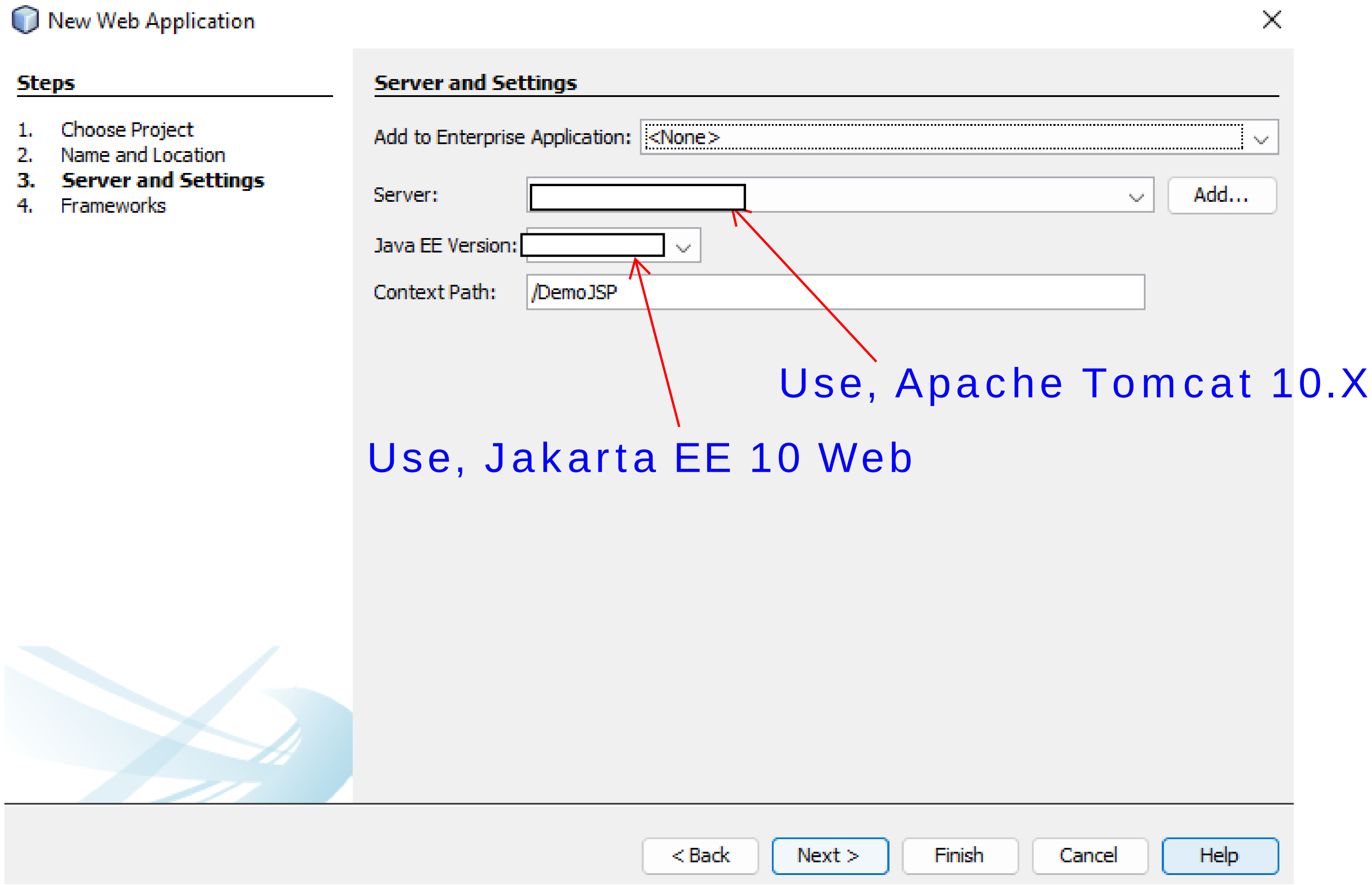
Project Folder: C:\Users\harry\Documents\NetBeansProjects\DemoJSP

☑ Use Dedicated Folder for Storing Libraries

Libraries Folder: .\lib

Different users and projects can share the same compilation libraries (see Help for details).

