

Practical 1

Q.1 a) Create a simple calculator application using servlet.

CODE:

index.html

```
<html>

    <body>

        <form method=post action="CalcServlet">

            NO-1 <input type=text name="t1">

            NO-2 <input type=text name="t2"> <br> <br>

            <input type=submit value="+" name="btn">

            <input type=submit value="-" name="btn">

            <input type=submit value="*" name="btn">

            <input type=submit value="/" name="btn">

        </form>

    </body>

</html>
```

CalcServlet.java

```
import java.io.*;

import javax.servlet.*;

import javax.servlet.http.*;

public class CalcServlet extends HttpServlet

{
    public void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException

    {
        response.setContentType("text/html");

        PrintWriter out = response.getWriter();

        int a=Integer.parseInt(request.getParameter("t1"));
```

```
int b=Integer.parseInt(request.getParameter("t2"));
```

```
int c=0;
```

```
String op=request.getParameter("btn");
```

```
if (op.equals("+"))
```

```
    c=a+b;
```

```
else if (op.equals("-"))
```

```
    c=a-b;
```

```
else if (op.equals("*"))
```

```
    c=a*b;
```

```
else if (op.equals("/"))
```

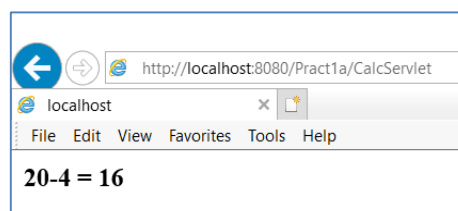
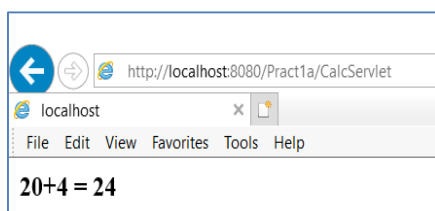
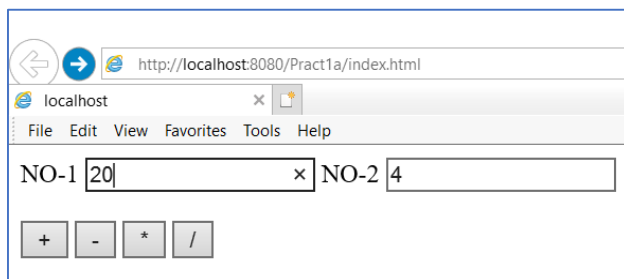
```
    c=a/b;
```

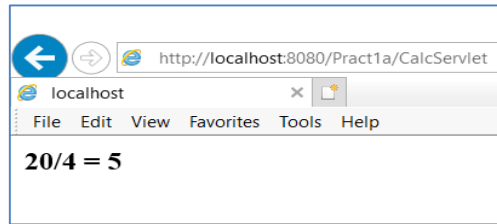
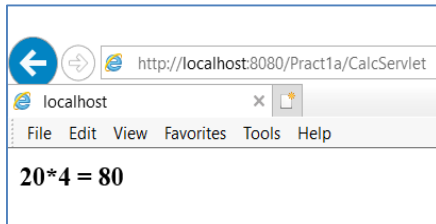
```
out.println("<b>" +a+op+b+" = "+c+"<b>");
```

```
}
```

```
}
```

OUTPUT:





Q.1 b) Create a servlet for a login page. If the username and password are correct then it says message “Hello ” else a message “login failed”.

CODE:

index.html

```
<html>

<body>

    <form action="LoginServlet" method="post">

        UserName : <input type="text" name="uname"><br>

        Password : <input type="password" name="pw"> <br>

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

    </form>

</body>

</html>
```

LoginServlet.java

```
import java.io.*;

import javax.servlet.*;

import javax.servlet.http.*;

public class LoginServlet extends HttpServlet {

    public void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException

    {

        response.setContentType("text/html");

        PrintWriter out = response.getWriter();

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

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

        String msg="";
```

```
if (username .equals("admin") && password.equals("admin123"))
```

```
    msg="Hello "+username;
```

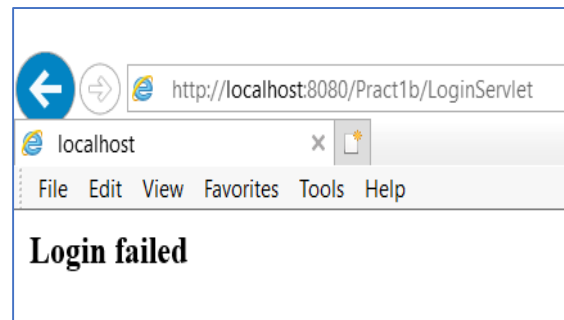
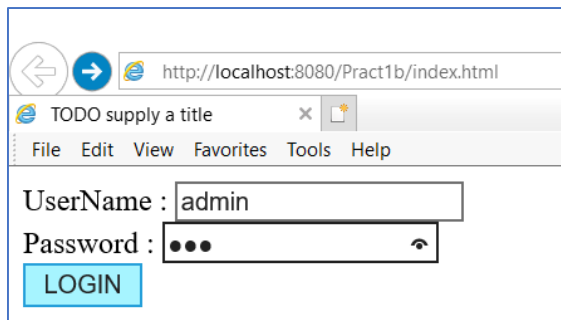
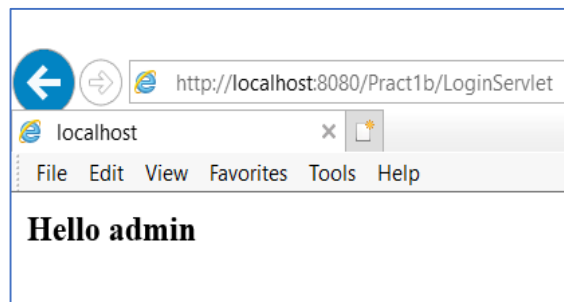
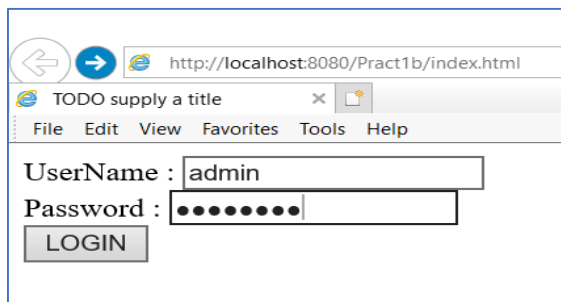
```
else
```

```
    msg="Login failed";
```

```
out.println("<b>"+msg+"<b>");
```

```
}
```

```
}
```

OUTPUT:

Q.1 c) Create a registration servlet in Java using JDBC. Accept the details such as Username, Password, Email, and Country from the user using HTML Form and store the registration details in the database.

Code:

MySql Command from mysql software:-

1. Select **services** -> expand **databases** -> right click on **MySQL server at localhost:3306[disconnected]** -> click on **connect** -> enter password (**tiger**) -> OK
2. Again right click on **MySQL server at localhost:3306** -> select **Create database** -> enter database name and select the check box to grant permission.
3. Right click on **Table** under your database
4. Enter table name **user** by replacing untitled. Click on **Add column**, name -> **username**, type-> **varchar**, size-> **20**, select checkbox of primary key, again click on **Add column** password **varchar** size **20**, again click on **Add column** emailid **varchar(20)**, again click **Add column** country **varchar 10**;
5. add mysql-connector to library folder of the current application

index.html

<html>

<body>

<form action="RegistrationServlet" method="post">

User name : <input type="text" name="uname">

Password : <input type="password" name="pw">

Email Id : <input type="text" name="email">

Country : <select name="coun">

<option>select...

