**A Project Report**

**On**

**HOTEL MANAGEMENT SYSTEM**

Submitted in partial fulfillment of the requirement of the award of the degree of

Diploma in Computer Science and Engineering

Session 2022 - 2025

******

***Under the Guidance of*** ***Submitted By***

**Mr. KANCHAN Raju MONIKA KUMARI**

**Briztech Infosystems Pvt. Ltd. Reg. No. 22102060022**

Diploma in Computer Science and Engineering

**GOVERNMENT WOMEN`S POLYTECHNIC RANCHI, JHARKHAND**

**ACKNOWLEDGEMENT**

This thesis work has been an intellectually invigorating experience for me. I am sure that the knowledge and experience gathered during the course of this work wil make me stand in good stead in feature.

With immense pleasure and due respect, I express my sincere gratitude to **prof NEHA TRIPATHI, in charge, GOVERNMENT WOMEN’S POLYTECHNIC RANCHI, JHARKHAND** for all his support and co-operation in successfully completing this thesis work by providing excellent facilities.

I would also like to extend my sincere gratitude to all faculties’ members and staff for helping me in my college during my Diploma course.

I would like to take this opportunity to extend my sincere gratitude and thanks to my pioneer **Mr. KANCHAN RAJU SIR,** firstly for coming up with such an innovative thesis idea. He has not only made us to work but guided us to orient toward research. It has been real pleasure working under his guidance and it is chiefly his encouragement and motivation that has made this thesis a reality.

Last, but not the last L am heartily thankful to almighty God for showring his blessing forever during my entire life and also to my family members for providing me a great support.

**MONIKA KUMARI**

**Reg. No. 22102060022**

**Diploma in Computer Science and Engineering**

**GOVERNMENT WOMEN’S POLYTECHNIC RANCHI,JHARKHAND**

**CERTIFICATE FOR PROJECT**

This is to certify that this is a bona-fide record of the project HOTEL MANAGEMENT SYSTEMS done satisfactory at **“BRIZTECH INFOSYSTEMS PVT. LTD”** by **MONIKA KUMARI, Roll No.22102060022,** in partial fulfillment of Diploma Examination.

This report or similar report on the topic has not been submitted for any other examination and doesn’t form part of any other course undergone by the candidate.

Date: **Mr.KANCHAN RAJU**

Place: RANCHI GUIDE, BRIZTECH INFOSYSTEM

RANCHI, JHARKHAND

**CERTIFICATE OF APPROVAL**

The foregoing project entitled **Hotel management** is hereby approved as a creditable **MONIKA KUMARI, Roll No 22102060022** worked on the **HOTEL MANAGEMENT SYSTEMS** and has been presented in satisfactory manner to warrant its acceptance as prerequisite to the degree for which it has been submitted.

It is understood that by this approval, the undersigned do not necessarily endorse any conclusion drawn or opinion expressed therein, but approve the project for the purpose for which it is submitted.

**(Internal Examiner) (External Examiner) Diploma in Computer Science and Engineering**

**GOVERNMENT WOMEN’S POLYTECHNIC RANCHI, JHARKHAND**

**DECLARATION BY THE CANDIDATE**

By MONIKA KUMARI, Roll No. 22102060022, hereby declare that the work, which is being presented in the dissertion, entitled “HOTEL MANAGEMENT,” in partial fulfillment of requirement for the award of the degree of “Diploma in Computer Science and Engineering” submitted in GOVERNMENT WOMEN’S POLYTECHNIC RANCHI, JHARKHAND Iis an authentic record of our work carried out under the guidance of **Mr. KANCHAN RAJU SIR**

We have not submitted the matter embodied in this dissertation for the award of any other degree.

Date:

Place: RANCHI

**Table of Contents**

1. Acknowledgments
2. Introduction
3. System requirements
4. About Java
5. Features of Java
6. Glashfish server
7. JDBC
8. About JSP
9. About html
10. About MySQL database
11. Data dictionary
12. Modules of the project with description
13. System design
14. Flow chat
15. E-R diagram
16. Project coding and snapshots
17. Limitation of the project
18. Advantage of tour and travels project
19. Scope of project
20. Bibliography

**INDRODUCTION OF PROJECT**

This is the project about hotel management where all the customers will come and Booking manager will book the room for them.as far as the software is concerned There is java programming language with jsp is used.java is a high performance programming language heavily used into the software development industry. For the robust performance java is used.there will be web application which will be used to Book the customers hotel room and store their records into the mariadb database for the future use.

**OBJECTIVE**

The objectives of the system are-

* To provide hotel facility to the customers.
* Reduced operational booking and registration.
* Increased operational efficiency.
* Data security**.**

This software package can be readily used by non-programming personal avoiding human handled chance of error. This project is used by companies for hotels service providers.

Frequent customers can create their accounts and see the available rooms and book them As per their requirement. They can select other options for hotels.

Application can be upgraded according to the company requirements and can be changed .

**MODIFICATION AND IMPROVEMENT OVER THE EXISTING IMPLEMENTATION**

**Present State:**

* No Software present at all.
* Manual booking hotels for customers.
* Headache of booking hotels
* More time taken to registration of customers for hotel.

**After implementation of project:**

* Easy to use GUI.
* customers can register with no rush for any place.
* Respective eligible customers can get information of hotels.
* List of hotels be made on single click.
* Easily tracking system for hotels and rooms.
* Report of hotels according to need of customers can be generated easily.
* Removal of burden of booking hotels.
* Better hotel facility to the customers.

**SCOPE**

This project has a large scope as it has the following features which help in making it easy to use, understand and modify it:

* Automation of hotel booking system.
* No Need to do Paper Work.
* To save the environment by using paper free work.
* To increase the accuracy and efficiency of the hotels.
* Better Management of hotels.
* Secure hotel facility to the all customers.

This software package can be readily used by non-programming personal avoiding human handled chance of error. This project is used by

1. Hotel Service providers

**MAIN IMPORTANT POINTS ARE :**

* Simplified Management of hotel booking service
* customer management.
* Customer details for hotel customers.
* Hotel details.
* Hotles and customers timing details are available online.
* Rooms available for per hotels.

**PROJECT ANALYSIS**

The significance of the project is to provide the following benefits:

* The significance of project is to automate the hotel booking and billing system.
* There is no need to worry about the hotels.
* All hotel booking system and management are processed through proper channel.
* Customers can see the booked hotels in a single click.
* Hotel detail are available and can be upgraded as per the requirement.
* Bus timing and rates of hotels are available online.

**SIGNIFICANCE OF PROJECT**

The significance of the project is to provide the following benefits:

* The significance of project is to automate the hotel booking

using software.

* All hotels are registered with travel agency.
* hotels can see the all details of booked customers.
* hotels can see the all the details of customers on single click.
* Customers , hotel details are available and can be upgraded as per the requirement.