WRITING A JSP



New Web Application

Steps

1. Choose Project
2. Name and Location
3. **Server and Settings**
4. Frameworks

Server and Settings

Add to Enterprise Application: <None>

Server:

Java EE Version:

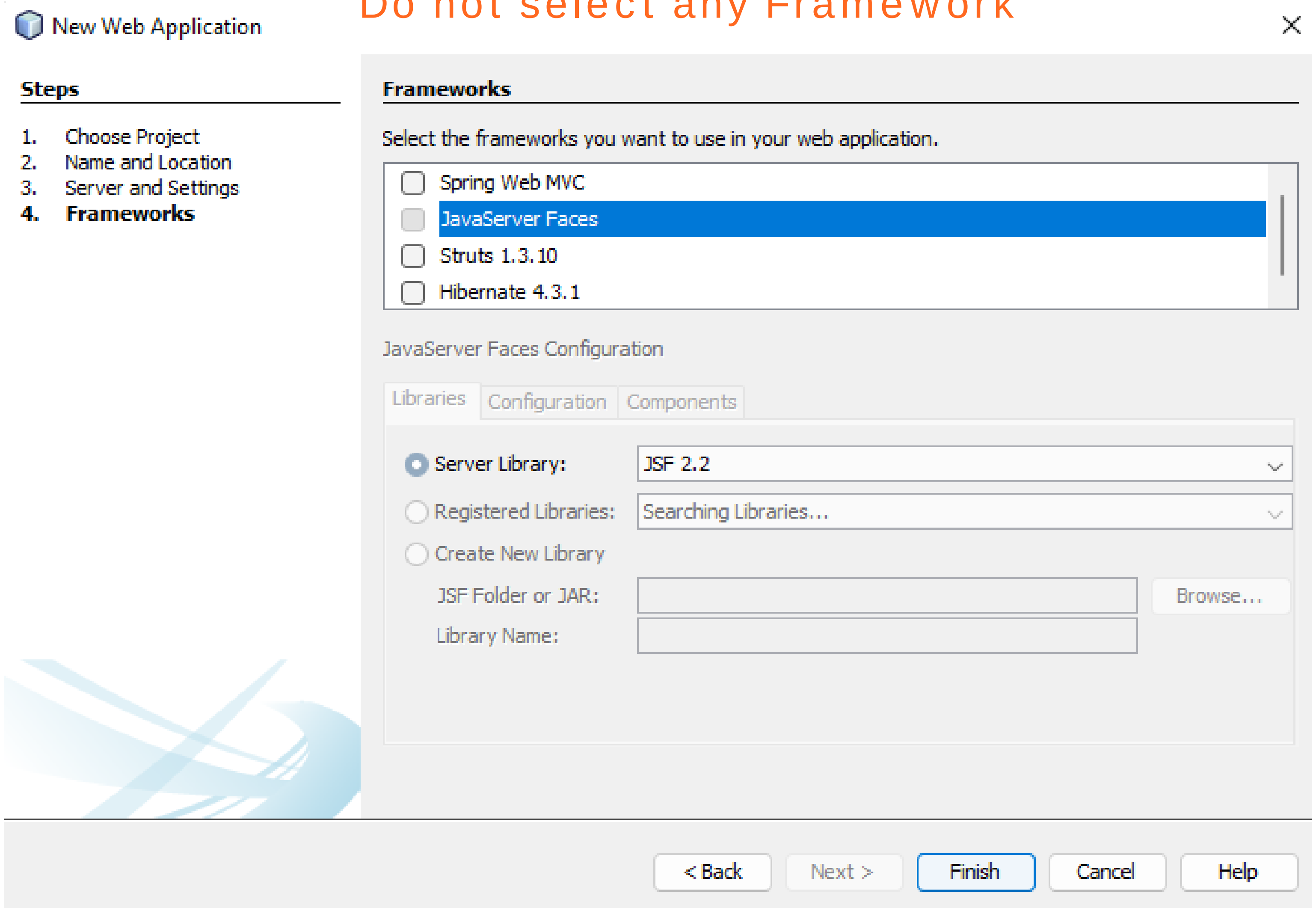
Context Path: /DemoJSP

Use, Apache Tomcat 10.X

Use, Jakarta EE 10 Web

WRITING A JSP

Do not select any Framework



New Web Application

Steps

1. Choose Project
2. Name and Location
3. Server and Settings
4. **Frameworks**

Frameworks

Select the frameworks you want to use in your web application.