<option> India

<option> Bangladesh

<option> Bhutan

<option> Canada

</select>

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

</form>

</html>

RegistrationServlet.java

```
import java.io.*;

import java.sql.*;

import javax.servlet.*;

import javax.servlet.http.*;

public class RegistrationServlet extends HttpServlet

{
    public void doPost(HttpServletRequest request, HttpServletResponse response) throws
IOException, ServletException

    {
        Connection con=null;

        PreparedStatement ps=null;

        response.setContentType("text/html");

        PrintWriter out = response.getWriter();

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

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

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

        String country=request.getParameter("coun");

        try

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

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

            out.println("connection done successfully...");

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

            ps.setString(1,username);

            ps.setString(2,password);

            ps.setString(3,emailid);

            ps.setString(4,country);

            ps.execute();
```

```
        out.print("Data inserted successfully!!!!");

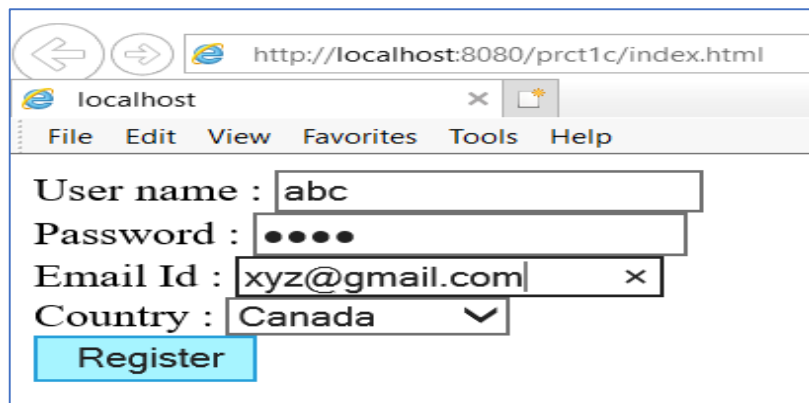
    }

    catch(Exception e) {    out.println(e);    }

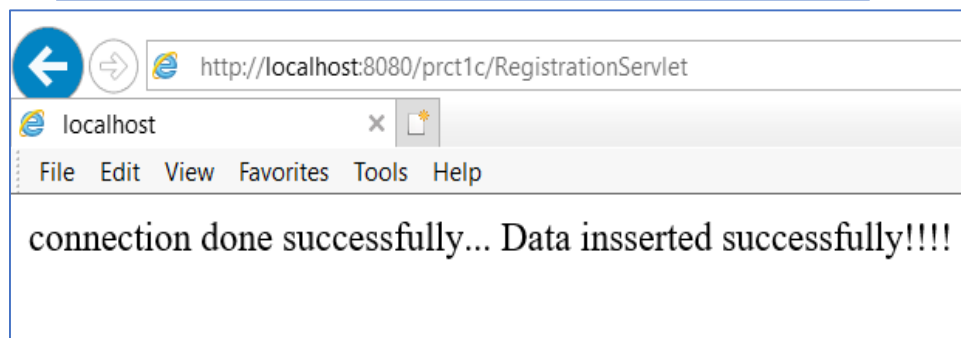
    out.println("<b>"+ "<b>");

}

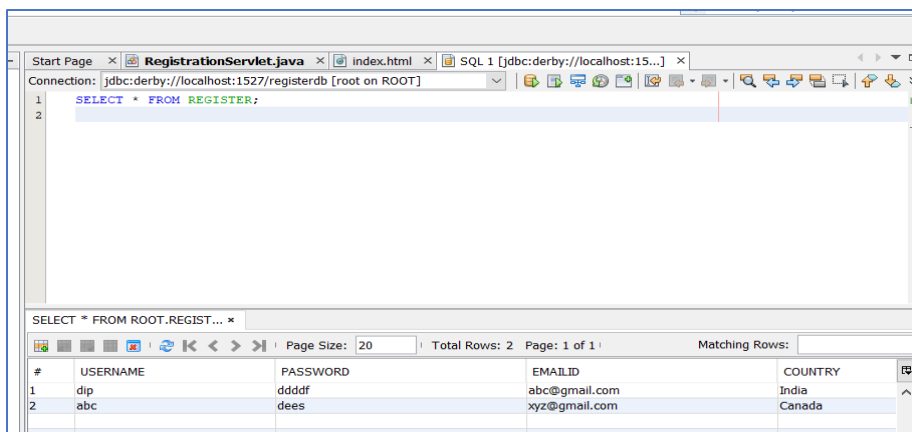
}
```

OUTPUT:

A screenshot of a web browser window showing a registration form. The address bar displays 'http://localhost:8080/prct1c/index.html'. The form includes fields for 'User name' (filled with 'abc'), 'Password' (masked with dots), 'Email Id' (filled with 'xyz@gmail.com'), and 'Country' (a dropdown menu showing 'Canada'). A blue 'Register' button is located below the form fields.



A screenshot of a web browser window showing a success message. The address bar displays 'http://localhost:8080/prct1c/RegistrationServlet'. The main content area displays the text 'connection done successfully... Data inserted successfully!!!!'.



A screenshot of a database management tool window showing a query result. The query is 'SELECT * FROM ROOT.REGIST...'. The result is displayed in a table with 5 columns: #, USERNAME, PASSWORD, EMAILID, and COUNTRY. There are 2 rows of data.

#	USERNAME	PASSWORD	EMAILID	COUNTRY
1	dip	dddf	abc@gmail.com	India
2	abc	dees	xyz@gmail.com	Canada

PRACTICAL 2

Q.2 a) Using Request Dispatcher Interface create a Servlet which will validate the password entered by the user, if the user has entered "Servlet" as password, then he will be forwarded to Welcome Servlet else the user will stay on the index.html page and an error message will be displayed.

CODE:

Index.html

```
<html>

  <body>

    <form method="post" action="ValidateServlet">

      User Name: <input type="text" name="un"><br>

      Password: <input type="password" name="pw"><br>

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

    </form>

  </body>

</html>
```

ValidateServlet.java

```
import java.io.*;

import javax.servlet.*;

import javax.servlet.http.*;

public class ValidateServlet extends HttpServlet

{
    public void doPost(HttpServletRequest req, HttpServletResponse res)throws IOException,
    ServletException
```

```
{
    res.setContentType("text/html");

    PrintWriter out=res.getWriter();

    String username=req.getParameter("un");

    String password=req.getParameter("pw");

    if(password.equals("Servlet"))

    {

        req.setAttribute("s1username",username);

        req.setAttribute("s1password",password);

        RequestDispatcher rd= req.getRequestDispatcher("/WelcomeServlet");

        rd.forward(req, res);

    }

    else

    {

        out.print("Incorrect password");

        RequestDispatcher rd= req.getRequestDispatcher("/index.html");

        rd.include(req, res);

    }

}
```

WelcomeServlet.java

```
import java.io.*;

import javax.servlet.*;

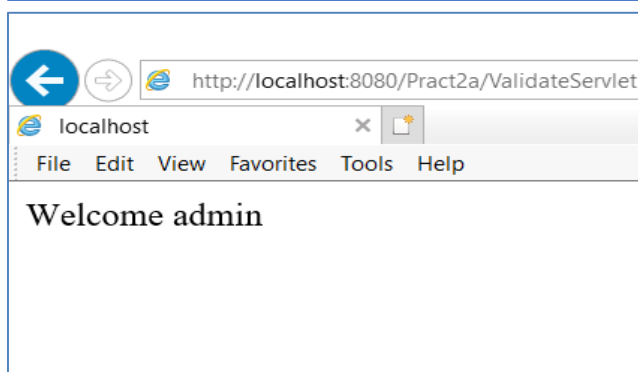
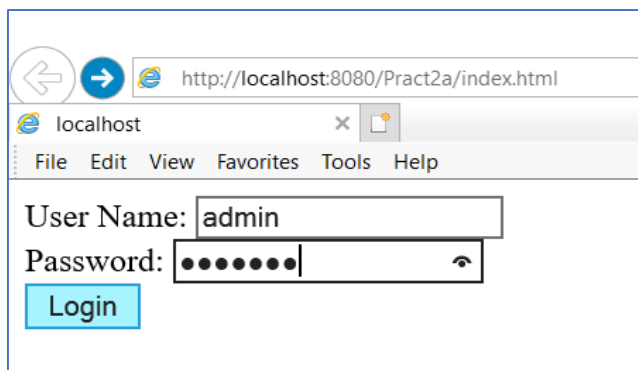
import javax.servlet.http.*;

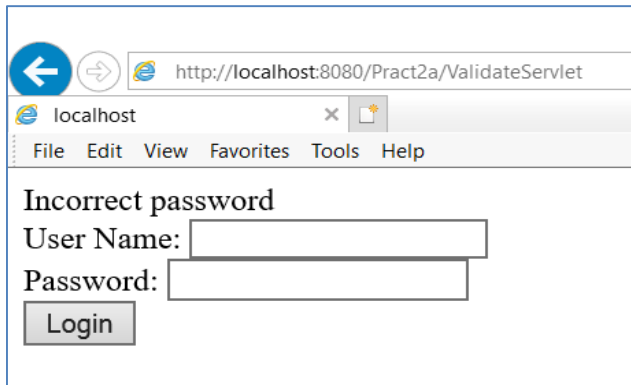
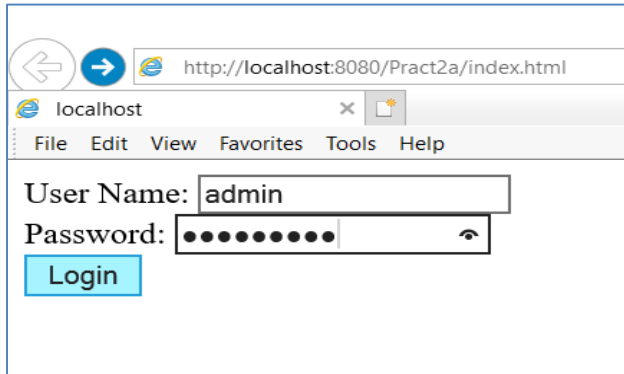
public class WelcomeServlet extends HttpServlet

{

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