**System Requirements**

* Hardware Requirements:

**The hardware requirements for this project are:-**

* + Hard Disk 500 GB or above
  + RAM 4GB or above
  + CPU I3 or above
  + Keyboard Standard keyboard of 104 keys
  + Printer HP LaserJet P1007
* Software Requirements:

**The software needed to develop this software is:-**

* + Operating System Windows 10,11
  + Web-browser Google chrome.
  + Web Server Glashfish server
  + MariaDB database For creating Backend.
  + Web technology JSP
  + Programming language java 8.0
  + IDE- Netbeans

Front End HTML,CSS,JAVASCRIPT

**Introduction of java**

Java is a programming language created by James gosling, Mike Sheridon and Patrick Naughton initiated the java language project in June 1991.java was originally designed for interactive television, but it was too advanced for the digital cable television industry at the time. The language was initially called awk after an awk tree that stood outside Gosling office later the project went by the name green and was finally renamed Java from “java coffee Gosling designed java with a C/C++ style syntax that system and application programmer would find familiar.

‘Sun Microsystems’ released the first public implementation as java 1.0in 1995.it promised “write once run anywhere (WORA) providing no cost run times on popular platforms. In 1997 sun Microsystems approached the ISO/IEC IIC1 standard body and later the Ecma international to formalize java. But it soon withdrew from the process.java software runs on everything from laptops to data centers, game consoles of scientific super computer.

Before embarking on the road to Java programmer certification, it is important to

Understand the basic terminology and concepts in object-oriented programming (OOP). In this chapter, the emphasis is on providing an introduction rather than Exhaustive coverage. In-depth coverage of the concepts follows in subsequent Chapters of the book. Java supports the writing of many different kinds of executables: applications, applets, and servlets. The basic elements of a Java application are introduced in this chapter. The old adage that practice makes perfect is certainly true when learning a programming language. To encourage programming on the computer, the mechanics of compiling and running a Java application are outlined.

There were five primary goal in the creations of the java language.

* It must be “simple, object orient and familiar”.
* It must be architecture neutral and portable.”
* It must execute with “high performance”
* It must be” interpreted ,threaded and dynamic”

**Features of Java**

1. Compiled and interpreted:-Usually a computer language is either compiled or interpreted but java combines two state systems. Firstly java compile translate source code(sc) instruction into byte code are not machine instruction therefore in the second stage the java interpreter generate the machine code i.e. directly executable by the machine.
2. Platform independent and portable:-java programming can be easily moved from one computer system to another anywhere and anytime it also a platform indepent programming language because we can execute the java program on different platform like on console, on windows on browser etc.
3. Robust and secure:-Java is a robust language because it provider many safe guards to insure retable codes.java also incorporator the concept of acception handling which captures series error and eliminates any risks crashing the system means it enables multiple programmer at multiple remote location to collaborate and work together on a single project.
4. Multithreaded:-multithreaded means handling multiple task simultaneously.
5. Simple, small and familiar:-Java is a small and simple language because many features of c and c++ are available here.
6. Object oriented:- java is a true object oriented language almost everything in java is an object. All programs code and data resides within object and classes.

**Basic structure of java programming:**

|  |
| --- |
| Document section |
| Package statement |
| Import statement |
| Interface statement |
| Main method  {  Main method definition  } |

Optional

Essential

**Documentation selection:-** the documentation selections comprise a set of comment line giving the name of the program.

**Package statement:-**The 1st statement allowed in a java file is a package name and informs the compiler that the classes defined here belongs to particular package.

**Import statement:-**The import statement of java is the just similar to the preprocessor directly of c and c++.

**Interface statement:-**It is an optional section and is used only when we want implement the multiple inheritance features in the programs.

**Main method:-** Each an every program of java one function necessary required having name the main function because the compilation and the execution process of the program always begins from here.

**Java virtual machine:-** The heart of the java platform is the concept of a “virual machine” that execute java byte code programs. This is the same no matter what hardware or operating system the program in running under. There JIT(just in time) compiler within the java virtual machine, or JVM. The JIT compiler translate the java byte codes into native processor instructions at runtime and caches the native code in memory during execution.

**How to install Java in system:**

* This guide is for installing the java runtime environment (JRE) for Browser.

For instruction on installing the developer tools (jdk), see this guide. Java is also different than java script. If you need to enable java script, see this guide.

* Visit the java home page. Java installs system files which all browser use, so need to follow special instruction for specific browsers. You can get to the java installer from the java home page.
* The java installer will download files during the installation process. If you need to install java on a device without a network connection, download the offline installer, available on the manual downloads page.
* Depending on your browser setting, you may need to accept the java installation download before it begins.
* For MAC OS x10.6, java comes preinstalled. For OS X 10.7.3 or newer to install java. You must also be using a 64-bit browser such as safari or Firefox(i.e. not chrome)
* For Linux, java needs to be downloaded, manually, and then enabled in order to work. See this guide for detailed instruction on installing java in Linux.
* Start the installer. Once the installer has finished downloading, run it to begin the installation. On OS X, double-click the .dmg file to begin the installation.
* Close any browser windows before starting the installation, as they will need to b e restated anyway after installation is complete.
* Follow the installation steps. read each screen of the installation program. Java will try to install extra software such as browser toolbars unless you uncheck the boxes. If you don’t want your browser changed, make sure to read each screen carefully.
* Test the installation. After you have finished installing java, test the installation to makes sure everything went ok. You can find the java testing applet on the java website, or by searching for java test and selecting the firsts result.
* You will need to allow the plug-in to run and you may be asked multiple times before it can load. This is because, in general, java can be a dangerous tool which can grant others access to your computer if you are not careful. Always be sure you trust the website that you are running java applets on.

**Glashfish Server**

Glashfish, often referred to as Glashfish, is an open-source web server glashfish implements several Java EE specifications including Java Servlet, Java Server Pages (JSP),Java EL, and Web Socket, and provides a "pure Java "HTTP web server environment for Java code to run in. glashfish is developed and maintained by an open community of developers under the auspices of the Apache Software Foundation, released under the Apache License2.0 license, and is open-source software.

**Web application:-** Ithas also added user- as well as system-based web applications enhancement to add support for deployment across the variety of environments. It also tries to manage sessions as well as applications across the network. Tomcat is building additional components. A number of additional components may be used with Glashfish. These components may be built by users should they need them or they can be downloaded from one of the mirrors.

* **Installing Java**:-Tomcat requires java in order to run. If your computer already has java installed, you can probably skip this step. However, make sure you have a recent version of java. Here I provide instructions for installing version 1.4.2 of the Java 2 Platform, Standard Edition (J2SE).

**JDBC**

JDBC is Java application programming interface that allows the Java programmers to access database management system from Java code. It was developed by JavaSoft a subsidiary of Sun Microsystems.

