SCTR's

Pune Institute of Computer Technology, Dhankawadi, Pune 411043

Department of Computer Engineering

Lab Manual of

Web Technology Lab (310257)

For T.E. (Computer Engineering)

Academic Year: 2025-2026

Semester: VI

Preparation of Lab Manual Contributed By

Sr No	Name of experiment	Faculty name
1	• Case study: • Visit various websites available online for different client projects, identify and note down the evaluation results in following format and note down and learn and conclude different website design issues that should be considered while designing a web site.	Prof. Ashwini Jewalikar
2	1. Implement a web page index.htm for any client website (e.g., a restaurant website project) using following: a. HTML syntax: heading tags, basic tags and attributes, frames, tables, images, lists, links for text and images, forms etc. b. Use of Internal CSS, Inline CSS, External CSS	Prof Ashwini Jewalikar
3	Design the XML document to store the information of the employees of any business organization and demonstrate the use of: a) DTD b) XML Schema And display the content in (e.g., tabular format) by using CSS/XSL.	Prof. Parag Jambhulkar
4	Implement an application in Java Script using following: a) Design UI of application using HTML, CSS etc. b) Include Java script validation c) Use of prompt and alert window using Java Script e.g., Design and implement a simple calculator using Java Script for operations like addition, multiplication, subtraction, division, square of number etc. a) Design calculator interface like text field for input and output, buttons for numbers and operators etc. b) Validate input valuesvPrompt/alerts for invalid values etc.	Prof. Parag Jambhulkar

5	Implement the sample program demonstrating the use of Servlet. e.g., Create a database table ebookshop (book_id, book_title, book_author, book_price, quantity) using database like Oracle/MySQL etc. and display (use SQL select query) the table content using servlet.	Prof. V S Gaikwad
6	Implement the program demonstrating the use of JSP. e.g., Create a database table students_info (stud_id, stud_name, class, division, city) using database like Oracle/MySQL etc. and display (use SQL select query) the table content using JSP.	Prof. V S Gaikwad
7	Build a dynamic web application using PHP and MySQL. a. Create database tables in MySQL and create connection with PHP. Create the add, update, delete and retrieve functions in the PHP web app interacting with MySQL database	Prof. V S Gaikwad
8	Design a login page with entries for name, mobile number email id and login button. Use struts and perform following validations a. Validation for correct names b. Validation for mobile numbers c. Validation for email id d. Validation if no entered any value e. Re-display for wrongly entered values with message Congratulations and welcome page upon successful entries	Prof. P. T. Kohok
9	Design an application using Angular JS. e.g., Design registration (first name, last name, username, password) and login page using Angular JS.	Prof. P. T. Kohok
10	Design and implement a business interface with necessary business logic for any web application using EJB. e.g., Design and implement the web application logic for deposit and withdraw amount transactions using EJB.	Prof. P. T. Kohok
11	**Mini Project**: Design and implement a dynamic web application for any business functionality by using web development technologies that you have learn in the above given assignments.	Prof. Parag Jambhulkar

Pune Institute of Computer Technology,
Dhankawadi, Pune - 411043
Department of Computer Engineering

Subject: Web Technology Lab (310257) **Academic Year:** 2025–26
Class: TE **Semester:** II

Index

<table>
<tr><th>Sr No.</th><th>Name of assignment</th></tr>
<tr><td>1</td><td>Case study:
Visit various websites available online for different client projects, identify and note down the evaluation results in following format and note down and learn and conclude different website design issues that should be considered while designing a web site.
<table>
<tr><td>Sr No.</td><td>Website URL</td><td>Purpose of website</td><td>Things liked in the website</td><td>Things disliked in the website</td><td>overall evaluation (Good/Bad)</td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table></td></tr>
<tr><td>2</td><td>Implement a web page index.htm for any client website (e.g., a restaurant website project) using following:
a. HTML syntax: heading tags, basic tags and attributes, frames, tables, images, lists, links for text and images, forms etc.
b. Use of Internal CSS, Inline CSS, External CSS</td></tr>
<tr><td>3</td><td>Design the XML document to store the information of the employees of any business organization and demonstrate the use of:
a) DTD
b) XML Schema
And display the content in (e.g., tabular format) by using CSS/XSL.</td></tr>
<tr><td>4</td><td>Implement an application in Java Script using following:
a) Design UI of application using HTML, CSS etc.
b) Include Java script validation
c) Use of prompt and alert window using Java Script
e.g., Design and implement a simple calculator using Java Script for operations like addition, multiplication, subtraction, division, square of number etc.
a) Design calculator interface like text field for input and output, buttons for numbers and operators etc.
b) Validate input values
c) Prompt/alerts for invalid values etc.</td></tr>
<tr><td>5</td><td>Implement the sample program demonstrating the use of Servlet.
e.g., Create a database table ebookshop (book_id, book_title, book_author, book_price, quantity) using database like Oracle/MySQL etc. and display (use SQL select query) the table content using servlet.</td></tr>
</table>

6	Implement the program demonstrating the use of JSP. e.g., Create a database table students_info (stud_id, stud_name, class, division, city) using database like Oracle/MySQL etc. and display (use SQL select query) the table content using JSP.
7	Build a dynamic web application using PHP and MySQL. a. Create database tables in MySQL and create connection with PHP. b. Create the add, update, delete and retrieve functions in the PHP web app interacting with MySQL database
8	Design a login page with entries for name, mobile number email id and login button. Use struts and perform following validations a. Validation for correct names b. Validation for mobile numbers c. Validation for email id d. Validation if no entered any value e. Re-display for wrongly entered values with message f. Congratulations and welcome page upon successful entries
9	Design an application using Angular JS. e.g., Design registration (first name, last name, username, password) and login page using Angular JS.
10	Design and implement a business interface with necessary business logic for any web application using EJB. e.g., Design and implement the web application logic for deposit and withdraw amount transactions using EJB.
11	**Mini Project**: Design and implement a dynamic web application for any business functionality by using web development technologies that you have learn in the above given assignments.

Subject Coordinator

Head of the Department

ASSIGNMENT NO: 1

TITLE	Case study:Web site analysis
PROBLEM STATEMENT /DEFINITION	Visit various websites available online for different client projects, identify and note down the evaluation results in following format and note down and learn and conclude different website design issues that should be considered while designing a web site.
OBJECTIVE	● To Analyse the website contents ● To understand web site design issues
S/W PACKAGES AND HARDWARE APPARATUS USED	Latest Version of 64 bit Operating Systems, Open Source Fedora-GHz. 8 G.B. RAM, 500 G.B. HDD, 15"Color Monitor, Keyboard, Mouse
REFERENCES	1. https://www.tutorialspoint.com/articles/tag/apache-tomcat-server 2. *www.w3schools.com*
STEPS	Refer to details
INSTRUCTIONS FOR WRITING JOURNAL	● Title ● Problem Definition ● Objectives ● Theory ● Installation steps of all the servers ● Test cases ● Output ● Conclusion

Designing Websites

Define Your Sites Purpose and Strategy. ...

Research the Latest Web Design Trends. ...

Choose Your Platform. ...

Select a Template / Theme. ...

Decide on Your Branding. ...

Add In and Optimize Your Content. ...

Publish Your Website. ...

Analyze and Improve. ...

Selection of Server:

Web Server and Application server

Most of the times these terms Web Server and Application server are used interchangeably.

Following are some of the key differences in features of Web Server and Application Server:

- Web Server is designed to serve HTTP Content. App Server can also serve HTTP Content but is not limited to just HTTP. It can be provided other protocol support such as RMI/RPC
- Web Server is mostly designed to serve static content, though most Web Servers have plugins to support scripting languages like Perl, PHP, ASP, JSP etc. through which these servers can generate dynamic HTTP content.
- Most of the application servers have Web Server as integral part of them, that means App Server can do whatever Web Server is capable of. Additionally App Server have components and features to support Application level services such as Connection Pooling, Object Pooling, Transaction Support, Messaging services etc.
- As web servers are well suited for static content and app servers for dynamic content, most of the production environments have web server acting as reverse proxy to app server. That means while servicing a page request, static contents (such as images/Static HTML) are served by web server that interprets the request. Using some kind of filtering technique (mostly extension of requested resource) web server identifies dynamic content request and transparently forwards to app server

Components of Tomcat

1. Catalina : It is the Servlet Container of Tomcat.
2. Coyote : Coyote acts as a connector and supports HTTP 1.1
3. Jasper : It is the Tomcat‘s JSP Engine.
4. Cluster : A component for load balancing to manage large applications.
5. High availability : A Tomcat component to schedule system upgrades and changes without affecting live environment.
6. Web Application : Manage Sessions, Support deployment across different environments.

This article will walk you throughout the process of installing Apache Tomcat 8 (i.e. 8.5.14) on Linux systems, which includes RHEL, CentOS, Fedora, Debian, Ubuntu, etc.

Step 1: Installing Java 8

1. Before installing Tomcat make sure you have the latest version of Java Development Kit (JDK) installed and configured on the system. It is preferred to use oracle Java.
To install latest Oracle Java JDK (jdk-8u131) on Linux, you may like to refer our recent posts on Oracle jdk/jre/jar

Step 2: Download and Install Apache Tomcat 8
2. Once latest Java installed and configured correctly on the system, we will move forward to download and install latest stable version of Tomcat 8

Steps for Tomcat installation

1. Download tomcat tar file.
2. create tomcat installtion directory anywhere using command mkdir command.

e.g. mkdir /opt/tomcat_installation

3. cp apache-tomcat-{version}.tar.gz

/opt/tomcat_installation 4.cd /opt/tomcat_installation

5. tar -xvf apache-tomcat-{version}.tar.gz
6. It will extract apache-tomcat-{version}.tar.gz
7. It will also include bin directory where there are binaries for

tomcat. 8.cd bin.

9../startup.sh

It will show Tomcat started.

10. Now copy html file that you want to host on tomcat server

into tomcat_installation/webapps/

11. Open browser type http://localhost:8080/hello. if hello.html is copied in step number 10

in /opt/tomcat_installation/ webapps/ROOT/

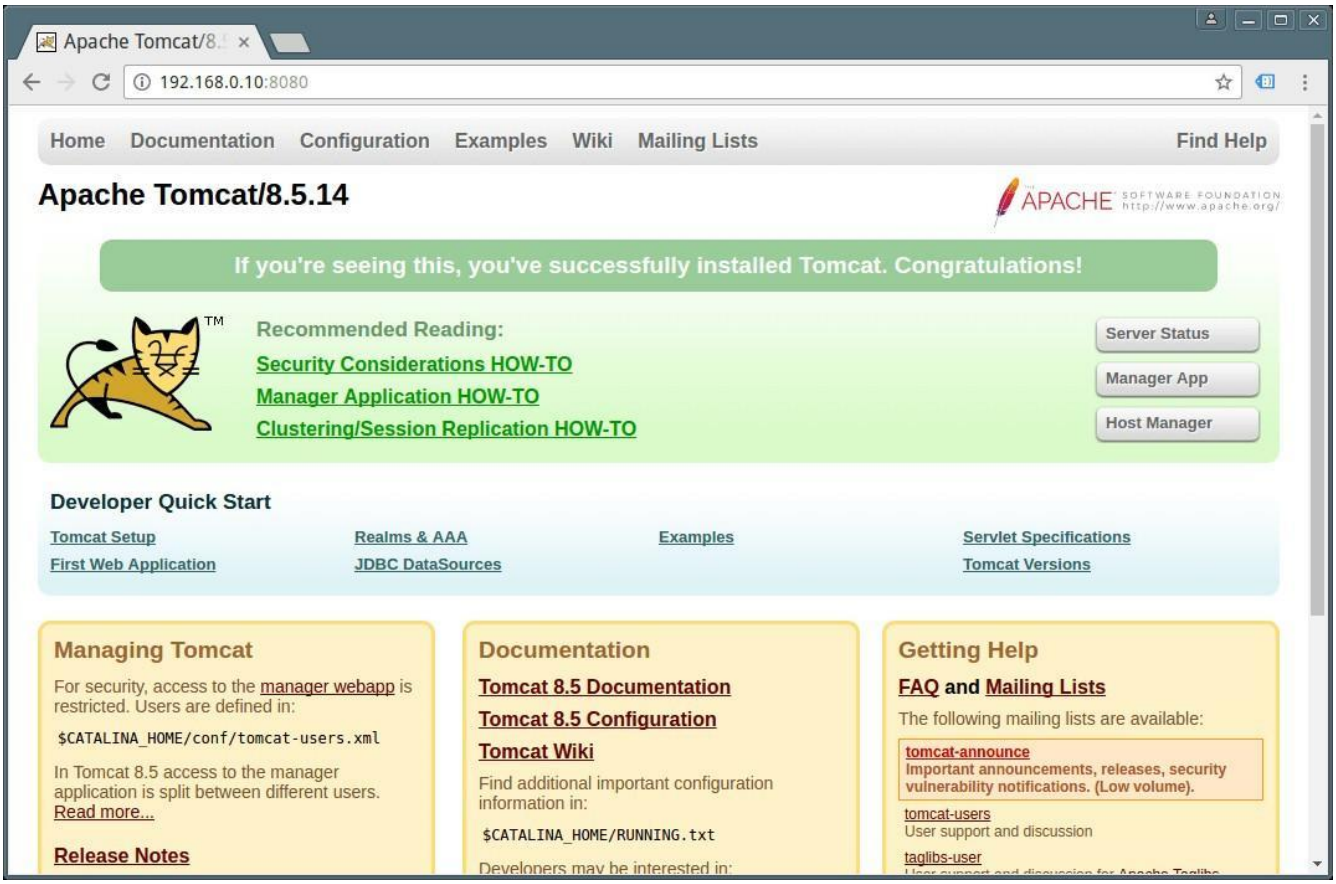
12. Create lib directory using mkdir command inside tomcat_installation/webapps/
13. Now in order to host jsp pages which will connect to database we have to copy jstl-1.2.jar and mysql-

connector.jar to tomcat_installation/webaps/ftp://192.168.4.87/pub.

14. shutdown tomcat server using ./shutdown.sh command from bin

directory. 15.cp jsp files tomcat_installation/webapps/

Once _Tomcat Started_, you can point your browser to http://127.0.0.1:8080 and you should see something as:



Getting started with JBoss AS 7 in Fedora

From a terminal, install JBoss AS 7 using dnf or yum:

sudo dnf -y install jboss-as
sudo yum -y install jboss-as

Start the JBoss AS 7 system service:

sudo systemctl start jboss-as.service

Connect to the JBoss AS 7 management console (for the system instance):

sudo -u jboss-as sh -c "jboss-cli -c"

When you connect to the management console, the server will send a secret key challenge to the client. The client can only pass the challenge if it has physical direct access to the file system and the same permissions as the user running the server. Otherwise, you'd need to create and use a proper management user.

Stop the JBoss AS 7 system service:

sudo systemctl stop jboss-as.service

Create a user instance of JBoss AS 7:

jboss-as-cp -l $HOME/jboss-as-user-instance

Start the JBoss AS 7 user instance:

$HOME/jboss-as-user-instance/bin/standalone.sh

standalone.sh is a script generated by jboss-as-cp that effectively runs this command

JBOSS_BASE_DIR=$HOME/jboss-as-user-instance /usr/share/jboss-as/bin/standalone.sh

-c standalone-web.xml

Connect to the JBoss AS 7 management console (for the user instance):

jboss-cli -c

GLASSFISH:

And between the best GlassFish Advantages you find: it‘s Free, No vendor Lock-in and Easy Adaptability (Supports MySQL, .NET, Eclipse and NetBeans Integration).

GlassFish is the Java EE reference implementation

Support Latest version of the JEE 7 Specification

Download GlassFish 4 JEE 7 App Server for Linux:

Here Get GlassFish 4 latest-glassfish-unix.sh

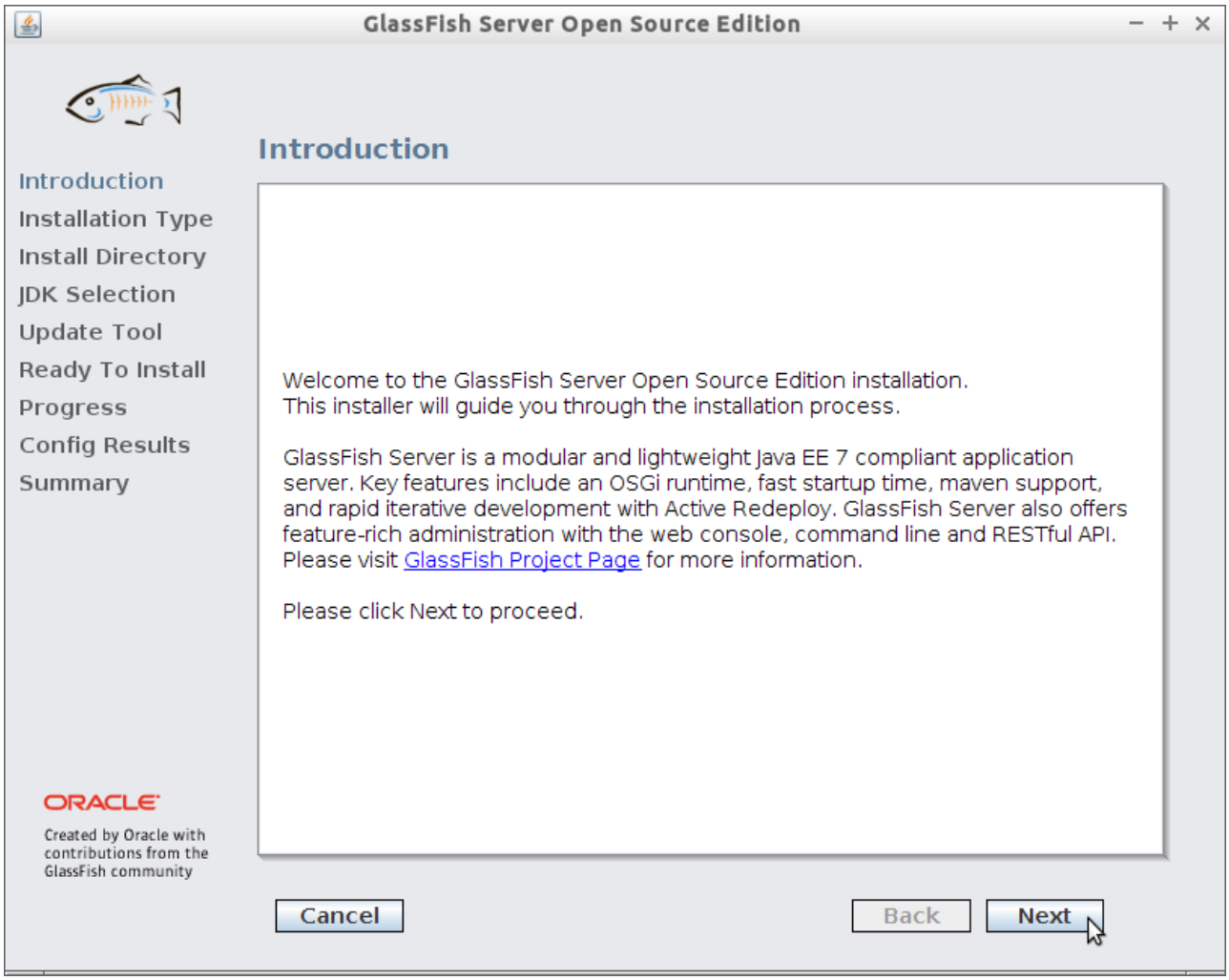
Start GlassFish 4 Installation

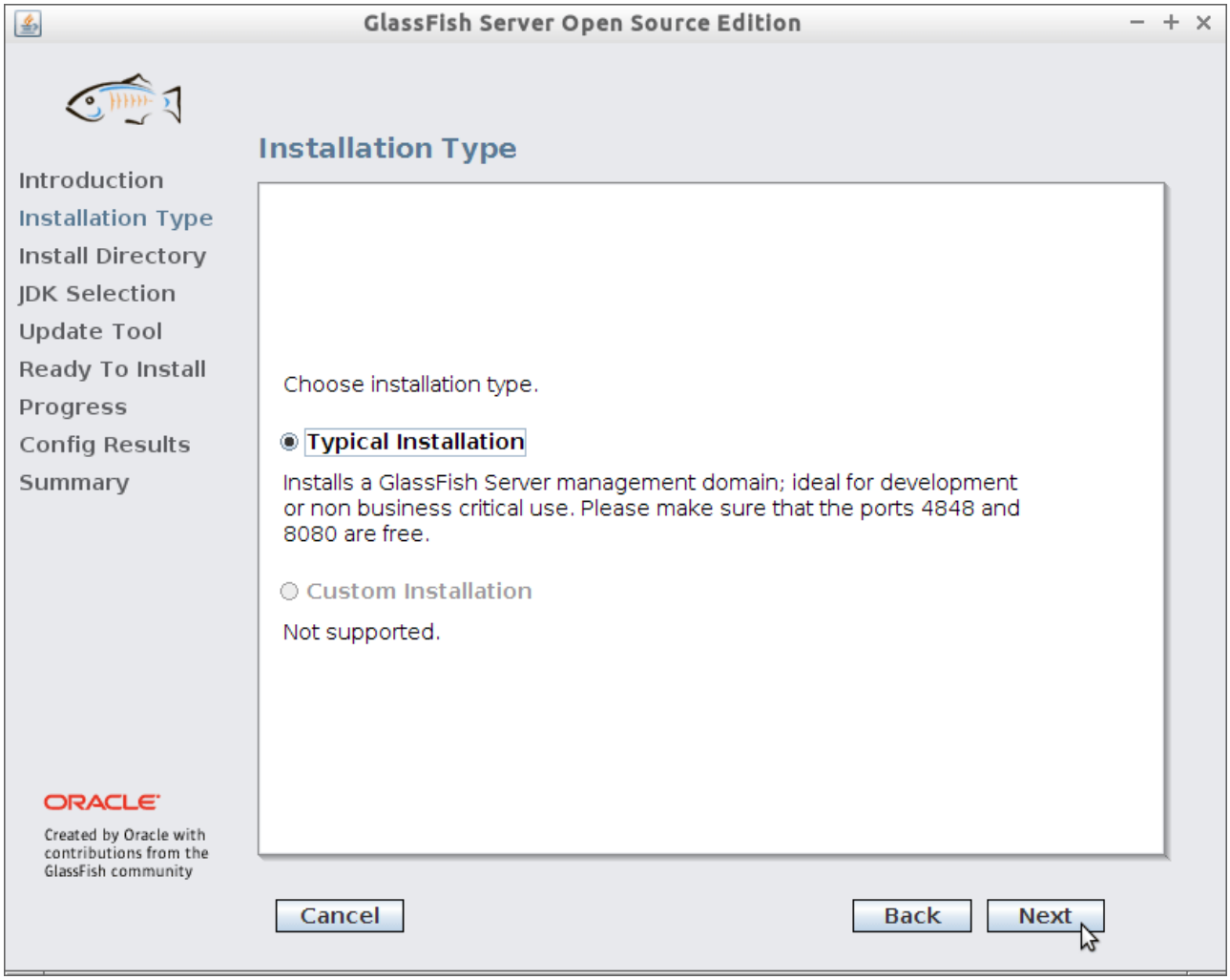
```
sudo su
```

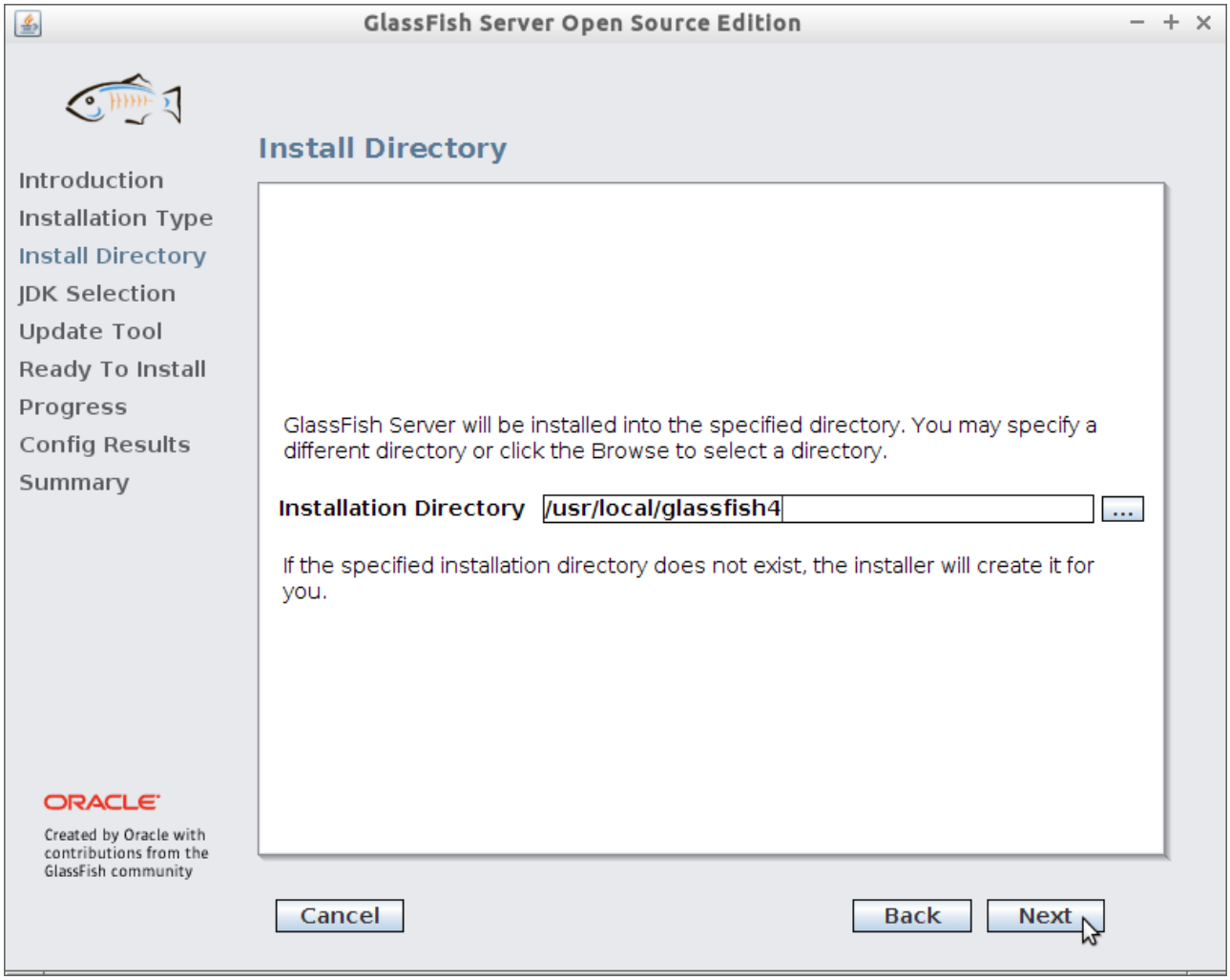
If Got —User is Not in Sudoers file‖ then Look: Solution