```
{  
  
    res.setContentType("text/html");  
  
    try (PrintWriter out = res.getWriter()) {  
  
        String s2username = (String)req.getAttribute("s1username");  
  
        String s2password = (String)req.getAttribute("s2password");  
  
        out.println("Welcome "+s2username);  
  
    }  
}  
  
}
```

OUTPUT:



Q.2 b) Create a servlet that uses Cookies to store the number of times a user has visited servlet.

CODE:

CookieServlet.java

/*

* To change this license header, choose License Headers in Project Properties.

* To change this template file, choose Tools | Templates

* and open the template in the editor.

*/

package pract2;

import java.io.*;

import javax.servlet.*;

import javax.servlet.http.*;

public class CookieServlet extends HttpServlet

{

```
private int i=1;

public void doGet(HttpServletRequest request, HttpServletResponse response)

    throws IOException, ServletException

{
    response.setContentType("text/html");

    PrintWriter out = response.getWriter();

    String k=String.valueOf(i);

    Cookie c = new Cookie("visit",k);

    response.addCookie(c);

    int j=Integer.parseInt(c.getValue());

    if(j==1)

    {

        out.println("This is the first time you are visiting this page");

    }

    else

    {

        synchronized(CookieServlet.this)

        {
            out.println("You visited this page "+i+" times");

        }

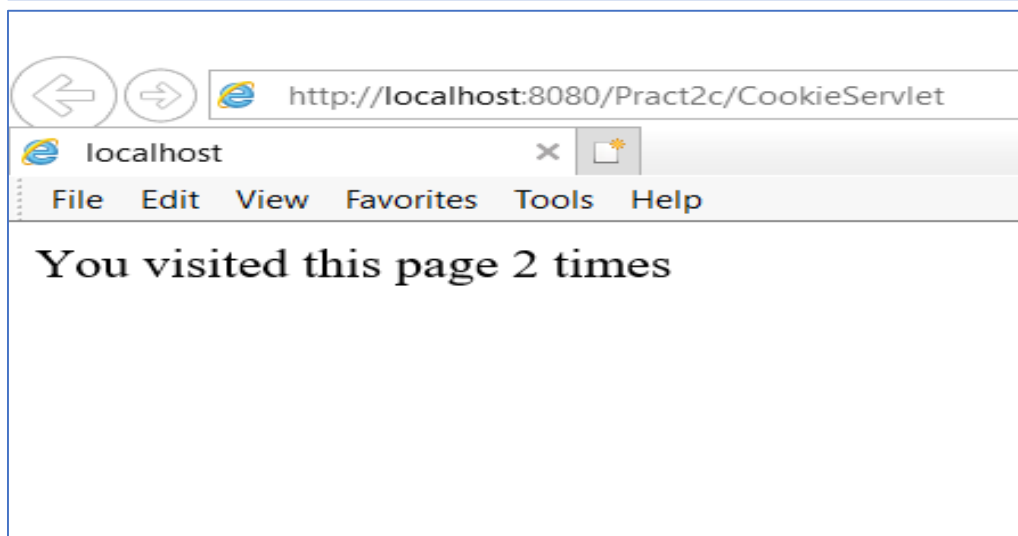
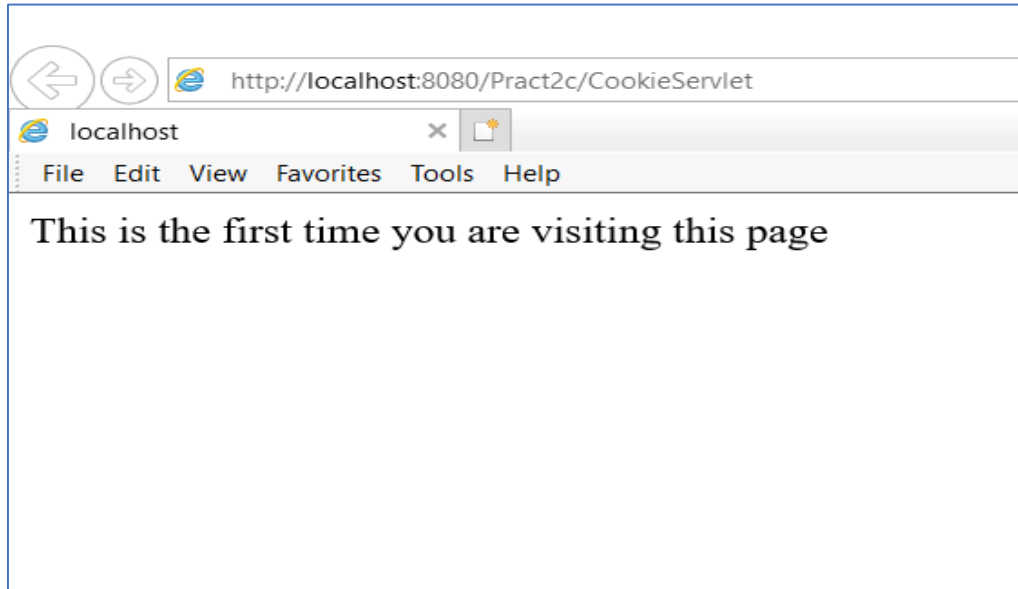
    }

    i++;

}

}
```

OUTPUT:



Q.2 c) Create a servlet demonstrating the use of session creation and destruction. Also check whether the user has visited this page first time or has visited earlier also using sessions.

CODE:

CalculationVisitServlet.java

```
package sessionapp;
```

```
/*
```

```
* To change this license header, choose License Headers in Project Properties.
```

```
* To change this template file, choose Tools | Templates
```

```
* and open the template in the editor.
```

```
*/
```

```
package pract2;
```

```
/*
```

```
* To change this license header, choose License Headers in Project Properties.
```

```
* To change this template file, choose Tools | Templates
```

```
* and open the template in the editor.
```

```
*/
```

```
import java.io.*;
```

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

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

```
public class HttpSessionServlet extends HttpServlet
```

```
{
```

```
    private int counter;
```

```
    public void doGet(HttpServletRequest request, HttpServletResponse response) throws  
ServletException, IOException
```

```
    {
```

```
        response.setContentType("text/html");
```

```
        PrintWriter out = response.getWriter();
```

```
        HttpSession session=request.getSession(true);
```

```
        if(session.isNew())
```

```
        {
```

```
            out.print("This is the first time you are visiting this page");
```

```
            ++counter;
```

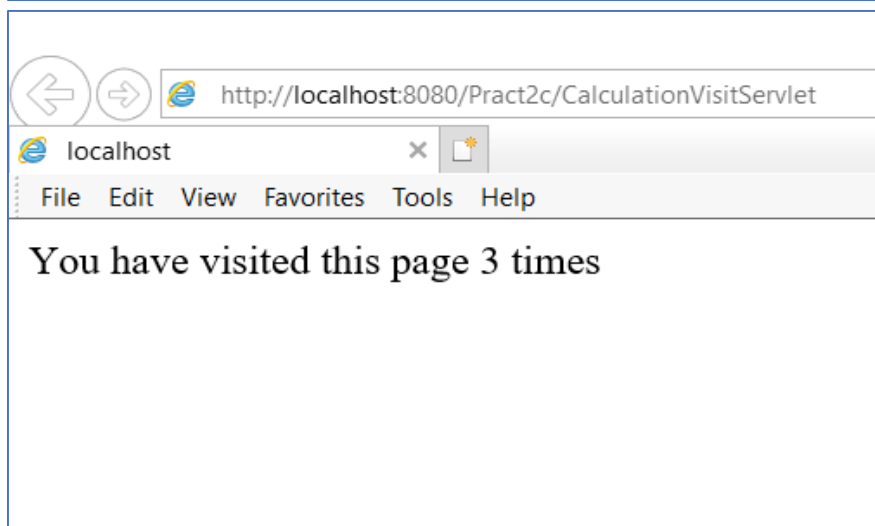
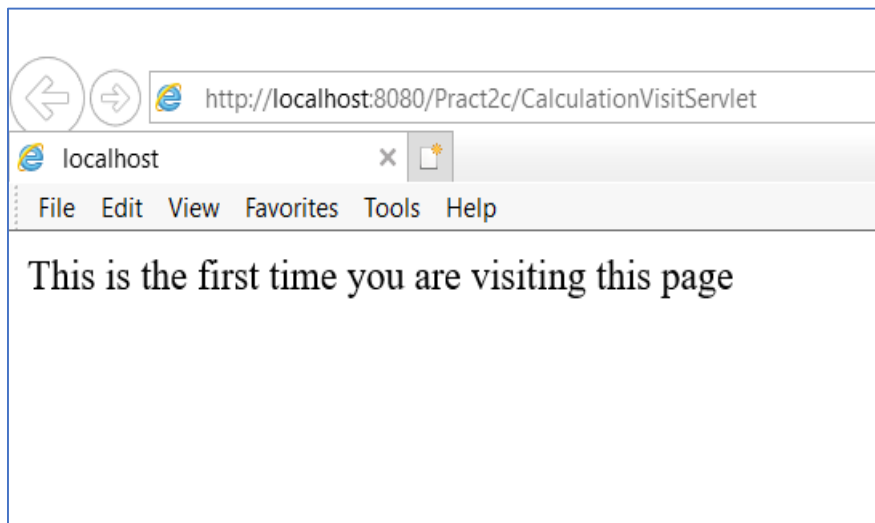
```
        }
```

