Practical no: 10

Problem Statement: Design, Develop & Deploy web.

Name: Aditi Dinesh Mulay

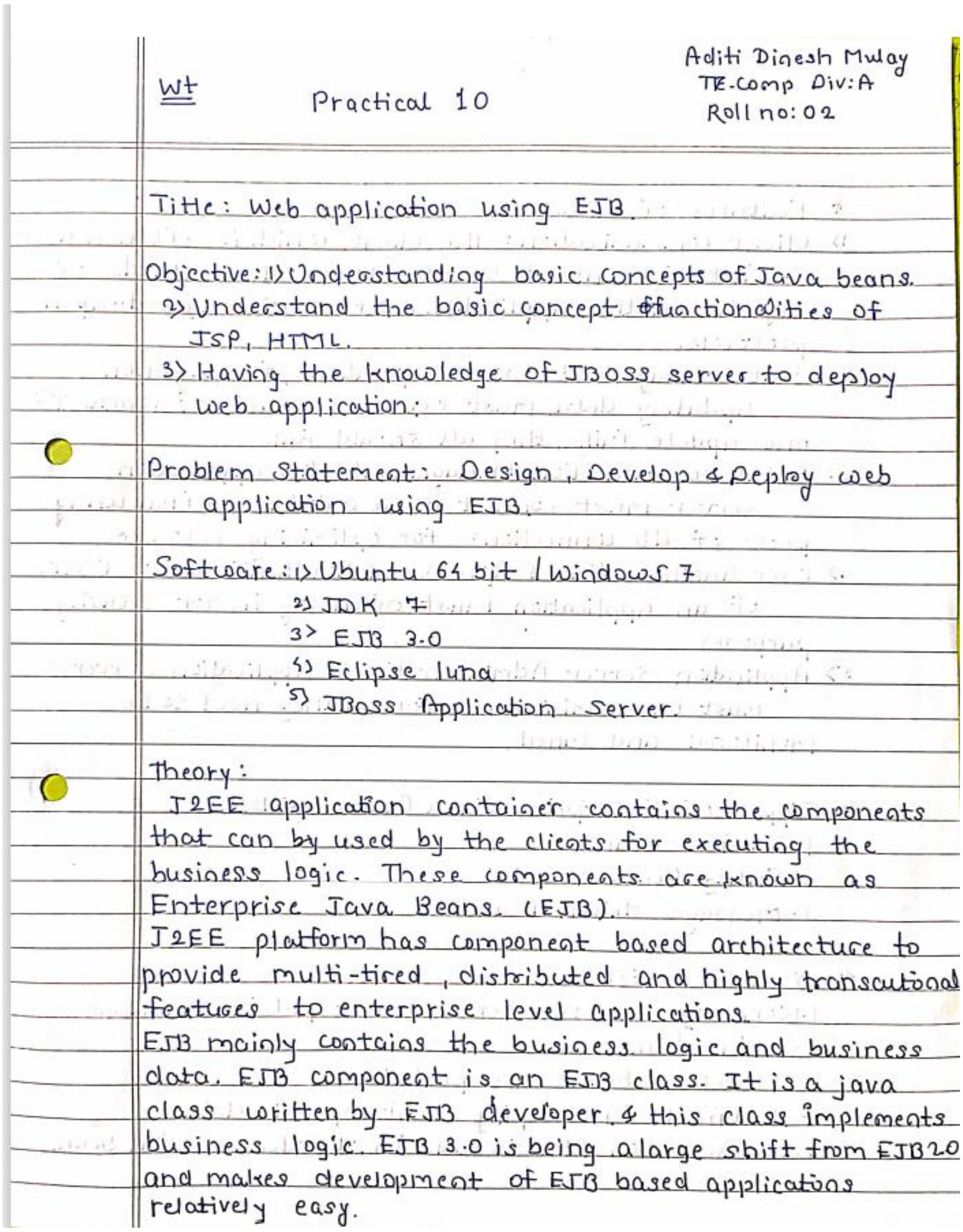
Class: T.E. Computer

Subject: Web Technology

Div: A

Roll no: 02

PRN No. 71918146B



Wt

Practical 10

Aditi Dinesh Mulay
TE-Comp Div:A
Roll no: 02

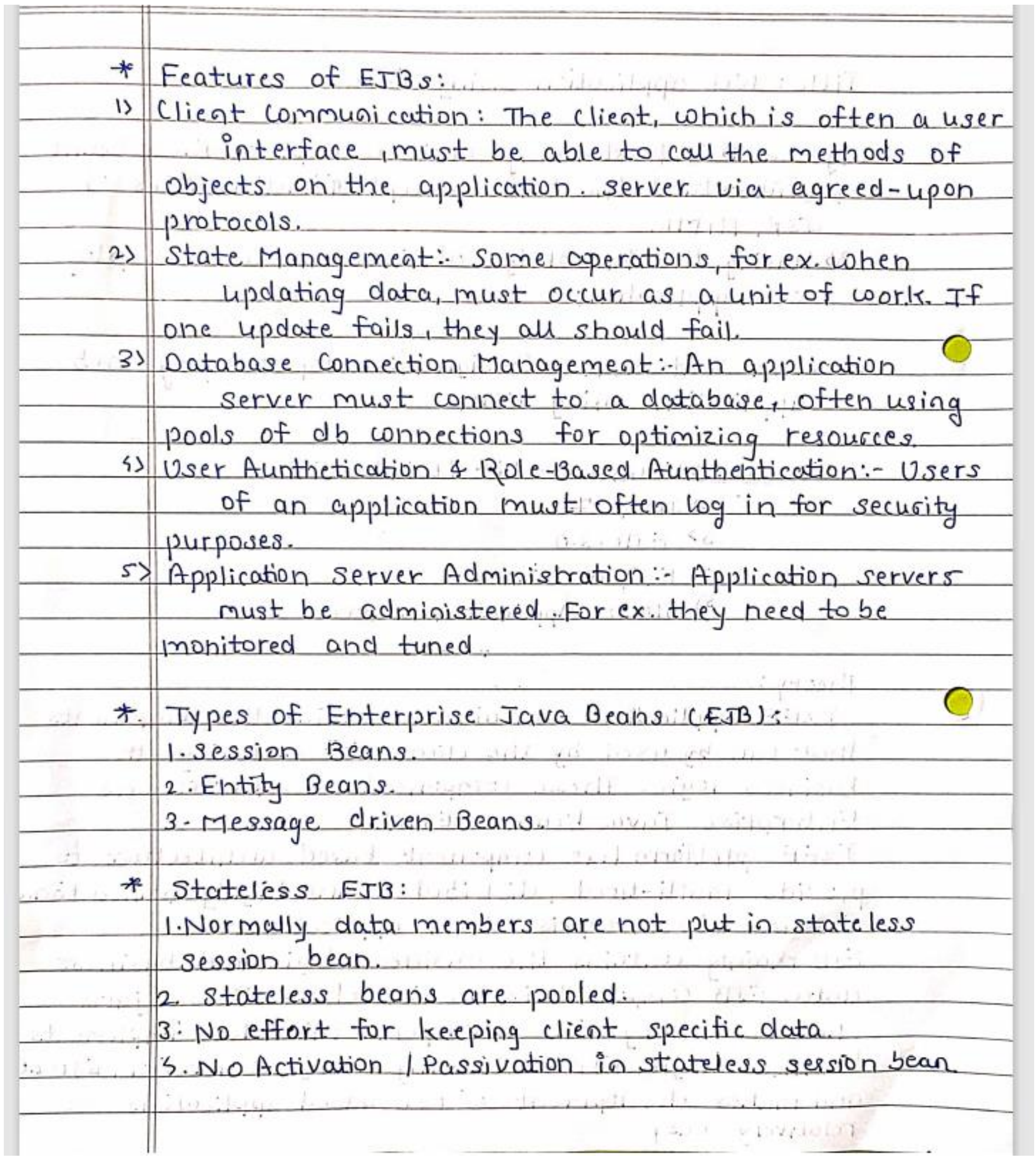
Title: Web application using EJB.