- □ Spring Web MVC
- □ JavaServer Faces
- □ Struts 1.3.10
- □ Hibernate 4.3.1

JavaServer Faces Configuration

● Server Library: JSF 2.2

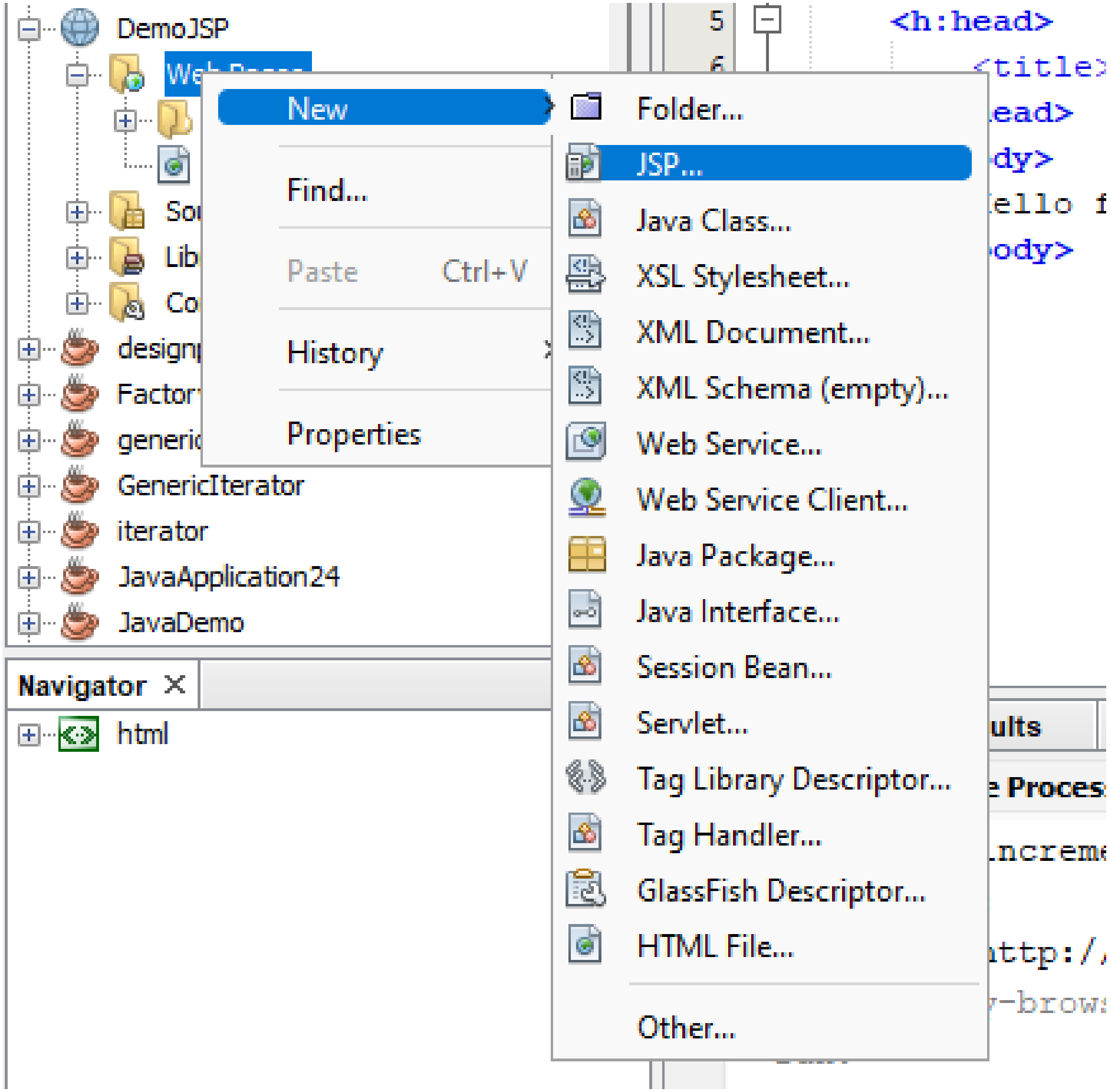
○ Registered Libraries: Searching Libraries...