```
        else
```

```
        {
```

```
synchronized(HttpSessionServlet.this)
```

```
{  
    if(counter==10)  
    {  
        session.invalidate();  
        counter=0;  
        request.getSession(false);  
    }  
    else  
        out.print("You have visited this page "+(++counter)+ " times");  
}  
}  
}
```

OUTPUT:

PRACTICAL 3

Q.3 a) Create a Servlet application to upload and download a file.

CODE:

Uploading a file

Index.html

```
<form action="FileUploadServlet" method="post" enctype="multipart/form-data">

Select File to Upload:<input type="file" name="file" id="file">

Destination <input type="text" value="/tmp" name="destination">

<br>

<input type="submit" value="Upload file" name="upload" id="upload">

</form>
```

FileUploadServlet.java

```
package fileservletapp;

import java.io.*;

import javax.servlet.*;

import javax.servlet.annotation.MultipartConfig;

import javax.servlet.http.*;

@MultipartConfig

public class FileUploadServlet extends HttpServlet {

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

    {

        res.setContentType("text/html");

        PrintWriter out = res.getWriter();
```

```
String path=req.getParameter("destination");

Part filePart=req.getPart("file");

String sfilePart=req.getPart("file").toString();

out.print("<br> filePart: "+sfilePart);

String filename=filePart.getSubmittedFileName().toString();

out.print("<br><br><hr> file name: "+filename);

OutputStream os=null;

InputStream is=null;

try {

    os=new FileOutputStream(new File(path+File.separator+filename));

    is=filePart.getInputStream();

    int read=0;

    byte[] b=new byte[1024];

    while ((read = is.read(b)) != -1) {

        os.write(b, 0, read);

    }

    out.println("<br>file uploaded sucessfully...!!!");

}

catch(FileNotFoundException e){out.print(e);}

} }
```

Downloading a file

Index.html

```
<body>

    <h1>File Download Application</h1>

    Click <a href="DownloadServlet?filename=SampleChapter.pdf">Sample Chapter</a>

    <br/><br/>
```

</body>

DownloadServlet.java

```
package filedownloadapp;
```

```
import java.io.*;
```

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

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

```
public class DownloadServlet extends HttpServlet
```

```
{    public void doGet(HttpServletRequest request, HttpServletResponse response)
```

```
        throws ServletException, IOException
```

```
{        response.setContentType("APPLICATION/OCTET-STREAM");
```

```
        String filename = request.getParameter("filename");
```

```
        ServletContext context = getServletContext();
```

```
        InputStream is = context.getResourceAsStream("/") + filename);
```

```
        ServletOutputStream os = response.getOutputStream();
```

```
        response.setHeader("Content-Disposition","attachment; filename=\"" + filename + "\"");
```

```
// if comment this statement then it will ask you about the editor with which you want to open the file
```

```
        int i;
```

```
        byte b[]=new byte[1024];
```

```
        while ((i=is.read(b)) != -1) {
```

```
            os.write(b);
```

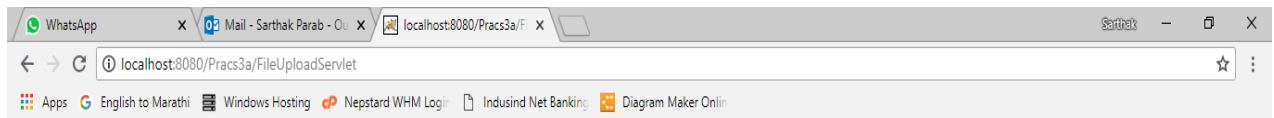
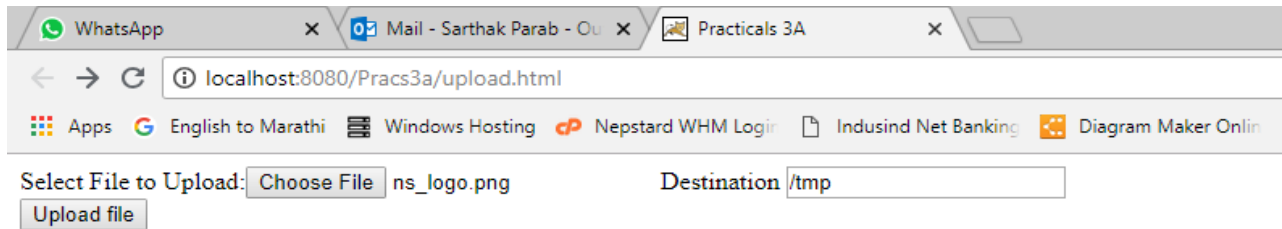
```
        }
```

```
        is.close();
```

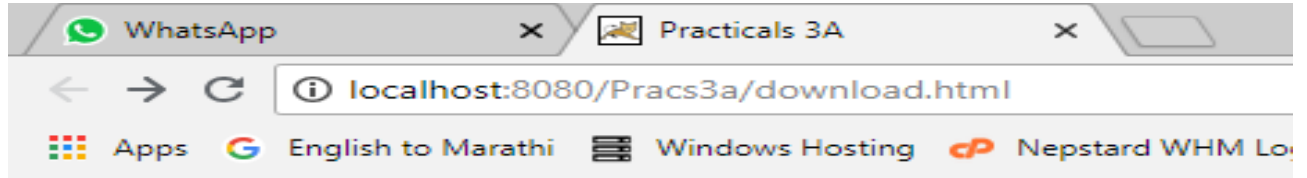
```
        os.close();
```

```
}
```

OUTPUT:



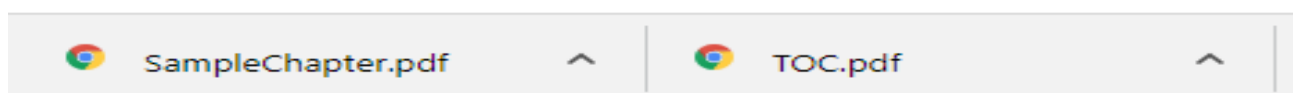
filePart: File name=ns_logo.png, StoreLocation=C:\Users\Sarthak\AppData\Roaming\NetBeans\8.0.2\config\GF_4.1\domain1\generated\jsp\Pracs3a\upload__665e6b4f_1665e440109_7#b_00000000.tmp, size=159983bytes, isFormField=false, FieldName=file



File Download Application

Click [Sample Chapter](#)

Click [Table Of Contents](#)



Q.3 b) Develop Simple Servlet Question Answer Application using Database.**Create a table in mysql**

- Click on 'Services' tab
- Create a database
- Database name: queansdb
- Table name: queans
- Fields:
 - queno integer primary key
 - question varchar 200
 - opt1 varchar 100
 - opt2 varchar 100
 - opt3 varchar 100
 - opt4 varchar 100
 - anskey varchar 1

Insert min 2 records

Right click on table-> click on 'view data' -> right click on empty dataset -> insert a record

> click on 'Add Row' -> OK

add mysql connector to Libray

- click on projects tab
- right click on libraries
- click on add jar
- browse the connector 'mysql-connector-java-5.1.23-bin'
in folder: C:\Program Files\NetBeans 8.0\ide\modules\ext

click on OK

CODE:**QueAnsDBServlet.java**

```
package dbapp;

import javax.servlet.*;

import javax.servlet.http.*;

import java.io.*;

import java.sql.*;

public class QueAnsDBServlet extends HttpServlet

{
```

public void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException

```
{  
  
    response.setContentType("text/html");  
  
    PrintWriter out = response.getWriter();  
  
    try  
  
    {  
  
        out.print("<html><body><br>");  
  
        out.println("<form method='post' action='Marks'>");  
  
        Class.forName("com.mysql.jdbc.Driver");  
  
        Connection con=DriverManager.getConnection("jdbc:mysql://localhost/queansdb","root","tiger");  
  
        Statement st = con.createStatement();  
  
        String sql="select * from queans";  
  
        ResultSet rs = st.executeQuery(sql);  
  
        int i=0;  
  
        out.print("<center>Online Exam</center>");  
  
        while(rs.next())  
  