Objective: 1) Understanding basic concepts of Java beans.
2) Understand the basic concept ~~&~~ functionalities of JSP, HTML.
3) Having the knowledge of JBOSS server to deploy web application.

Problem Statement: Design, Develop & Deploy web application using EJB.

Software: 1) Ubuntu 64 bit / Windows 7
2) JDK 7
3) EJB 3.0
4) Eclipse luna
5) JBoss Application Server.

Theory:
J2EE application container contains the components that can by used by the clients for executing the business logic. These components are known as Enterprise Java Beans. (EJB).
J2EE platform has component based architecture to provide multi-tired, distributed and highly transactional features to enterprise level applications.
EJB mainly contains the business logic and business data. EJB component is an EJB class. It is a java class written by EJB developer & this class implements business logic. EJB 3.0 is being a large shift from EJB 2.0 and makes development of EJB based applications relatively easy.



* Features of EJBs:

1) Client Communication: The client, which is often a user interface, must be able to call the methods of objects on the application server via agreed-upon protocols.

2) State Management:- Some operations, for ex. when updating data, must occur as a unit of work. If one update fails, they all should fail.

3) Database Connection Management:- An application server must connect to a database, often using pools of db connections for optimizing resources.

4) User Aunthetication & Role-Based Aunthentication:- Users of an application must often log in for security purposes.

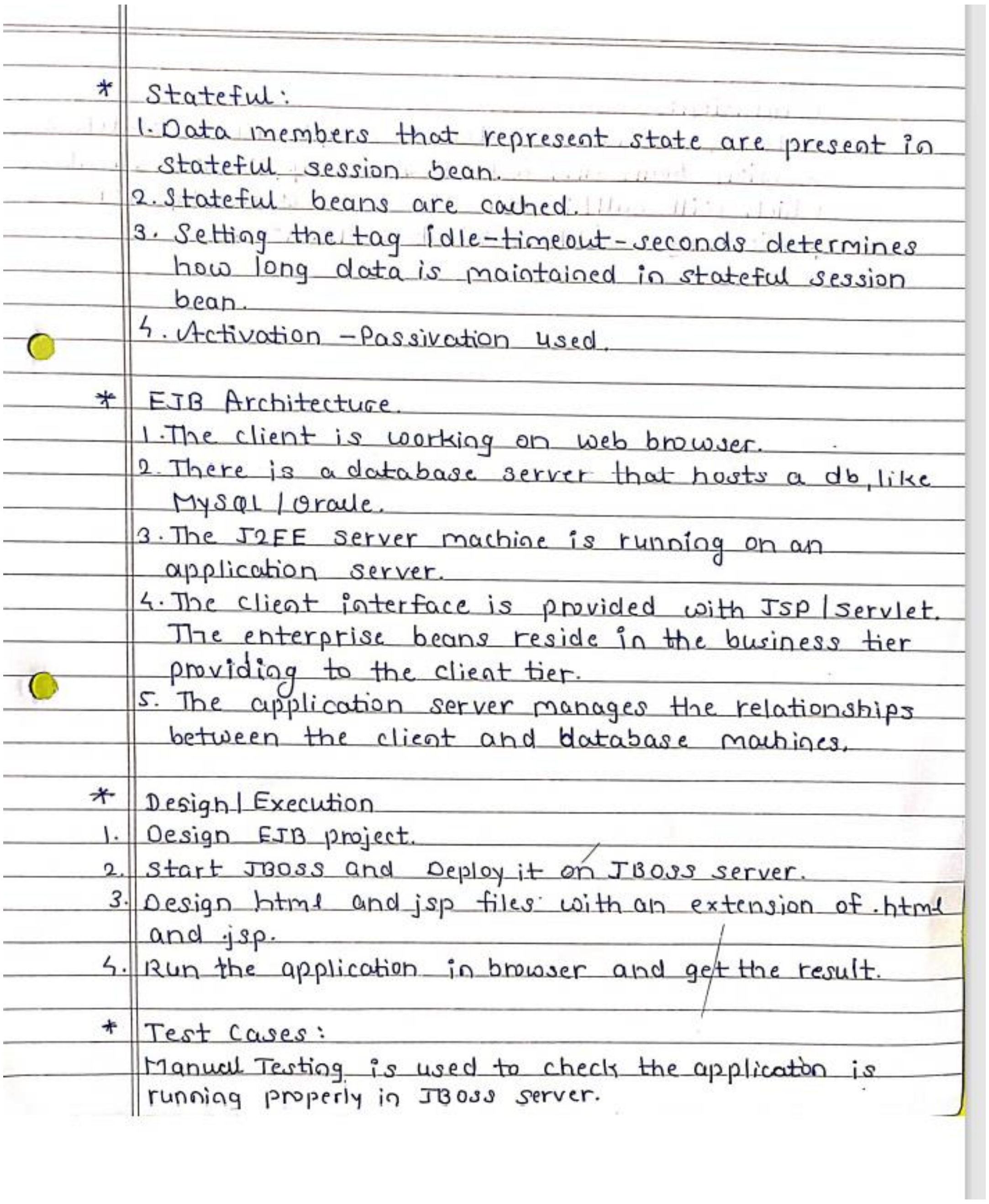
5) Application Server Administration:- Application servers must be administered. For ex. they need to be monitored and tuned.