**Definition**:- Java Database Connectivity in short called as JDBC. It is a java API which enables the java programs to execute SQL statements. It is an application programming interface that defines how a java programmer can access the database in tabular format from Java code using a set of standard interfaces and classes written in the Java programming language.JDBC has been developed under the Java Community Process that allows multiple implementations to exist and be used by the same application. JDBC provides methods for querying and updating the data in Relational Database Management system such as SQL, Oracle etc.

The Java application programming interface provides a mechanism for dynamically loading the correct Java packages and drivers and registering them with the JDBC Driver Manager that is used as a connection factory for creating JDBC connections which supports creating and executing statements such as SQL INSERT, UPDATE and DELETE. Driver Manager is the backbone of the JDBC architecture.

Generally all Relational Database Management System supports SQL and we all know that Java is platform independent, so JDBC makes it possible to write a single database application that can run on different platforms and interact with different Database Management Systems.

Java Database Connectivity is similar to Open Database Connectivity (ODBC) which is used for accessing and managing database, but the difference is that JDBC is designed specifically for Java programs, whereas ODBC is not depended upon any language.

In short JDBC helps the programmers to write java applications that manage these three programming activities:

1. It helps us to connect to a data source, like a database.
2. It helps us in sending queries and updating statements to the database and
3. Retrieving and processing the results received from the database in terms of answering to your query.

**JSP**

Filename extension .jsp

Internet media type application/jsp

Latest release 2.2

Standard JSR 245

Website Java Server Pages Technology

Java Server Pages (JSP) is a technology that helps software developers create dynamically generated web pages based on HTML, XML, or other document types. Released in 1999 by Sun Microsystems,[1]JSP is similar to PHP and ASP, but it uses the Java programming language.

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

**Overview**

Architecturally, JSP may be viewed as a high-level abstraction of Java servlets. JSPs are translated into servlets at runtime; each JSP servlet is cached and re-used until the original JSP is modified.

JSP can be used independently or as the view component of a server-side model–view–controller design, normally with Java Beans as the model and Java servlets (or a framework such as Apache Struts) as the controller. This is a type of Model architecture.

JSP allows Java code and certain pre-defined actions to be interleaved with static web markup content, such as HTML, with the resulting page being compiled and executed on the server to deliver a document. The compiled pages, as well as any dependent Java libraries, contain Java byte code rather than machine code. Like any other Java program, they must be executed within a Java virtual machine (JVM) that interacts with the server's host operating system to provide an abstract, platform-neutral environment.

JSPs are usually used to deliver HTML and XML documents, but through the use of Output Stream, they can deliver other types of data as well.

**HTML**

HTML is a language for describing web pages. It stands for Hyper Text Markup Language. It is a set of special codes that can be embedded in text to add formatting and linking information. The html tags describe document content and HTML documents contain HTML tags and plain text.

* ***HTML Tags:-***

A Markup tags are usually called HTML tags. Tags are instructions that are embedded directly into the text of the document. An html tag is a signal to a browser that it should do something other than just throw text up on the screen. By convention all html tags begin with an open angle bracket(<) and end with a close angle bracket(>).

eg:- <html>,<b>,<h1>.

It is of two types…

1. **Paired tags:**

* HTML tags normally come in pairs like <b> and </b>.
* The first tag in a pair is the start tag, the second tag is the end tag.
* The end tag is written like the start tag, with a forward slash before the tag name
* Start and end tags are also called opening tags and closing tags

<tagname>content</tagname>

1. **Singular tags:**

* This is a stand-alone tag.
* It does not have a companion tag.
* Eg:<br>,<frameset>.
* ***HTML Elements:-***
* "HTML tags" and "HTML elements" are often used to describe the same thing.
* An HTML element is everything between the start tag and the end tag, including the tags:

Eg: <p>This is a paragraph.</p>

* ***HTML Page Structure:***

Below is a visualization of an HTML page structure:

<html>

<body>

<h1>This a heading</h1>

<p>This is a paragraph.</p>

<p>This is another paragraph.</p>

</body>

</html>

* ***HTML Attributes:***
* HTML elements can have attributes
* Attributes provide additional information about an element
* Attributes are always specified in the start tag
* Attributes come in name/value pairs like: name="value"
* Attribute Example
* HTML links are defined with the <a> tag. The link address is specified in the href attribute:

Eg: <a href="http://www.w3schools.com">This is a link</a>

**HTML Text Formatting Tags:**

***Tags*** ***Description***

<b> Defines bold text

<em> Defines emphasized text

<small> Defines smaller text

<strong> Defines important text

<sub> Defines subscripted text

<sup> Defines superscripted text

<ins> Defines inserted text

<del> Defines deleted text

<i> defines a part of text in an alternate voice or mood

* ***HTML List:***

The most common HTML lists are ordered and unordered lists.

* **HTML Unordered Lists:-** An unordered list starts with the <ul> tag. Each list item starts with the <li> tag.The list items are marked with bullets (typically small black circles).

Eg: <ul>

<li>Coffee</li>

<li>Milk</li></ul>

* **HTML Ordered Lists:-** An ordered list starts with the <ol> tag. Each list item starts with the <li> tag. The list items are marked with numbers.

Eg: <ol>

<li>Coffee</li>

<li>Milk</li>

</ol>

**HTML Definition List:**

A definition list is a list of items, with a description of each item. The <dl> tag defines a definition list. The <dl> tag is used in conjunction with <dt> (defines the item in the list) and <dd> (describes the item in the list):

<dl>

<dt>Coffee</dt>

<dd>- black hot drink</dd>

<dt>Milk</dt>

<dd>- white cold drink</dd>

</dl>

**Essential HTML Tags**:-

**Defination:- <html></html>** This basically defines the document as web page. It also identifies the begging and end of the HTML document. All other tags must fall between the html tags.

There are four essential of HTML tags that form the basic structure needed for every HTML file:

* <html></html>
* <head></head>
* <tittle></tittle>
* <body></body>

**Header<head></head>:-**The header contains information about he document that will not appear on the actual page, such as the tittle of the document, the author which style sheet to use and also meta tags.

**Tittle <tittle></tittle>:-**The tittle tag defines the tittle that will appear in the actual must appear between the head tags**.**

**Body<body></body>:-**The body tags contain all the information and other visible content on the page. All your image, links and plain text must go between the <body> and </body>tags.

***Example*:-**below is a html document. Notice that everything falls between the html tags, the title appears within the head of the document, and that the body comes after the head**.**

<html>

<head>

<tittle>this is the tittle </title>

</head>

This is everything that goes in the document!