        {  
  
            i++;  
  
            out.print("<br><br><hr>"+rs.getInt(1)+"  ");  
  
            out.print(rs.getString(2));  
  
            out.print("<br><input type=radio name="+i+" value=1>"+rs.getString(3));  
  
            out.print("<br><input type=radio name="+i+" value=2>"+rs.getString(4));  
  
            out.print("<br><input type=radio name="+i+" value=3>"+rs.getString(5));  
  
            out.print("<br><input type=radio name="+i+" value=4>"+rs.getString(6));  
  
            String ans="ans"+i;
```

```
        out.println("<br><input type=hidden name="+ans+" value="+rs.getString(7)+">");

    }

    out.println("<br><input type=hidden name=total value="+i+">");

    out.println("<input type=submit value=submit>");


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

    out.print("</body></html>");

}

catch(Exception e)

{

    out.println("ERROR "+e.getMessage());

}

}

}
```

Marks.java

```
package dbapp;

import javax.servlet.*;

import javax.servlet.http.*;

import java.io.*;

public class Marks extends HttpServlet

{

    public void doPost(HttpServletRequest request, HttpServletResponse response) throws

ServletException, IOException

    {

        response.setContentType("text/html");

        PrintWriter out = response.getWriter();
```



```
try

{   out.print("<html><body>");

    int total=Integer.parseInt(request.getParameter("total"));

    int marks=0;

    for(int i=1; i<=total; i++)

    {

        String sel=request.getParameter(new Integer(i).toString());

        String ans=request.getParameter("ans"+i);

        if (sel.equals(ans)) marks++;

    }

    out.println("Total Marks : "+marks);

    out.print("</body></html>");

}

catch(Exception e)

{

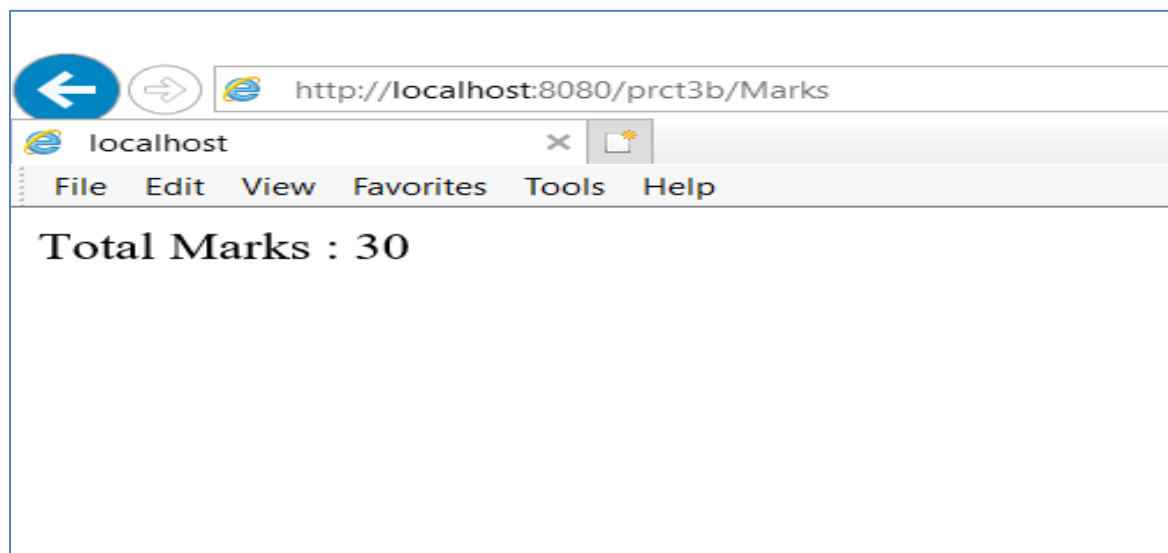
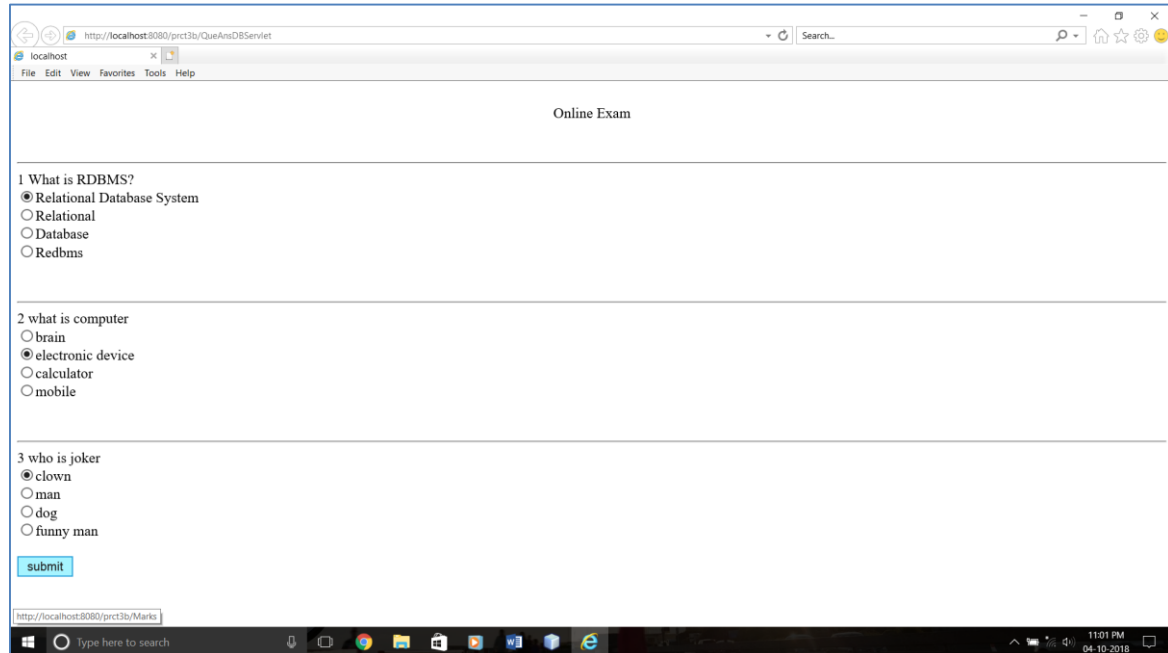
    out.println("ERROR "+e.getMessage());

}

}
```

Rightclick on QueAnsDbServlet and Run

OUTPUT:



Q.3 c) Create simple Servlet application to demonstrate Non-Blocking Read Operation.

CODE:

Index.html

`<html>`

Non Blocking Servlet

</body>

</html>

ReadingListener.java

```
package nonblkapp;

import java.io.*;

import java.util.logging.Level;

import java.util.logging.Logger;

import javax.servlet.*;

public class ReadingListener implements ReadListener

{
    ServletInputStream input = null;

    AsyncContext ac = null;

    ReadingListener(ServletInputStream in, AsyncContext c) {

        input = in;

        ac = c;

    }

    @Override

    public void onDataAvailable() {

    }

    public void onAllDataRead()

    {
        ac.complete();

    }

    public void onError(Throwable t)

    {
        ac.complete();

        t.printStackTrace();
    }
}
```

}

ReadingNonBlockingServlet.java

```
package nonblkapp;

import java.io.*;

import javax.servlet.*;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.*;

@WebServlet (name = "ReadingNonBlockingServlet", urlPatterns =
{"/ReadingNonBlockingServlet"},asyncSupported = true )

public class ReadingNonBlockingServlet extends HttpServlet {

@Override

    protected void service(HttpServletRequest request, HttpServletResponse response)

        throws ServletException, IOException

    {    response.setContentType("text/html");

        AsyncContext ac = request.startAsync();

        ServletInputStream in=request.getInputStream();

        in.setReadListener(new ReadingListener(in,ac));

    }

}
```

NonBlockingServlet.java

```
package nonblkapp;

import java.io.*;
```

```
import java.net.HttpURLConnection;

import java.net.URL;

import java.util.logging.Level;

import java.util.logging.Logger;

import javax.servlet.*;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.*;

@WebServlet(name = "NonBlockingServlet", urlPatterns = { "/NonBlockingServlet" })

public class NonBlockingServlet extends HttpServlet {

    @Override

    protected void service(HttpServletRequest request, HttpServletResponse response) throws

ServletException, IOException {

        response.setContentType("text/html");

        PrintWriter out = response.getWriter();

        String filename = "booklist.txt";

        ServletContext c = getServletContext();

        InputStream is = c.getResourceAsStream("/"+filename);

        InputStreamReader isr = new InputStreamReader(is);

        BufferedReader br = new BufferedReader(isr);

        String path = "http://" + request.getServerName() + ":" + request.getServerPort() +

request.getContextPath() + "/ReadingNonBlockingServlet";

        out.println("<h1>File Reader</h1>");

        //out.flush();

        URL url = new URL(path);

        HttpURLConnection hc = (HttpURLConnection) url.openConnection();

        hc.setChunkedStreamingMode(2); //2bytes at a time

        hc.setDoOutput(true); // true if URL connection done
```

```
hc.connect();