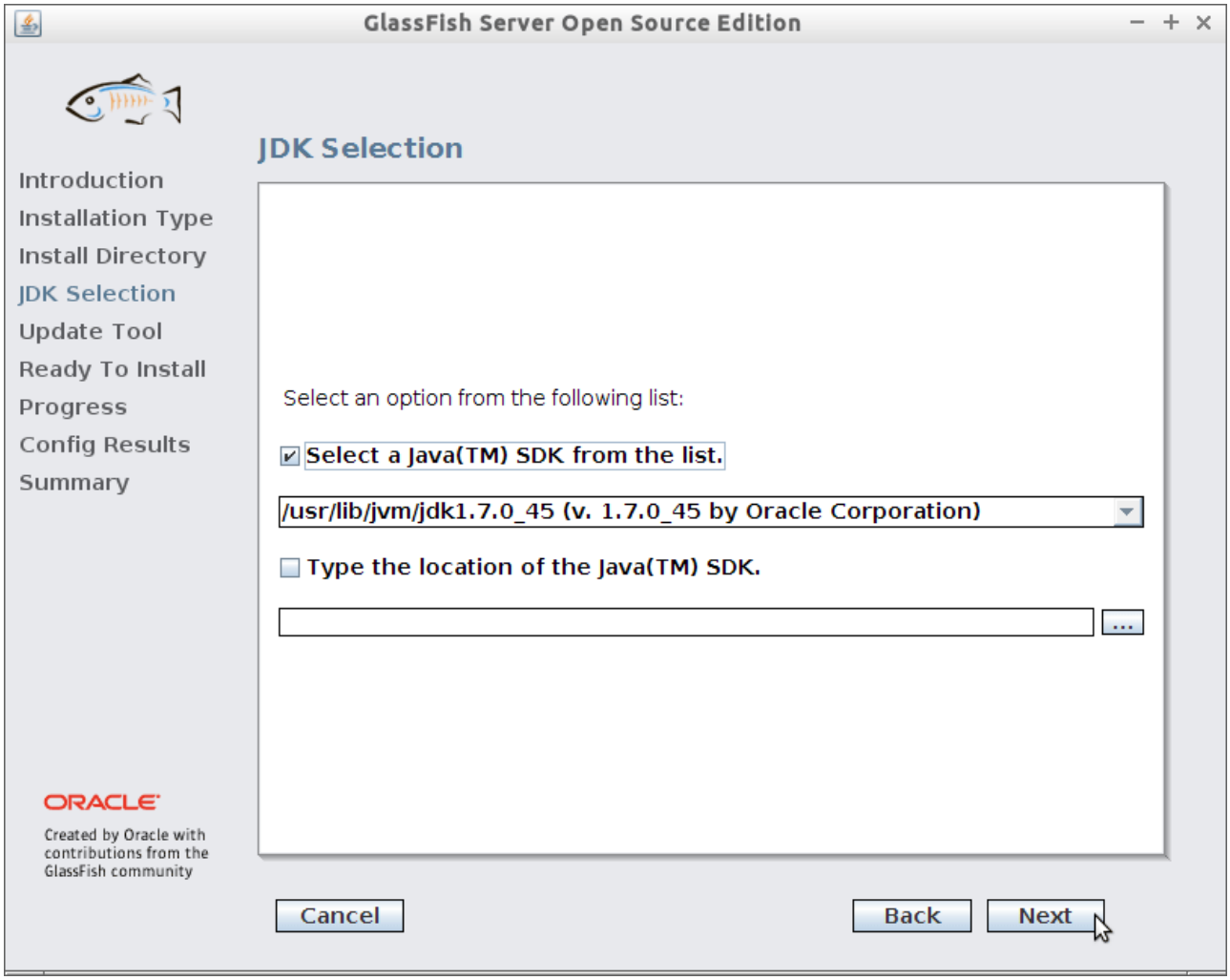
```
cd </path/2>/latest-glassfish-unix.sh
chmod +x ./latest-glassfish-unix.sh
```

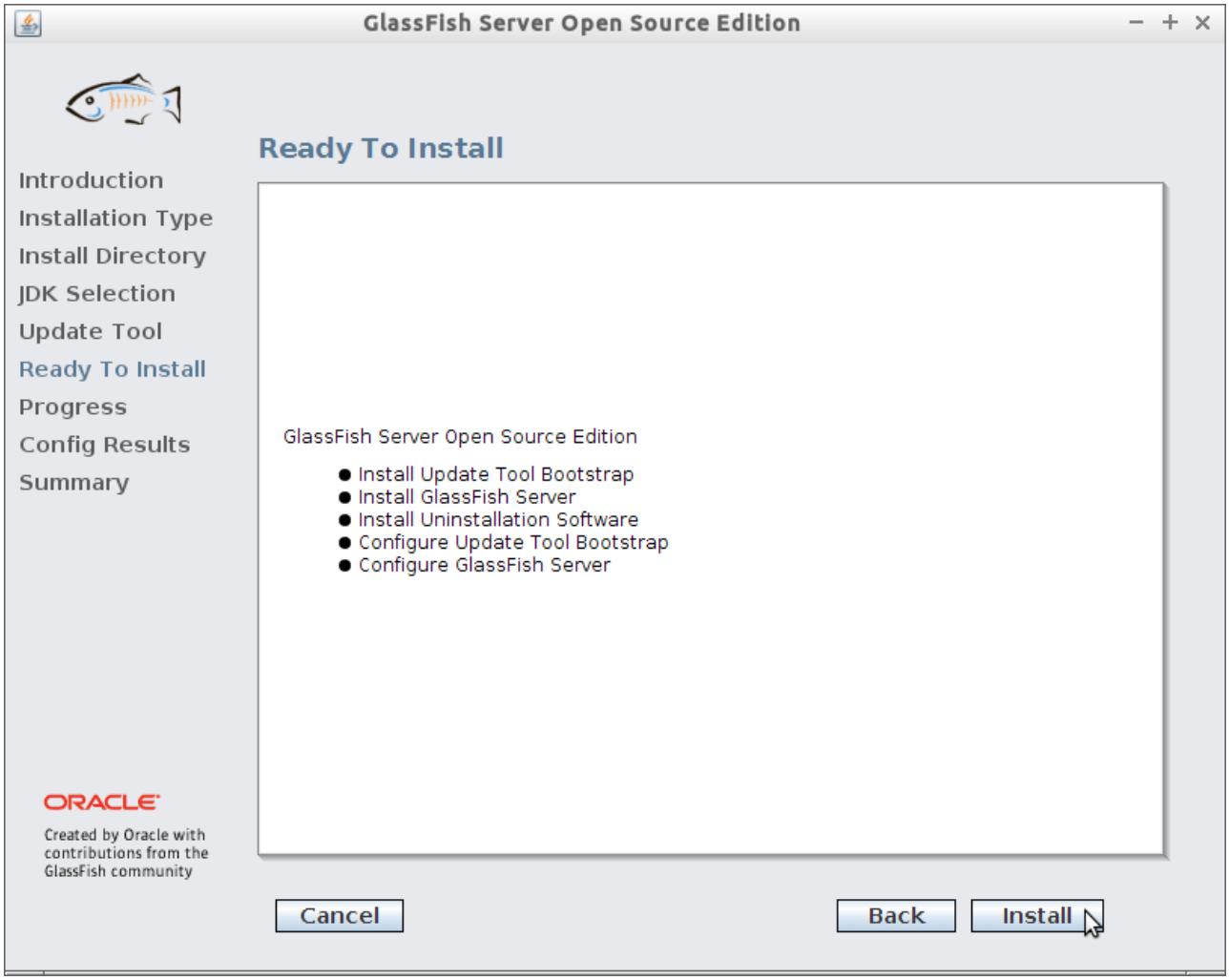
Installing GlassFish 4 JEE 7 App Server

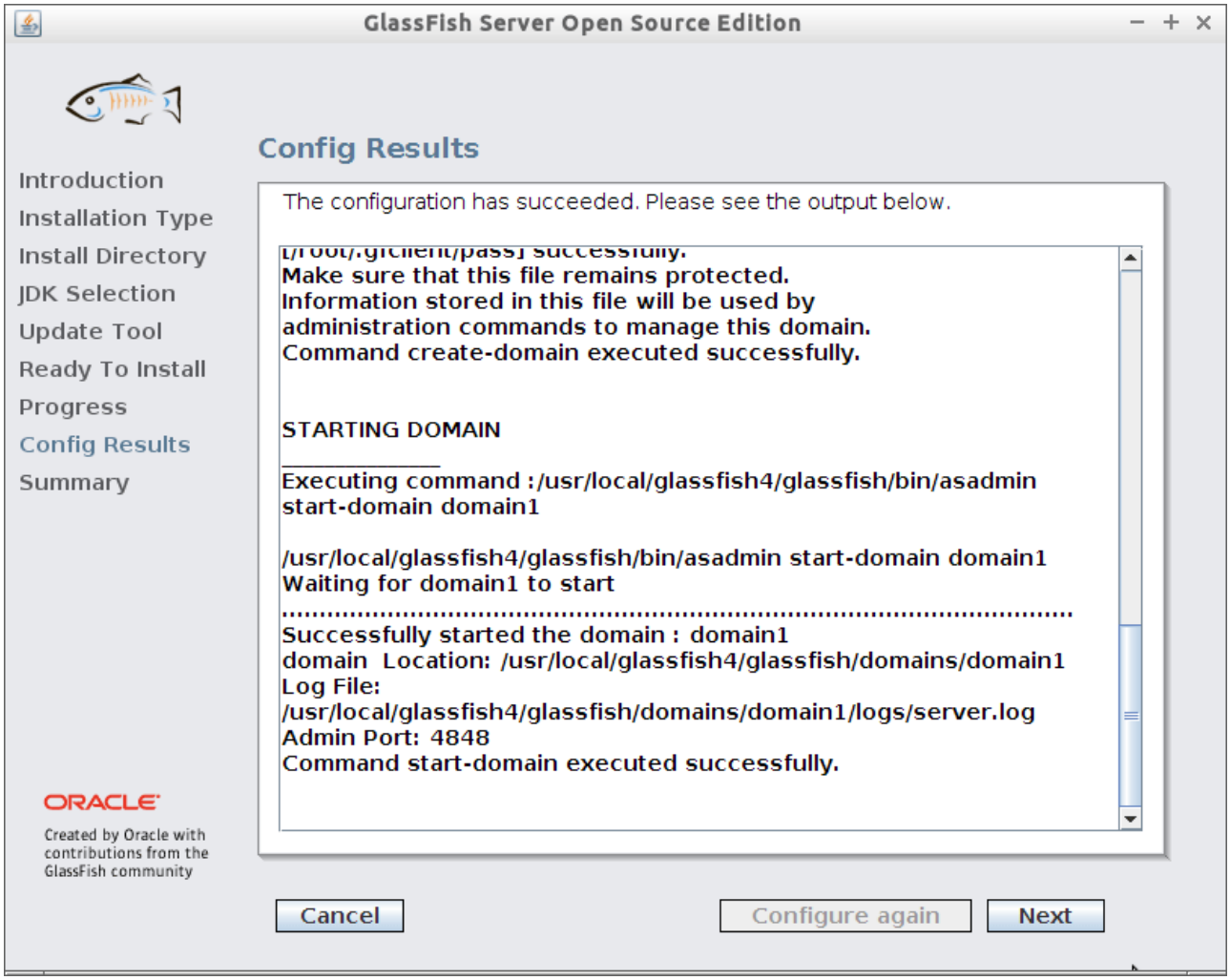


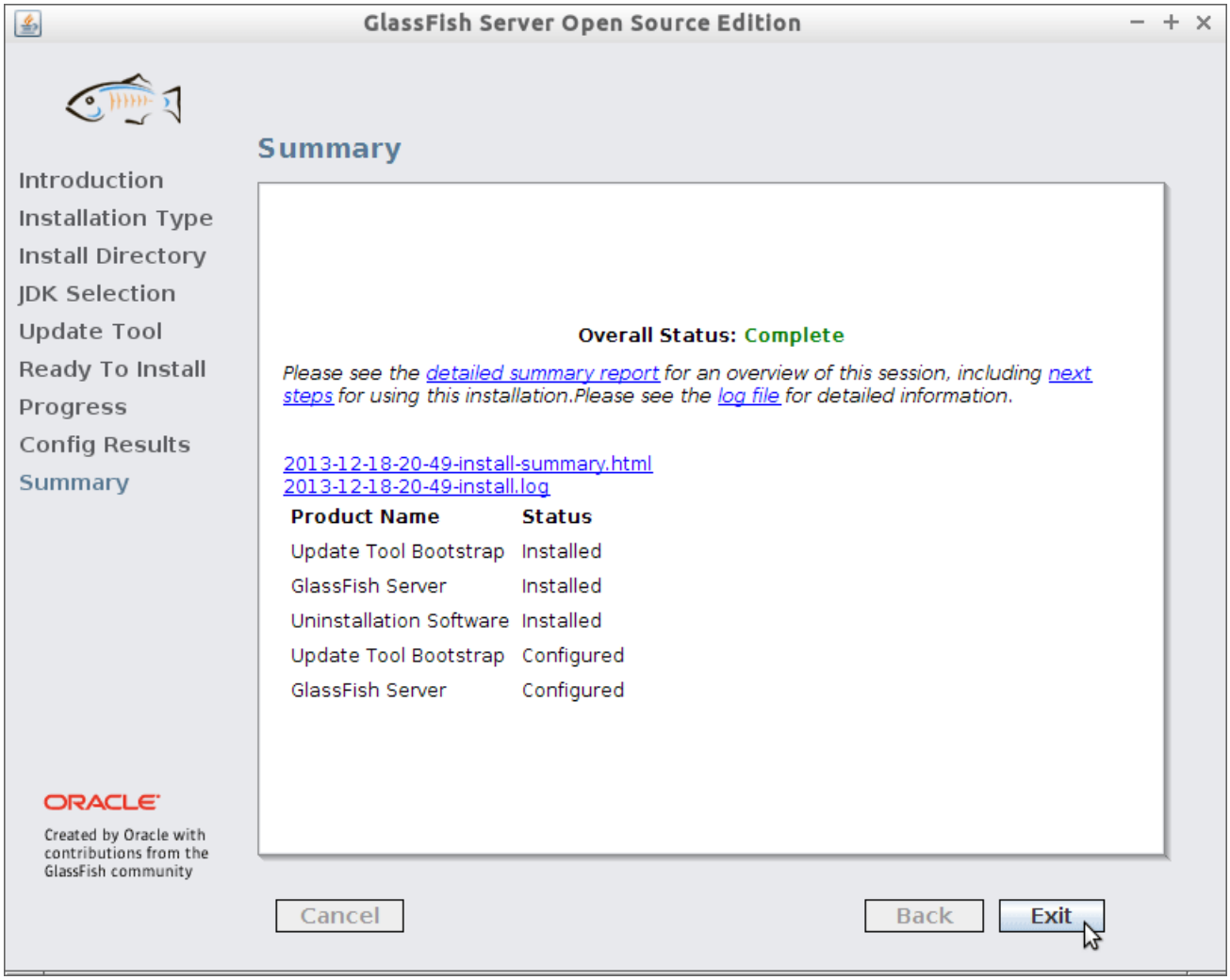

GlassFish Server Open Source Edition
Introduction
Installation Type
Install Directory
JDK Selection
Update Tool
Ready To Install
Progress
Config Results
Summary
Installation Type
Choose installation type.
Typical Installation
Installs a GlassFish Server management domain; ideal for development or non business critical use. Please make sure that the ports 4848 and 8080 are free.
Custom Installation
Not supported.
ORACLE
Created by Oracle with contributions from the GlassFish community
Cancel
Back
Next


GlassFish Server Open Source Edition
Introduction
Installation Type
Install Directory
JDK Selection
Update Tool
Ready To Install
Progress
Config Results
Summary
Install Directory
GlassFish Server will be installed into the specified directory. You may specify a different directory or click the Browse to select a directory.
Installation Directory
/usr/local/glassfish4
...
If the specified installation directory does not exist, the installer will create it for you.
ORACLE
Created by Oracle with contributions from the GlassFish community
Cancel
Back
Next


GlassFish Server Open Source Edition
JDK Selection
Introduction
Installation Type
Install Directory
JDK Selection
Update Tool
Ready To Install
Progress
Config Results
Summary
Select an option from the following list:
Select a Java(TM) SDK from the list.
/usr/lib/jvm/jdk1.7.0_45 (v. 1.7.0_45 by Oracle Corporation)
Type the location of the Java(TM) SDK.
...
ORACLE
Created by Oracle with contributions from the GlassFish community
Cancel
Back
Next


GlassFish Server Open Source Edition
Ready To Install
Introduction
Installation Type
Install Directory
JDK Selection
Update Tool
Ready To Install
Progress
Config Results
Summary
GlassFish Server Open Source Edition
Install Update Tool Bootstrap
Install GlassFish Server
Install Uninstallation Software
Configure Update Tool Bootstrap
Configure GlassFish Server
ORACLE
Created by Oracle with contributions from the GlassFish community
Cancel
Back
Install


GlassFish Server Open Source Edition
Config Results
Introduction
Installation Type
Install Directory
JDK Selection
Update Tool
Ready To Install
Progress
Config Results
Summary
The configuration has succeeded. Please see the output below.
t/root/.glclient/pass] successfully.
Make sure that this file remains protected.
Information stored in this file will be used by
administration commands to manage this domain.
Command create-domain executed successfully.
STARTING DOMAIN
Executing command :/usr/local/glassfish4/glassfish/bin/asadmin
start-domain domain1
/usr/local/glassfish4/glassfish/bin/asadmin start-domain domain1
Waiting for domain1 to start
Successfully started the domain : domain1
domain Location: /usr/local/glassfish4/glassfish/domains/domain1
Log File:
/usr/local/glassfish4/glassfish/domains/domain1/logs/server.log
Admin Port: 4848
Command start-domain executed successfully.
ORACLE
Created by Oracle with contributions from the GlassFish community
Cancel
Configure again
Next



Access Glassfish Admin Console on Browser: http://localhost:4848

References:

1. https://tutorialforlinux.com/how-to-install-apache-tomcat-java-ee-server-on-linux-distributions-easy-guides/
2. https://www.tutorialspoint.com/articles/tag/apache-tomcat-server
3. *www.w3schools.com*

Oral Questions:

1. How to start / stop Apache tomcat?

2. What is the default port for HTTP and HTTPs
3. List the important configuration file name of tomcat.
4. How to check version of running Apache tomcat?
5. How to know if web server is running?
6. How to install Apache web server?
7. Differentiate between web server and application server.
8. What is Glassfish application server?
9. How to start and stop the Default Domain?
10. How to start administration console in GlassFish?

ASSIGNMENT NO: 2

TITLE	***Design with CSS, HTML.***
PROBLEM STATEMENT /DEFINITION	Implement a web page index.htm for any client website (e.g., a restaurant website project) using following: 1. HTML syntax: heading tags, basic tags and attributes, frames, tables, images, lists, links for text and images, forms etc. 2. Use of Internal CSS, Inline CSS, External CSS
OBJECTIVE	● To develop web pages using HTML ● To optimize page styles & layout with CSS
S/W PACKAGES AND HARDWARE APPARATUS USED	Operating System open source Fedora 20 or higher equivalent or Windows Networked computer with internet access Editor : IDE , Eclipse or any simple equivalent editor (i.e. text based & WYSIWYG based) Web browser / Internet explorer 7
REFERENCES	1. Web enabled commercial application development using HTML , DHTML , JavaScript , Perl CGI by Ivan Bayross 2. Musiciano C. Kennedy B., _ HTML & XHTML ‘ , 5th or higher edition , O‘Reilly / SPD Publications , ISBN B1 – 7366 – 517 – 1 3. Learning XML by Erik T. Ray , O‘reilly 4. Internet & World Wide Web , How to Programe , 3rd or higher edition , H. M. Deitel , P. J. Deitel , A. B. Goldberg , Pearson education 5. McKinnon A., McKinnon L., _ XML ‘ , Vikas Publishing House , 2004 , ISBN 981 – 254 – 299 – X 6. https://www.w3schools.com/xml/
STEPS	Refer to steps below
INSTRUCTIONS FOR WRITING JOURNAL	● Title ● Problem Statement ● Description (HTML vs XML & various tags in it CSS) ● Block diagrams (design part)

	● Troubleshooting (if any) ● Conclusion ● References

Problem Statement:

Implement a web page index.htm for any client website (e.g., a restaurant website project) using following:

1. HTML syntax: heading tags, basic tags and attributes, frames, tables, images, lists, links for text and images, forms etc.
2. Use of Internal CSS, Inline CSS, External CSS

Objectives:

1) To develop web pages using HTML
2) To optimize page styles & layout with CSS

Outcomes:

1) Define the key terms relevant to coding HTML and CSS, including: tag, attribute, element, entity, selector, header, table, ordered list, unordered list, link, heading, paragraph ; et cetra .
2) Describe the function of common tags & styles in short snippets of code & predict the output of the same .
3) Apply Internal,external and inline CSS to file
4) Define & compare the concepts of multimedia , hypermedia & hypertext .

Theory:

Introduction:

Files that travel across the largest network in the world , the Internet , & carry information from _ Server ‘ to _ Client ‘ that requested them are called _ **Web pages** / HTML documents ‘ . Individual who develops these web pages is called _ **Web Developer** ‘ . Web Pages are created using HTML syntax . The organization of web pages into directories & files stored on the HDD of a computer is called _ **Web Site** ‘ creation . As studied in previous assignment , the Server Computer runs special software called _ **Web Server** ‘ software that allows :

- Web Site Management
- Accept a client‘s request for information
- Respond to a client‘s request by providing the page with the required information

Computers that offer the facility to read information stored in web pages are called _ **Web Clients** ‘ . Web Clients run special software called a _ **Browser** ‘ that allows to :

- Connect to an appropriate Server

- Query the Server for the information to be read
- Provides an interface to read the information returned by the Server

Following points emphasize the requirements of a good web site :

First Impression – Did the initial page grab the attention ?

Interface Design – Is the menu interface interactive enough & visually interesting ?

Corporate Mildew – Is the site trapped in a web of corporate look , feel & canned marketing speak ?

Coriolis Effect – Does the site generate enough currents of interest based on design & content for the user to comeback ?

HTML :

The language used to develop web pages is called **H**yper**T**ext **M**arkup **L**anguage which is interpreted by a Browser . HTML is a set of special codes that can be embedded in text to add formatting & linking information . **HTML Tags** are instructions that are embedded directly into text of document . It is a signal to a browser that it should do something other than just throw text up on the screen . HTML tags can be of two types :

Paired Tags Singular Tags

Some HTML tags require additional information to be supplied to them that are known as *Attributes* of a tag . Attribute(s) are written immediately following the tag , separated by a *space* . The creation of textual content of Web Site is done in any editor viz; Notepad / Eclipse / IDE ; et cetra & saved as *filename.**htm / .html*** file .