</body>

</html>

**JSP Overview**

In Java, **JSP** stands for **Jakarta Server Pages(** (**JSP**; formerly **JavaServer Pages**)**)**. It is a server-side technology that is used for creating web applications. It is used to create dynamic web content. JSP consists of both HTML tags and JSP tags. In this, JSP tags are used to insert JAVA code into HTML pages. It is an advanced version of **Servlet**Technology i.e. a web-based technology that helps us to create dynamic and platform-independent web pages. In this, Java code can be inserted in HTML/ XML pages or both. JSP is first converted into a servlet by the JSP container before processing the client’s request. JSP has various features like JSP Expressions, JSP tags, JSP Expression Language, etc.

JSP allows embedding Java code in HTML pages, making it easier to build interactive web applications. To further enhance your knowledge of Java web technologies, consider enrolling in the [**Java Backend**](https://gfgcdn.com/tu/S6G/) course, which covers essential concepts for backend development

## ****How is JSP more Advantageous than Servlet?****

* They are easy to maintain.
* No recompilation or redeployment is required.
* Less coding is required in JSP.
* JSP has access to the entire API of JAVA.
* JSP are extended version of Servlet.

## ****Features of JSP****

* **Coding in JSP is Easy:** As it involves adding Java code to HTML/XML.
* **Easy to Use and Learn:** It is straightforward and accessible for both Java and non-Java programmers.
* **It Does Not Require Advanced Knowledge of Java:** Suitable for users with basic Java skills.
* **Reduction in the Length of Code:** JSP uses action tags, custom tags, etc., to minimize code length.
* **Java Scriplets:** Allows embedding Java code, variables, and expressions within JSP pages.
* **JSP Expression:** Evaluates expressions and converts them to strings.
* **Declaration Tag:** Used to declare variables and methods within JSP pages.
* **Implicit Objects:** Provides built-in objects that reduce the length of code.
* **Make Interactive Websites:** Facilitates the creation of dynamic web pages that interact with users in real-time.
* **Connection to Database is Easier:** Simplifies connecting to databases and allows for easy data reading and writing.
* **Extension to Servlet:** Inherits all features of servlets and includes implicit objects and custom tags.
* **Portable, Powerful, Flexible, and Easy to Maintain:** Browser and server independent, making it versatile and easy to manage.
* **No Redeployment and No Re-Compilation:** JSP is dynamic, secure, and platform-independent, so no need for redeployment or recompilation.
* **Create HTML Page from Where Request Will Be Sent:** For example, try.html for sending requests to the server.
* **To Handle User Requests, Create .jsp File:** For instance, new.jsp for handling requests.
* **Create Project Folder Structure:** Organize files and resources into a project structure.
* **Create XML File:** For configuration or data, e.g., my.xml.
* **Create WAR File:** Package the application into a WAR (Web Application Archive) file for deployment.
* **Start Tomcat:** Ensure the servlet container (Tomcat) is running.
* **Run Application:** Deploy and test the JSP application.
* **It is Capable of Handling Exceptions:** JSP can manage errors and exceptions effectively.
* **Difficult to Debug for Errors:** Debugging JSP pages can be challenging due to their mixed nature of HTML and Java code.
* **First-Time Access Leads to Wastage of Time:** Initial access can be slower as JSP pages need to be compiled.
* **Output is HTML Which Lacks Features:** JSP generates HTML, which may not support advanced features.

## Creating a simple JSP Page

**hello.JSP :**

JSP simply puts Java inside HTML pages. You can take any existing HTML page and change its extension to “.jsp” instead of “.html”. In fact, this is the perfect exercise for your first JSP.

Take the HTML file you used in the previous exercise. change its extension from “.html” to  “jsp”. Now load the new file, with the “.jsp” extension, in your browser.

You will see the same output, but it will take longer! But only the first time. If you reload it again, it will load normally.

What is happening behind the scenes is that your JSP is being turned into a Java file, compiled, and loaded. This compilation only happens once, so after the first load, the file doesn’t take long to load anymore. (But every time  you change the JSP file, it will be re-compiled again.)

Of course, it is not very useful to just write HTML pages with a .jsp extension! We now proceed to see what makes JSP so useful.

### ****Adding dynamic content via expressions:****

As we saw in the previous section, any HTML file can be turned into a JSP file by changing its extension to .jsp . Of course , what makes JSP useful is the ability to embed Java. Put the following text in a file. jsp extension (let us call it hello.jsp) , place it in your JSP directory, and view it in a browser.

<HTML>  
<BODY>  
 Hello! The time is now <%= new java.util.Date() %>  
</BODY>  
</HTML>

Notice that each time you reload the page in the browser, it comes up with the current time. The character sequence.

<%= and %> enclose Java expressions, which are evaluated at run time.

This is what makes it possible to use JSP to generate dynamic HTML pages that change in response to user actions or vary from user to user.

### ****Explain JSP Elements:****

We will learn about the various elements available in JSP with suitable examples. In JSP elements can be divided into 4 different types.

These are:

* Expression
* Scriplets
* Directives
* Declarations

### Expression:

We can use this tag to output any data on the generated page. These data are automatically converted to string and printed on the output stream.

**Syntax:**

JSP Expressions are : <%="Anything" %>

*NOTE : JSP Expressions start with Syntax of JSP Scriptles are with <%=and ends with %>.  Between these, you can put anything that will convert to the String and that will be displayed.*

**Example:**

<%="HelloWorld!" %>

### ****Scriplets:****

In this tag we can insert any amount of valid java code and these codes are placed in the \_jsp Service method by the JSP engine.

**Syntax:**

<%//java codes%>

*NOTE : JSP Scriptlets begins with <% and ends %> . We can embed any amount of Java code in the JSP Scriptlets. JSP Engine places these codes in the \_jspService() method.*

**Variables available to the JSP Scriptlets are:**

* Request
* Response
* Session
* Out

### ****Directives:****

A JSP “directive” starts with <%@ characters. In the directives, we can import  packages , and define error-handling pages or the session information of the JSP page.

**Syntax:**

<%@directive attribute="value"% >

* page
* include
* taglib

### ****Declarations :****

This tag is used for defining the functions and variables to be used in the JSP.