String text = "";

System.out.println("Reading started...");

BufferedWriter bw = new BufferedWriter(new OutputStreamWriter(hc.getOutputStream()));

while ((text = br.readLine()) != null)

{

    bw.write(text);

    bw.flush();

    out.println(text+"<br>");

    out.flush();

    try

    {

        Thread.sleep(1000);

    }

    catch (Exception ex)

    {

        out.print(ex);

    }

}

bw.write("Reading completed...");

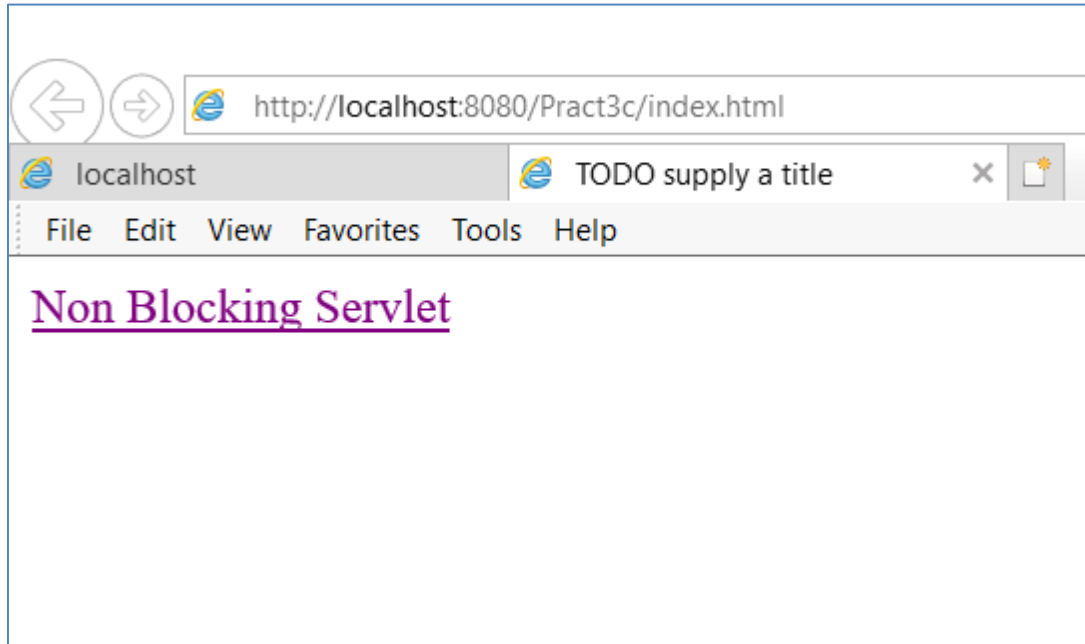
bw.flush();

bw.close();

}

}
```

OUTPUT:



PRACTICAL 4

Q.4 a) Develop a simple JSP application to display values obtained from the use of intrinsic objects of various types.

index.jsp

```
<html>

<body>

<form action="implicitObjectEx.jsp">

Enter your name:<input type="text" name="myname"><br>

Enter your email id:<input type="text" name="mymailid"><br>

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

</form>

</body>

</html>
```

implicitObjectEx.jsp

```
<% @page contentType="text/html" pageEncoding="UTF-8"%>

<html>

<head>

<title>JSP Page</title>

</head>

<body>

<h1>Use of Intrinsic Objects in JSP</h1>

<h1>Request Object</h1>

Query String<%=request.getQueryString() %><br>

Context Path<%=request.getContextPath() %><br>

Remote Host<%=request.getRemoteHost() %><br>

<h1>Response Object</h1>

Character Encoding Type<%=response.getCharacterEncoding() %><br>

Content Type <%=response.getContentType() %><br>

Locale <%=response.getLocale() %><br>
```


ID<%=session.getId() %>

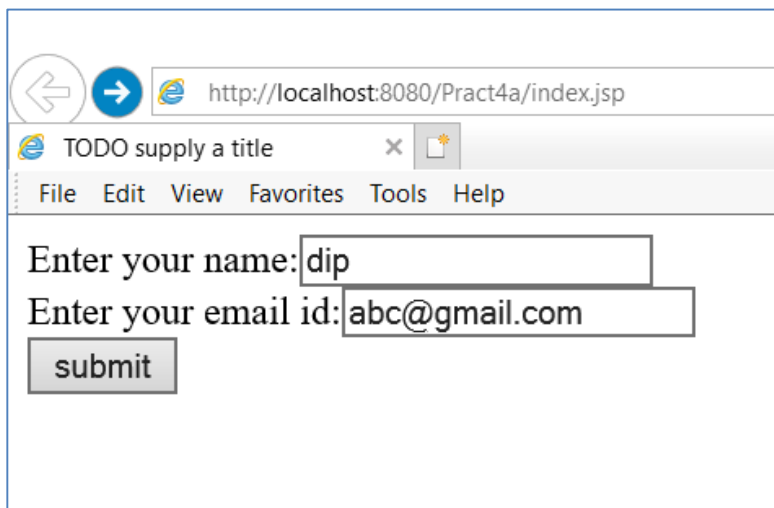
Creation Time<%=new java.util.Date(session.getCreationTime()) %>

Last Access Time<%=new java.util.Date(session.getLastAccessedTime()) %>

</body>

</html>

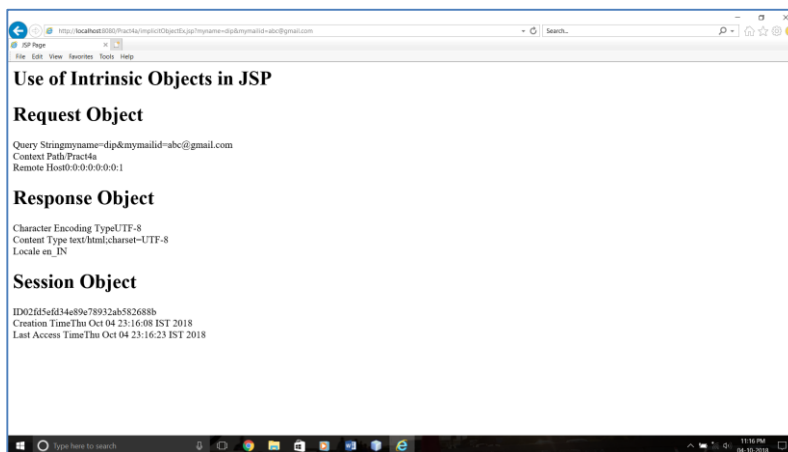
OUTPUT:



Enter your name: dip

Enter your email id: abc@gmail.com

submit



Q.4 b) Develop a simple JSP application to pass values from one page to another with validations. (Name-txt, age-txt, hobbies-checkbox, email-txt, gender-radio button).

CODE:

Index.jsp

```
<html>

<body>

<form action="Validate.jsp">
Enter Your Name <input type="text" name="name"><br>
Enter Your Age <input type="text" name="age"><br>
Select Hobbies <input type="checkbox" name="hob" value="Singing">Singing
<input type="checkbox" name="hob" value="Reading">Reading Books
<input type="checkbox" name="hob" value="Football">Playing Football<br>
Enter E-mail<input type="text" name="email"><br>
Select Gender <input type="radio" name="gender" value="male">Male
<input type="radio" name="gender" value="female">Female
<input type="radio" name="gender" value="other">Other<br>
<input type="hidden" name="error" value="">
<input type="submit" value="Submit Form">

</form >

</body>

</html>
```

CheckerBean.java

```
package mypack;

import java.beans.*;
import java.io.Serializable;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

public class CheckerBean
{
String name,hob,email,gender,error;
int age;
public CheckerBean()
{
name="";
hob="";
email="";
gender="";
error="";
age=0;
}

public void setName(String n)
```

```
{  
name=n;  
}  
public String getName()  
{  
return name;  
}  
public void setAge(int a)  
{  
age=a;  
}  
public int getAge()  
{  
return age;  
}  
public void setHob(String h)  
{  
hob=h;  
}  
public String getHob()  
{  
return hob;  
}  
public void setEmail(String e)  
{  
email=e;  
}  
public String getEmail()  
{  
return email;  
}  
public void setGender(String g)  
{  
gender=g;  
}  
public String getGender()  
{  
return gender;  
}  
public String getError()
```