Tag	Name	Example
<!--	comment	<!--This can be viewed in the HTML part of a document-->
<a –	anchor	 Visit PICT
	bold	Example
<big>	big (text)	<big> Example </big>
<body>	body of HTML document	<body> The content of your HTML page </body>
 	line break	The contents of your page The contents of your page
	center	This will center your contents
<dd> <dl> <dt>	definition description definition list definition term	<dl> <dt> definition term </dt> <dd> definition of the term </dd> </dl>
	Emphasis	This is an example of using emphasis tag
<embed>	embed object	<embed src = ―your file‖ width = ―100%‖ height = ―60‖ align =

		—center‖>
	Font	 Example
<h1> <h2> <h3> <h4> <h5> <h6>	heading 1 heading 2 heading 3 heading 4 heading 5 heading 6	Heading 1 Example Heading 2 Example Heading 3 Example Heading 4 Example Heading 5 Example Heading 6 Example
head	heading of HTML document	<head>Contains elements describing the document</head>
<hr>	horizontal rule	<hr width="50%" size="3" noshade />
<html>	hypertext markup language	<html> <head> <meta> <title>Title of your web page</title> </head> <body>HTML web page contents </body> </html>
	Italic	Example
<img>	Image	
<input>	input field	**Ex:** <form method=post action="/cgibin/example.cgi"> <table border="0" cellspacing="0" cellpadding="2"><tr><td bgcolor="#8463ff"><input type="text" size="10" maxlength="30"></td><td bgcolor="#8463ff" valign="Middle"> <input type="image" name="submit" src="yourimage.gif"></td></tr> </table> </form> **Ex:** <form method=post action="/cgibin/example.cgi"> Select an option: <input type="radio" name="option"> Option 1 <input type="radio" name="option" checked> Option 2 <input type="radio" name="option"> Option 3 Select an option: <input type="checkbox" name="selection"> Selection 1 <input type="checkbox" name="selection" checked>Selection 2 <input type="checkbox" name="selection"> Selection 3 <input type="Submit" value="Submit"> <input type="Reset" value="Clear"> </form>
<menu>	Menu	<menu> <li type="disc">List item 1

 	list item ordered list	<li type="circle">List item 2 <li type="square">List item 3 </MENU> <ol type="i"> - List item 1 - List item 2
<link>	Link	<head> <link rel="stylesheet" type="text/css" href="style.css" /> </head>
<marquee>	scrolling text	<marquee bgcolor="#cccccc" loop="-1" scrollamount="2" width="100%">Example Marquee</marquee>
<meta>	meta	<meta http-equiv="Pragma" content="nocache">
<option>	listbox option	<form method=post action="/cgibin/example.cgi"> Select an option: <select> <option>option 1</option> <option selected>option 2</option> </select> </form>
<p>	paragraph	<p align="center"> This is an example displaying the use of the paragraph tag
<small>	small (text)	Example
<strike>	deleted text	<strike>Example</strike>
<table>	table	<table cellpadding="2" cellspacing="2" width="100%"> <tr> <th>Column 1</th> <td bgcolor="#cccccc">Column 1</td> <td bgcolor="#cccccc">Column 2</td> </tr> <tr> <td>Row 2</td> <td>Row 2</td> </tr> </table>
<title>	document title	<title>Title of your HTML page</title>
<u>	underline	<u>Example</u>
<tt>	teletype	<tt>Example</tt>
<style>	CSS	

DHTML (Dynamic HTML):

It combines HTML with Cascading Style Sheets (CSS) & Scripting Languages . HTML specifies a web page‘s element lie table , frame , paragraph , bulleted list ; etc. CSS can be used to determine an element‘s size , color , position & number of other features .

CSS (Cascading Style Sheets) :

Style Sheets are powerful mechanism for adding styles to Web documents that enforces standards & uniformity throughout a web site & provide numerous attributes to create dynamic effects . Style information can be associated with the web page in several ways :

- by embedding the style information directly through a STYLE attribute
- by embedding the style information directly through a < STYLE > header
- by embedding the style information directly through < LINK > element

Order of importance for adding style sheets into the document :

I. Inline styles
II. Embedded styles
III. Linked styles
IV. Imported styles
V. Default browser styles

Advantages:

- ✓ ability to make global changes to all documents from a single location
- ✓ greater author control over appearance of text & its placement on the page
- ✓ reduced clutter of multiple opening & closing tags on individual text elements
- ✓ simplified modification of page design through style editing
- ✓ eliminating the need for clumsy HTML workarounds to achieve basic layout effects
- ✓ great improvement of the design potential for HTML pages without introducing a large no. of new proprietary tags or compromising ability of other browser to effectively display the document text

XML – Nuts & Bolts:

- DTD
- XSD – eXtensible Schema Definition
- XSL – eXtensible Style Languages
- XML Linking Languages (XPath , Xlink & Xpointer)
- XML Namespaces

Advantages of Schemas:

- ✓ easier to validate the correctness of data
- ✓ easier to work with data from database
- ✓ easier to define data facets & data patterns
- ✓ easier to convert data between different data types

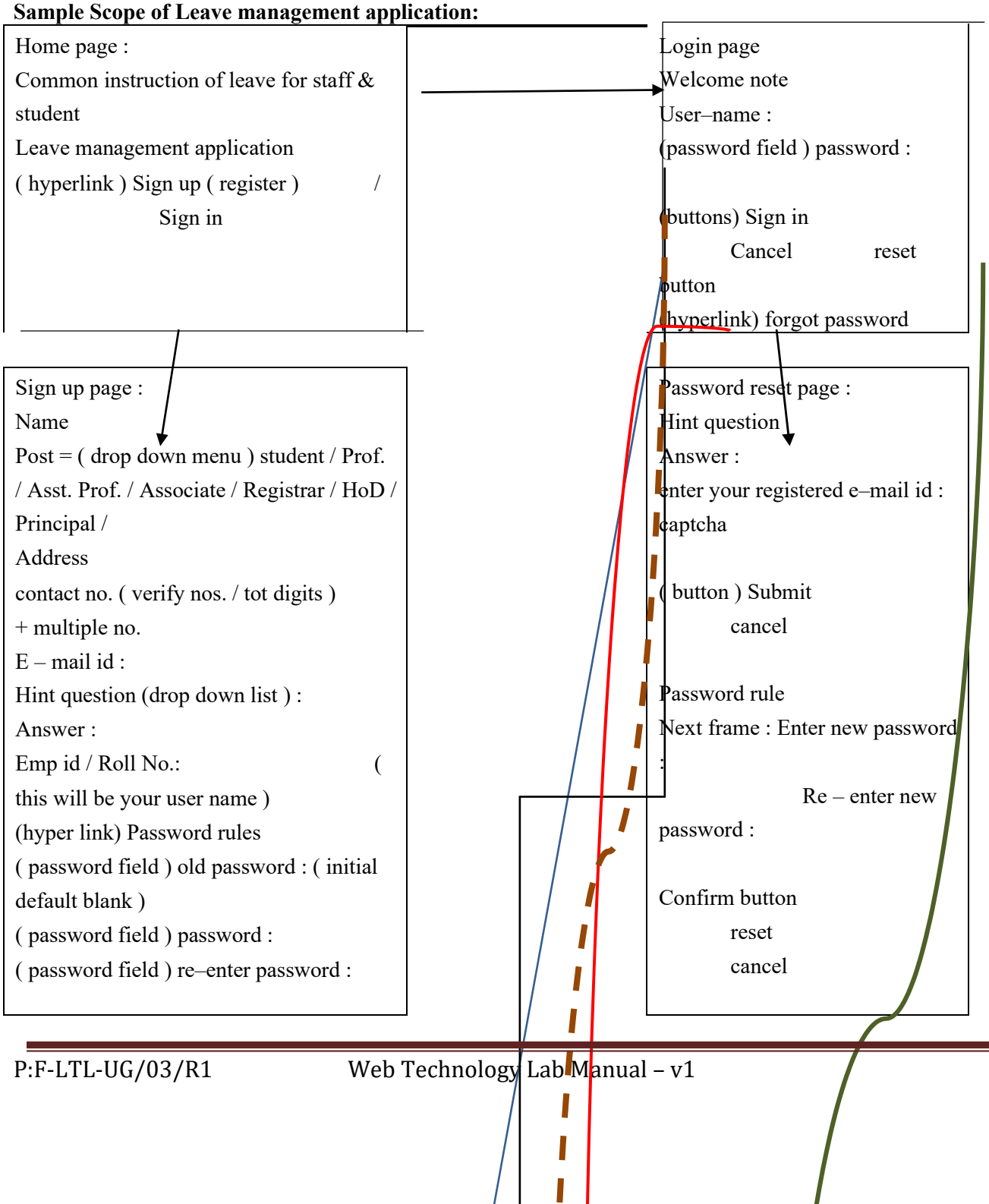
HTML vs XML

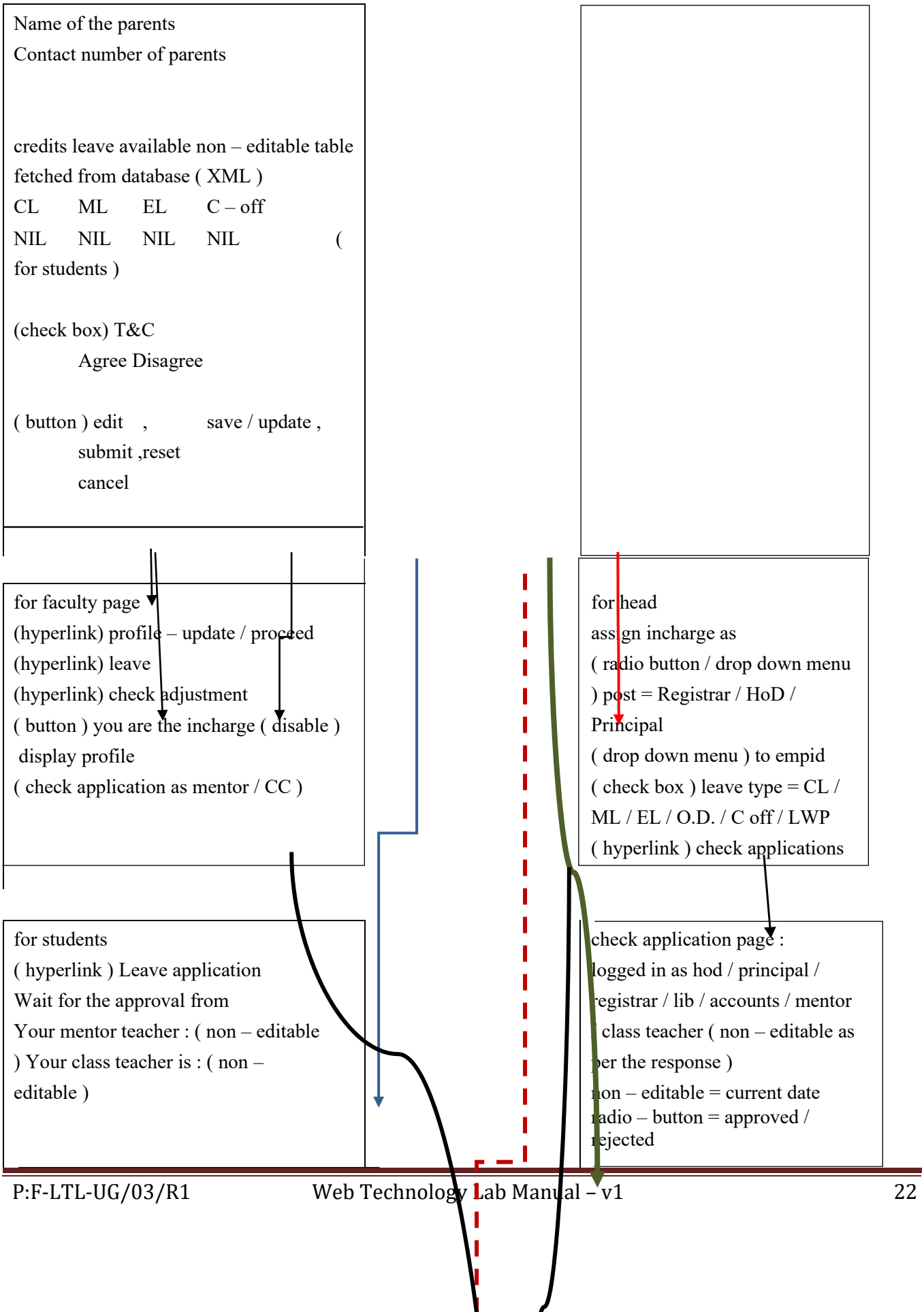
HyperText Markup Language	**Extensible Markup Language**
used to display data	used to transport & store data
it is markup language	provides framework to define markup languages

not case sensitive	case sensitive

it is presentation language	neither presentation nor programming lang.
predefined tags	custom tag
no strict rules	strict rules
Static	dynamic
does not preserve white spaces	preserve white spaces
list out the limitations	list out the limitations

Sample Scope of Leave management application:




Name of the parents
Contact number of parents
credits leave available non – editable table fetched from database (XML)
CL ML EL C – off
NIL NIL NIL NIL (
for students)
(check box) T&C
Agree Disagree
(button) edit , save / update ,
submit ,reset
cancel
for faculty page
(hyperlink) profile – update / proceed
(hyperlink) leave
(hyperlink) check adjustment
(button) you are the incharge (disable)
display profile
(check application as mentor / CC)
for head
assign incharge as
(radio button / drop down menu
) post = Registrar / HoD /
Principal
(drop down menu) to empid
(check box) leave type = CL /
ML / EL / O.D. / C off / LWP
(hyperlink) check applications
for students
(hyperlink) Leave application
Wait for the approval from
Your mentor teacher : (non – editable
) Your class teacher is : (non –
editable)
check application page :
logged in as hod / principal /
registrar / lib / accounts / mentor
class teacher (non – editable as
per the response)
non – editable = current date
radio – button = approved /
rejected

verified as (radio button)
original / in-charge for
if incharge , enable (radio button / drop down with single selection) = HoD / Principal / Registrar / Lib / accounts

(rich text) remark is any for approval / rejection

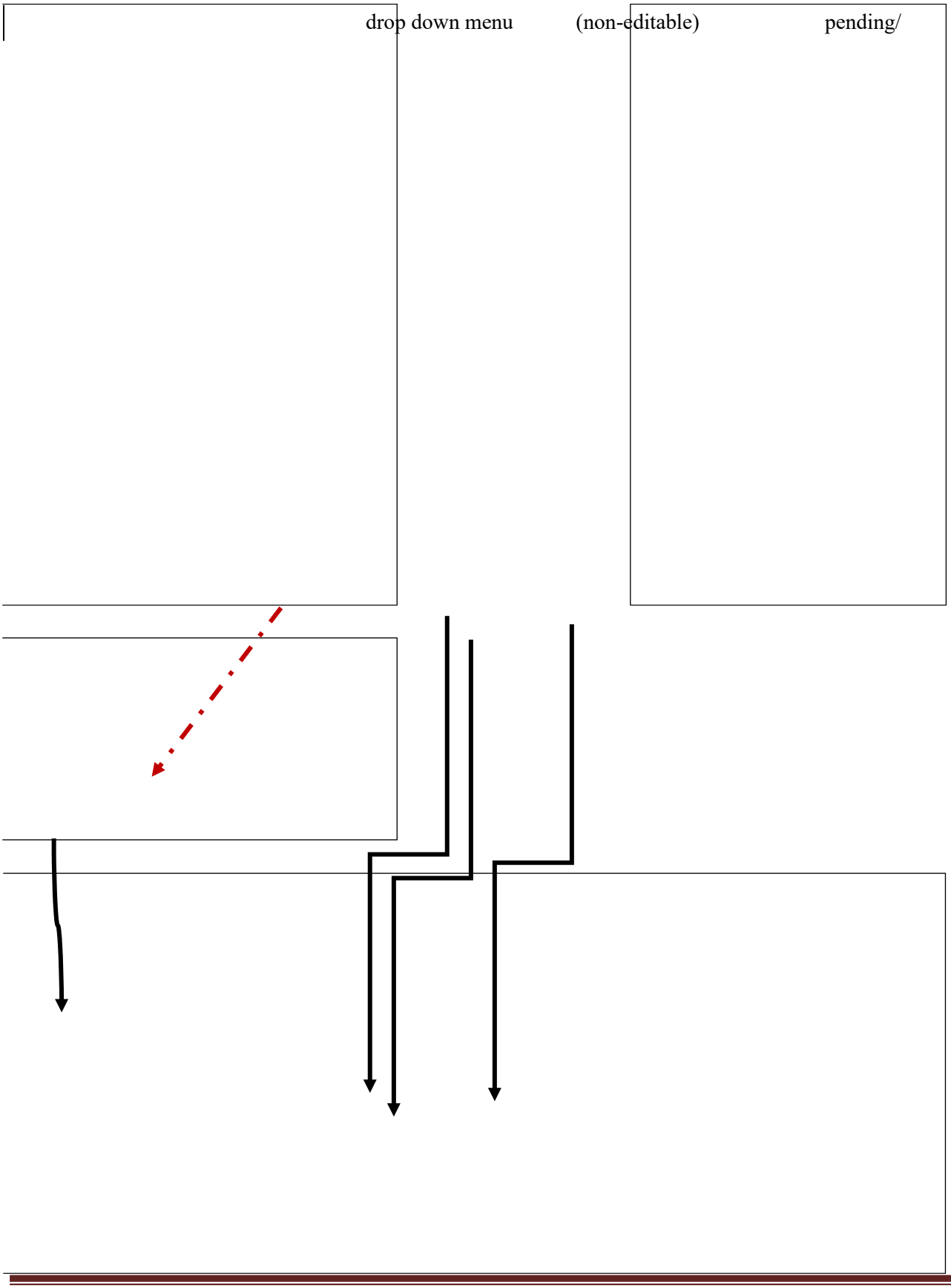
button save , reset
if rejected , on click of submit button reject response to applicant
if approved , on click of submit button send for further higher level of approval

for admin page
Overall management Assign roles & responsibilities
disable CL / EL / C off for all during high instruction (e.g. audit)
cancellation of leave / O.D.

Leave frame
(check box) leave type = CL / ML / EL / O.D. / C off / LWP
duration of leave = calendar (half day / fullday/ time)
tot. no. of days
reason / nature of work
leave address
on leave phone no.
balance leave : (non – editable text / label)

Adjustment table (enabled only when faculty)
Date Class Th/Pracadjusted with name send button status

drop down menu (non-editable) pending/



changes as
per response
from adjuster

upload proofs

Note :

- *on every final button pop – up for confirmation*
- *on every page there must be buttons of back , Home page & Logout & accordingly pages must be interlinked*
- *make use of CSS & XML DTD / Schema wherever applicable*

Algorithmic steps:

1. Finalize the scope of given list of web application
2. Task distribution
3. Design the respective application in lab note book & get it verified by respective lab subject teacher
4. Write the code snippet in the editor (HTML , CSS , XML) & visualize the same in browser
5. Testing

Testing:

Test id	Test case	Expected o/p	Actual o/p
1.	All web pages must be properly interlinked		
2.	Every page must have back button , home page button & logout with appropriate navigation		
3.	After the click of final button , pop – up confirmation is required		
4.	Web document is expected to be user –		

	friendly with formal design		
5.	Style overloading		
6.	Provide the alternate values to attributes of tags w.r.t. compatibility of browser . explorer		

References:

1. Web enabled commercial application development using HTML , DHTML , JavaScript , Perl CGI by Ivan Bayross
2. Musiciano C. Kennedy B., _ HTML & XHTML ‘ , 5th or higher edition , O‘Reilly / SPD Publications , ISBN B1 – 7366 – 517 – 1
3. Learning XML by Erik T. Ray , O‘reilly
4. Internet & World Wide Web , How to Programe , 3rd or higher edition , H. M. Deitel , P. J. Deitel , A. B. Goldberg , Pearson education
5. McKinnon A., McKinnon L., _ XML ‘ , Vikas Publishing House , 2004 , ISBN 981 – 254 – 299 – X https://www.w3schools.com/xml/

Oral Questions:

1. Compare HTML with XML.
2. What is the difference between form get and form post?
3. What is the importance of the HTML DOCTYPE?
4. What is web application?
5. What is markup language?
6. What is DOM document & XPath?
7. Can we have empty XML tag?
8. Can we replace HTML with XML?

Extra Assignments for practice:

1) Create a specimen of a corporate web page . Divide the browser screen into two frames . The frame on the left will be a menu consisting of hyperlinks . Clinking on any one of these links will lead to a new page , which must open in the target frame , which is on RHS .

2) Design following frame :

Content of Frame 1	Content of Frame 3

	Content of Frame 4
Content of Frame 2	
	Content of Frame 5

Solution :

```
<html>
<head> <title> Nested frames </title>              </head>
<frameset        cols = — 40% ‖ , * >     <framset        rows = — 50% ‖ , * >
        <frame          src = —frame1.html‖/>          <frame src = —frame2.html/>‖
</frameset>
<frameset        rows = — 20% ‖ , — 35% ‖ , *>
        <frame          src = —frame3.html‖/>          <frame          src = —frame4.html‖/>
        <frame          src = —frame5.html‖/>          </frameset>     </frameset>     </html>
```

3) Design following table which includes caption , border , cellpadding & cellspacing

NAME	MARKS		
	PowerBuilder	VisualBasic	Developer2000
Shilpa	21	45	30
Vaishali	26	30	40

Answer =

```
< HTML >
<HEAD> <TITLE>Working With Table</TITLE>       </HEAD>
<BODY BGCOLOR=LIGHTGREY>
<B>Specifing ROWSPAN and COLSPAN Attributes !</B>      <BR><BR><BR><BR>
<CENTER>     <TABLE BORDER=1 WIDTH=50% ALIGN=CENTER>
<TR>            <TH ROWSPAN=2>NAME              <TH COLSPAN=3>MARKS    </TR>
<TR>   <TH>PowerBuilder              <TH>VisualBasic              <TH>Developer2000    </TR>
<TR ALIGN=CENTER>        <TD> Shilpa    <TD> 21        <TD> 45        <TD> 30        </TR>
<TR ALIGN=CENTER>        <TD> Vaishali <TD> 26        <TD> 30        <TD> 40 </TR>
<CAPTION ALIGN=bottom><B><BR>Mark Sheet</B></CAPTION>          </TABLE>
</CENTER>    </BODY>       </HTML>
```

4) Create a document with two links to an external document . The first link should lead to the beginning of the external document . The second link should lead to a particular section in the external document . In the external document specify a link that will lead to a particular section within it .

5) XML design

6) Design a Web page for CYBERSHOP INC, using style sheets with the following Specifications : i. Define a style class _.Maxx‘ with the attributes viz; font – size , color , font – weight & font – family i. Use the defined Style class wherever the text _ CYBERSHOP INC ‘ appears on the web document i. Use unordered listing giving the list of services offered by CYBERSHOP INC . Define three segments using <DIV> … </DIV> tags with background colors Blue , Green & Goldenrod positioned accordingly with the some text .	<html> <head> <title>Cybershop Inc</title> <style type="text/css"> ul{List-style:square} .MAXX{Font-Size:26pt;Color:green;Font-weight:BOLD;Font-family:CURSIVE} </style> <body bgcolor="pink"> <P CLASS="MAXX">CYBERSHOP INC</P> Chatting Printing Hacking class Ebusiness <b> Chat Print Hacking class <b> </body>

ASSIGNMENT NO: 3

TITLE	***Employee database using XML schema***
PROBLEM STATEMENT /DEFINITION	1. Design the XML document to store the information of the employees of any business organization and demonstrate the use of: 2. a) DTD 3. b) XML Schema 4. And display the content in (e.g., tabular format) by using CSS/XSL.
OBJECTIVE	● To understand in depth working of XML and DTD ● To use CSS and XSL for providing validations, animation and effects to any web application.
S/W PACKAGES AND HARDWARE APPARATUS USED	Operating System open source Fedora 20 Networked computer with internet access Editor : IDE : Netbeans 8.1 Web browser Mozilla Firefox, Google Chrome
REFERENCES	https://www.w3schools.com/xml/xml_schema.asp https://www.geeksforgeeks.org/displaying-xml-using-css/
STEPS	Refer to steps below
INSTRUCTIONS FOR WRITING JOURNAL	● Title ● Problem Statement ● Theory ● Design part ● Troubleshooting (if any) ● Conclusion ● References

XML

An Introduction to XML

Let‘s see what‘s so special about XML, what we can achieve with it and which other technologies play well with it.

What is XML?

XML is a software- and hardware-independent tool for storing and transporting data.

- XML stands for eXtensible Markup Language
- markup language much like HTML
- designed to store and transport data
- designed to be self-descriptive

Why XML?

It is a dynamic markup language. It is used to transform data from one form to another form.

An XML file can be displayed using two ways. These are as follows :-

1. Cascading Style Sheet
2. Extensible Stylesheet Language Transformation
   Displaying XML file using CSS :
   CSS can be used to display the contents of the XML document in a clear and precise manner. It gives the design and style to whole XML document.
3. **Basic steps in defining a CSS style sheet for XML :**
   For defining the style rules for the XML document, the following things shoulde be done :-
   a. Define the style rules for the text elements such as font-size, color, font-weight, etc.
   b. Define each element either as a block, inline or list element, using the display property of CSS.
   c. Identify the titles and bold them.
4. **Linking XML with CSS :**
   In order to display the XML file using CSS, link XML file with CSS. Below is the syntax for linking the XML file with CSS:

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

Example 1.

In this example, the XML file is created that contains the information about five books and displaying the XML file using CSS.