**Syntax:**

<%!  
 //java codes  
%>

*NOTE : JSP Declaratives begins with <%! and ends %> with We can embed any amount of java code in the JSP Declaratives. Variables and functions defined in the declaratives are class-level and can be used anywhere on the JSP page.*

**Example :**

**<%@page import="java.util.\*"%>**  
<HTML>  
<BODY>  
<%!  
Date theDate = new Date(); // Corrected the unwanted space in the declaration  
Date getDate() {  
 System.out.println("In getDate() method");  
 return theDate;  
}  
%>  
Hello! The time is now <%=getDate()%>  
</BODY>  
</HTML>

### ****Example of a JSP Web Page:****

<HTML>  
 <HEAD>  
 <TITLE>A Web Page</TITLE>  
 </HEAD>  
 <BODY>  
 <%out.println("Hello there!");%>  
 </BODY>  
 </HTML>

## Run a Simple JSP Page:

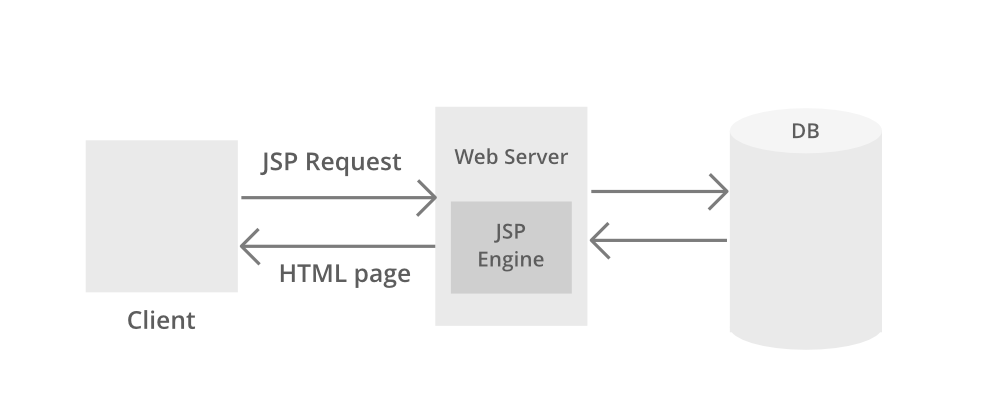
* **Step 1:** Save the JSP file using “.jsp” extension (ex- “hello.jsp”)
* **Step 2:**Start the server
* **Step 3:**Place your application inside a folder
* **Step 4:** To execute the JSP script, simply start tomcat server and use a browser to browse an URL of the JSP page i.e. http://localhost:portnumber/YourApplicationContextRoot/jspfile then you will see the jsp file is being compiled.

Want to be a master in **Backend Development with Java** for building robust and scalable applications? Enroll in [**Java Backend and Development Live Course**](https://gfgcdn.com/tu/Q8Q/) by GeeksforGeeks to get your hands dirty with Backend Programming. Master the key **Java concepts, server-side programming, database integration**, and more through hands-on experiences and **live projects**. Are you new to Backend development or want to be a Java Pro? This course equips you with all you need for building high-performance, heavy-loaded backend systems in Java. Ready to take your Java Backend skills to the next level? Enroll now and take your development career to sky highs.

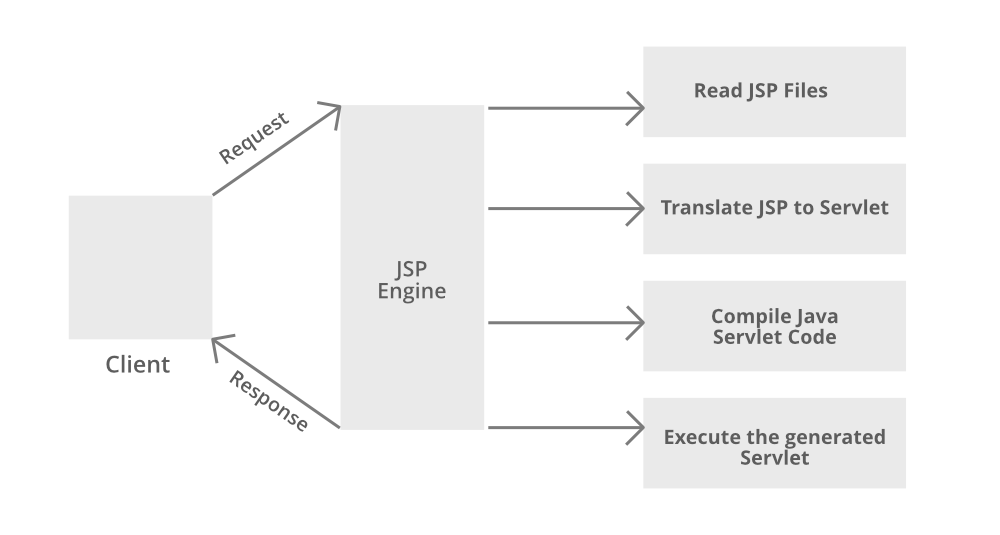
# JSP Architecture

**Last Updated :**20 Jul, 2021

[JSP](https://www.geeksforgeeks.org/introduction-to-jsp/) architecture gives a high-level view of the working of JSP. JSP architecture is a 3 tier architecture. It has a Client, Web Server, and Database. The client is the web browser or application on the user side. Web Server uses a JSP Engine i.e; a container that processes JSP. For example, Apache Tomcat has a built-in JSP Engine. JSP Engine intercepts the request for JSP and provides the runtime environment for the understanding and processing of JSP files. It reads, parses, build Java Servlet, Compiles and Executes Java code, and returns the HTML page to the client. The webserver has access to the Database. The following diagram shows the architecture of JSP.



Now let us discuss JSP which stands for Java Server Pages. It is a server-side technology. It is used for creating web applications. It is used to create dynamic web content. In this JSP tags are used to insert JAVA code into HTML pages. It is an advanced version of Servlet Technology. It is a Web-based technology that helps us to create dynamic and platform-independent web pages. In this, Java code can be inserted in HTML/ XML pages or both. JSP is first converted into a servlet by JSP container before processing the client’s request. **JSP Processing**is illustrated and discussed in sequential steps prior to which a pictorial media is provided as a handful pick to understand the JSP processing better which is as follows:



**Step 1:** The client navigates to a file ending withthe[***.jsp extension***](https://www.geeksforgeeks.org/difference-between-servlet-and-jsp/)and the browser initiates an HTTP request to the webserver. For example, the user enters the login details and submits the button. The browser requests a status.jsp page from the webserver.

**Step 2:** If the compiled version of JSP exists in the web server, it returns the file. Otherwise, the request is forwarded to the JSP Engine. This is done by recognizing the URL ending with **.jsp**extension.

**Step 3:** The JSP Engine loads the JSP file and translates the JSP to Servlet(Java code). This is done by converting all the template text into println() statements and JSP elements to Java code. This process is called **translation.**

**Step 4:** The JSP engine compiles the Servlet to an executable **.class**file. It is forwarded to the Servlet engine. This process is called **compilation**or **request processing phase.**

**Step 5:** The **.class**file is executed by the Servlet engine which is a part of the Web Server. The output is an HTML file. The Servlet engine passes the output as an HTTP response to the webserver.

**Step 6:** The web server forwards the HTML file to the client’s browser.

**MariaDB Overview**

A database application exists separate from the main application and stores data collections. Every database employs one or multiple APIs for the creation, access, management, search, and replication of the data it contains.