* Types of Enterprise Java Beans (EJB):

1. Session Beans.
2. Entity Beans.
3. Message driven Beans.

* Stateless EJB:

1. Normally data members are not put in stateless session bean.
2. Stateless beans are pooled.
3. No effort for keeping client specific data.
4. No Activation / Passivation in stateless session bean



* Stateful:

1. Data members that represent state are present in stateful session bean.
2. Stateful beans are cached.
3. Setting the tag idle-timeout-seconds determines how long data is maintained in stateful session bean.
4. Activation -Passivation used.

* EJB Architecture.

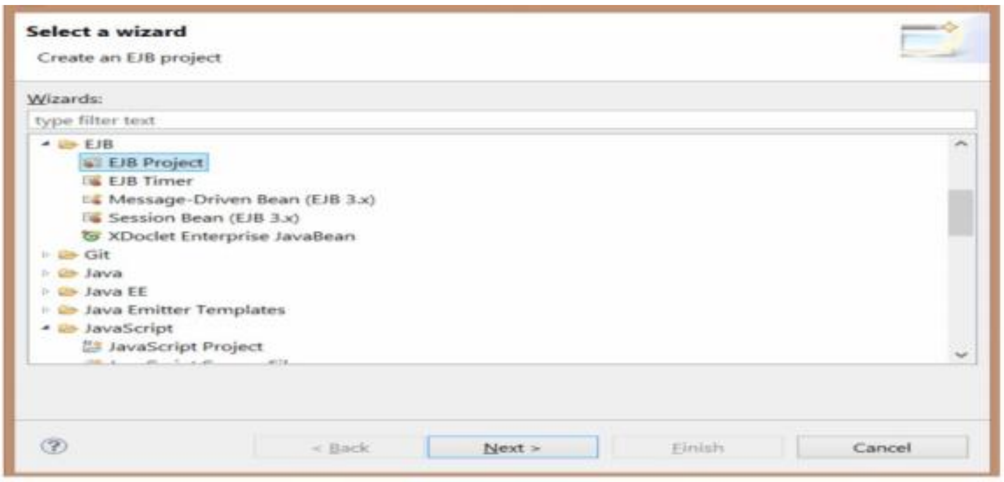
1. The client is working on web browser.
2. There is a database server that hosts a db, like MySQL / Oracle.
3. The J2EE server machine is running on an application server.
4. The client interface is provided with JSP / Servlet. The enterprise beans reside in the business tier providing to the client tier.
5. The application server manages the relationships between the client and database machines.

* Design / Execution

1. Design EJB project.
2. Start JBOSS and Deploy it on JBOSS server.
3. Design html and jsp files with an extension of .html and jsp.
4. Run the application in browser and get the result.

* Test Cases:

Manual Testing is used to check the application is running properly in JBoss server.

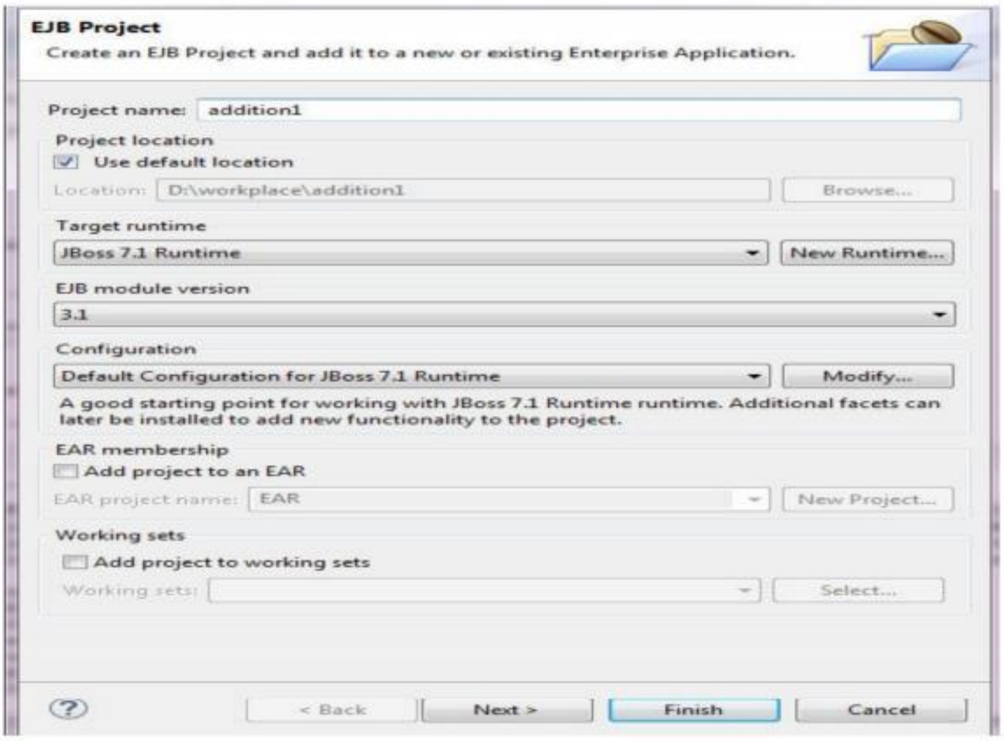

Select a wizard

Create an EJB project

Wizards:

type filter text

- EJB
  - EJB Project
  - EJB Timer
  - Message-Driven Bean (EJB 3.x)
  - Session Bean (EJB 3.x)
  - XDoclet Enterprise JavaBean
- Git
- Java
- Java EE
- Java Emitter Templates
- JavaScript
  - JavaScript Project


EJB Project

Create an EJB Project and add it to a new or existing Enterprise Application.

Project name: addition1

Project location

☑ Use default location

Location: D:\workplace\addition1

Target runtime

JBoss 7.1 Runtime

EJB module version

3.1

Configuration

Default Configuration for JBoss 7.1 Runtime

A good starting point for working with JBoss 7.1 Runtime runtime. Additional facets can later be installed to add new functionality to the project.