- **XML file :**

Creating employees.xml as :-

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/css" href="Rule.css"?>
<employees>
   <heading>Welcome To PICT </heading>
   <employee>
      <title>department -: Web Programming</title>
      <author>staff -: ABC</author>
       <age> age -: 30</age>
   </employee>
   <employee>
      <title>Title -: Internet world-wide-web</title>
      <author>staff -: XYZ</author>
      <age>age -: 40</age>
   </employee>

</employees>
```

- In the above example, employees.xml is linked with Rule.css which contains the corresponding style sheet rules.
- **CSS FILE :**
  Creating Rule.css as:-

```
employees {
    color: white;
    background-color : gray;
    width: 100%;
}
 heading {
    color: green;
    font-size : 40px;
    background-color : powderblue;
}
 heading, title, author, age {
    display : block;
}
 title {
    font-size : 25px;
    font-weight : bold;
}
```

Advantages of displaying XML using CSS:

1. CSS is used in XML or HTML to decorate the pages.

2. CSS is used for interactive interface, so it is understandable by user.
3. CSS enable multiple pages to share formatting, and reduce complexity and repetition in the structural content. So page loader is faster.

Disadvantages of displaying XML using CSS :

1. Using CSS, no transformation can be applied to the XML documents.
2. CSS uses different dimensions with different browsers. So the programmer has to run the code in different browser and test its compatibility to post it live.
3. CSS have different level of versions, so it is confusing for the browser and user.

DTD

An XML document with correct syntax is called "Well Formed".

An XML document validated against a DTD is both "Well Formed" and "Valid".

What is a DTD?

DTD stands for Document Type Definition.

A DTD defines the structure and the legal elements and attributes of an XML document.

Valid XML Documents

A "Valid" XML document is "Well Formed", as well as it conforms to the rules of a DTD:

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE note SYSTEM "Note.dtd">

<note>

<to>Tove</to>

<from>Jani</from>

<heading>Reminder</heading>

<body>Don't forget me this weekend!</body>

</note>

the DOCTYPE declaration above contains a reference to a DTD file. The content of the DTD file is shown and explained below.

XML DTD

The purpose of a DTD is to define the structure and the legal elements and attributes of an XML document:

<!DOCTYPE note

[

<!ELEMENT note (to,from,heading,body)>

<!ELEMENT to (#PCDATA)>

<!ELEMENT from (#PCDATA)>

<!ELEMENT heading (#PCDATA)>

<!ELEMENT body (#PCDATA)>

]>

The DTD above is interpreted like this:

- !DOCTYPE note - Defines that the root element of the document is note
- !ELEMENT note - Defines that the note element must contain the elements: "to, from, heading, body"
- !ELEMENT to - Defines the to element to be of type "#PCDATA"
- !ELEMENT from - Defines the from element to be of type "#PCDATA"
- !ELEMENT heading - Defines the heading element to be of type "#PCDATA"
- !ELEMENT body - Defines the body element to be of type "#PCDATA"

Tip: #PCDATA means parseable character data.

When to Use a DTD?

With a DTD, independent groups of people can agree to use a standard DTD for interchanging data.

With a DTD, you can verify that the data you receive from the outside world is valid.

You can also use a DTD to verify your own data.

When NOT to Use a DTD?

XML does not require a DTD.

When you are experimenting with XML, or when you are working with small XML files, creating DTDs may be a waste of time.

If you develop applications, wait until the specification is stable before you add a DTD. Otherwise, your software might stop working because of validation errors.

XML SCHEMA

An XML Schema describes the structure of an XML document, just like a DTD.

An XML document with correct syntax is called "Well Formed".

An XML document validated against an XML Schema is both "Well Formed" and "Valid".

XML Schema

XML Schema is an XML-based alternative to DTD:

```
<xs:element name="note">

<xs:complexType>
  <xs:sequence>
    <xs:element name="to" type="xs:string"/>
    <xs:element name="from" type="xs:string"/>
    <xs:element name="heading" type="xs:string"/>
    <xs:element name="body" type="xs:string"/>
  </xs:sequence>
</xs:complexType>

</xs:element>
```

The Schema above is interpreted like this:

- <xs:element name="note"> defines the element called "note"
- <xs:complexType> the "note" element is a complex type
- <xs:sequence> the complex type is a sequence of elements
- <xs:element name="to" type="xs:string"> the element "to" is of type string (text)
- <xs:element name="from" type="xs:string"> the element "from" is of type string
- <xs:element name="heading" type="xs:string"> the element "heading" is of type string
- <xs:element name="body" type="xs:string"> the element "body" is of type string

XML Schemas are More Powerful than DTD

- XML Schemas are written in XML
- XML Schemas are extensible to additions
- XML Schemas support data types
- XML Schemas support namespaces

Why Use an XML Schema?

With XML Schema, your XML files can carry a description of its own format.

With XML Schema, independent groups of people can agree on a standard for interchanging data.

With XML Schema, you can verify data.

XML Schemas Support Data Types

One of the greatest strengths of XML Schemas is the support for data types:

- It is easier to describe document content
- It is easier to define restrictions on data
- It is easier to validate the correctness of data
- It is easier to convert data between different data types

XML Schemas use XML Syntax

Another great strength about XML Schemas is that they are written in XML:

- You don't have to learn a new language
- You can use your XML editor to edit your Schema files
- You can use your XML parser to parse your Schema files
- You can manipulate your Schemas with the XML DOM
- You can transform your Schemas with XSLT

References:

https://www.w3schools.com/xml/xml_schema.asp
https://www.geeksforgeeks.org/displaying-xml-using-css/

Oral Questions:

1. What is xml?
2. What is css?
3. What is dtd?
4. How xml and dtd are different?
5. Is xml a W3C standard?
6. Which is the starting point of code execution in xml?

ASSIGNMENT NO: 4

TITLE	Add dynamic web application features to previously selected application using java script,html and css .
PROBLEM STATEMENT /DEFINITION	Implement an application in Java Script using following: a) Design UI of application using HTML, CSS etc. b) Include Java script validation c) Use of prompt and alert window using Java Script e.g., Design and implement a simple calculator using Java Script for operations like addition, multiplication, subtraction, division, square of number etc. a) Design calculator interface like text field for input and output, buttons for numbers and operators etc. b) Validate input values c) Prompt/alerts for invalid values etc
OBJECTIVE	● To understand java script applications ● To explore the usage of html, css in java script ● To understand form validation using java script
S/W PACKAGES AND HARDWARE APPARATUS USED	Operating System open source Fedora 20 Networked computer with internet access Editor : IDE : Netbeans 8.1 Web browser Mozilla Firefox, Google Chrome
REFERENCES	1. https://www.javatpoint.com/javascript-calculator 2.https://codeburst.io/making-a-calculator-with-basic-html-css-and-javascript-part-1-1e4288f0bea1 3. https://www.geeksforgeeks.org/design-a-tip-calculator-using-html-css-and-javascript/
STEPS	Refer to steps below
INSTRUCTIONS FOR WRITING JOURNAL	● Title ● Problem Statement ● Theory ● Architecture diagrams (design part) ● Troubleshooting (if any) ● Conclusion ● References

Theory:

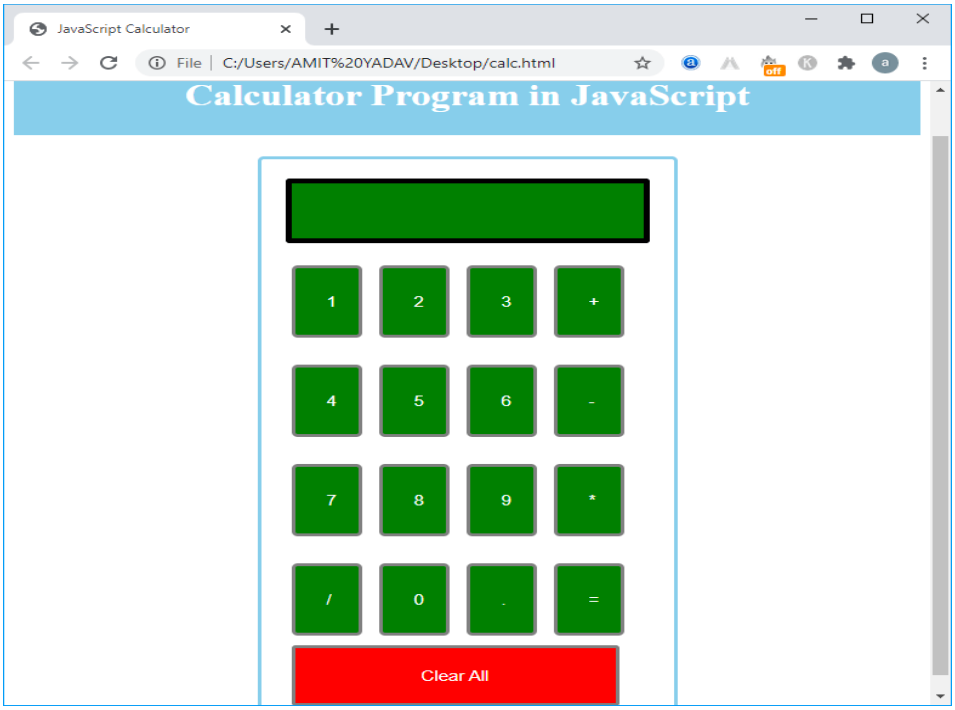
JavaScript Calculator

As we know, the Calculator is a portable device used in our daily life to perform various mathematical functions such as addition, subtraction, multiplication, division, root, etc. However, we have scientific or sophisticated calculators used to solve complex tasks such as trigonometry functions, degrees, exponential operators, log functions, hyperbolic functions, square root, and so on. In this topic, we will create a calculator program in JavaScript.

```
<!-- Create a simple Program to build the Calculator in JavaScript using with HTML and CSS web languages. -->
<!DOCTYPE html>
<html lang = "en">
<head>
<title> JavaScript Calculator </title>

<style>
h1 {
    text-align: center;
    padding: 23px;
    background-color: skyblue;
    color: white;
    }

#clear{
width: 270px;
border: 3px solid gray;
    border-radius: 3px;
    padding: 20px;
    background-color: red;
}

.formstyle
{
width: 300px;
height: 530px;
margin: auto;
border: 3px solid skyblue;
border-radius: 5px;
padding: 20px;
}


input
```

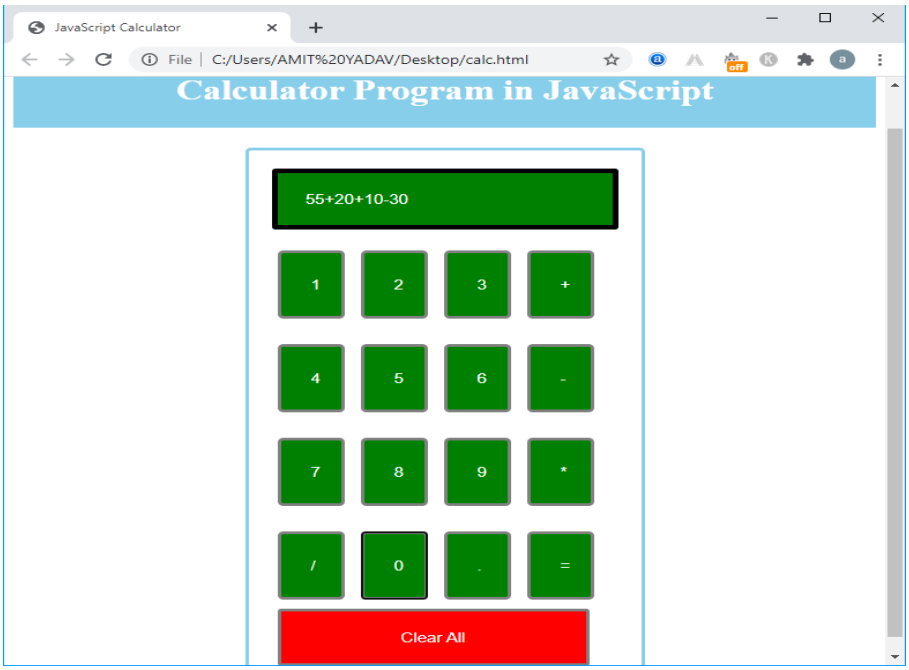
```
{
width: 20px;
background-color: green;
color: white;
border: 3px solid gray;
    border-radius: 5px;
    padding: 26px;
    margin: 5px;
    font-size: 15px;
}


#calc{
width: 250px;
border: 5px solid black;
    border-radius: 3px;
    padding: 20px;
    margin: auto;
}

</style>

</head>
<body>
<h1> Calculator Program in JavaScript </h1>
<div class= "formstyle">
<form name = "form1">

    <!-- This input box shows the button pressed by the user in calculator. -->
  <input id = "calc" type ="text" name = "answer"> <br> <br>
  <!-- Display the calculator button on the screen. -->
  <!-- onclick() function display the number prsses by the user. -->
  <input type = "button" value = "1" onclick = "form1.answer.value += '1' ">
  <input type = "button" value = "2" onclick = "form1.answer.value += '2' ">
  <input type = "button" value = "3" onclick = "form1.answer.value += '3' ">
   <input type = "button" value = "+" onclick = "form1.answer.value += '+' ">
  <br> <br>

  <input type = "button" value = "4" onclick = "form1.answer.value += '4' ">
  <input type = "button" value = "5" onclick = "form1.answer.value += '5' ">
  <input type = "button" value = "6" onclick = "form1.answer.value += '6' ">
  <input type = "button" value = "-" onclick = "form1.answer.value += '-' ">
  <br> <br>

  <input type = "button" value = "7" onclick = "form1.answer.value += '7' ">
  <input type = "button" value = "8" onclick = "form1.answer.value += '8' ">
```

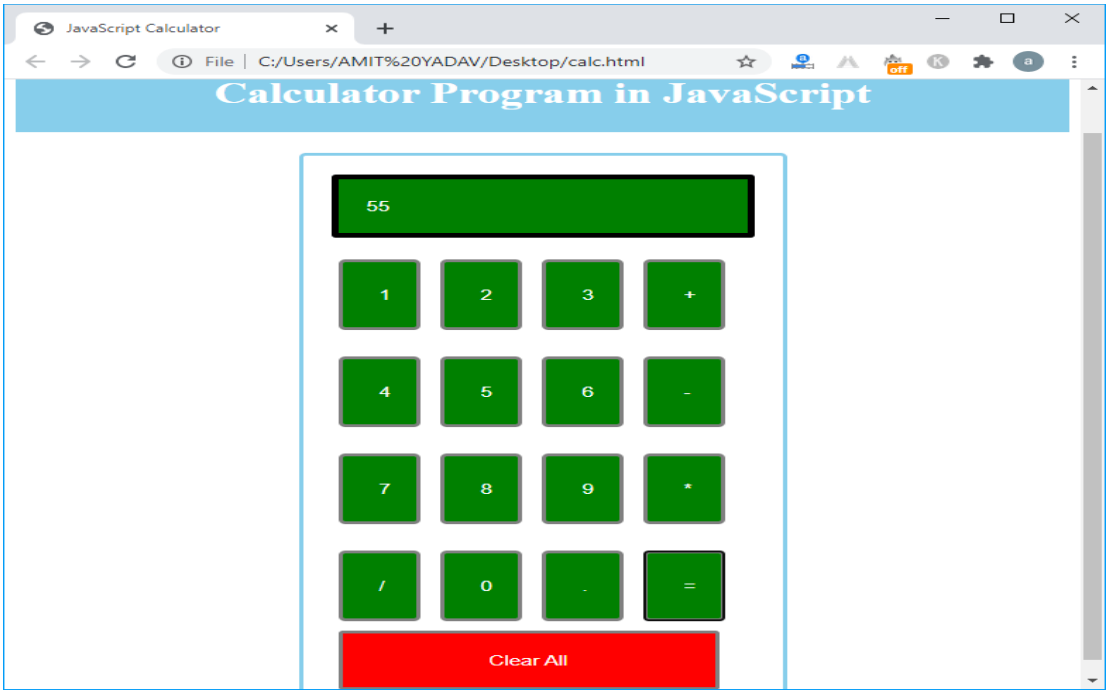
```
<input type = "button" value = "9" onclick = "form1.answer.value += '9' ">
<input type = "button" value = "*" onclick = "form1.answer.value += '*' ">
<br> <br>


<input type = "button" value = "/" onclick = "form1.answer.value += '/' ">
<input type = "button" value = "0" onclick = "form1.answer.value += '0' ">
  <input type = "button" value = "." onclick = "form1.answer.value += '.' ">
  <!-- When we click on the '=' button, the onclick() shows the sum results on the calculator screen. -
->
<input type = "button" value = "=" onclick = "form1.answer.value = eval(form1.answer.value) ">
<br>
<!-- Display the Cancel button and erase all data entered by the user. -->
<input type = "button" value = "Clear All" onclick = "form1.answer.value = ' ' " id= "clear" >
<br>

</form>
</div>
</body>
        </html>
```



Here we have created a Calculator program using the JavaScript language, including HTML and CSS web programming. In this Calculator, we can perform basic operations like addition, multiplication, subtraction, and division.

Now we perform some operation in the JavaScript program, as shown below.





References:

1. https://www.javatpoint.com/javascript-calculator

2.https://codeburst.io/making-a-calculator-with-basic-html-css-and-javascript-part-1-1e4288f0bea1
3. https://www.geeksforgeeks.org/design-a-tip-calculator-using-html-css-and-javascript/

Oral Questions:

1. What is the use of html?
2. How to include a file to html page?
3. What is difference between include and require?
4. Require_once(), require(), include(), what is difference between them?
5. How to declare an array in javascript?
6. What is use of in_array() in javascript?

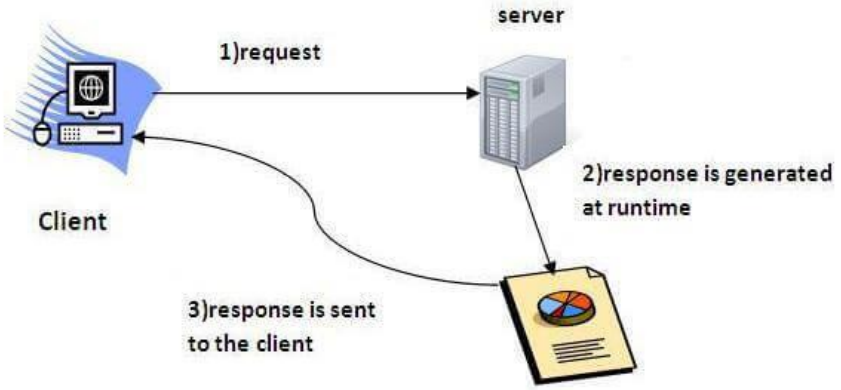
ASSIGNMENT NO: 5

TITLE	**Implement the sample program demonstrating the use of Servlet.**
PROBLEM STATEMENT /DEFINITION	Develop a web application using Servlets e.g. Create a database table ebookshop (book_id, book_title, book_author, book_price, quantity) using database like Oracle/MySQL etc. and display (use SQL select query) the table content using servlet.
OBJECTIVE	● To understand server side programming using Servlets ● To explore the usage of Servlets in web applications
S/W PACKAGES AND HARDWARE APPARATUS USED	Operating System open source Fedora 20 Networked computer with internet access Editor : IDE : Netbeans 8.1/ Eclipse for web development Web browser Mozilla Firefox, Google Chrome
REFERENCES	1. https://www.javatpoint.com 2. https://dzone.com/tutorials
STEPS	Refer to steps below
INSTRUCTIONS FOR WRITING JOURNAL	● Title ● Problem Statement ● Theory ● Architecture diagrams (design part) ● Test Cases(if any) ● Conclusion ● References

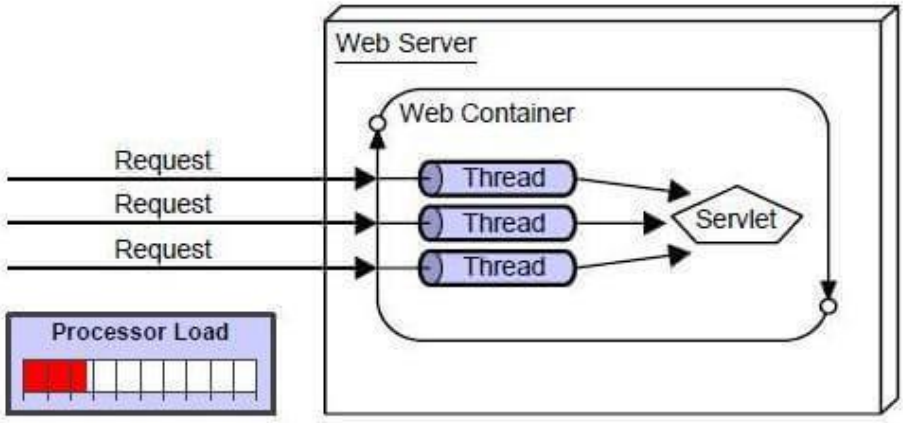
Theory:

What is a Servlet?

- Servlet is a technology which is used to create a web application.
- Servlet is an API that provides many interfaces and classes including documentation.
- Servlet is an interface that must be implemented for creating any Servlet.
- Servlet is a class that extends the capabilities of the servers and responds to the incoming requests. It can respond to any requests.
- Servlet is a web component that is deployed on the server to create a dynamic web page.



Advantages of Servlet:



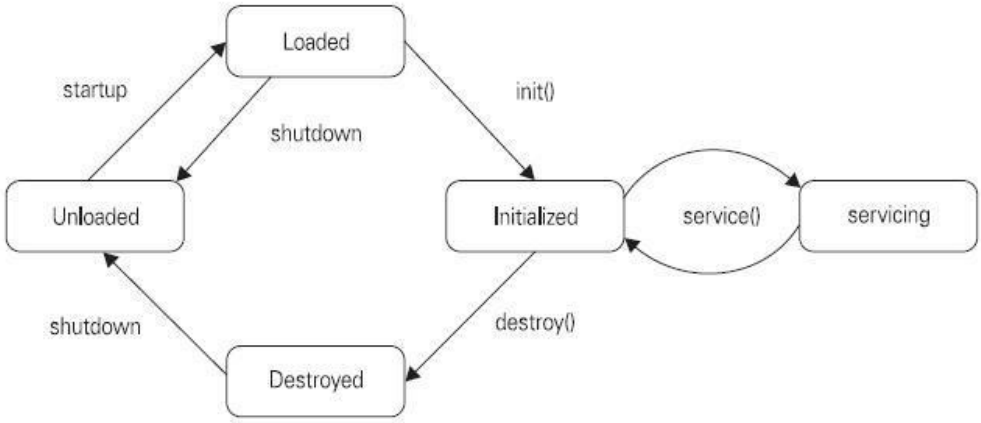
There are many advantages of Servlet over CGI. The web container creates threads for handling the multiple requests to the Servlet. Threads have many benefits over the Processes such as they share a common memory area, lightweight, cost of communication between the threads are low. The advantages of Servlet are as follows:

1. Better performance: because it creates a thread for each request, not process.
2. Portability: because it uses Java language.
3. Robust: JVM manages Servlets, so we don't need to worry about the memory leak, garbage collection, etc.
4. Secure: because it uses java language.

Life Cycle of a Servlet (Servlet Life Cycle)

The web container maintains the life cycle of a servlet instance. The life cycle of the servlet has following phases:

1. Servlet class is loaded
2. Servlet instance is created
3. init method is invoked
4. service method is invoked
5. destroy method is invoked



Servlet state transition diagram.

1) Servlet class is loaded

The classloader is responsible to load the servlet class. The servlet class is loaded when the first request for the servlet is received by the web container.

2) Servlet instance is created

The web container creates the instance of a servlet after loading the servlet class. The servlet instance is created only once in the servlet life cycle

3) init method is invoked

The web container calls the init method only once after creating the servlet instance. The init method is used to initialize the servlet. It is the life cycle method of the javax.servlet.Servlet interface.

4) service method is invoked

The web container calls the service method each time when request for the servlet is received. If servlet is not initialized, it follows the first three steps as described above then calls the service method. If servlet is initialized, it calls the service method. Notice that servlet is initialized only once.

5) destroy method is invoked

The web container calls the destroy method before removing the servlet instance from the service. It gives the servlet an opportunity to clean up any resource for example memory, thread etc.

Steps to create a servlet example:

The servlet example can be created by three ways:

1. By implementing Servlet interface,
2. By inheriting GenericServlet class, (or)
3. By inheriting HttpServlet class

The HttpServlet class is widely used to create the servlet because it provides methods to handle http requests such as doGet(), doPost, doHead() etc. In this example we are going to create a servlet that extends the HttpServlet class. In this example, we are inheriting the HttpServlet class and providing the implementation of the doGet() method. Notice that get request is the default request.

DemoServlet.java:

```
import javax.servlet.http.*;

import javax.servlet.*;

import java.io.*;

public class DemoServlet extends HttpServlet{

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

{

res.setContentType("text/html");//setting the content type

PrintWriter pw=res.getWriter();//get the stream to write the data

 //writing html in the stream

pw.println("<html><body>");

pw.println("Welcome to servlet");

pw.println("</body></html>");

pw.close();//closing the stream

}

}
```

Conclusion: Hence, we have studied and implemented Servlets for creating web applications.

References:

https://www.javatpoint.com/steps-to-create-a-servlet-using-tomcat-server

ASSIGNMENT NO: 6

TITLE	**Implement the program demonstrating the use of JSP.**
PROBLEM STATEMENT /DEFINITION	Develop a web application using JSP e.g. Create a database table students_info (stud_id, stud_name, class, division, city) using database like Oracle/MySQL etc. and display (use SQL select query) the table content using JSP.
OBJECTIVE	● Understand the basic functionalities of JSP ● Analyze the difference between JSP and Servlets for server side programming
S/W PACKAGES AND HARDWARE APPARATUS USED	Operating System open source Fedora 20 Networked computer with internet access Editor : IDE : Netbeans 8.1/ Eclipse for web development Web browser Mozilla Firefox, Google Chrome
REFERENCES	1. https://www.javatpoint.com 2. https://dzone.com/tutorials
STEPS	Refer to steps below
INSTRUCTIONS FOR WRITING JOURNAL	● Title ● Problem Statement ● Theory ● Architecture diagrams (design part) ● Test Cases(if any) ● Conclusion ● References

Theory:

Java Server Pages (JSP):

It is a server side programming technology that is used to create dynamic web-based applications. JSP have right to use the complete Java APIs, including the JDBC API to access the databases.

It is a technology that helps software developers to create dynamic web pages based on HTML, XML and other document types. It was released in 1999 by Sun Microsystems. It is just like a PHP and ASP, but it uses the Java programming language.

A JSP element is a type of java servlet that is designed to accomplish the role of a user interface for a java web application. Web developers write JSPs as text files that combine HTML or XHTML code, XML elements, and rooted JSP actions and commands.

Using JSP, you can collect input from users through web page forms, current records from a database or another source and create web pages dynamically.JSP tags can be used for different purposes, such as retrieving information from a database or registering user preferences, accessing JavaBeans components, passing control between pages, and sharing information between requests, pages etc.

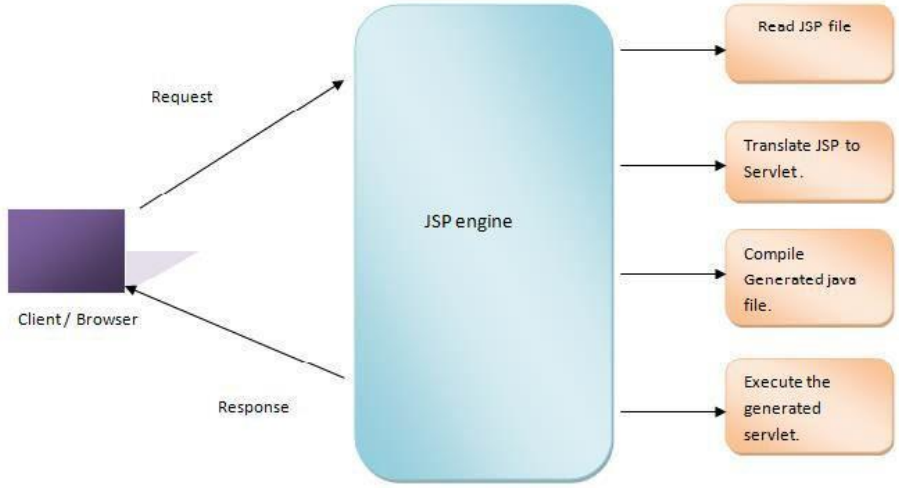
Why we need JSP?

JSP is used for the design of dynamic web page and servlet is used to code the logic that is present i.e. in the MVC (Model-View-Controller) architecture, the servlet is the controller and the JSP is the view.

Architecture of JSP

1. The request / response part of a JSP is defined in below architecture

2. The client initiated request for a JSP file using browser
3. Webs server (i.e, JSP Engine) invokes the JSP file and interpret the JSP file produce a java code. The created java code will be a Servlet.
4. Once Servlet is created, JSP engine compiles the servlet. Compilation errors will be detected in this phase.
5. Now servlet class is loaded by the container and executes it.
6. Engine sends the response back to the client.



Architecture of JSP

Syntax of JSP:

JSP declarations is used to declare variables and methods as shown below,
<% text %>

Following is the simple and first example for JSP:

```
//Hello.jsp
<html>
<head>
<title> JSP File</title>
</head>
<body>
<%
out.println("Welcome to JSP Class");
%>
</body>
</html>
```

Output:
Welcome to JSP Class

List of JSP Tags:

JSP tags are an essential part of Java Server Pages, a server-side technology. Tags in JSP create a container for Java code, insulating and providing separation of dynamic content from static design elements in your site. Although many resemble those used in Hyper Text Markup Language – HTML - JSP tags are not part of a scripting language. Instead, tags in JSP are just one element in a technology focused on generating dynamic Web content. JSP tags determine how the code within them will behave. A list of common JSP tags can be a good reference to ensure you choose and use tags correctly.

JSP Declaration Tags

Declaration tags in JSP function as identification containers for the functions, methods and variables in JSP pages. Because these tags identify rather than generate output, you will most often find declaration tags working in combination with expression or scriptlet tags. Syntax options include the simple "<%! jsp declaration %>" and the XML alternative "jsp:declaration…."

JSP Expression Tags

Expression tags signal JSP to convert a Java statement – also called an expression – into a string and display the output. Syntax options include the simple "<%= Java statement %>" and the XML alternative "jsp:expression…."

JSP Directive Tags

Directives – or message tags – are instructional tags that contain two parts: type and attribute. Type can be "page," which gives page-specific processing directions, "Include," which provides specific file names or "Tag Library," which identifies the tag library you want to use on the current page. Syntax options include the simple "<%@ dir-type dir-attr %> and the XML alternative "<jsp:directive.dir_type dir_attr />."

JSP Scriptlet Tags

Scriptlet tags allow you to embed any valid Java source code in JSP server pages. The code within the tags executes in consecutive order on the server side and is available for client access through a Web browser. Syntax options include the simple "<% Java code %> and the XML alternative "jsp:scriptlet Java code ."

JSP Flow Control Tags

Flow control tags function the same as – and are an alternative to – scriptlets. Unlike scriptlets, however, flow control tags allow you to control the order in which statements run. The conditional tags "if" and "choose" and the iterator tags "forEach" and "forTokens" are all examples of JSP flow control tags. The syntax framework for each includes <c:tag_type tag_attribute> Java code . For example, when you use an "if" statement, the correct syntax is <c: if test= "$test parameters"> Java code </c: if > where test= is the type and "$test parameters" identifies tag attributes.

JSP Action Tags

Action tags can tell JSP to transfer control between pages, set or get properties, facilitate browser independent support for Java applets and make it possible to use server-side JavaBeans. Of the many available action tags, the most common are the include directive, the forward tag, which transfers control to a dynamic or static URL and the useBean tag, which allows a JSP to create or receive an instance of a reusable software component that works with Java called a JavaBean. The only syntax option for an action tag is the XML version: "jsp:useBean Java body ."

JSP Comment Tags

Comment tags are for "information only" and do not appear on JSP pages. Use them for clarification or documentation and view them by right clicking on a Web page and accessing the "view source" option. The only syntax option for a comment tag is the simple version: <%/ *comments go here* />.

Conclusion: Hence, we have performed the dynamic web application using JSP.

References:

https://www.techwalla.com/articles/list-of-jsp-tags

Questions:

1. What is JSP?
2. What is Servlet?
3. What is the purpose of MySQL?
4. What is database?
5. What is the syntax of JSP?
6. How do we connect JSP file to database?

ASSIGNMENT NO: 7

TITLE	**Build a dynamic web application using PHP and MySQL.**
PROBLEM STATEMENT /DEFINITION	Develop a web application using PHP a. Create database tables in MySQL and create connection with PHP. b. Create the add, update, delete and retrieve functions in the PHP web app interacting with MySQL database
OBJECTIVE	● To understand the principles and methodologies of PHP web based applications development process
S/W PACKAGES AND HARDWARE APPARATUS USED	Operating System open source Fedora 20 Networked computer with internet access Editor : IDE : Netbeans 8.1/ Eclipse for web development XAMPP Server Web browser Mozilla Firefox, Google Chrome
REFERENCES	3. https://www.javatpoint.com 4. https://dzone.com/tutorials
STEPS	Refer to steps below
INSTRUCTIONS FOR WRITING JOURNAL	● Title ● Problem Statement ● Theory ● Architecture diagrams (design part) ● Test Cases(if any) ● Conclusion ● References

Theory:

PHP:

The PHP Hypertext Preprocessor (PHP) began as a little open source venture that advanced as an ever increasing number of individuals discovered how valuable it was. Rasmus Lerdorf released the principal form of PHP route in 1994. PHP is a recursive acronym for "PHP: Hypertext Preprocessor".

PHP is a server side scripting dialect that is installed in HTML. It is utilized to oversee dynamic substance, databases, session following, even form whole internet business locales. It is incorporated with various prevalent databases, including MySQL, PostgreSQL, Oracle, Sybase, Informix, and Microsoft SQL Server.

PHP is pleasingly zippy in its execution, particularly when gathered as an Apache module on the Unix side. The MySQL server, once begun, executes even extremely complex questions with colossal outcome sets in record-setting time.

PHP bolsters a substantial number of real conventions, for example, POP3, IMAP, and LDAP. PHP4 included help for Java and conveyed question designs (COM and CORBA), making n- level improvement a plausibility

out of the blue. PHP is excusing: PHP dialect tries to be as pardoning as would be prudent. PHP Syntax is C-Like.

PHP performs framework capacities, i.e. from documents on a framework it can make, open, read, compose, and close them. PHP can deal with frames, i.e. accumulate information from records, spare information to a document; through email you can send information, return information to the client.

You include, erase, adjust components inside your database through PHP. Access treats factors and set treats. Utilizing PHP, you can confine clients to get to a few pages of your site. It can encode information.

Example:
"Hello World" Script in PHP

To get a feel for PHP, first start with simple PHP scripts. Since "Hello, World!" is an essential example, first we will create a friendly little "Hello, World!" script.
As mentioned earlier, PHP is embedded in HTML. That means that in amongst your normal HTML (or XHTML if you're cutting-edge) you'll have PHP statements like this –

```
<html>
<head>
<title> Hello World</title>
</head>
<body>
<?php echo (“Hello Php”); ?>
</body>
</html>
```

To create and run PHP Web pages three fundamental parts should be introduced on your PC framework.

Web Server – PHP will work with for all intents and purposes all Web Server programming, including Microsoft's Internet Information Server (IIS) however then regularly utilized is unreservedly accessible Apache Server. Download Apache for nothing here – https://httpd.apache.org/download.cgi
Database – PHP will work with for all intents and purposes all database programming, including Oracle and Sybase yet most regularly utilized is uninhibitedly accessible MySQL database. Download MySQL for nothing here – https://www.mysql.com/downloads/
PHP Parser – keeping in mind the end goal to process PHP content directions a parser must be introduced to create HTML yield that can be sent to the Web Browser. This instructional exercise will manage you how to introduce PHP parser on your PC.

Open a Connection to MySQL:

Before we can access data in the MySQL database, we need to be able to connect to the server using following code:

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";

// Create connection
$conn = new mysqli($servername, $username, $password);
// Check connection
if ($conn->connect_error) {
```

```
  die("Connection failed: " . $conn->connect_error);
}
echo "Connected successfully";
?>
```

Close the Connection:

```
$conn->close();
```

PHP MySQL Insert Data:

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
}

$sql = "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('John', 'Doe', 'john@example.com')";

if ($conn->query($sql) === TRUE) {
  echo "New record created successfully";
} else {
  echo "Error: " . $sql . "<br>" . $conn->error;
}

$conn->close();
?>
```

PHP MySQL Select Data:

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
}
```

```php
$sql = "SELECT id, firstname, lastname FROM MyGuests";
$result = $conn->query($sql);

if ($result->num_rows > 0) {
  // output data of each row
  while($row = $result->fetch_assoc()) {
        echo "id: " . $row["id"]. " - Name: " . $row["firstname"]. " " . $row["lastname"]. "<br>";
  }
} else {
  echo "0 results";
}
$conn->close();
?>
```

Conclusion: Hence, we have studied how to design and develop small web application using PHP script and MySQL as backend.

Reference: https://www.w3schools.com/php

Questions:

1. What is the use of "echo" in php?
2. How to include a file to a php page?
3. Differences between GET and POST methods ?
4. What is the use of 'print' in php?
5. What is the difference between Session and Cookie?
6. What are the different errors in PHP?
7. How to print current date and time?
8. What is the difference between sql and Mysql?
9. Why do we use GROUP BY and ORDER BY function in mysql?
10. What is JOIN in MySQL? What are the different types of join?

ASSIGNMENT NO: 7

TITLE	**Develop and deploy web application using AngularJS.**

PROBLEM STATEMENT /DEFINITION	Develop and deploy one of the following web applications: 1. Online pizza order application 2. Student information system for training & placement department 3. Leave management application 4. Blogging platform 5. Meeting room booking application 6. Exam cell automation application
OBJECTIVE	● To understand in depth working of AngularJS ● To use AngularJS to develop any web application.
S/W PACKAGES AND HARDWARE APPARATUS USED	Operating System: open source Fedora 20 Networked computer with Internet access Editor : IDE : Netbeans 8.1 Web browser: Mozilla Firefox, Google Chrome
REFERENCES	1. https://angular.io/ 2. https://angular.io/tutorial 3. https://www.w3schools.com/angular/angular_intro.asp 4. https://www.tutorialspoint.com/angularjs/index.htm 5. https://www.javatpoint.com/angularjs-tutorial
STEPS	Refer to steps below
INSTRUCTIONS FOR WRITING JOURNAL	● Title ● Problem Statement ● Theory ● Design part ● Troubleshooting (if any) ● Conclusion ● References

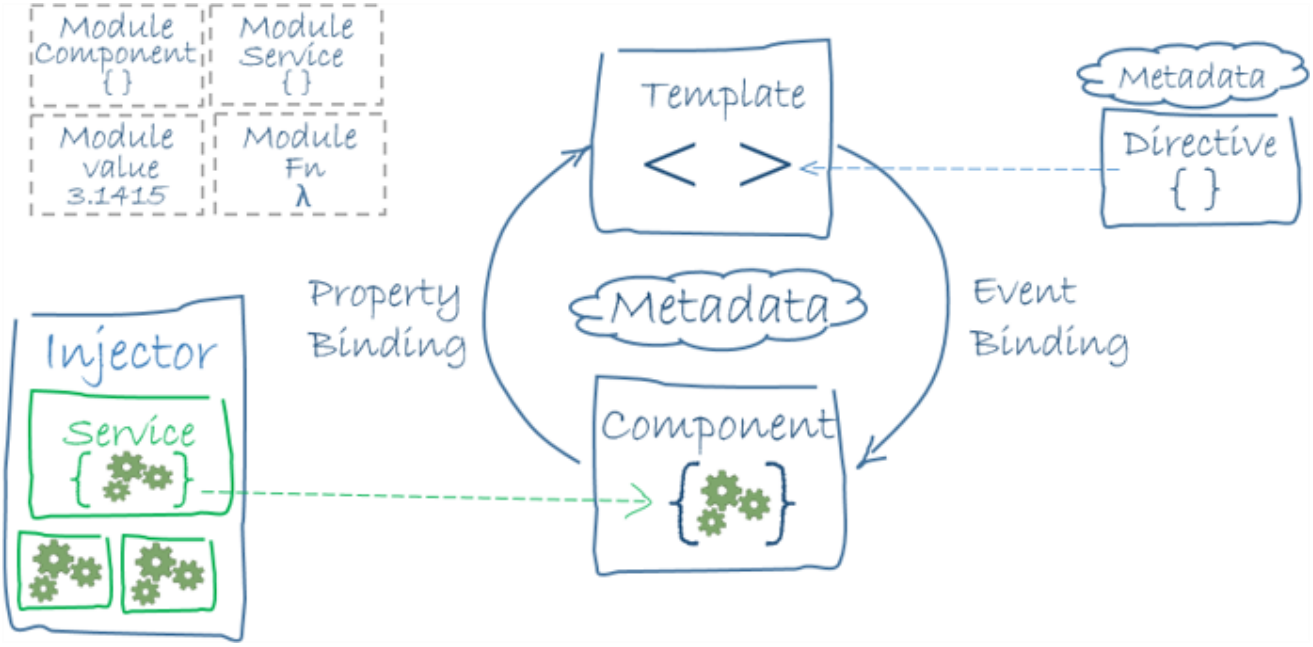
AngularJS

AngularJS (commonly referred to as "Angular.js" or "AngularJS 1.X") is a JavaScript-based open-source front-end web application framework mainly maintained by Google and by a community of individuals and corporations to address many of the challenges encountered in developing single-page applications. The JavaScript components complement Apache Cordova, the framework used for developing cross-platform mobile apps. It aims to simplify both the development and the testing of such applications by providing a framework for client-side model–view–controller (MVC) and model–view–viewmodel (MVVM) architectures, along with components commonly used in rich Internet applications.

Angular (commonly referred to as "Angular 4" or "Angular 2") is a TypeScript-based open-source front-end web application platform led by the Angular Team at Google and by a community of individuals and corporations. Angular is a complete rewrite from the same team that built AngularJS.

Architecture Overview

Angular is a framework for building client applications in HTML and either JavaScript or a language like TypeScript that compiles to JavaScript.
The framework consists of several libraries, some of them core and some optional.
You write Angular applications by composing HTML *templates* with Angularized markup, writing *component* classes to manage those templates, adding application logic in *services*, and boxing components and services in *modules*.
Then you launch the app by *bootstrapping* the *root module*. Angular takes over, presenting your application content in a browser and responding to user interactions according to the instructions you've provided.



Modules

Angular apps are modular and Angular has its own modularity system called *NgModules*. NgModules are a big deal. Every Angular app has at least one NgModule class, the *root module*, conventionally named AppModule. While the *root module* may be the only module in a small application, most apps have many more *feature modules*, each a cohesive block of code dedicated to an application domain, a workflow, or a closely related set of capabilities. An NgModule, whether a *root* or *feature*, is a class with an @NgModule decorator.

NgModule is a decorator function that takes a single metadata object whose properties describe the module. The most important properties are:

- declarations - the *view classes* that belong to this module. Angular has three kinds of view classes: components, directives, and pipes.
- exports - the subset of declarations that should be visible and usable in the component templates of other modules.
- imports - other modules whose exported classes are needed by component templates declared in *this* module.
- providers - creators of services that this module contributes to the global collection of services; they become accessible in all parts of the app.
- bootstrap - the main application view, called the *root component*, that hosts all other app views. Only the *root module* should set this bootstrap property.

Components

A *component* controls a patch of screen called a *view*.

For example, the following views are controlled by components:

- The app root with the navigation links.
- The list of heroes.
- The hero editor.

You define a component's application logic—what it does to support the view—inside a class. The class interacts with the view through an API of properties and methods.

```
export class HeroListComponent implements OnInit {
  heroes: Hero[];
  selectedHero: Hero;

  constructor(private service: HeroService) { }

  ngOnInit() {
    this.heroes = this.service.getHeroes();
  }

  selectHero(hero: Hero) { this.selectedHero = hero; }
}
```

Templates

You define a component's view with its companion **template**. A template is a form of HTML that tells Angular how to render the component.

```
<h2>Hero List</h2>

<p><i>Pick a hero from the list</i></p>
<ul>
  <li *ngFor="let hero of heroes" (click)="selectHero(hero)">
    {{hero.name}}
  </li>
</ul>

<app-hero-detail *ngIf="selectedHero" [hero]="selectedHero"></app-hero-detail>
```

Although this template uses typical HTML elements like <h2> and <p>, it also has some differences. Code like *ngFor, {{hero.name}}, (click), [hero], and <app-hero-detail> uses Angular's template syntax.
In the last line of the template, the <app-hero-detail> tag is a custom element that represents a new component, HeroDetailComponent.
The HeroDetailComponent is a *different* component than the HeroListComponent you've been reviewing. The HeroDetailComponent (code not shown) presents facts about a particular hero, the hero that the user selects from the list presented by the HeroListComponent. The HeroDetailComponent is a **child** of the HeroListComponent.

Metadata

Metadata tells Angular how to process a class. Looking back at the code for HeroListComponent, you can see that it's just a class. There is no evidence of a framework, no "Angular" in it at all. In fact, HeroListComponent really is *just a class*. It's not a component until you *tell Angular about it*. To tell Angular that HeroListComponent is a component, attach **metadata** to the class. In TypeScript, you attach metadata by using a **decorator**. Here's some metadata for HeroListComponent

```
@Component({
  selector:    'app-hero-list',
  templateUrl: './hero-list.component.html',
  providers:  [ HeroService ]
})
export class HeroListComponent implements OnInit {
/* . . . */
}
```

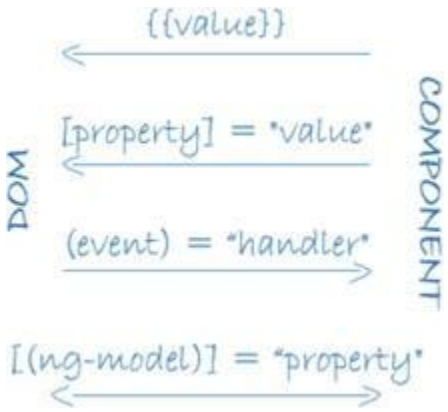
Here is the @Component decorator, which identifies the class immediately below it as a component class. The @Component decorator takes a required configuration object with the information Angular needs to create and present the component and its view.
Here are a few of the most useful @Component configuration options:

- selector: CSS selector that tells Angular to create and insert an instance of this component where it finds a <app-hero-list> tag in *parent* HTML. For example, if an app's HTML contains <app-hero-list></app-hero-list>, then Angular inserts an instance of the HeroListComponent view between those tags.
- templateUrl: module-relative address of this component's HTML template, shown above.
- providers: array of **dependency injection providers** for services that the component requires. This is one way to tell Angular that the component's constructor requires a HeroService so it can get the list of heroes to display.

The metadata in the @Component tells Angular where to get the major building blocks you specify for the component. The template, metadata, and component together describe a view. @Injectable, @Input, and @Output are a few of the more popular decorators.

Data binding

Without a framework, you would be responsible for pushing data values into the HTML controls and turning user responses into actions and value updates. Writing such push/pull logic by hand is tedious, error-prone, and a nightmare to read as any experienced jQuery programmer can attest. Angular supports **data binding**, a mechanism for coordinating parts of a template with parts of a component. Add binding markup to the template HTML to tell Angular how to connect both sides.
As the diagram shows, there are four forms of data binding syntax. Each form has a direction — to the DOM, from the DOM, or in both directions.



Directives

Angular templates are *dynamic*. When Angular renders them, it transforms the DOM according to the instructions given by **directives**. A directive is a class with a @Directive decorator. A component is a *directive- with-a-template*; a @Component decorator is actually a @Directive decorator extended with template-oriented features.
Two *other* kinds of directives exist: *structural* and *attribute* directives. They tend to appear within an element tag as attributes do, sometimes by name but more often as the target of an assignment or a binding.
Structural directives alter layout by adding, removing, and replacing elements in DOM.

Services

Service is a broad category encompassing any value, function, or feature that your application needs. Almost anything can be a service. A service is typically a class with a narrow, well-defined purpose. It should do something specific and do it well.

Examples include:

- logging service
- data service
- message bus
- tax calculator
- application configuration

There is nothing specifically *Angular* about services. Angular has no definition of a service. There is no service base class, and no place to register a service. Yet services are fundamental to any Angular application. Components are big consumers of services.

Dependency injection

Dependency injection is a way to supply a new instance of a class with the fully-formed dependencies it requires. Most dependencies are services. Angular uses dependency injection to provide new components with the services they need.

Angular can tell which services a component needs by looking at the types of its constructor parameters.

References:

1. https://angular.io/
2. https://angular.io/tutorial
3. https://www.w3schools.com/angular/angular_intro.asp
4. https://www.tutorialspoint.com/angularjs/index.htm
5. https://www.javatpoint.com/angularjs-tutorial

Oral Questions:

1. What is data binding in AngularJS?
2. What is scope in AngularJS?
3. What are the controllers in AngularJS?
4. What are the services in AngularJS?
5. What are the filters in AngularJS?
6. Explain directives in AngularJS.
7. Explain templates in AngularJS.
8. What is routing in AngularJS?
9. What is deep linking in AngularJS?
10. What are the advantages of AngularJS?
11. What are the disadvantages of AngularJS?
12. Which are the core directives of AngularJS?
13. Explain AngularJS boot process.
14. What is MVC?
15. Explain ng-app directive.
16. Explain ng-model directive.
17. Explain ng-bind directive.
18. Explain ng-controller directive.
19. How AngularJS integrates with HTML?

20. Explain ng-init directive.

21. Explain ng-repeat directive.
22. What are AngularJS expressions?
23. Explain uppercase filter.
24. Explain lowercase filter.
25. Explain currency filter.
26. Explain filter filter.
27. Explain orderby filter.
28. Explain ng-disabled directive.
29. Explain ng-show directive.
30. Explain ng-hide directive.
31. Explain ng-click directive.
32. How angular.module works?
33. How to validate data in AngularJS?
34. Explain ng-include directive.
35. How to make an ajax call using Angular JS?
36. What is use of $routeProvider in AngularJS?
37. What is $rootScope?
38. What is scope hierarchy in AngularJS?
39. What is a service?
40. What is service method?
41. What is factory method?
42. What are the differences between service and factory methods?
43. Which components can be injected as a dependency in AngularJS?
44. What is provider?
45. What is constant?
46. Is AngularJS extensible?
47. On which types of component can we create a custom directive?
48. What is internationalization? How to implement internationalization in AngularJS?

Assignment No. 8

TITLE	**Design, develop and deploy web application using EJB**
PROBLEM STATEMENT /DEFINITION	Design, develop and deploy web application using EJB, JSP & Servlet from the given list : 1. Online pizza order application 2. Student information system for training & placement department 3. Leave management application 4. Blogging platform 5. Meeting room booking application 6. Exam cell automation application
OBJECTIVE	● To understand working of EJB ● To explore the usage of EJB with JSP and Servlet
S/W PACKAGES AND HARDWARE APPARATUS USED	Operating System open source Fedora 20 Networked computer with internet access Editor : IDE : Netbeans 8.1 Web browser Mozilla Firefox, Google Chrome
REFERENCES	● http://www.ejbtutorial.com/category/ejb ● https://www.javatpoint.com/ejb-tutorial ● https://netbeans.org/kb/docs/javaee/entappclient.html ● https://examples.javacodegeeks.com/enterprise-java/ejb3/ejb-tutorial-beginners-example/ ● https://netbeans.org/kb/docs/javaee/javaee-entapp-ejb.html ● https://wowjava.wordpress.com/2011/01/14/a-simple-ejb-3-0-example-with- jsp-and-servlet/ ● https://www.youtube.com/watch?v=uI8TGqv-5hk&feature=related ● http://www.pavanjaiswal.com/2018/03/simple-ejb-jsp-servlet- application.html
STEPS	Refer to steps below
INSTRUCTIONS FOR WRITING JOURNAL	● Title ● Problem Statement ● Theory ● Architecture diagrams (design part) ● Troubleshooting (if any) ● Conclusion ● References

EJB

EJB (Enterprise Java Bean) is used to develop scalable, robust and secured enterprise applications in java.

Unlike RMI, middleware services such as security, transaction management etc. are provided by **EJB Container** to all EJB applications.

The current version of EJB is EJB 3.2. The development of EJB 3 is faster than EJB 2 because of simplicity and annotations such as @EJB, @Stateless, @Stateful, @ModelDriven, @PreDestroy, @PostConstruct etc.

To run EJB application, you need an application server (EJB Container) such as Jboss, Glassfish, Weblogic, Websphere etc. It performs:

a. life cycle management,
b. security,
c. transaction management, and
d. object pooling.

EJB application is deployed on the server, so it is called server side component also.

EJB is like COM (Component Object Model) provided by Microsoft. But, it is different from Java Bean, RMI and Web Services.

When to use EJB?

1. **Application needs Remote Access**. In other words, it is distributed.
2. **Application needs to be scalable**. EJB applications supports load balancing, clustering and fail-over.
3. **Application needs encapsulated business logic**. EJB application is separated from presentation and persistent layer.