Databases also use non-relational data sources such as objects or files. However, databases prove the best option for large datasets, which would suffer from slow retrieval and writing with other data sources.

Relational database management systems, or RDBMS, store data in various tables.Relationships between these tables are established using primary keys and foreign keys.

RDBMS offers the following features −

* They enable you to implement a data source with tables, columns, and indices.
* They ensure the integrity of references across rows of multiple tables.
* They automatically update indices.
* They interpret SQL queries and operations in manipulating or sourcing data from tables.

## RDBMS Terminology

Before we begin our discussion of MariaDB, let us review a few terms related to databases.

* **Database** − A database is a data source consisting of tables holding related data.
* **Table** − A table, meaning a spreadsheet, is a matrix containing data.
* **Column** − A column, meaning data element, is a structure holding data of one type; for example, shipping dates.
* **Row** − A row is a structure grouping related data; for example, data for a customer. It is also known as a tuple, entry, or record.
* **Redundancy** − This term refers to storing data twice in order to accelerate the system.
* **Primary Key** − This refers to a unique, identifying value. This value cannot appear twice within a table, and there is only one row associated with it.
* **Foreign Key** − A foreign key serves as a link between two tables.
* **Compound Key** − A compound key, or composite key, is a key that refers to multiple columns. It refers to multiple columns due to a column lacking a unique quality.
* **Index** − An index is virtually identical to the index of a book.
* **Referential Integrity** − This term refers to ensuring all foreign key values point to existing rows.

Ezoic

## MariaDB Database

MariaDB is a popular fork of MySQL created by MySQL's original developers. It grew out of concerns related to MySQL's acquisition by Oracle. It offers support for both small data processing tasks and enterprise needs. It aims to be a drop-in replacement for MySQL requiring only a simple uninstall of MySQL and an install of MariaDB. MariaDB offers the same features of MySQL and much more.

### Key Features of MariaDB

The important features of MariaDB are −

* All of MariaDB is under GPL, LGPL, or BSD.
* MariaDB includes a wide selection of storage engines, including high-performance storage engines, for working with other RDBMS data sources.
* MariaDB uses a standard and popular querying language.
* MariaDB runs on a number of operating systems and supports a wide variety of programming languages.
* MariaDB offers support for PHP, one of the most popular web development languages.
* MariaDB offers Galera cluster technology.
* MariaDB also offers many operations and commands unavailable in MySQL, and eliminates/replaces features impacting performance negatively.

All downloads for MariaDB are located in the [Download](https://downloads.mariadb.org/) section of the official MariaDB foundation website. Click the link to the version you would like, and a list of downloads for multiple operating systems, architectures, and installation file types is displayed.

## Installing on LINUX/UNIX

If you have intimate knowledge of Linux/Unix systems, simply download source to build your install. Our recommended way of installing is to utilize distribution packages. MariaDB offers packages for the following Linux/Unix distributions −

* RedHat/CentOS/Fedora
* Debian/Ubuntu

The following distributions include a MariaDB package in their repositories −

* openSUSE
* Arch Linux
* Mageia
* Mint
* Slackware

Follow these steps to install in an Ubuntu environment −

**Step 1** − Login as a root user.

**Step 2** − Navigate to the directory containing the MariaDB package.

**Step 3** − Import the GnuPG signing key with the following code −

sudo apt-key adv --recv-keys --keyserver keyserver.ubuntu.com 0xcbcb082a1bb943db

**Step 4** − Add MariaDB to the **sources.list** file. Open the file, and add the following code −

sudo add-apt-repository 'deb http://ftp.osuosl.org/pub/mariadb/repo/5.5/ubuntuprecise main'

**Step 5** − Refresh the system with the following −

sudo apt-get update

**Step 6** − Install MariaDB with the following −

sudo apt-get install mariadb-server

Ezoic

## Installing on Windows

After locating and downloading an automated install file (MSI), simply double click the file to start the installation. The installation wizard will walk you through every step of installation and any necessary settings.

Test the installation by starting it from the command prompt. Navigate to the location of the installation, typically in the directory, and type the following at the prompt −

mysqld.exe --console

If the installation is successful, you will see messages related to startup. If this does not appear, you may have permission issues. Ensure that your user account can access the application. Graphical clients are available for MariaDB administration in the Windows environment. If you find the command line uncomfortable or cumbersome, be sure to experiment with them.

[Explore our](https://www.tutorialspoint.com/latest/courses?utm_source=tutorialspoint&utm_medium=tutorials_3p&utm_campaign=internal" \t "_blank)**[latest online courses](https://www.tutorialspoint.com/latest/courses?utm_source=tutorialspoint&utm_medium=tutorials_3p&utm_campaign=internal" \t "_blank)**[and learn new skills at your own pace. Enroll and become a certified expert to boost your career.](https://www.tutorialspoint.com/latest/courses?utm_source=tutorialspoint&utm_medium=tutorials_3p&utm_campaign=internal" \t "_blank)

## Testing the Installation

Perform a few simple tasks to confirm the functioning and installation of MariaDB.

**Use the Admin Utility to Get Server Status**

View the server version with the mysqladmin binary.

[root@host]# mysqladmin --version

It should display the version, distribution, operating system, and architecture. If you do not see the output of that type, examine your installation for issues.

**Execute Simple Commands with a Client**

Bring up the command prompt for MariaDB. This should connect you to MariaDB and allow execution of commands. Enter a simple command as follows −

mysql> SHOW DATABASES;

Ezoic

## Post- Installation

After successful installation of MariaDB, set a root password. A fresh install will have a blank password. Enter the following to set the new password −

mysqladmin -u root password "[enter your password here]";

Enter the following to connect to the server with your new credentials −

mysql -u root -p

Enter password:\*\*\*\*\*\*\*

## Upgrading on Windows

If you already have MySQL installed on your Windows system, and want to upgrade to MariaDB; do not uninstall MySQL and install MariaDB. This will cause a conflict with the existing database. You must instead install MariaDB, and then use the upgrade wizard in the Windows installation file.

The options of your MySQL my.cnf file should work with MariaDB. However, MariaDB has many features, which are not found in MySQL.

Consider the following conflicts in your my.cnf file −

* MariaDB uses Aria storage engine by default for temporary files. If you have a lot of temporary files, modify key buffer size if you do not use MyISAM tables.
* If your applications connect/disconnect frequently, alter the thread cache size.
* If you use over 100 connections, use the thread pool.