```
{
return error;
}

public boolean validate()
{
boolean res=true;
if(name.trim().equals(""))
{
error+="
```

Validate.jsp

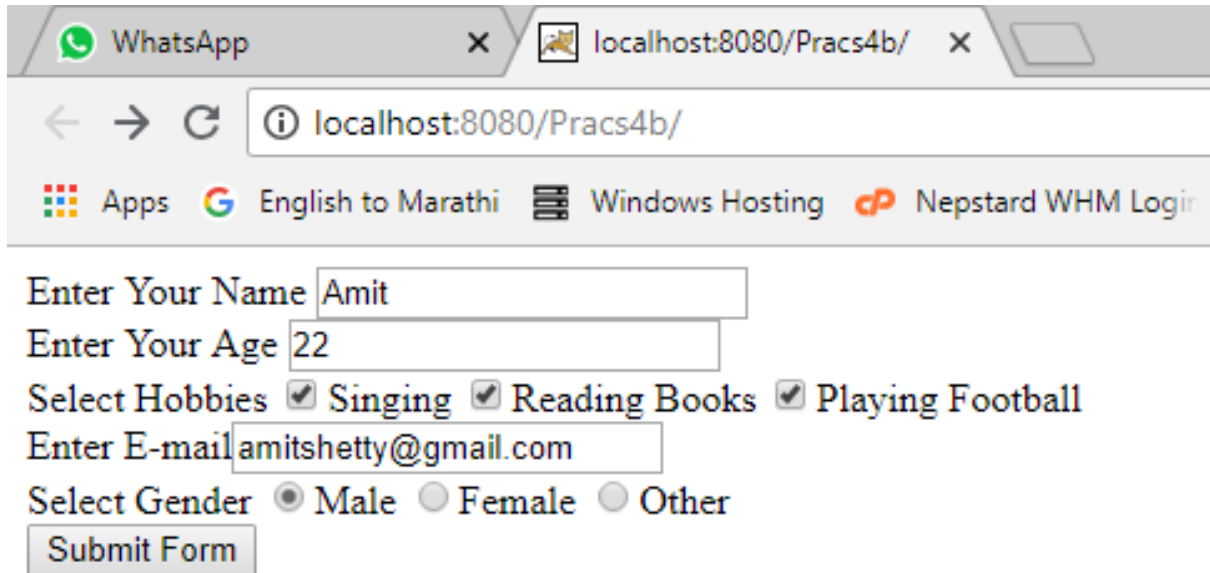
```
<% @page contentType="text/html" pageEncoding="UTF-8" import="mypack.*"%>
<html>
<head>
<title>JSP Page</title>
</head>
<body>
<h1>Validation Page</h1>
<jsp:useBean id="obj" scope="request"
class="mypack.CheckerBean" >
<jsp:setProperty name="obj" property="*" />
```

```
</jsp:useBean>
<%if(obj.validate())
{ %>
<jsp:forward page="successful.jsp"/>
<% }
else { %>
<jsp:include page="index.html"/>
<% } %>
<%=obj.getError() %>
</body>
</html>
```

successful.jsp

```
<% @page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPEhtml>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>JSP Page</title>
</head>
<body>
<h1>DATA VALIDATED SUCCESSFULLY</h1>
</body>
</html>
```

OUTPUT:



WhatsApp x localhost:8080/Pracs4b/ x

localhost:8080/Pracs4b/

Apps English to Marathi Windows Hosting Nepstard WHM Login

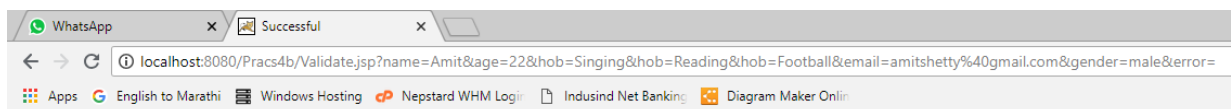
Enter Your Name

Enter Your Age

Select Hobbies ☒ Singing ☒ Reading Books ☒ Playing Football

Enter E-mail

Select Gender ☒ Male ☐ Female ☐ Other



DATA VALIDATED SUCCESSFULLY

Q.4 c) Create a registration and login JSP application to register and authenticate the user based on username and password using JDBC.

Index.html

```
<html>
<head>
<title>New User Registration Page</title>
</head>
<body>
<form action="Registration.jsp">
<h1>New User Registration Page</h1>
Enter User Name<input type="text" name="txtName"><br>
Enter Password<input type="password" name="txtPass1"><br>
Re-Enter Password<input type="password" name="txtPass2"><br>
Enter Email<input type="text" name="txtEmail"><br>
Enter Country Name<select name="txtCon">
<option>India</option>
<option>France</option>
<option>England</option>
<option>Argentina</option>
</select><br>
<input type="submit" value="REGISTER"><input type="reset">
</form>
</body>
</html>
```

Registration.jsp

```
<% @page contentType="text/html" import="java.sql.*"%>
<html><body>
<h1>Registration JSP Page</h1>
<%
String uname=request.getParameter("txtName");
String pass1=request.getParameter("txtPass1");
String pass2=request.getParameter("txtPass2");
String email=request.getParameter("txtEmail");
String ctry=request.getParameter("txtCon");
if(pass1.equals(pass2))
{
try
{
Class.forName("com.mysql.jdbc.Driver");
```

Connection

```
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/logindb","root","tiger");
PreparedStatement stmt=con.prepareStatement("insert into userpass values(?,?,?,?)");
stmt.setString(1,uname);
stmt.setString(2,pass1);
stmt.setString(3,email);
stmt.setString(4,ctry);
int row=stmt.executeUpdate();
if(row==1)
{
out.println("Registration Successful");}
else
{
out.println("Registration FAILED!!!!");
%>
<jsp:include page="index.html"></jsp:include>
<%
}
}catch(Exception e){out.println(e);}
}
else
{
out.println("<h1>Password Mismatch</h1>");
%>
<jsp:include page="index.html"></jsp:include>
<% }
%>
</body>
</html>
```

Login.html

```
<html>
<body>
<h1>Login Page</h1>
<form action="Login.jsp">
Enter User Name<input type="text" name="txtName"><br>
Enter Password<input type="password" name="txtPass"><br>
<input type="submit" value="~~~LOGIN~~~"><input type="reset">
</form>
```


</body>

</html>

Login.jsp

<% @page contentType="text/html" import="java.sql.*"%>

<html><body>

<h1>Registration JSP Page</h1>

<%

String uname=request.getParameter("txtName");

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

ResultSet rs=null;

try{

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

Connection

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

Statement stmt=con.createStatement();

rs=stmt.executeQuery("select password from userpass where username='"+uname+"'");

rs.next();

if(pass.equals(rs.getString(1)))

{

out.println("<h1>~~~LOGIN SUCCESSFULL~::~~</h1>");

}

else

{

out.println("<h1>password does not match!!!!</h1>");

%>

<jsp:include page="index.html"></jsp:include>

<%

}

}catch(Exception e){

out.println("<h1>User does not exist!!!!</h1>");

%>

<jsp:include page="index.html"></jsp:include>

<%

}

%>

</body>

</html>

OUTPUT:

A screenshot of a web browser window. The address bar shows 'http://localhost:8080/prct4c/index.html'. The browser has a single tab titled 'New User Registration Page'. The page content includes a title 'New User Registration Page' and a registration form with the following fields: 'Enter User Name' (containing 'dip1'), 'Enter Password' (masked with dots), 'Re-Enter Password' (masked with dots), 'Enter Email' (containing 'abc@gmail.com'), and 'Enter Country Name' (a dropdown menu showing 'India'). At the bottom of the form are two buttons: 'REGISTER' and 'Reset'.

A screenshot of a web browser window. The address bar shows 'http://localhost:8080/prct4c/Registration.jsp?txtName=dip1&txtP...'. The browser has a single tab titled 'localhost'. The page content includes a title 'Registration JSP Page' and a message 'Registration Successful'.

A screenshot of a database management tool window. The top pane shows a SQL query: `SELECT * FROM USERPASS;`. The bottom pane shows the results of the query in a table format. The table has 5 columns: #, UNAME, PASS1, EMAIL, and CTRY. There is one row of data.

#	UNAME	PASS1	EMAIL	CTRY
1	dip1	hello	abc@gmail.com	India

PRACTICAL 5

Q.5 a) Create an html page with fields, eno, name, age, desg, salary. Now on submit this data to a JSP page which will update the employee table of database with matching eno.

Index.html

```
<html>
<body>
<form action="UpdateEmp.jsp" >
Enter Employee Number<input type="text" name="txtEno" ><br>
Enter Salary to update<input type="text" name="txtSal" ><br>
<input type="reset" ><input type="submit">
</form>
</body>
</html>
```

UpdateEmp.jsp