○ Create New Library

JSF Folder or JAR:

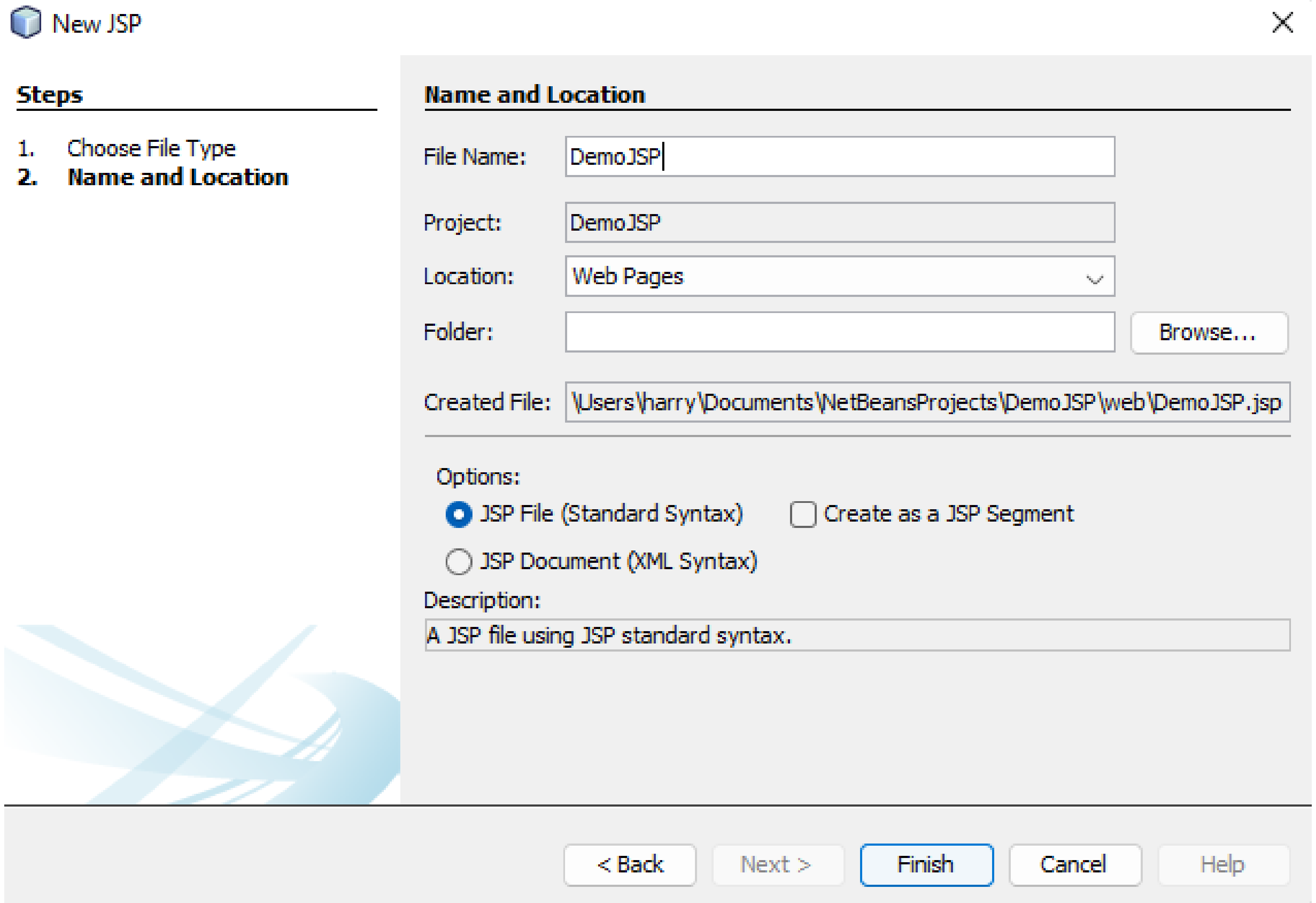
Library Name:

WRITING A JSP



```
5    <h:head>
6        <title>
     ead>
     dy>
     ello f
     ody>
```

WRITING A JSP



Steps

1. Choose File Type
2. **Name and Location**

Name and Location

File Name: DemoJSP

Project: DemoJSP

Location: Web Pages

Folder:

Created File: \Users\harry\Documents\NetBeansProjects\DemoJSP\web\DemoJSP.jsp

Options:

● JSP File (Standard Syntax) □ Create as a JSP Segment

○ JSP Document (XML Syntax)