### Compatibility

MySQL and MariaDB are essentially identical. However, there are enough differences to create issues in upgradation. Review more of these key differences in the [MariaDB](https://mariadb.com/kb/en/mariadb/mariadb-vs-mysql-compatibility/" \t "_blank)

**DATA DICTIONARY**

**Data Dictionary**

**Table –User**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **DATA TYPE** | **SIZE** | **CONSTRAINT** |
| **NAME** | **VARCHAR2** | **20** | **Not null** |
| **GENDER** | **VARCHAR2** | **20** | **Not null** |
| **ADDRESS** | **VARCHAR2** | **20** | **Not null** |
| **CITY** | **VARCHAR2** | **20** | **Not null** |
| **PHNUMBER** | **VARCHAR2** | **20** | **Not null** |
| **EMAILID** | **VARCHAR2** | **30** | **Not null** |
| **USERNAME** | **VARCHAR2** | **20** | **Not null** |
| **PASSWORD** | **VARCHAR2** | **20** | **Not null** |

**Table –Customers**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **DATA TYPE** | **SIZE** | **CONSTRAINT** |
| **YNAME** | **VARCHAR2** | **20** | **Not null** |
| **FNAME** | **VARCHAR2** | **20** | **Not null** |
| **RNO** | **VARCHAR2** | **20** | **Not null** |
| **RATE** | **VARCHAR2** | **20** | **Not null** |
| **ID** | **INT** | **10** | **PRIMARY KEY** |

**Source Code**

<!DOCTYPE html>

<!-- Created By CodingLab - www.codinglabweb.com -->

<html lang="en" dir="ltr">

<style>

@import url('https://fonts.googleapis.com/css2?family=Poppins:wght@200;300;400;500;600;700&display=swap');

\*{

margin: 0;

padding: 0;

box-sizing: border-box;

font-family: 'Poppins',sans-serif;

}

body{

height: 100vh;

display: flex;

justify-content: center;

align-items: center;

padding: 10px;

background: linear-gradient(135deg, #71b7e6, #9b59b6);

}

.container{

max-width: 700px;

width: 100%;

background-color: #fff;

padding: 25px 30px;

border-radius: 5px;

box-shadow: 0 5px 10px rgba(0,0,0,0.15);

}

.container .title{

font-size: 25px;

font-weight: 500;

position: relative;

}

.container .title::before{

content: "";

position: absolute;

left: 0;

bottom: 0;

height: 3px;

width: 30px;

border-radius: 5px;

background: linear-gradient(135deg, #71b7e6, #9b59b6);

}

.content form .user-details{

display: flex;

flex-wrap: wrap;

justify-content: space-between;

margin: 20px 0 12px 0;

}

form .user-details .input-box{

margin-bottom: 15px;

width: calc(100% / 2 - 20px);

}

form .input-box span.details{

display: block;

font-weight: 500;

margin-bottom: 5px;

}

.user-details .input-box input{

height: 45px;

width: 100%;

outline: none;

font-size: 16px;

border-radius: 5px;

padding-left: 15px;

border: 1px solid #ccc;

border-bottom-width: 2px;

transition: all 0.3s ease;

}

.user-details .input-box input:focus,

.user-details .input-box input:valid{

border-color: #9b59b6;

}

form .gender-details .gender-title{

font-size: 20px;

font-weight: 500;

}

form .category{

display: flex;

width: 80%;

margin: 14px 0 ;

justify-content: space-between;

}

form .category label{

display: flex;

align-items: center;

cursor: pointer;

}

form .category label .dot{

height: 18px;

width: 18px;

border-radius: 50%;

margin-right: 10px;

background: #d9d9d9;

border: 5px solid transparent;

transition: all 0.3s ease;

}

#dot-1:checked ~ .category label .one,

#dot-2:checked ~ .category label .two,

#dot-3:checked ~ .category label .three{

background: #9b59b6;

border-color: #d9d9d9;

}

form input[type="radio"]{

display: none;

}

form .button{

height: 45px;

margin: 35px 0

}

form .button input{

height: 100%;

width: 100%;

border-radius: 5px;

border: none;

color: #fff;

font-size: 18px;

font-weight: 500;

letter-spacing: 1px;

cursor: pointer;

transition: all 0.3s ease;

background: linear-gradient(135deg, #71b7e6, #9b59b6);

}

form .button input:hover{

/\* transform: scale(0.99); \*/

background: linear-gradient(-135deg, #71b7e6, #9b59b6);

}

@media(max-width: 584px){

.container{

max-width: 100%;

}

form .user-details .input-box{

margin-bottom: 15px;

width: 100%;

}

form .category{

width: 100%;

}

.content form .user-details{

max-height: 300px;

overflow-y: scroll;

}

.user-details::-webkit-scrollbar{

width: 5px;

}

}

@media(max-width: 459px){

.container .content .category{

flex-direction: column;

}

}

</style>

<head>

<meta charset="UTF-8">

<!---<title> Responsive Registration Form | CodingLab </title>--->

<link rel="stylesheet" href="style.css">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

</head>

<body>

<div class="container">

<div class="title">New User Registration</div>

<div class="content">

<form action="newuser.jsp" method="post">

<div class="user-details">

<div class="input-box">

<span class="details">Full Name</span>

<input type="text" name="name" placeholder="Enter your name" required>

</div>

<div class="input-box">

<span class="details">Username</span>

<input type="text" name="username" placeholder="Enter your username" required>

</div>

<div class="input-box">

<span class="details">Email</span>

<input type="text" name="emailid" placeholder="Enter your email" required>

</div>

<div class="input-box">

<span class="details">Phone Number</span>

<input type="text" name="phnumber" placeholder="Enter your number" required>

</div>

<div class="input-box">

<span class="details">Password</span>

<input type="text" name="password" placeholder="Enter your password" required>

</div>

<div class="input-box">

<span class="details">Confirm Password</span>

<input type="text" name="cpassword" placeholder="Confirm your password" required>

</div>

<div class="input-box">

<span class="details">City</span>

<input type="text" name="city" placeholder="Enter your city" required>

</div><div class="input-box">

<span class="details">Address</span>

<input type="text" name="address" placeholder="Enter your address" required>

</div>

</div>

<div class="gender-details">

<input type="radio" name="gender" value="male" checked id="dot-1">

<input type="radio" name="gender" value="female" id="dot-2">

<span class="gender-title">Gender</span>

<div class="category">

<label for="dot-1">

<span class="dot one"></span>

<span class="gender">Male</span>

</label>

<label for="dot-2">

<span class="dot two"></span>

<span class="gender">Female</span>

</label>

</div>

</div>

<div class="button">

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

</div>

</form>

</div>

</div>

</body>

</html>

**Newuser.jsp**

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

<%

Connection con;

PreparedStatement ps;

ResultSet rs;

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

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

String address=request.getParameter("address");

String city=request.getParameter("city");

String mno=request.getParameter("mno");

String emailid=request.getParameter("emailid");

String username=request.getParameter("username");

String password=request.getParameter("password");