EAR membership

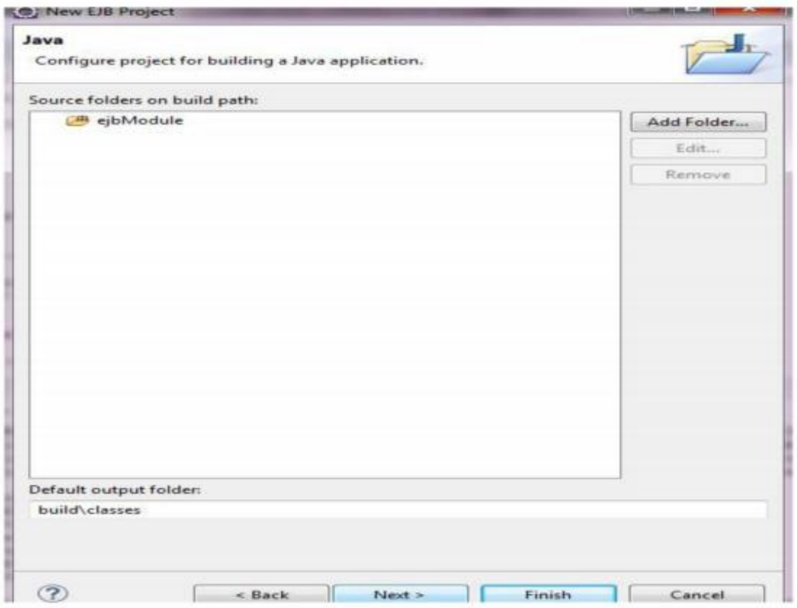
☐ Add project to an EAR

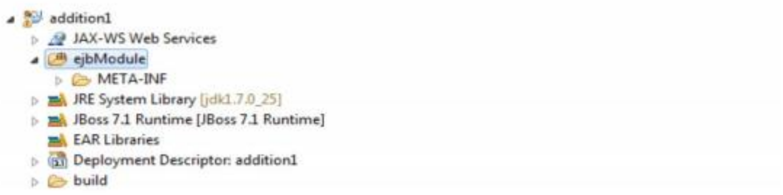
EAR project name: EAR

Working sets

☐ Add project to working sets

Working sets:

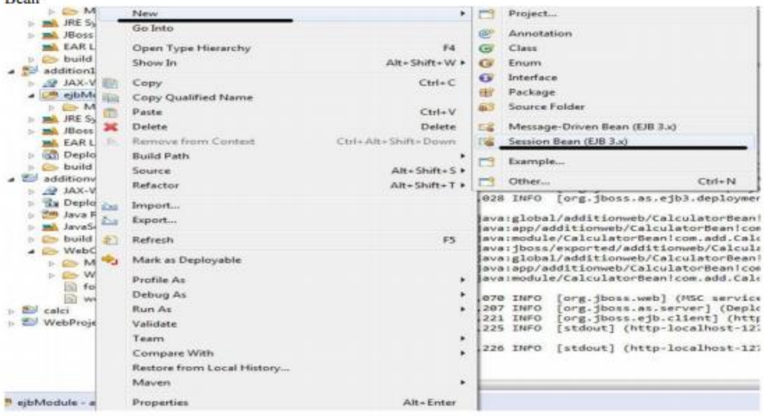



addition1
JAX-WS Web Services
ejbModule
META-INF
JRE System Library [jdk1.7.0_25]
JBoss 7.1 Runtime [JBoss 7.1 Runtime]
EAR Libraries
Deployment Descriptor: addition1
build

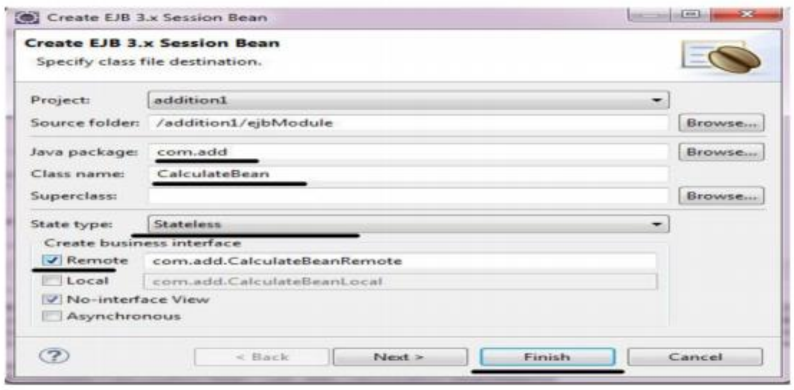
Step 2 :

Now create Stateless session bean with its remote interface. Expand project –> expande ejbModule –> Right click Session Bean –> New –> Session

Bean



```
028 INFO  [org.jboss.as.ejb3.deploymer

ava:global/additionweb/CalculatorBean!
ava:app/additionweb/CalculatorBean!com
ava:module/CalculatorBean!com.add.Calc
ava:jboss/exported/additionweb/Calcula
ava:global/additionweb/CalculatorBean!
ava:app/additionweb/CalculatorBean!com
ava:module/CalculatorBean!com.add.Calc

070 INFO  [org.jboss.web] (MSC service
207 INFO  [org.jboss.as.server] (Deplc
221 INFO  [org.jboss.ejb.client] (http
225 INFO  [stdout] (http-localhost-12

226 INFO  [stdout] (http-localhost-12
```

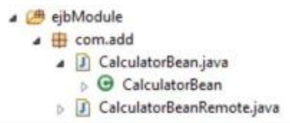


Create EJB 3.x Session Bean
Specify class file destination.

Project:	addition1
Source folder:	/addition1/ejbModule
Java package:	com.add
Class name:	CalculateBean
Superclass:	
State type:	Stateless

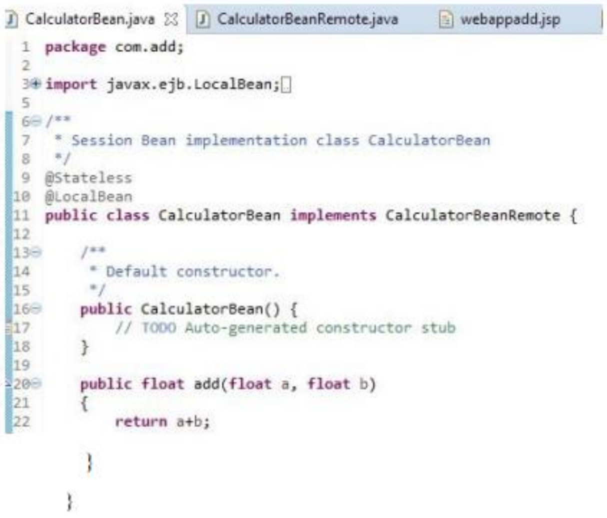
Create business interface
[✓] Remote com.add.CalculateBeanRemote
[ ] Local com.add.CalculateBeanLocal
[✓] No-interface View
[ ] Asynchronous

In ejbModule 2 java files are going to create after Finish button.