Types of EJB?

There are 3 types of enterprise bean in java.

Session Bean: Session bean contains business logic that can be invoked by local, remote or webservice client.

Message Driven Bean: Like Session Bean, it contains the business logic but it is invoked by passing message.

Entity Bean: It encapsulates the state that can be persisted in the database. It is deprecated. Now, it is replaced with JPA (Java Persistent API).

Difference between RMI and EJB

Both RMI and EJB, provides services to access an object running in another JVM (known as remote object) from another JVM. The differences between RMI and EJB are given below:

RMI	EJB
In RMI, middleware services such as security, transaction management, object pooling etc. need to be done by the java programmer.	In EJB, middleware services are provided by EJB Container automatically.
RMI is not a server-side component. It is not required to be deployed on the server.	EJB is a server-side component, it is required to be deployed on the server.
RMI is built on the top of socket programming.	EJB technology is built on the top of RMI.

EJB and Webservice

In EJB, bean component and bean client both must be written in java language.

If bean client need to be written in other language such as **.net**, **php** etc, we need to go with **webservices** (SOAP or REST). So EJB with web service will be better option.

Disadvantages of EJB

1. Requires application server
2. Requires only java client. For other language client, you need to go for webservice.
3. Complex to understand and develop ejb applications

Session Bean

Session bean encapsulates business logic only, it can be invoked by local, remote and webservice client.

It can be used for calculations, database access etc.

The life cycle of session bean is maintained by the application server (EJB Container).

Types of Session Bean

1. **Stateless Session Bean**: It doesn't maintain state of a client between multiple method calls.
2. **Stateful Session Bean**: It maintains state of a client across multiple requests.
3. **Singleton Session Bean**: One instance per application, it is shared between clients and supports concurrent access.

Stateless Session Bean

Stateless Session bean *is a business object that represents business logic* only. It doesn't have state (data).

In other words, *conversational state* between multiple method calls is not maintained by the container in case of stateless session bean.

The stateless bean objects are pooled by the EJB container to service the request on demand.

It can be accessed by one client at a time. In case of concurrent access, EJB container routes each request to different instance.

Annotations used in Stateless Session Bean

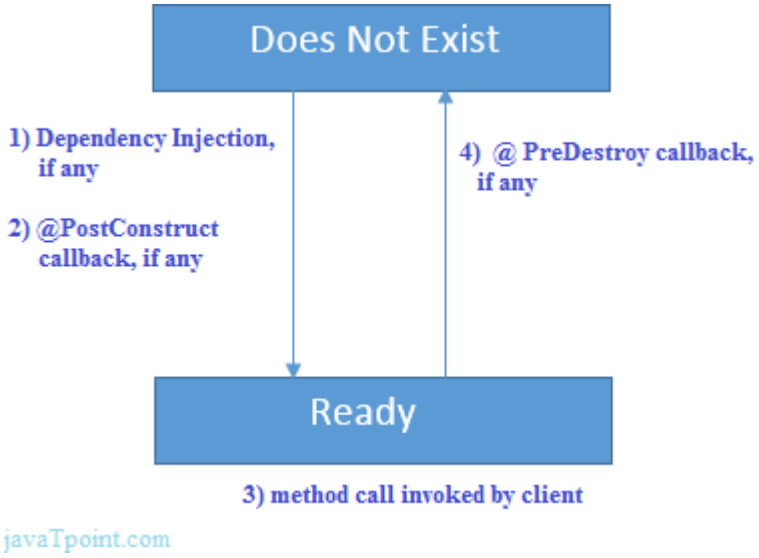
There are 3 important annotations used in stateless session bean:

1. @Stateless
2. @PostConstruct
3. @PreDestroy

Life cycle of Stateless Session Bean

There is only two states of stateless session bean: does not exist and ready. It is explained by the figure EJB Container creates and maintains a pool of session bean first. It injects the dependency if then calls the

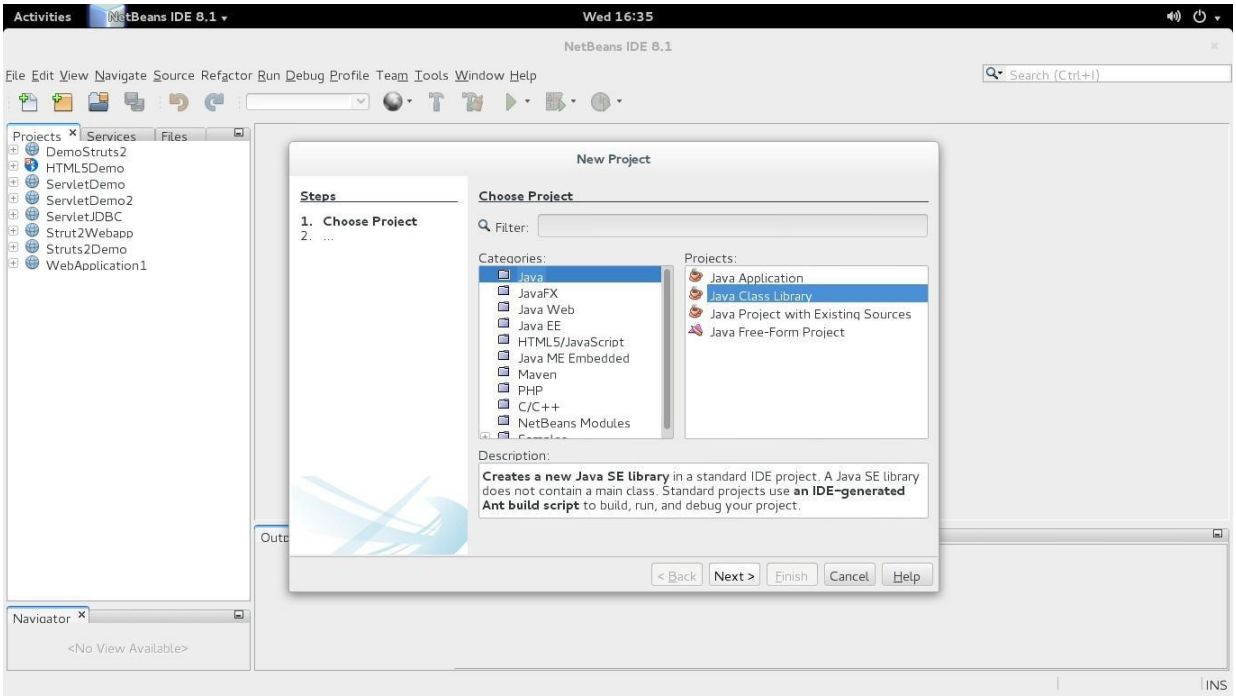
@PostConstruct method if any. Now actual business logic method is invoked by the client. Then, container calls @PreDestory method if any. Now bean is ready for garbage collection.



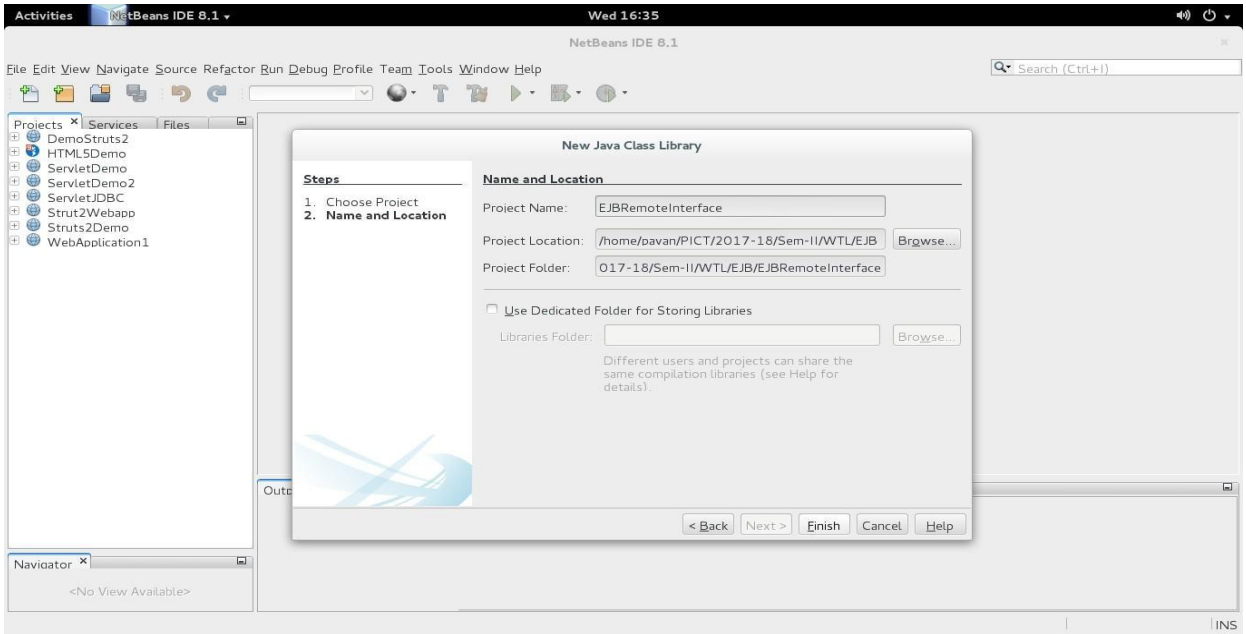
Example of Stateless Session Beans

IDE used: Netbeans 8.1
Application server: GlassFish-4.1.1
Jdk: 1.8
Operating System:Fedora
Follow the below mentioned steps. Its like **Hello World application** in EJB that take help of JSP and Servlet.

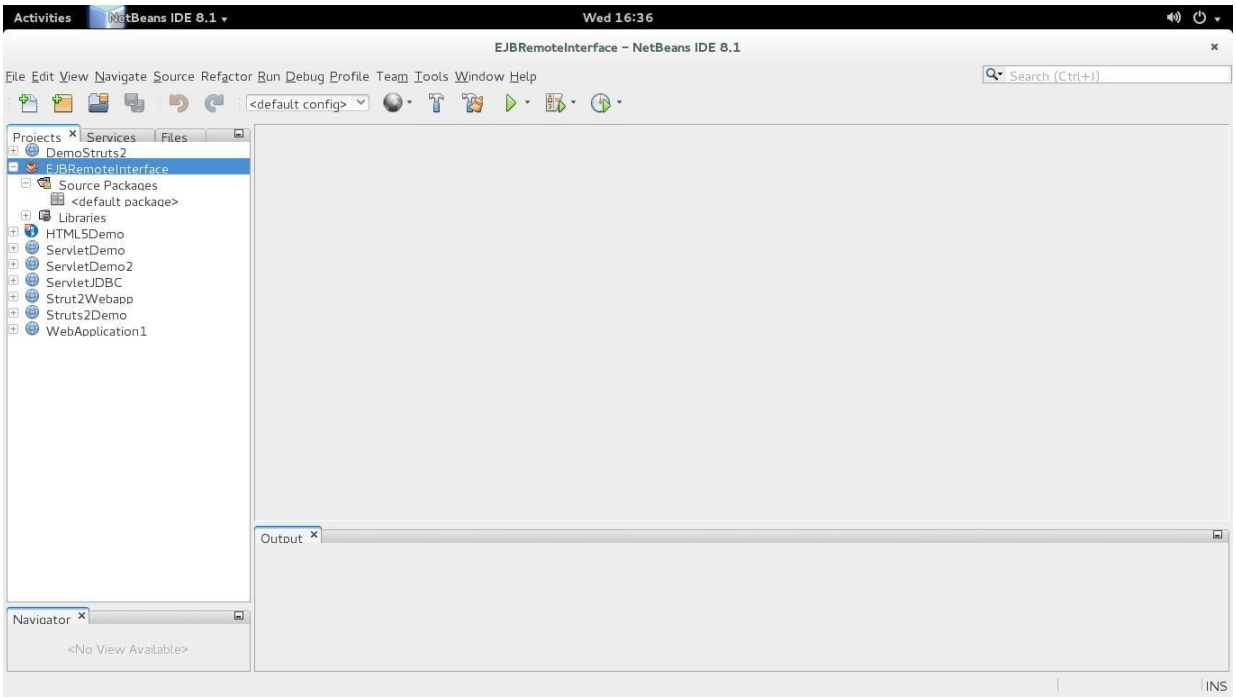
1. Create Java class library project



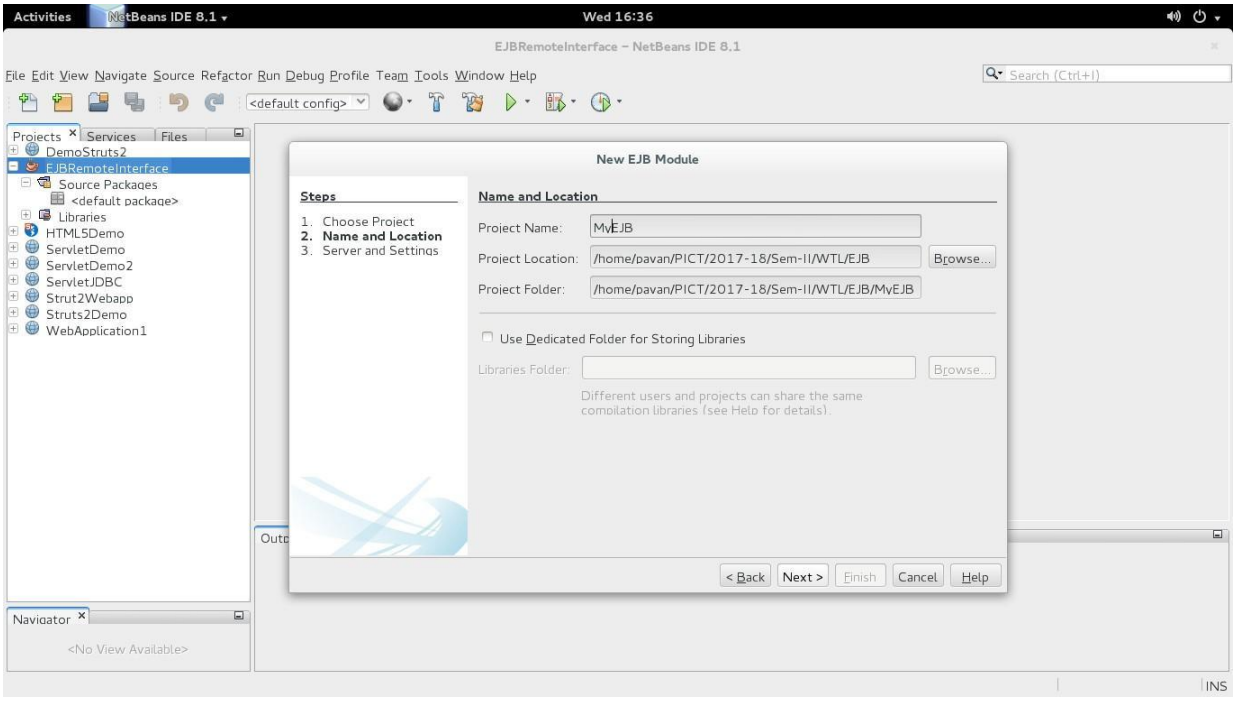
2.Provide name EJBRemoteInterface



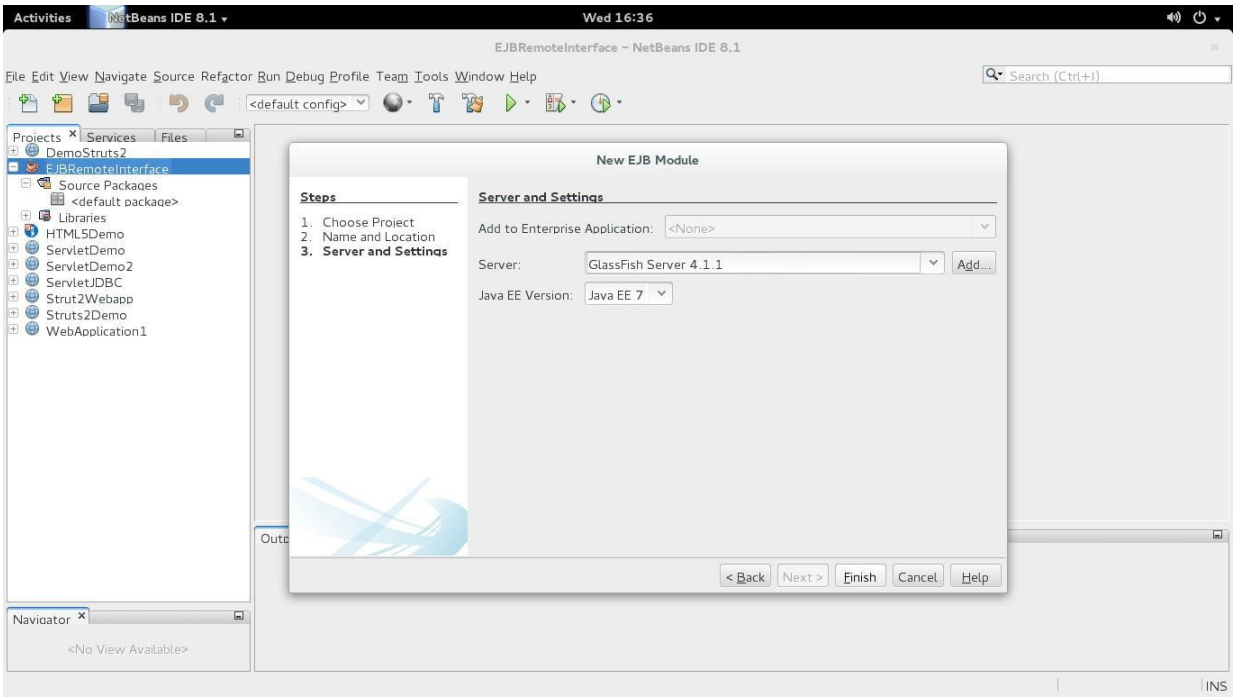
3. EJBRemoteInterface directory structure



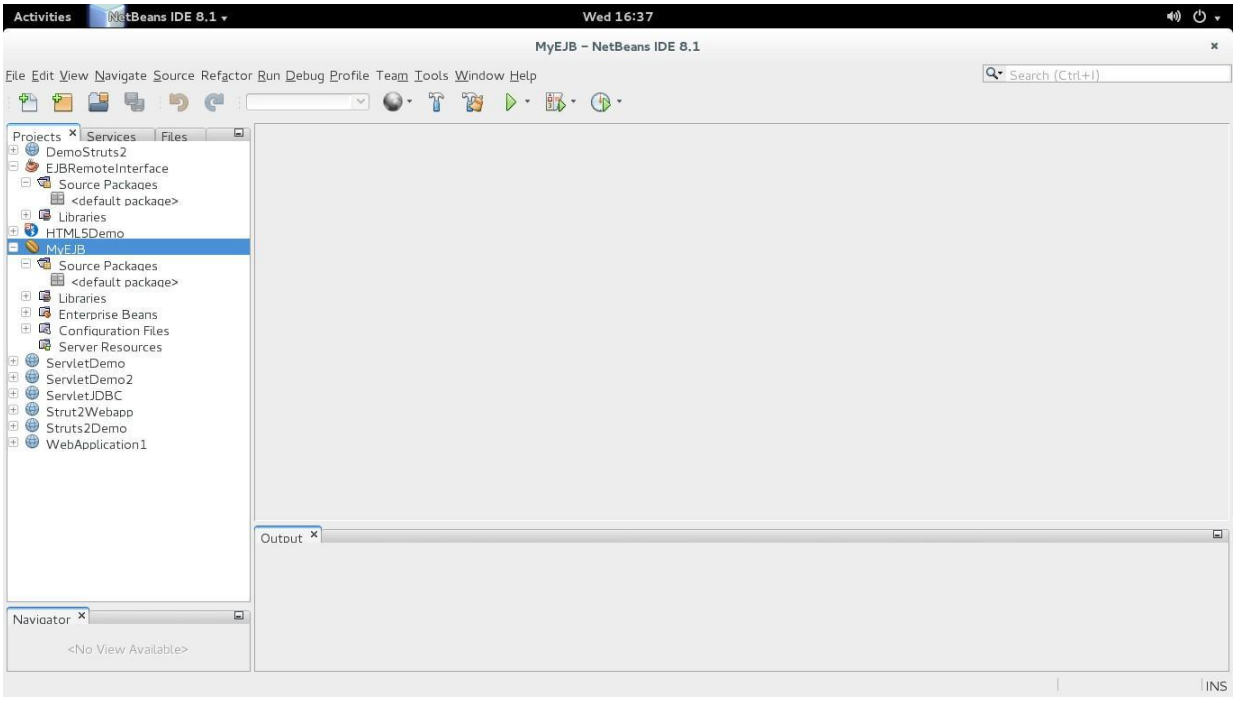
4. Create new JavaEE EJB module and provide name as MyEJB



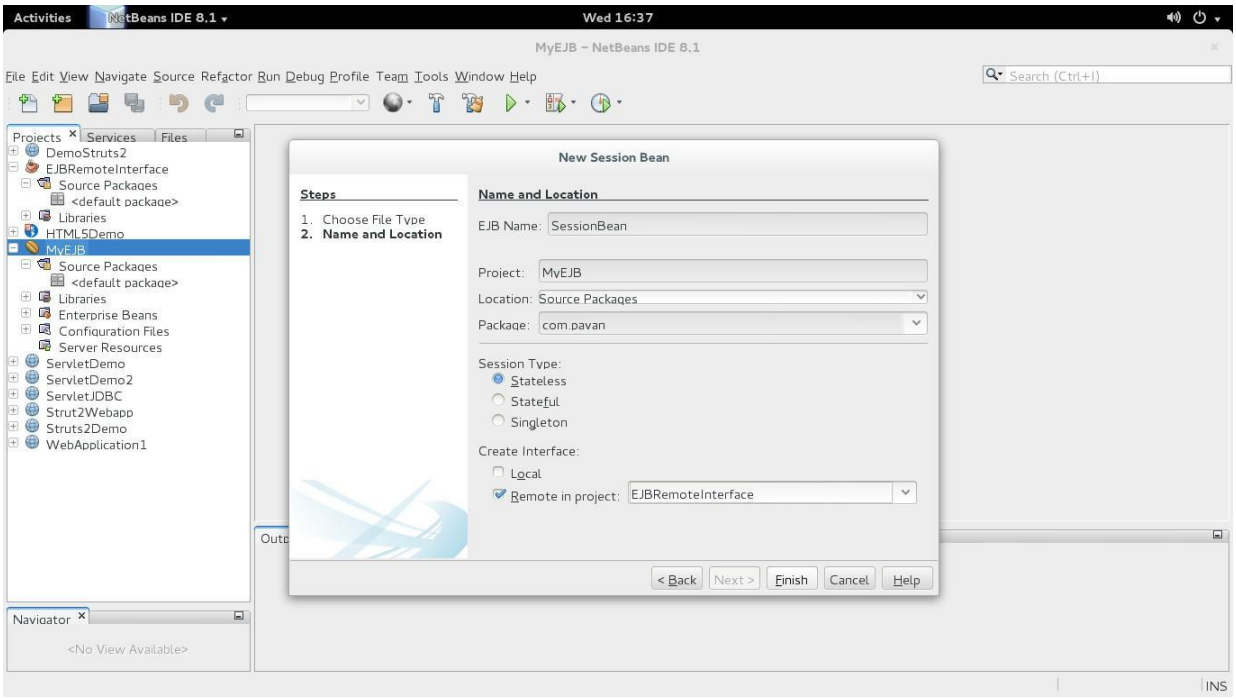
5. Select GlassFish Server



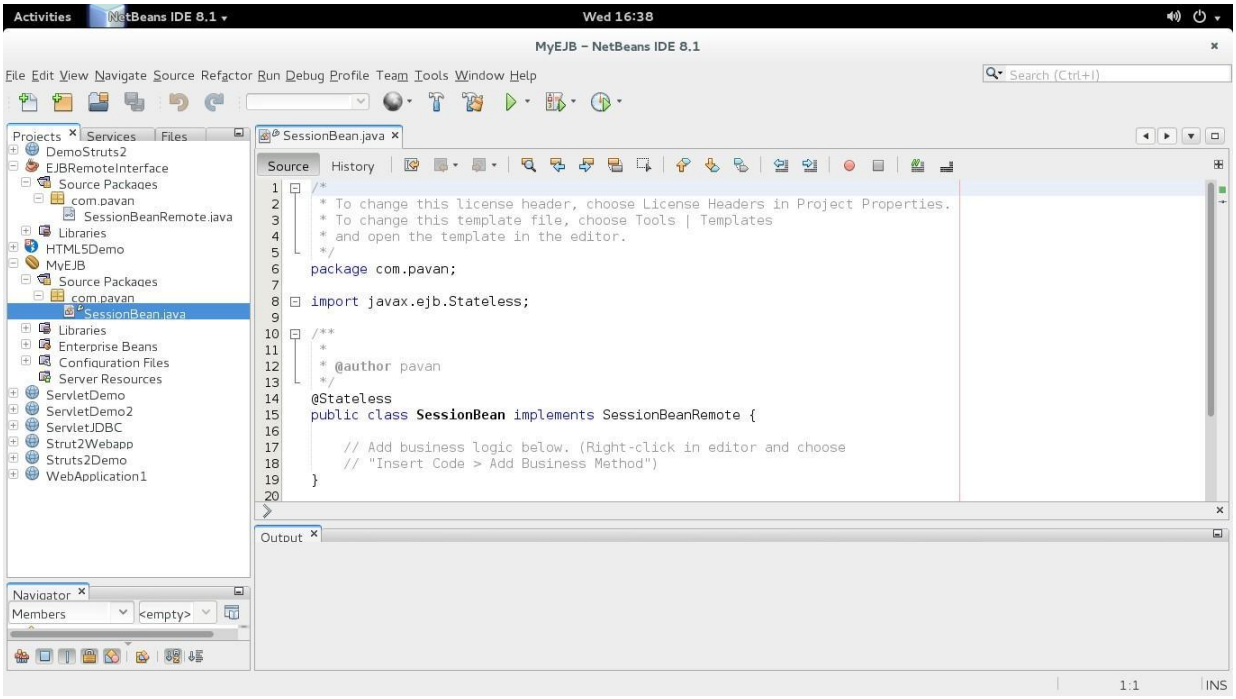
6. MyEJB directory structure



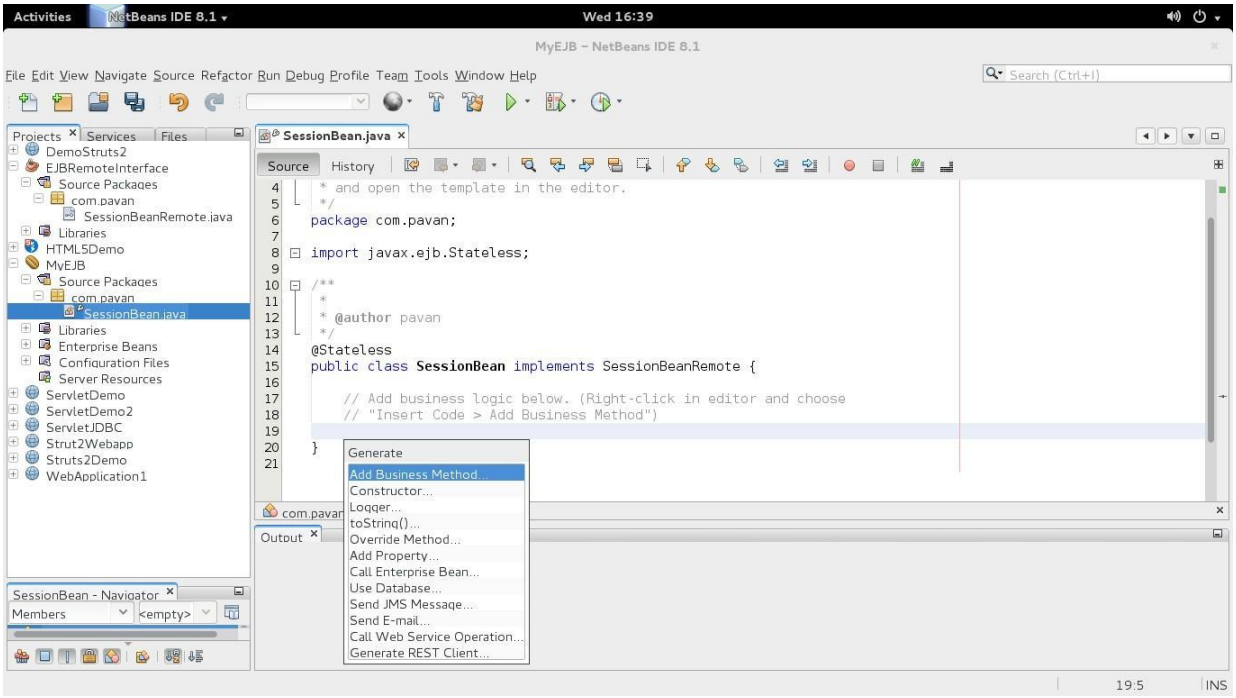
7. Right click on MyEJB and create new SessionBean having name: SessionBean