Class.forName("com.mysql.jdbc.Driver");

con=DriverManager.getConnection("jdbc:mysql://localhost:3306/kan","root","briztech");

ps=con.prepareStatement("select \* from users where username=?");

ps.setString(1,username);

rs=ps.executeQuery();

if(rs.next())

{

out.println("Already Registered User<br><br>") ;

}

else

{

ps=con.prepareStatement("insert into users values(?,?,?,?,?,?,?,?)");

ps.setString(1,name);

ps.setString(2,gender);

ps.setString(3,address);

ps.setString(4,city);

ps.setString(5,mno);

ps.setString(6,emailid);

ps.setString(7,username);

ps.setString(8,password);

ps.execute();

out.println("<b>user registered");

}

%>

<html>

<body>

<style>

a {

background-color: #04AA6D;

color: white;

padding: 14px 20px;

margin: 8px 0;

border: none;

cursor: pointer;

text-decoration: none;

}

a:hover {

opacity: 0.8;

}

</style>

<br>

</br>

</br>

<a href="login.html">Login</a>

</body>

</html>

**Login.jsp**

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

<html>

<body>

<%

Connection con;

PreparedStatement ps;

ResultSet rs;

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

String pass=request.getParameter("pass");

Class.forName("com.mysql.jdbc.Driver");

con=DriverManager.getConnection(

"jdbc:mysql://localhost:3306/kan","root","briztech");

ps=con.prepareStatement("select \* from users where username=? and password=?");

ps.setString(1,name);

ps.setString(2,pass);

rs=ps.executeQuery();

if(rs.next())

{

response.sendRedirect("welcome.jsp");

}

else

{

response.sendRedirect("error.html");

}

%>

</body>

</html>

**Displaybooking.jsp**

<html>

<body>

<style>

a {

background-color: #04AA6D;

color: white;

padding: 14px 20px;

margin: 8px 0;

border: none;

cursor: pointer;

text-decoration: none;

}

a:hover {

opacity: 0.8;

}

th{

color: blue;

}

table,td{

text-align: center;

}

</style>

<body>

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

<%

Connection con;

PreparedStatement ps;

ResultSet rs;

String id=request.getParameter("id");

Class.forName("com.mysql.jdbc.Driver");

con=DriverManager.getConnection(

"jdbc:mysql://localhost:3306/kan","root","briztech");

ps=con.prepareStatement("select \* from customers where id=?");

ps.setString(1,id);

rs=ps.executeQuery();

if(rs.next())

{

out.println("<center>");

out.println("<table border=1>");

out.println("<tr>");

out.println("<th colspan=5>");

out.println("Room Detail");

out.println("</th>");

out.println("</tr>");

out.println("<tr>");

out.println("<th>");

out.println("ID");

out.println("</th>");

out.println("<th>");

out.println("Name");

out.println("</th>");

out.println("<th>");

out.println("Father Name");

out.println("</th>");

out.println("<th>");

out.println("Room No.");

out.println("</th>");

out.println("<th>");

out.println("Room Charge");

out.println("</th>");

out.println("</tr>");

out.println("<tr>");

out.println("<td>");

out.println(rs.getString("id"));

out.println("</td>");

out.println("<td>");

out.println(rs.getString("yname"));

out.println("</td>");

out.println("<td>");

out.println(rs.getString("fname"));

out.println("</td>");

out.println("<td>");

out.println(rs.getString("rno"));

out.println("</td>");

out.println("<td>");

out.println(rs.getString("rate"));

out.println("</td>");

out.println("</tr>");

out.println("</table>");

out.println("</center>");

String rno=rs.getString("rno");

String rate=rs.getString("rate");

int room=Integer.parseInt(rno);

int rent=Integer.parseInt(rate);

out.println("Payable Amount "+(room\*rent));

}

else

{

out.println("CUSTOMERID not valid");

}

%>

</body>

</html>

<html>

<body>

<style>

a {

background-color: #04AA6D;

color: white;

padding: 14px 20px;

margin: 8px 0;

border: none;

cursor: pointer;

text-decoration: none;

}

a:hover {

opacity: 0.8;

}

</style>

<br>

</br>

</br>

<a href="welcome.jsp">Back To Home</a>

</body>

</html>

**Cancelbooking.jsp**

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

<%

String id=request.getParameter("id");

Connection con;

PreparedStatement ps;

ResultSet rs;

Class.forName("com.mysql.jdbc.Driver");

con=DriverManager.getConnection(

"jdbc:mysql://localhost:3306/kan","root","briztech");

ps=con.prepareStatement("SELECT \* FROM CUSTOMERS where id=?");

ps.setString(1,id);

rs=ps.executeQuery();

if(rs.next())

{

ps=con.prepareStatement("DELETE FROM customers where id=?");

ps.setString(1,id);

ps.execute();

out.println("Booking Cancelled.");

}

else

{

out.println("CUSTOMERID is INVALID");

}

%>

<html>

<body>

<style>

a {

background-color: #04AA6D;

color: white;

padding: 14px 20px;

margin: 8px 0;

border: none;

cursor: pointer;

text-decoration: none;

}

a:hover {

opacity: 0.8;

}

</style>

<br>

</br>

</br>

<a href="welcome.jsp">Back To Home</a>

</body>

</html>

**Updatebooking.jsp**

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

<%

String id=request.getParameter("id");

String yname=request.getParameter("yname");

String fname=request.getParameter("fname");

String rno=request.getParameter("rno");

String rate=request.getParameter("rate");

Connection con;

PreparedStatement ps;

ResultSet rs;

Class.forName("com.mysql.jdbc.Driver");

con=DriverManager.getConnection(

"jdbc:mysql://localhost:3306/kan","root","briztech");

ps=con.prepareStatement("select \* from customers where id=?");

ps.setString(1,id);

rs=ps.executeQuery();

if(rs.next())

{

ps=con.prepareStatement("update customers set yname=? fname=? rno=? where id=?");

ps.setString(1,id);

ps.setString(2,yname);

ps.setString(3,fname);

ps.setString(4,rno);

ps.execute();

out.println("Booking Successfully Updated");

}

else

{

out.println("ID is INVALID");

}

%>

<html>

<body>

<style>

a {

background-color: #04AA6D;

color: white;

padding: 14px 20px;

margin: 8px 0;

border: none;

cursor: pointer;

text-decoration: none;

}

a:hover {

opacity: 0.8;

}

</style>

<br>

</br>

</br>

<a href="welcome.jsp">Back To Home</a>

</body>

</html>

Images here only .html file snapshots

**BIBLIOGRAPHY**

**BOOKS and RFERENCES**

**HTML**:- McGraw Hill

**Java**:-E Balagurusamy

**JSP** :Gary Bollinger,Bharathi Natarajan

**WEB SITES :-**

•www.Google.com

•www.Codeguru.com

•www.tutorialspoint.com

•www.w3school.com

**THE END**