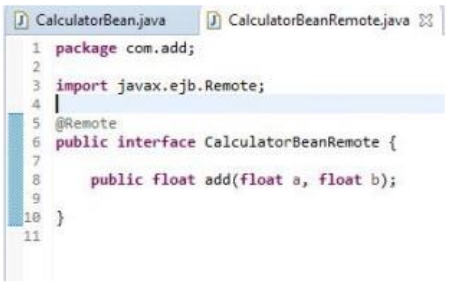
```
ejbModule
  com.add
    CalculatorBean.java
      CalculatorBean
    CalculatorBeanRemote.java
```

Write following code in CalculatorBean.java



```java
package com.add;

import javax.ejb.LocalBean;

/**
 * Session Bean implementation class CalculatorBean
 */
@Stateless
@LocalBean
public class CalculatorBean implements CalculatorBeanRemote {

    /**
     * Default constructor.
     */
    public CalculatorBean() {
        // TODO Auto-generated constructor stub
    }

    public float add(float a, float b)
    {
        return a+b;

     }

   }
```

Write Following code in CalculatorBeanRemote.java



```java
package com.add;

import javax.ejb.Remote;

@Remote
public interface CalculatorBeanRemote {

    public float add(float a, float b);

}

```

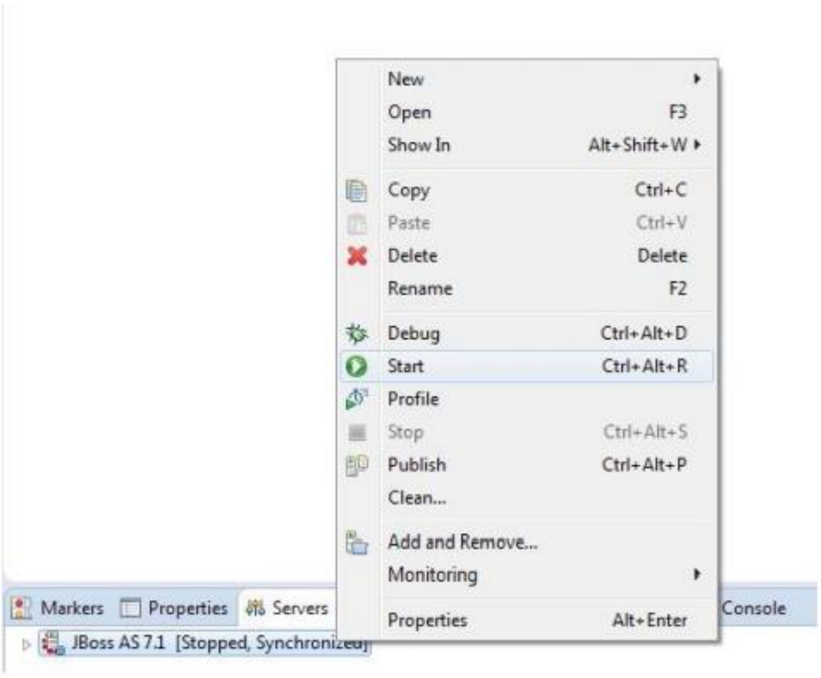
Step 3:

Deploying the project :

Now we need to deploy the our EJB "addition" on server. Follow the steps mentioned bellow to deploy this project on server.

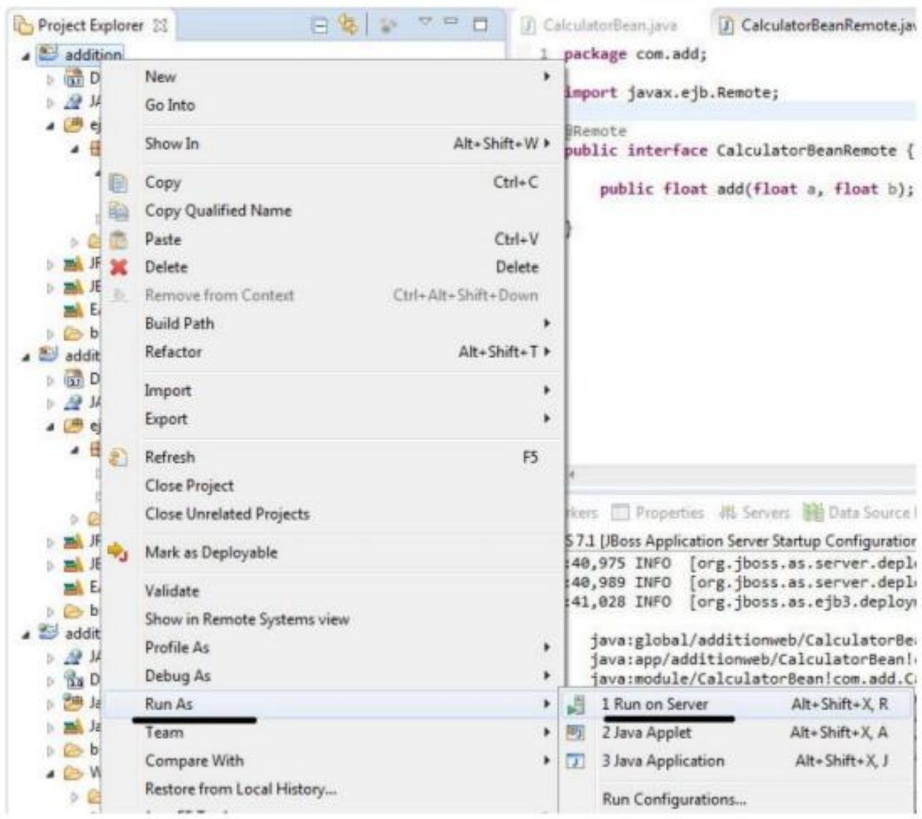
Strat the server

Right click on "JBoss 7.1 Runtime Server" from Servers view and click on Start.



Step 4:

Now next step Go to Project-> addition -> right click -> run-> Run on server

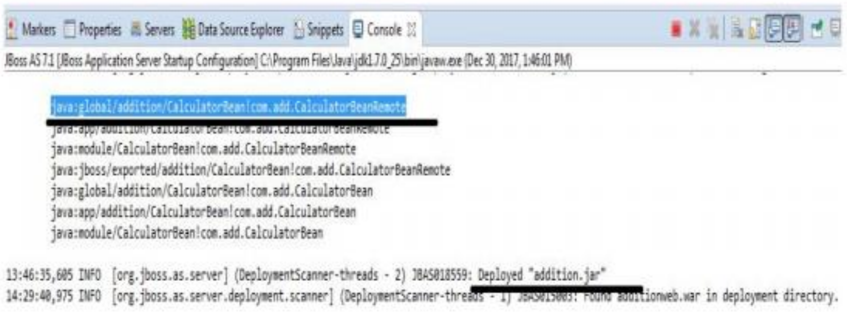


```
package com.add;

import javax.ejb.Remote;

@Remote
public interface CalculatorBeanRemote {

    public float add(float a, float b);
}
```

```
S 7.1 [JBoss Application Server Startup Configuratior
:40,975 INFO  [org.jboss.as.server.depl
:40,989 INFO  [org.jboss.as.server.depl
:41,028 INFO  [org.jboss.as.ejb3.deploy

    java:global/additionweb/CalculatorBe
    java:app/additionweb/CalculatorBean!
    java:module/CalculatorBean!com.add.C
```