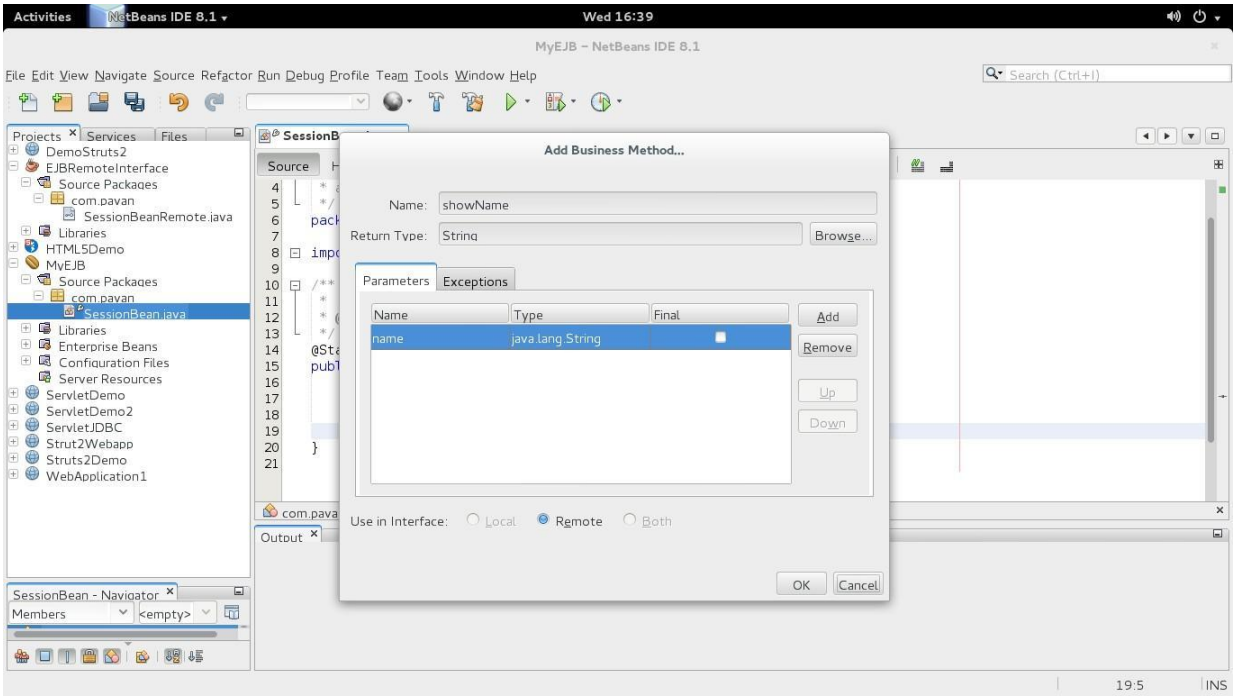
8. Empty SessionBean will look like as below



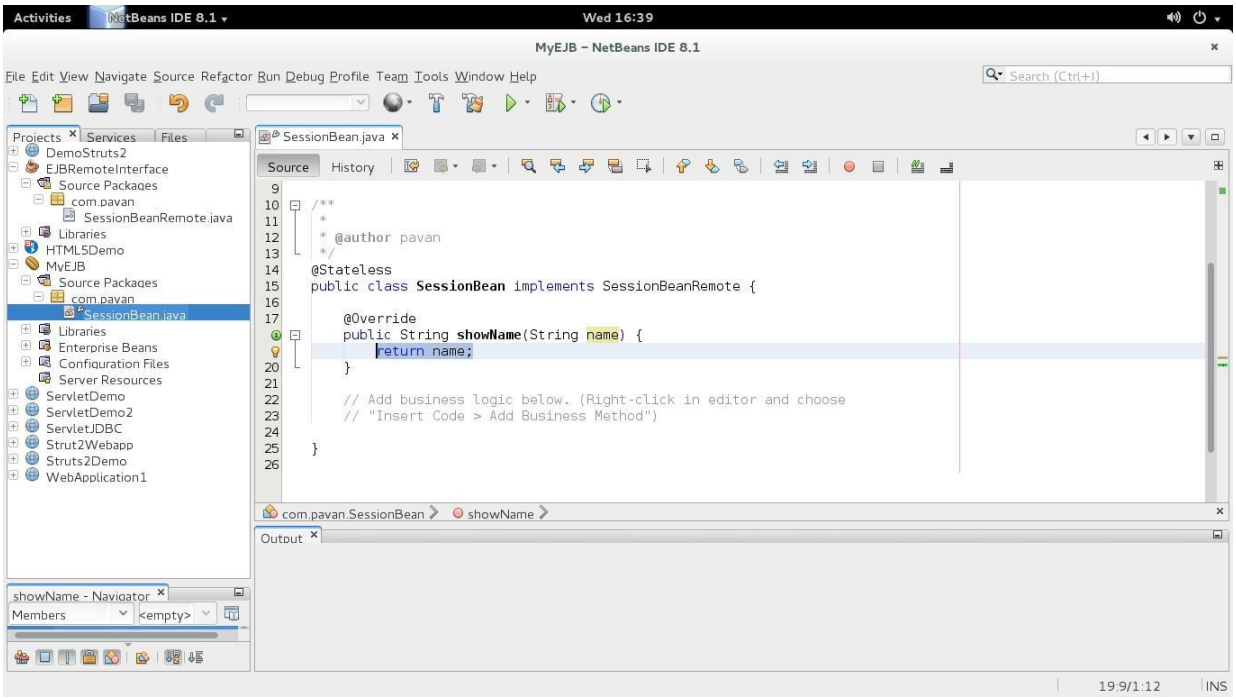
9. Add business method to newly created sessionbean by right clicking on file – insert code – business method



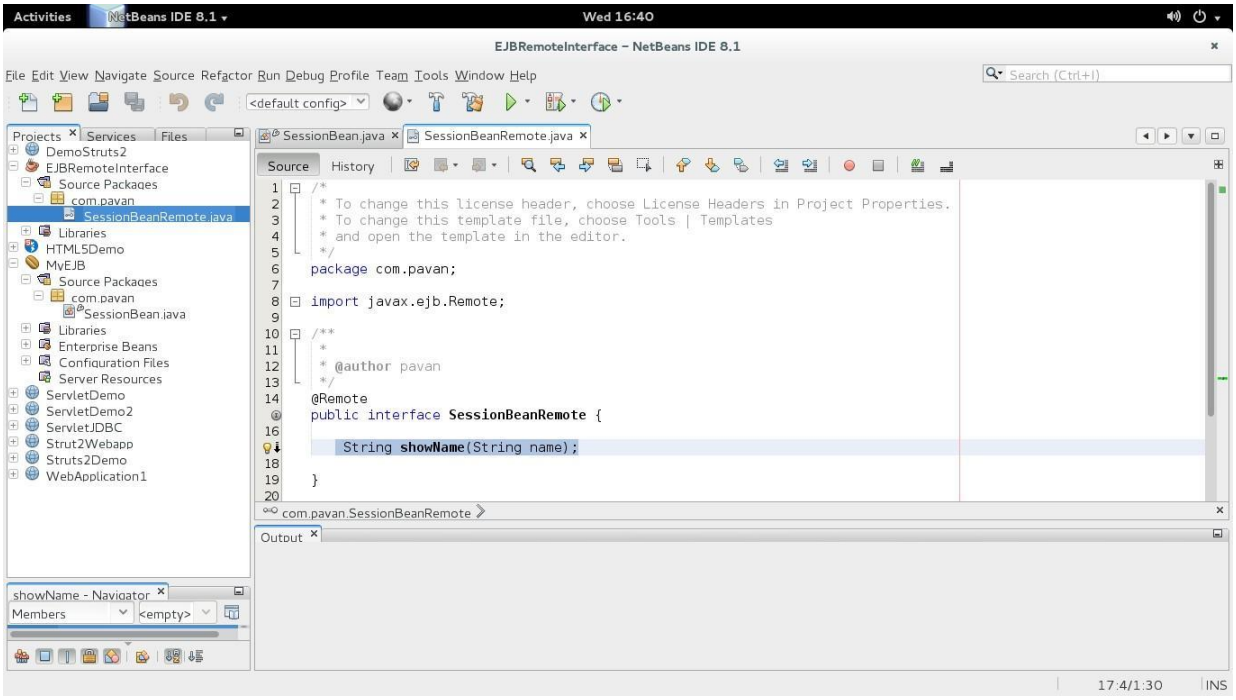
10. Providing name and parameters to business method: String showName(String)



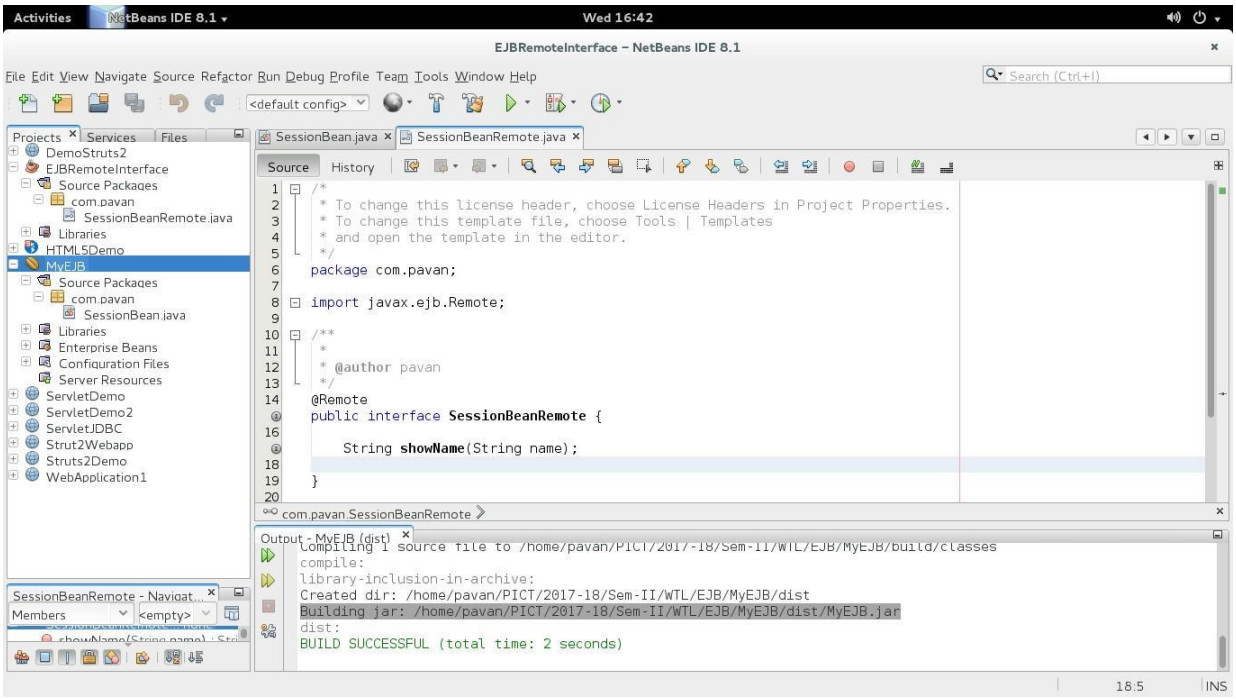
11. Final contents of SessionBean



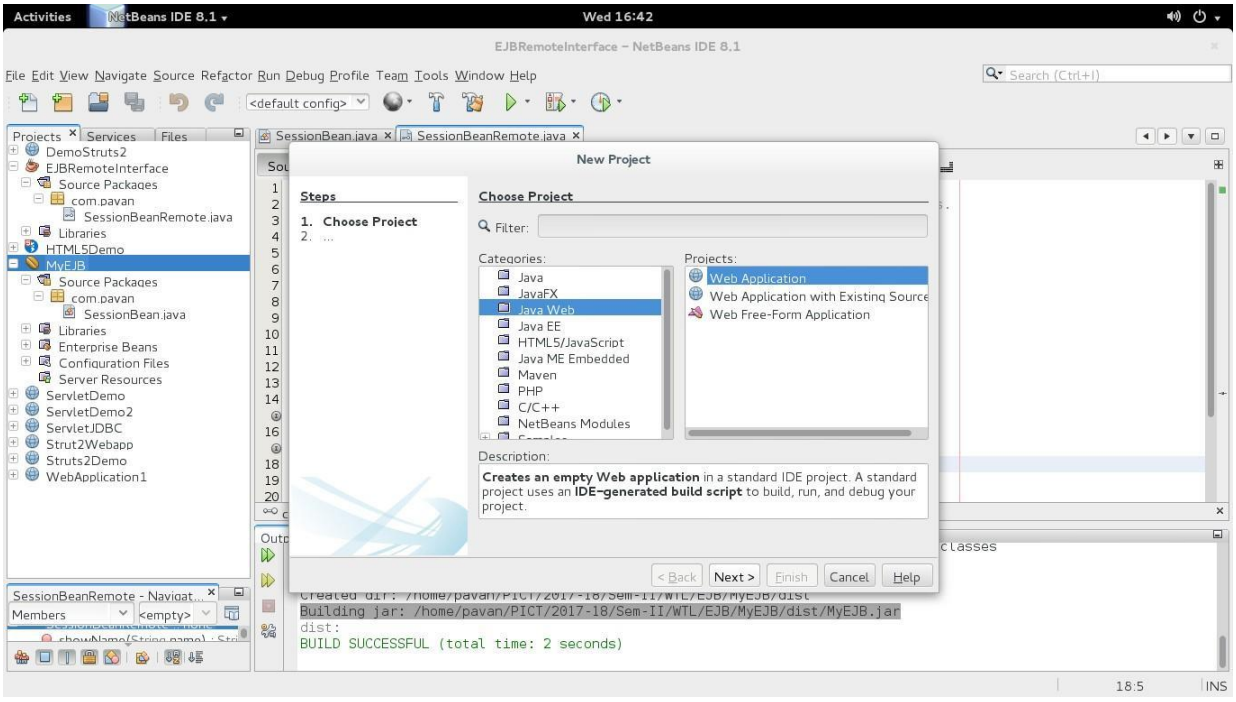
12. Contents of interface: SessionBeanRemote which has got created automatically in project: EJBRemoteInterface



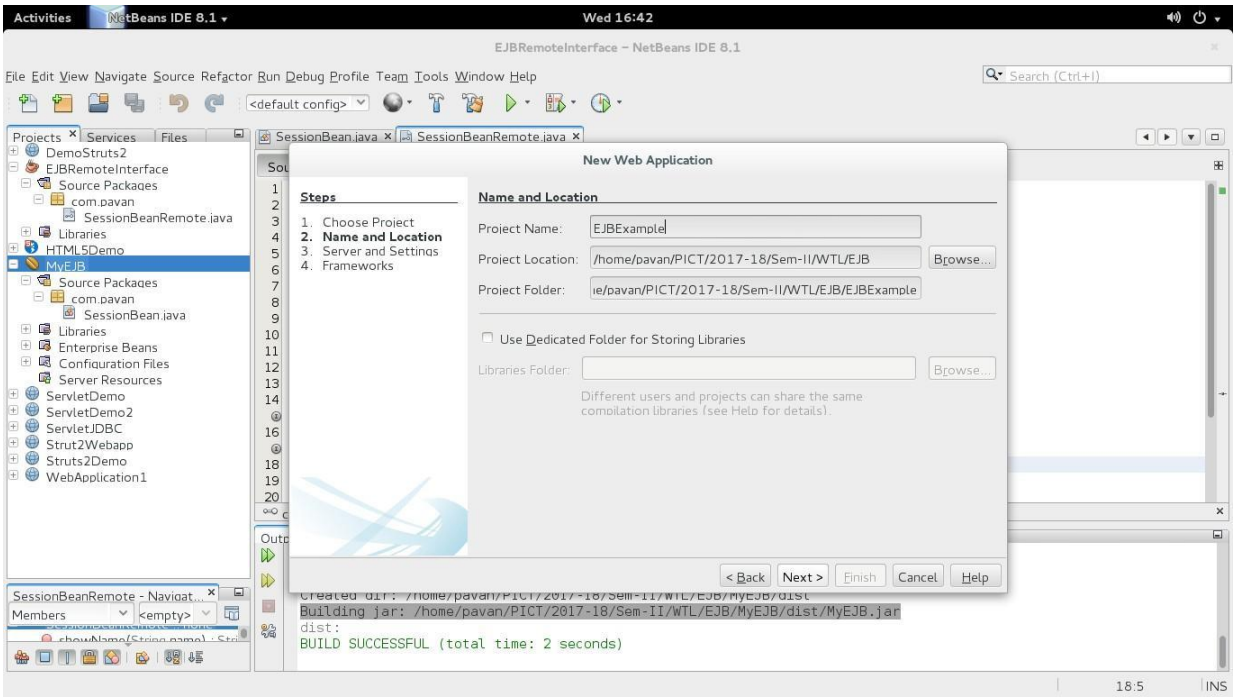
13. Build MyEJB. Right click on MyEJB and select Build option