Description:

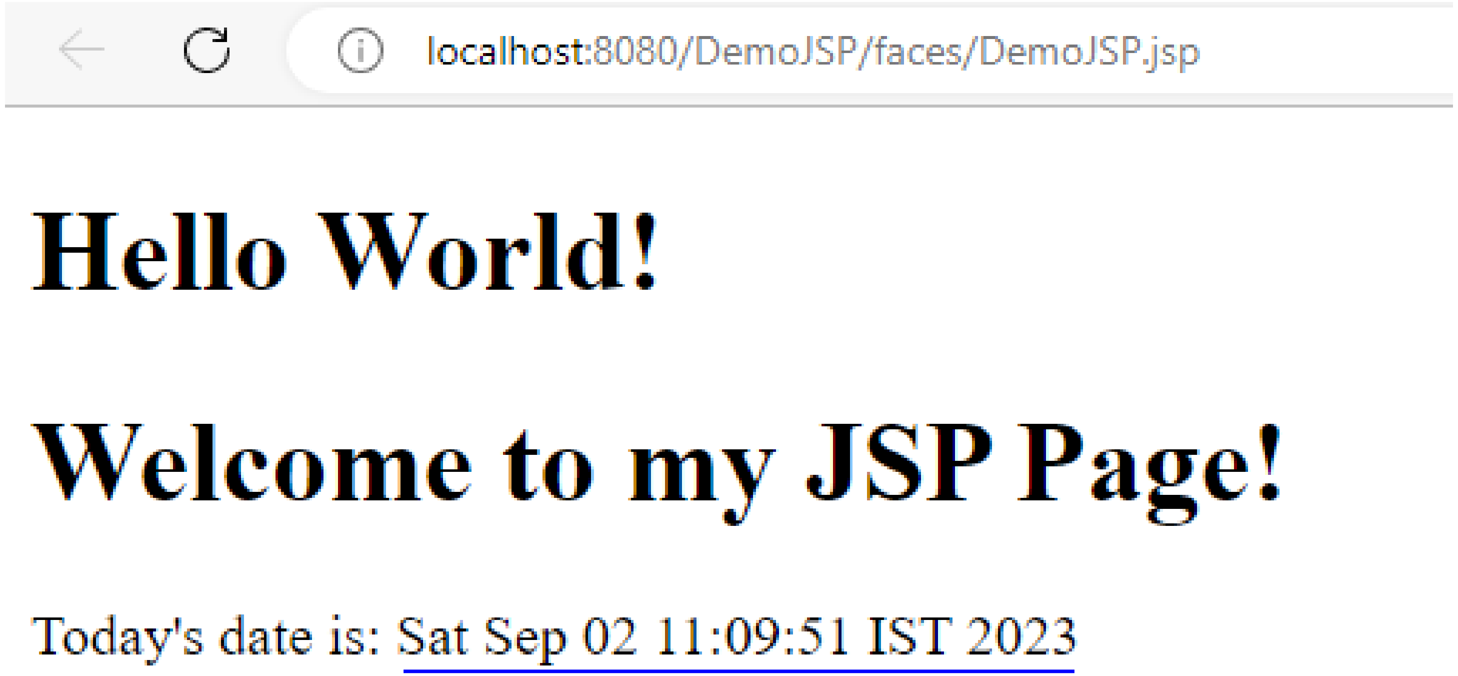
A JSP file using JSP standard syntax.

A JSP file that Displays the Date

```
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>JSP Page</title>
    </head>
    <body>
        <h1>Welcome to my JSP Page!</h1>
        <p>Today's date is: <%= new java.util.Date()%></p>
    </body>
</html>
```

Dynamic Code embedded in HTML Code

OUTPUT:



Hello World!

Welcome to my JSP Page!

Today's date is: Sat Sep 02 11:09:51 IST 2023

This is output of Dynamic Code.

REFERENCES

Bryan Basham, Kathy Sierra, Bert Bates, Head First Servlets & JSP, 2nd Edition, Publication: O-reilly.

Bruce W. Perry, Java Servlet & JSP Cookbook Publisher: O'Reilly

https://en.wikipedia.org/wiki/Jakarta_Server_Pages

https://jakarta.ee/specifications/pages/3.0/jakarta-server-pages-spec-3.0

https://javabeat.net/