Step 5:

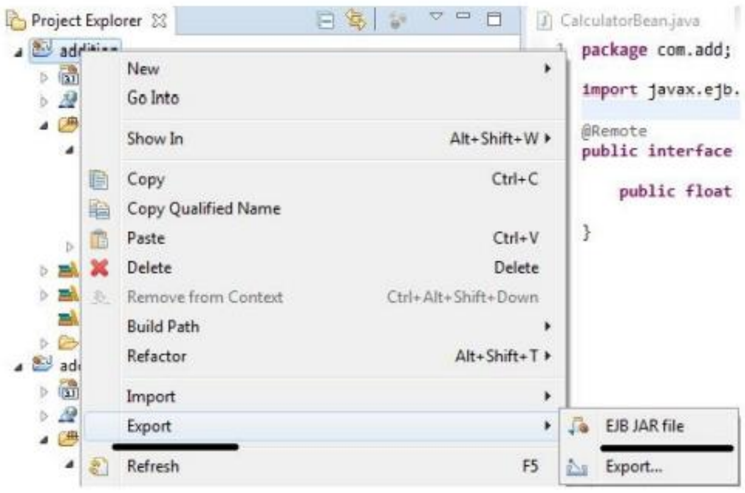
After running the program you can see following message on console



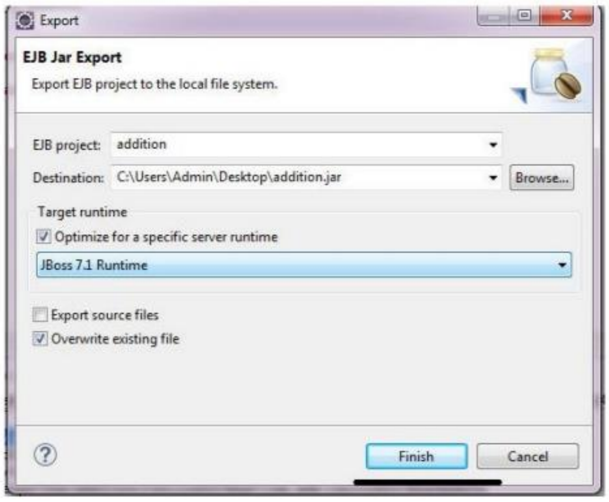
```
JBoss AS 7.1 [JBoss Application Server Startup Configuration] C:\Program Files\Java\jdk1.7.0_25\bin\javaw.exe (Dec 30, 2017, 1:46:01 PM)

        java:global/addition/CalculatorBean!com.add.CalculatorBeanRemote
        java:app/addition/CalculatorBean!com.add.CalculatorBeanRemote
        java:module/CalculatorBean!com.add.CalculatorBeanRemote
        java:jboss/exported/addition/CalculatorBean!com.add.CalculatorBeanRemote
        java:global/addition/CalculatorBean!com.add.CalculatorBean
        java:app/addition/CalculatorBean!com.add.CalculatorBean
        java:module/CalculatorBean!com.add.CalculatorBean

13:46:35,605 INFO  [org.jboss.as.server] (DeploymentScanner-threads - 2) JBAS018559: Deployed "addition.jar"
14:29:40,975 INFO  [org.jboss.as.server.deployment.scanner] (DeploymentScanner-threads - 1) JBAS015003: Found additionweb.war in deployment directory.
```

Step 6:

Once this jar file is deployed to server now export EJB jar file save it in desktop -> Finish.



```
package com.add;

import javax.ejb.

@Remote
public interface

    public float

}
```


Export

EJB Jar Export

Export EJB project to the local file system.

EJB project: addition

Destination: C:\Users\Admin\Desktop\addition.jar

Target runtime

[✓] Optimize for a specific server runtime

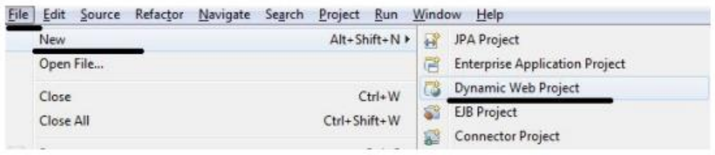
JBoss 7.1 Runtime

[ ] Export source files

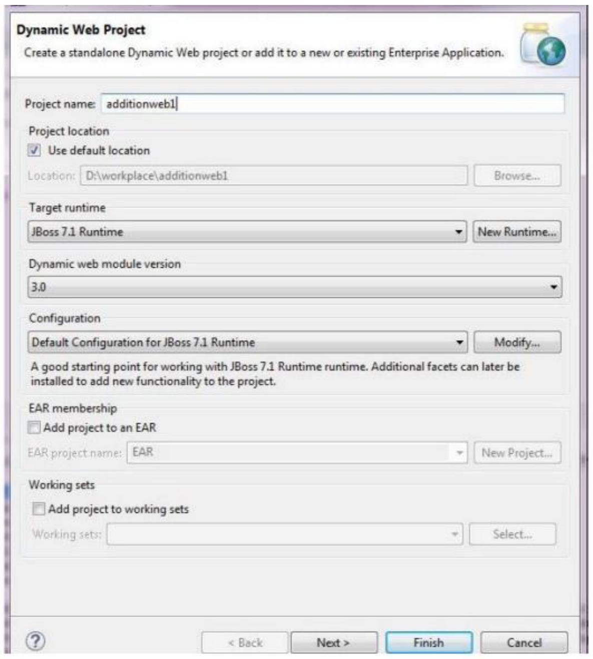
[✓] Overwrite existing file

Step 7:

Now create another project



Write project name-> as additionweb -> finish


Dynamic Web Project

Create a standalone Dynamic Web project or add it to a new or existing Enterprise Application.

Project name: additionweb1

Project location

[✓] Use default location

Location: D:\workplace\additionweb1

Target runtime

JBoss 7.1 Runtime

Dynamic web module version

3.0

Configuration

Default Configuration for JBoss 7.1 Runtime

A good starting point for working with JBoss 7.1 Runtime runtime. Additional facets can later be installed to add new functionality to the project.