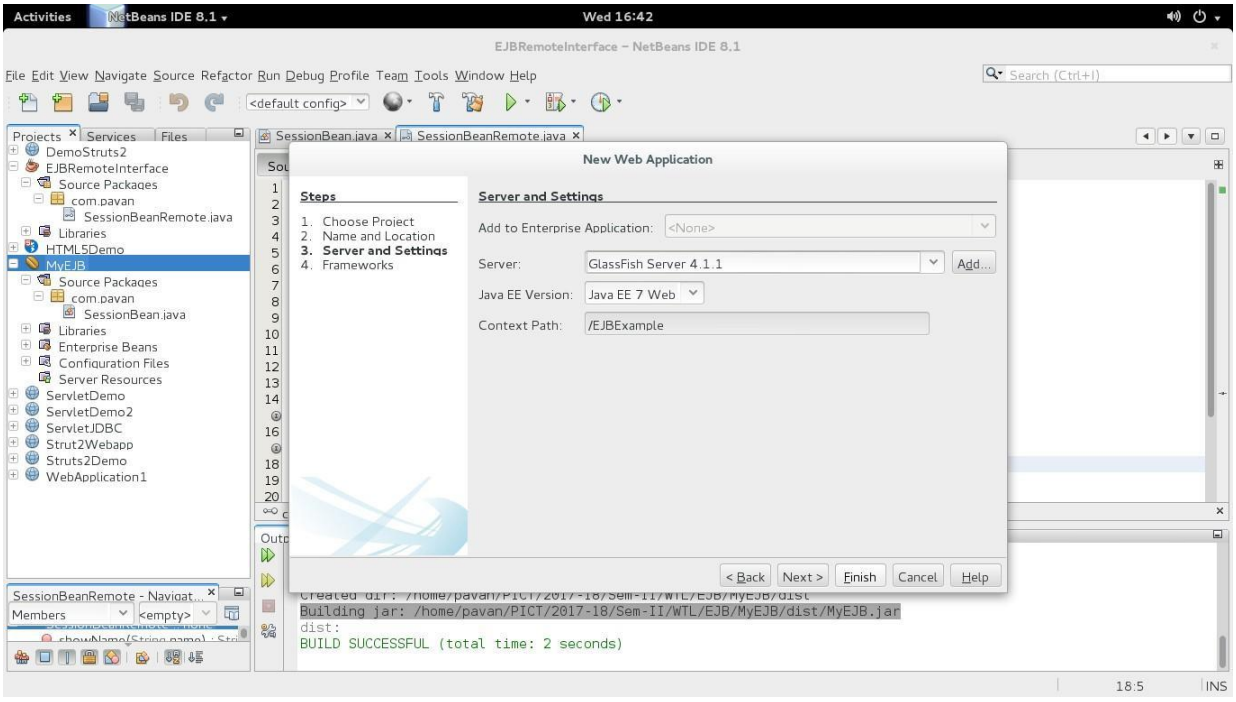
14. Create new Java Web application



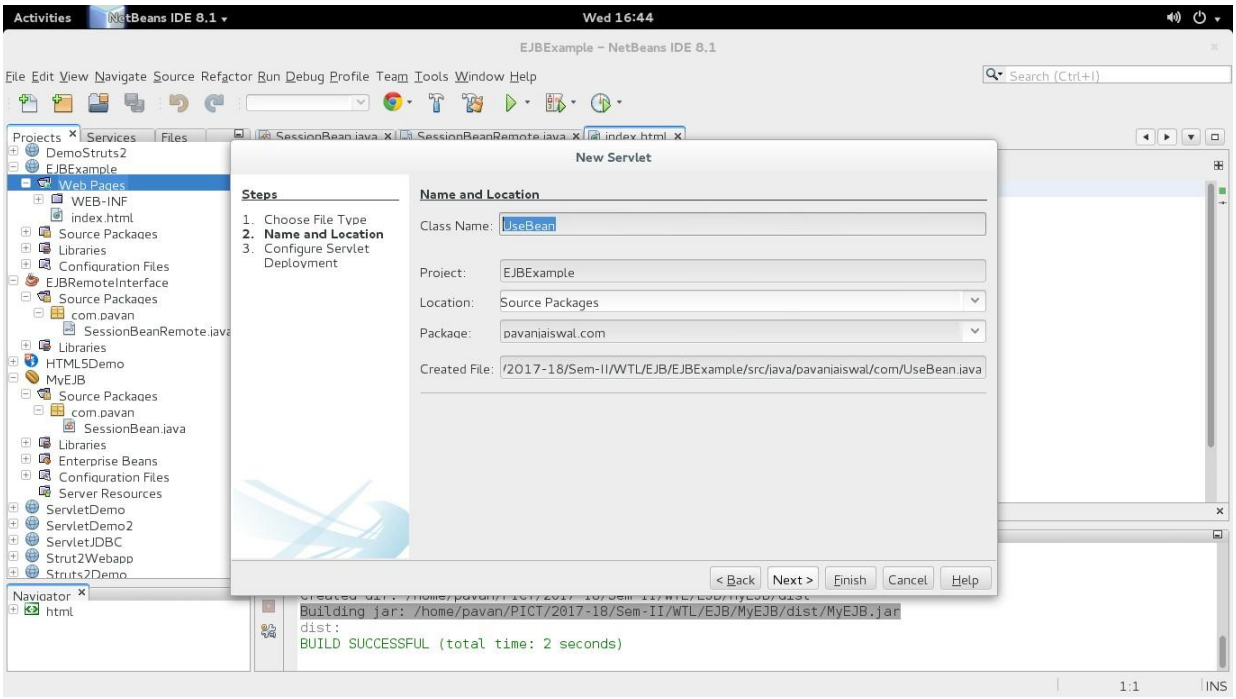
15. Provide name: EJBExample



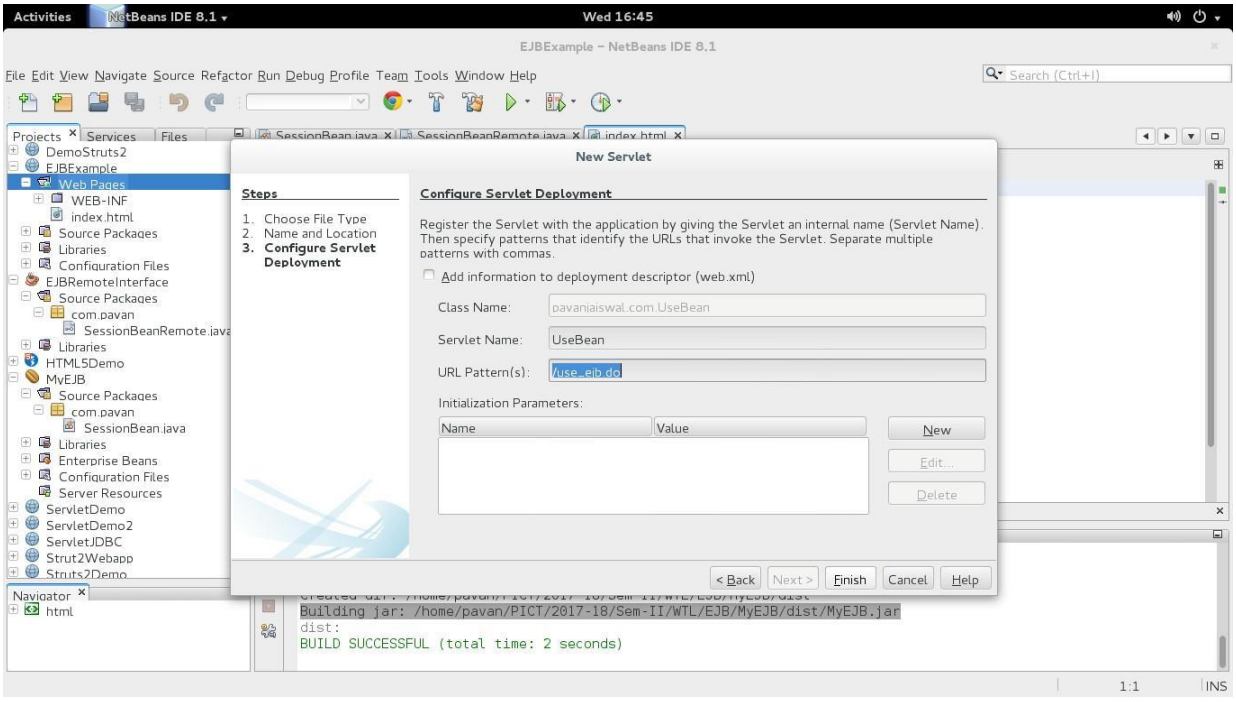
16. Glassfish server selection



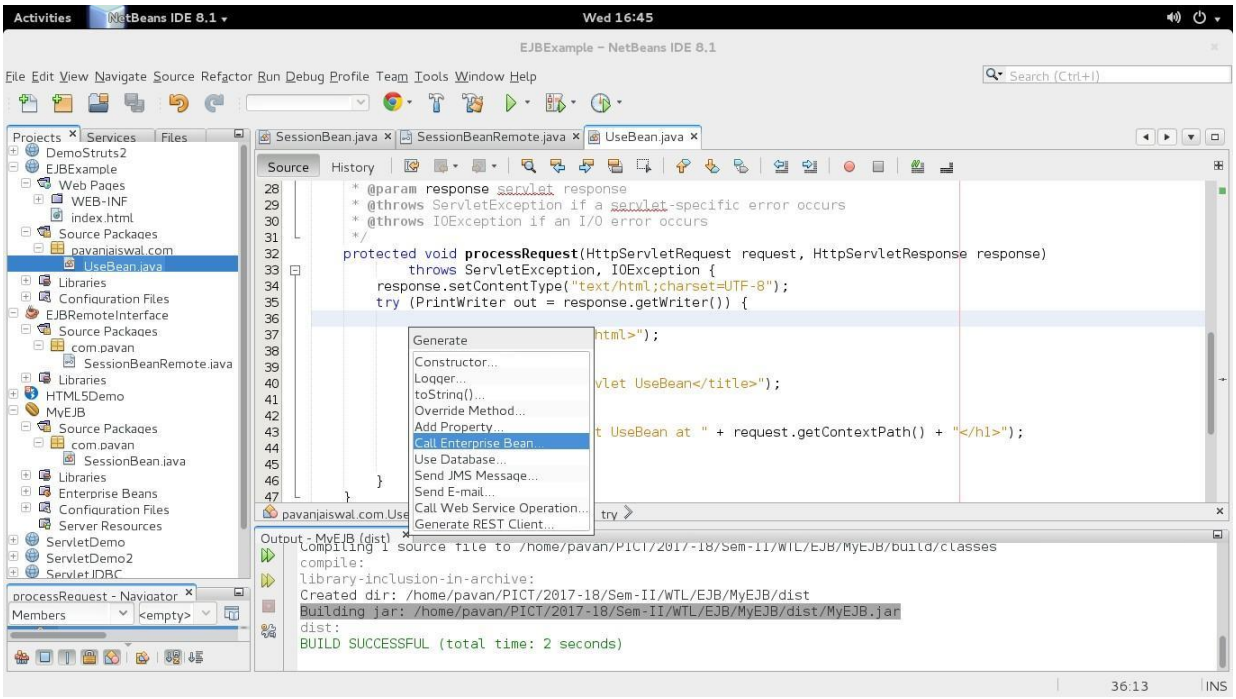
17. Create Servlet: UseBean



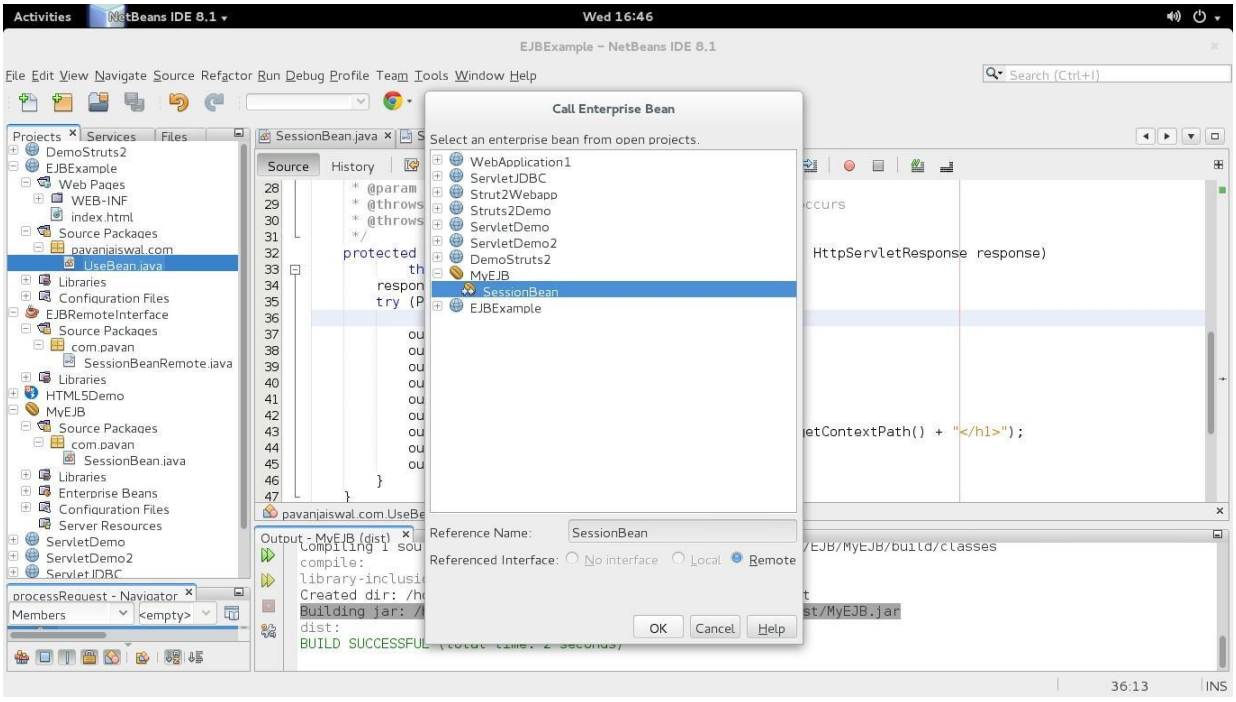
18. Provide url-pattern as: /use_ejb.do



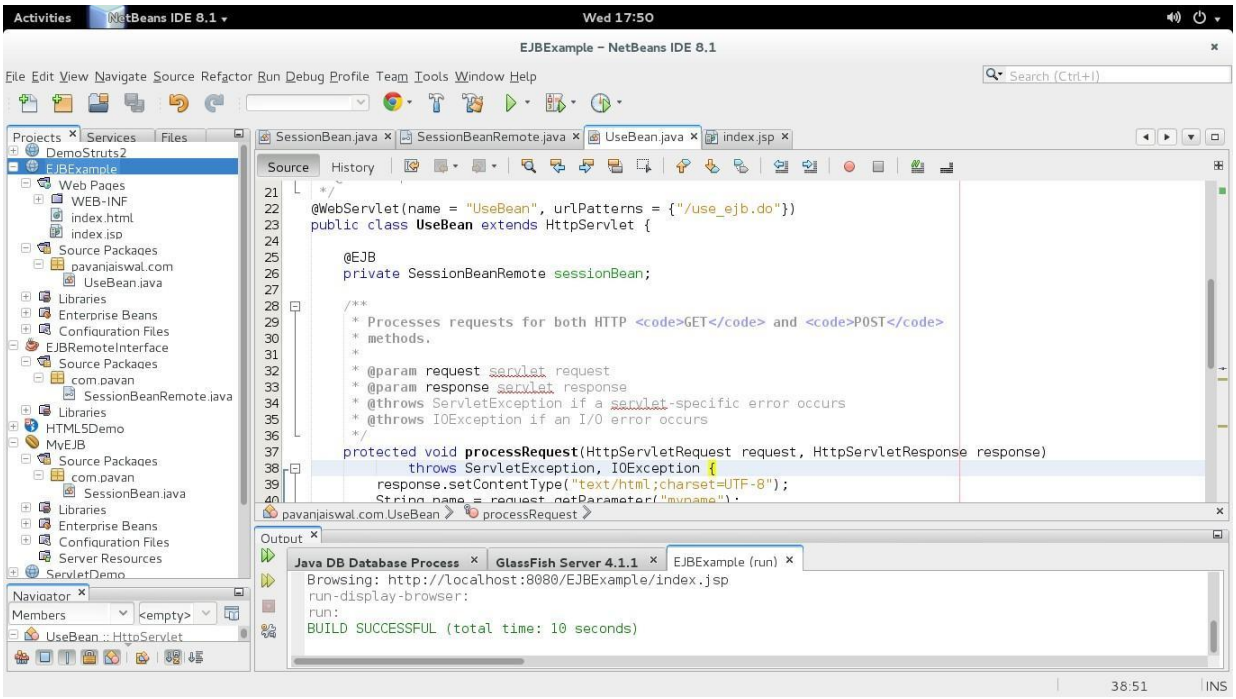
19. Call Enterprise Bean from Servlet ProcessRequest() method



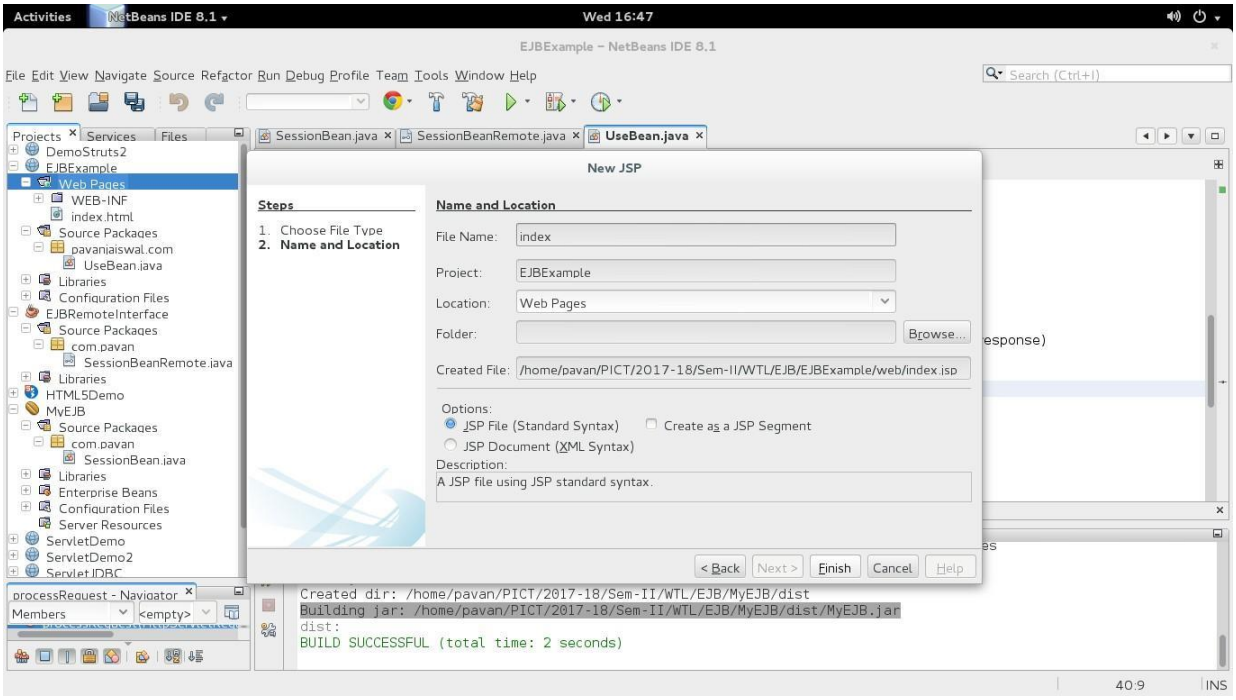
20. Select SessionBean from MyEJB drop down list



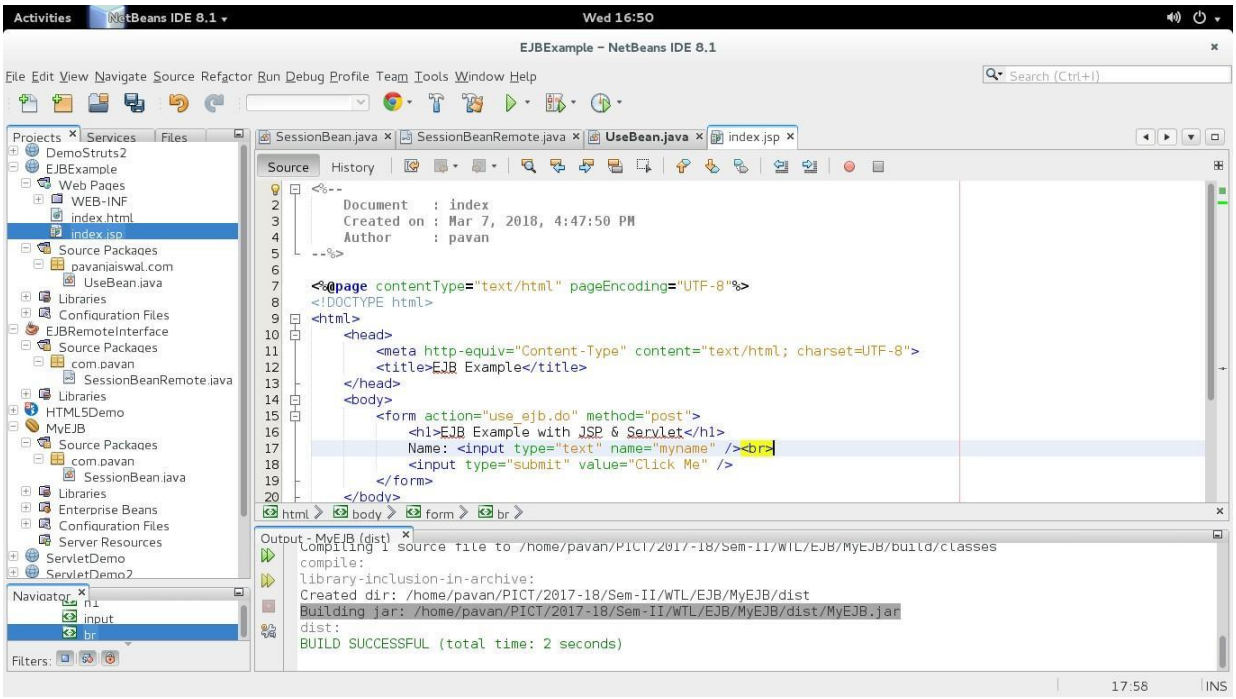
21. Instance of SessionBeanRemote created in servlet



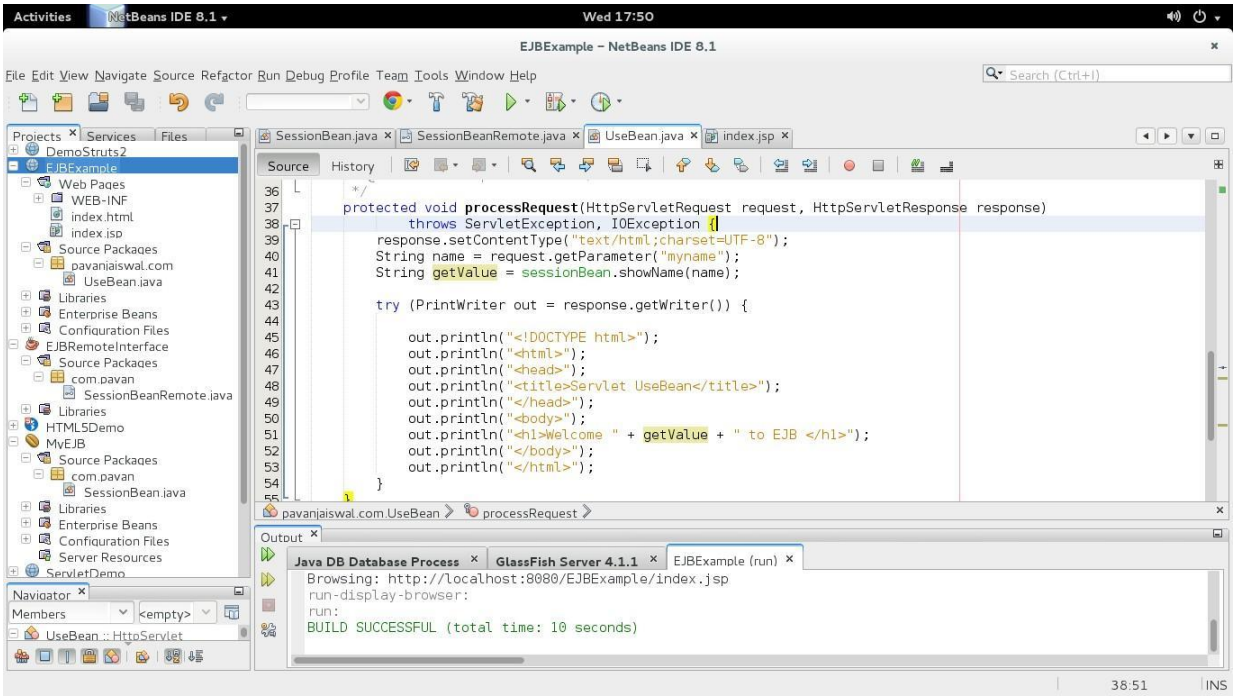
22. In EJBExample application create index.jsp which will have form to accept your name and submit event to: use_ejb.do servlet url-pattern



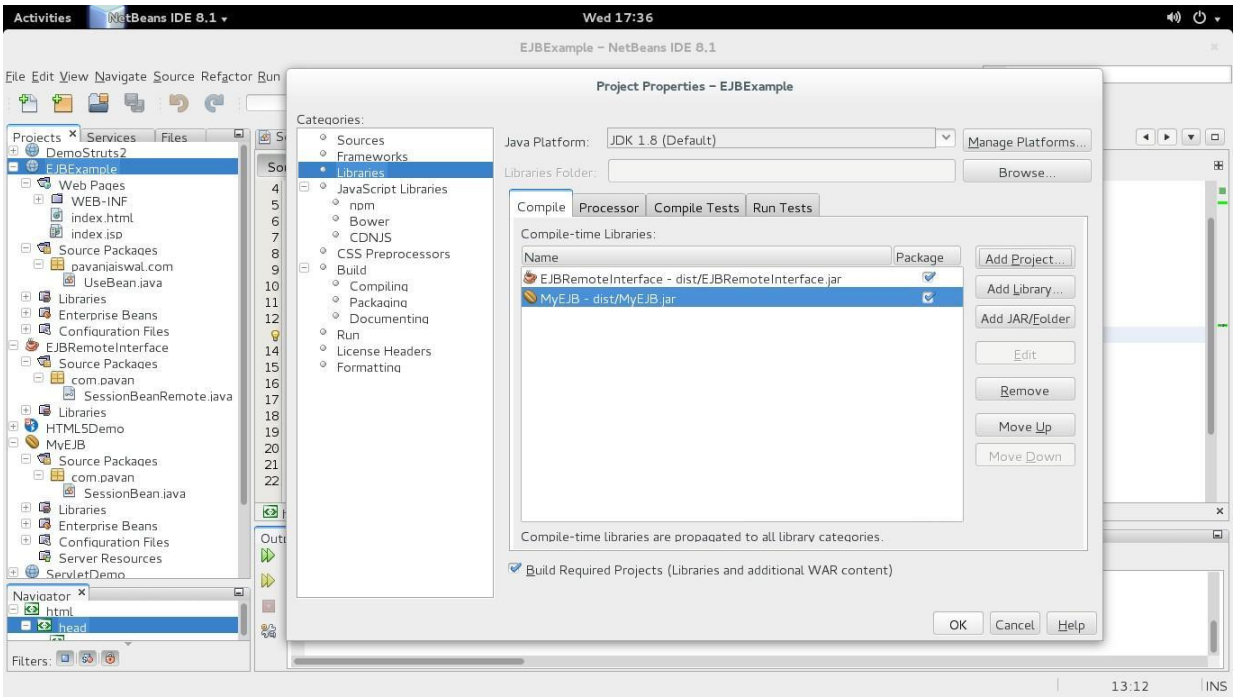
23. Contents of index.jsp



24. Contents of processRequest() method of UseBean servlet which in turns calling SessionBean method showName()



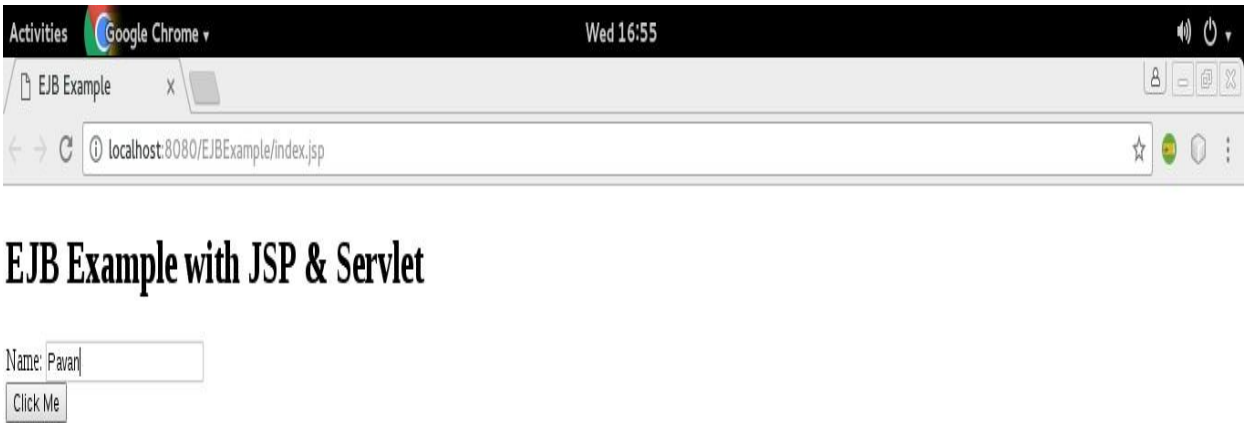
25. Right click to EJBExample – properties – library – add projects MyEJB and EJBRemoteInterface



26. Clean and build EJBExample

27. Deploy MyEJB. Just right click on it and select option Deploy

28. If everything goes fine, then run your project EJBExample. You will see index.jsp ready to accept your name in browser.



Congratulations!!! You are done with your First EJB-JSP-Servlet application.

References:

1. http://www.ejbtutorial.com/category/ejb
2. https://www.javatpoint.com/ejb-tutorial
3. https://netbeans.org/kb/docs/javaee/entappclient.html
4. https://examples.javacodegeeks.com/enterprise-java/ejb3/ejb-tutorial-beginners-example/
5. https://netbeans.org/kb/docs/javaee/javaee-entapp-ejb.html
6. https://wowjava.wordpress.com/2011/01/14/a-simple-ejb-3-0-example-with-jsp-and-servlet/
7. https://www.youtube.com/watch?v=uI8TGqv-5hk&feature=related
8. http://www.pavanjaiswal.com/2018/03/simple-ejb-jsp-servlet-application.html

Oral questions:

1. What is EJB?
2. What are the types of Enterprise Beans
3. What is session bean?
4. What is stateless session bean?
5. What is stateful session bean?
6. What is entity bean?
7. What are the benefits of EJB?
8. When to use local session bean and remote session bean?
9. Is Message Driven bean a stateless bean?
10. Explain @ annotations used in beans.