EAR membership

[ ] Add project to an EAR

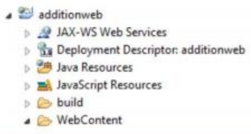
EAR project name: EAR

Working sets

[ ] Add project to working sets

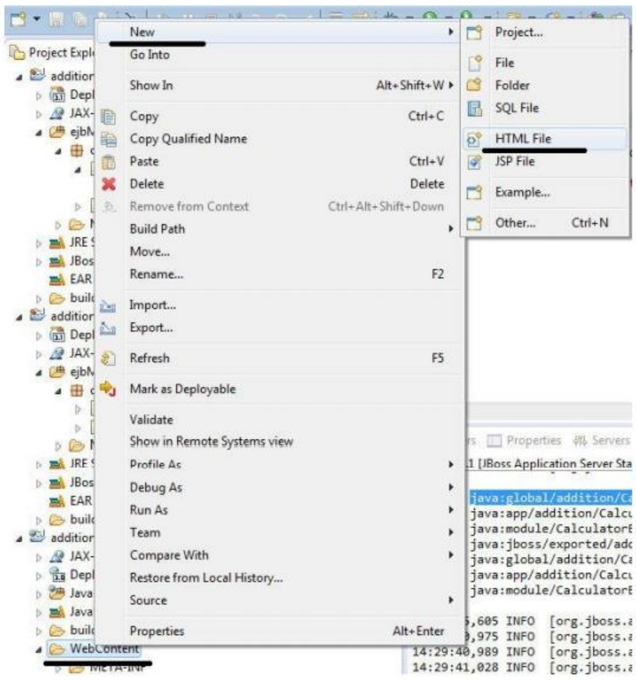
Working sets:

Get a file structure as follow



Step 8:

WebContent -> right click->new -> html page

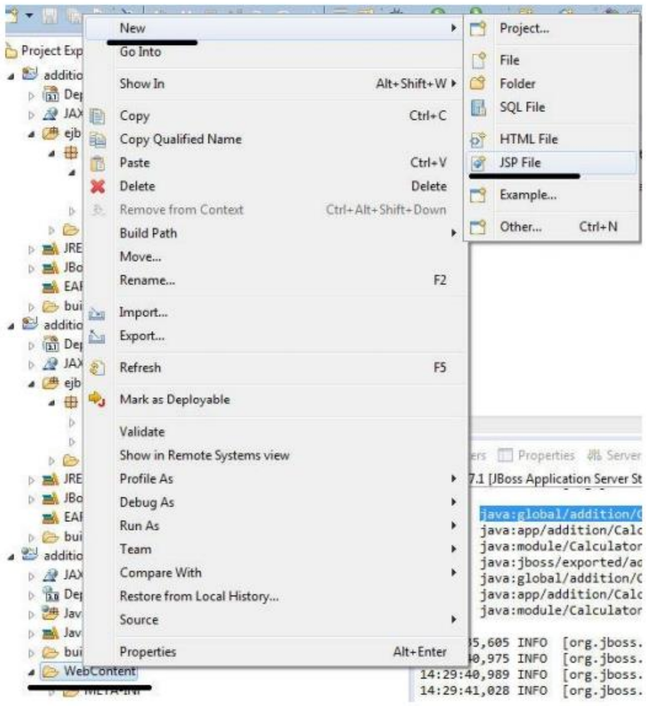


```
java:global/addition/C
java:app/addition/Calcu
java:module/CalculatorE
java:jboss/exported/add
java:global/addition/Ca
java:app/addition/Calcu
java:module/CalculatorE

,605 INFO  [org.jboss.a
,975 INFO  [org.jboss.a
14:29:40,989 INFO  [org.jboss.a
14:29:41,028 INFO  [org.jboss.a
```

Write file name -> form.html->Finish

Step 9:

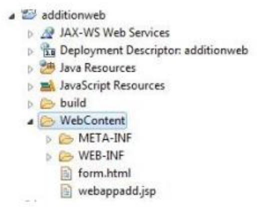
WebContent -> right click->new -> jsp page



```
java:global/addition/C
java:app/addition/Calc
java:module/Calculator
java:jboss/exported/ac
java:global/addition/C
java:app/addition/Calc
java:module/Calculator

5,605 INFO  [org.jboss.
0,975 INFO  [org.jboss.
14:29:40,989 INFO  [org.jboss.
14:29:41,028 INFO  [org.jboss.
```

Write file name -> webappadd.jsp->Finish

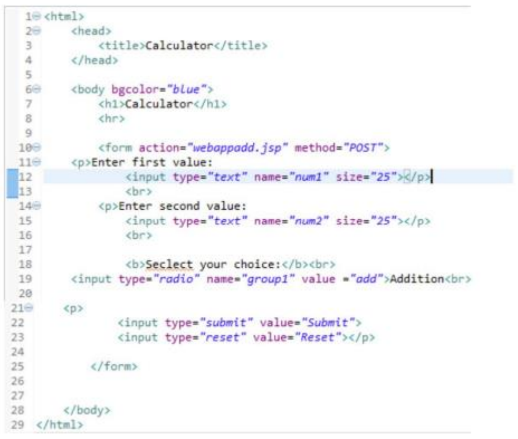
Get the file structure in project window as follows



```
additionweb
    JAX-WS Web Services
    Deployment Descriptor: additionweb
    Java Resources
    JavaScript Resources
    build
    WebContent
        META-INF
        WEB-INF
        form.html
        webappadd.jsp
```

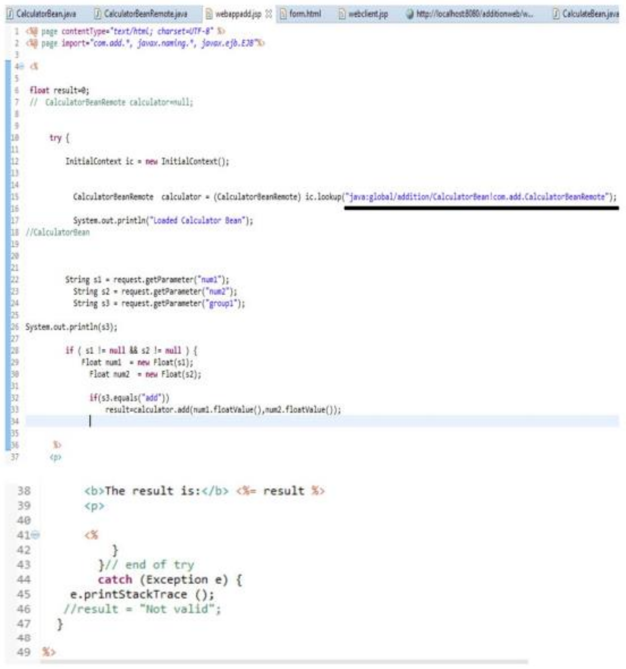
Write the following code in form.html

//form.html



```
<html>
    <head>
        <title>Calculator</title>
    </head>

    <body bgcolor="blue">
        <h1>Calculator</h1>
        <hr>

        <form action="webappadd.jsp" method="POST">
    <p>Enter first value:
            <input type="text" name="num1" size="25"></p>
            <br>
        <p>Enter second value:
            <input type="text" name="num2" size="25"></p>
            <br>

            <b>Seclect your choice:</b><br>
    <input type="radio" name="group1" value ="add">Addition<br>

   <p>
          <input type="submit" value="Submit">
          <input type="reset" value="Reset"></p>

        </form>


   </body>
</html>
```

Write following code in webappadd.jsp



```
<%@ page contentType="text/html; charset=UTF-8" %>
<%@ page import="com.add.*, javax.naming.*, javax.ejb.EJB"%>

<%

 float result=0;
 //  CalculatorBeanRemote calculator=null;


     try {

         InitialContext ic = new InitialContext();


          CalculatorBeanRemote  calculator = (CalculatorBeanRemote) ic.lookup("java:global/addition/CalculatorBean!com.add.CalculatorBeanRemote");

          System.out.println("Loaded Calculator Bean");
//CalculatorBean



         String s1 = request.getParameter("num1");
          String s2 = request.getParameter("num2");
          String s3 = request.getParameter("group1");

System.out.println(s3);

         if ( s1 != null && s2 != null ) {
             Float num1  = new Float(s1);
               Float num2  = new Float(s2);

               if(s3.equals("add"))
                   result=calculator.add(num1.floatValue(),num2.floatValue());



      %>
      <p>

        <b>The result is:</b> <%= result %>
        <p>

        <%
            }
          }// end of try
          catch (Exception e) {
      e.printStackTrace ();
     //result = "Not valid";
   }

%>
```

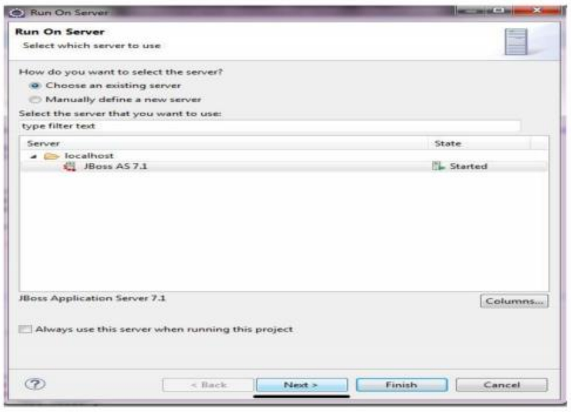
Step 10 :

Copy the url from step 5 and add that url in wepappadd code as given above.

Step 11 :

Running the application :

Right click on project addition-> run as -> run on server



Run On Server

Run On Server

Select which server to use

How do you want to select the server?

(●) Choose an existing server

( ) Manually define a new server

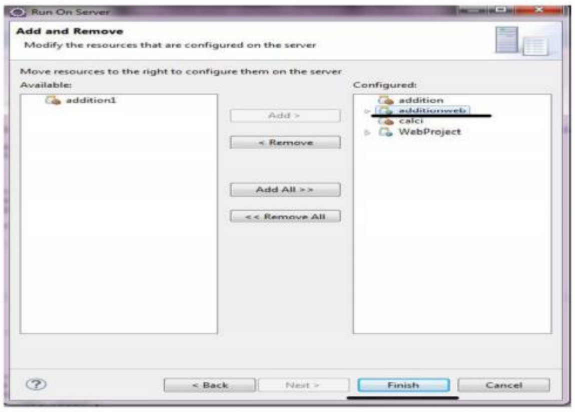
Select the server that you want to use:

type filter text

Server	State
localhost	
JBoss AS 7.1	Started

JBoss Application Server 7.1

[ ] Always use this server when running this project



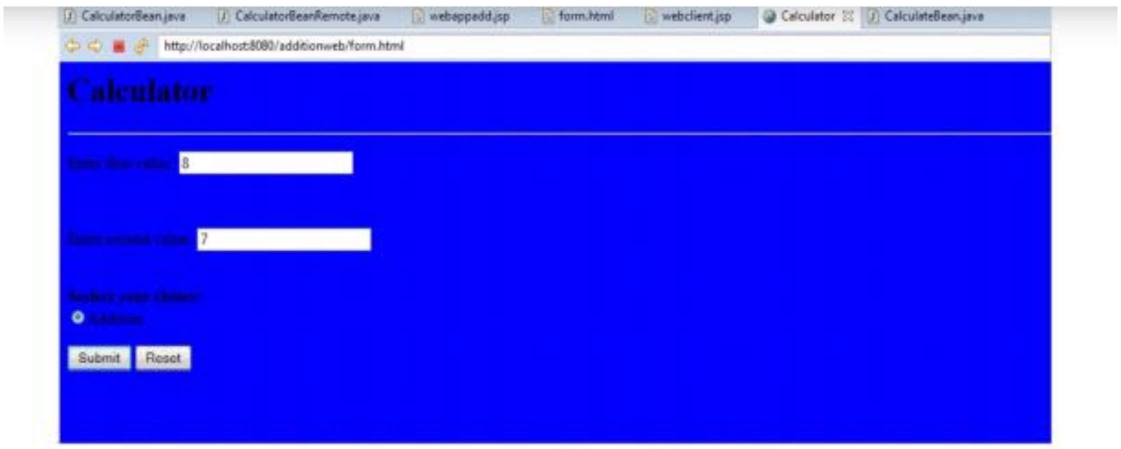
Run On Server

Add and Remove

Modify the resources that are configured on the server

Move resources to the right to configure them on the server

Available:	Configured:
addition1	addition
	additionweb
	calci
	WebProject



http://localhost:8080/additionweb/form.html

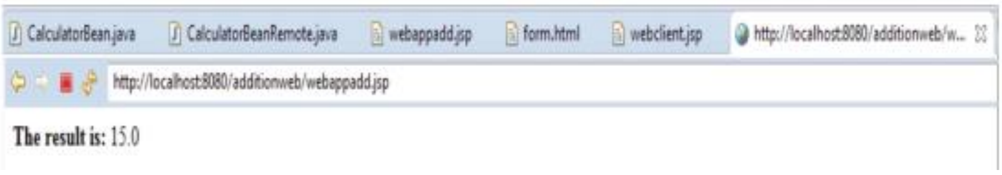
Calculator

Enter first value: 8

Enter second value: 7

Select your choice:

(●) Addition



CalculatorBean.java
CalculatorBeanRemote.java
webappadd.jsp
form.html
webclient.jsp
http://localhost:8080/additionweb/w...

http://localhost:8080/additionweb/webappadd.jsp

The result is: 15.0

Conclusion:

Hence, we have created a simple EJB 3 stateless session bean and a local Java application client which will call/invoke the bean to develop for performing addition of two numbers.