```
<% @page contentType="text/html" import="java.sql.*" %>
<html>
<body>
<h1>Updating Employee Record</h1>
<%
String eno=request.getParameter("txtEno");
String sal = request.getParameter("txtSal");
try{
Class.forName("com.mysql.jdbc.Driver");
Connection con =
DriverManager.getConnection("jdbc:mysql://localhost:3306/empdb","root","tiger");
PreparedStatement stmt = con.prepareStatement("select * from emp where
empno=?");
stmt.setString(1, eno);
ResultSet rs = stmt.executeQuery();
if(rs.next()){
out.println("<h1> Employee "+rs.getString(2)+" Exist </h1>");
PreparedStatement pst= con.prepareStatement("update emp set salary=? where
empno=?");
pst.setString(1, sal);
pst.setString(2, eno);
pst.executeUpdate();
out.println("<h1>Employee Record updated !!!!!</h1>");
}
else{
out.println("<h1>Employee Record not exist !!!!!</h1>");
}
```

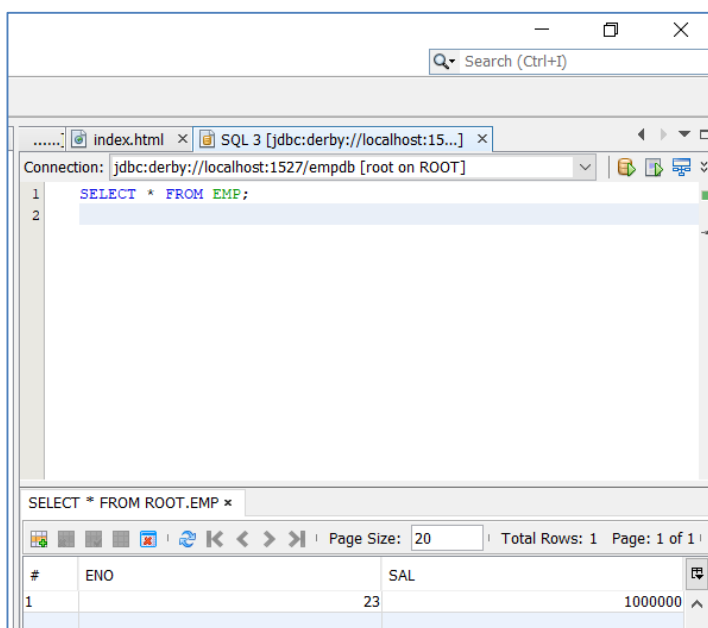
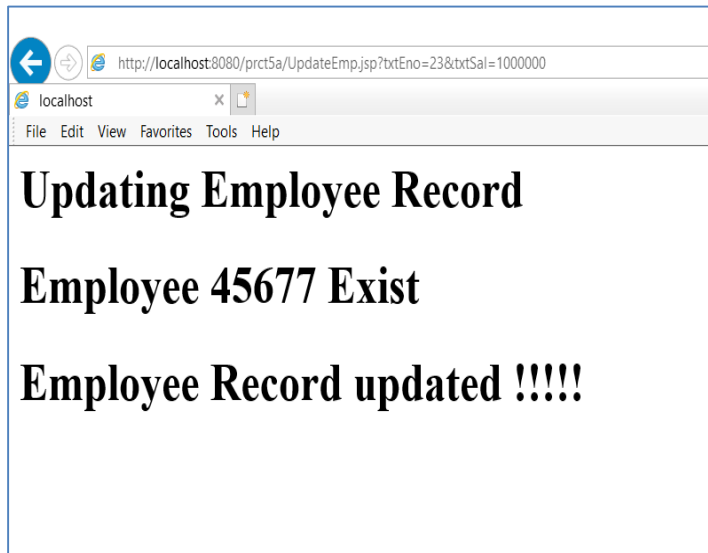
```
}  
}catch(Exception e){out.println(e);}   
%>  
</body>  
</html>
```

OUTPUT:

The top screenshot shows the SQL Developer interface. The query editor contains the SQL statement: `SELECT * FROM EMP;`. The results pane below shows a table with columns #, ENO, and SAL. The first row contains the values 1, 23, and 45677.

#	ENO	SAL
1	23	45677

The bottom screenshot shows a web browser window at the URL `http://localhost:8080/prct5a/index.html`. The browser displays a form with two input fields: "Enter Employee Number" with the value 23, and "Enter Salary to update" with the value 1000000. Below the input fields are two buttons: "Reset" and "Submit Query".



Q.5 b) Create a JSP page to demonstrate the use of Expression language.

CODE:

a. Index.jsp

```
<body>
<h3>welcome to index page</h3>
<%
session.setAttribute("user","Admin");
%>
<%
Cookie ck=new Cookie("name","mycookie");
response.addCookie(ck);
%>
<form action="ExpressionLanguage.jsp">
Enter Name:<input type="text" name="name" /><br/><br/>
<input type="submit" value="Submit"/>
</form>
</body>
```

b. ExpressionLanguage.jsp

```
<body>
Welcome, ${ param.name }
Session Value is ${ sessionScope.user }
Cookie name is , ${cookie.name.value}
</body>
```

c. ELArithmeticOperator.jsp

```
<body>
<%-- arithmetic op --%>
5*5+4: ${5*5+4} <br>
1.4E4+1.4: ${1.4E4+1.4}<br>
10 mod 4: ${10 mod 4}<br>
15 div 3: ${15 div 3}<br>
</body>
```

d. ELLogicalOperator.jsp

```
<body>
<!-- LogicalOperator --%>
<h2>Logical Operator</h2>
true and true: ${true and true}<br>
true && false: ${true && false}<br>
true or true: ${true or true}<br>
true || false: ${true || false}<br>
not true: ${not true}<br>
!false: ${!false}
</body>
```

e. ELRelationalOperator.jsp

```
<body>
<!-- RelationalOperator --%>
<h2>Relational Operator</h2>
10.0==10: ${10.0==10} <br>
10.0 eq 10: ${10.0 eq 10} <br>
((20*10)!= 200): ${((20*10)!= 200)} <br>
3 ne 3: ${3 ne 3} <br>
3.2>=2: ${3.2>=2} <br>
3.2 ge 2: ${3.2 ge 2} <br>
2<3: ${2<3} <br>
4 lt 6: ${4 lt 6} <br>
2 <= 4: ${2 <= 4} <br>
4 le 2: ${4 le 2}
</body>
```

f. ELconditional op

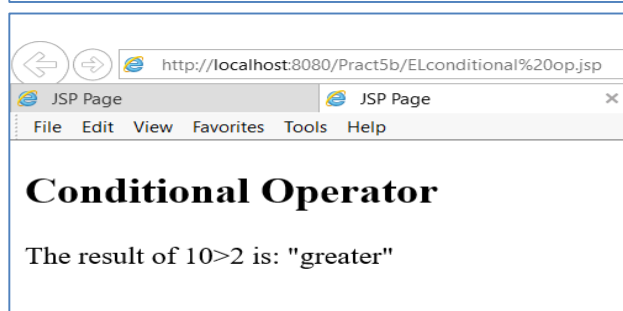
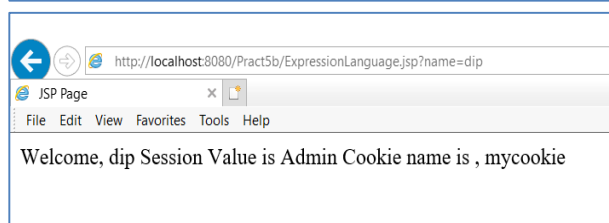
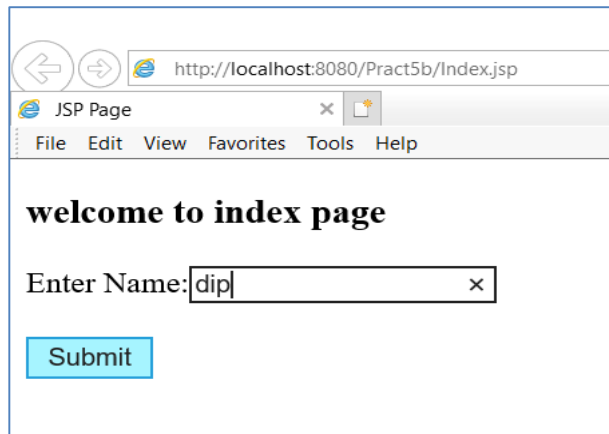
```
<body>
<h2>Conditional Operator</h2>

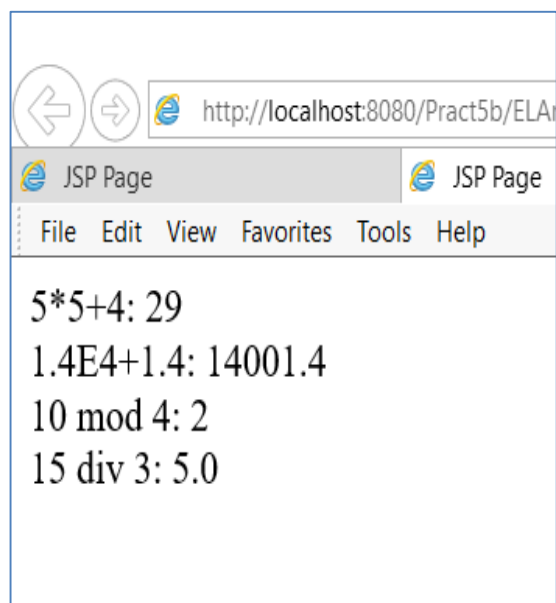
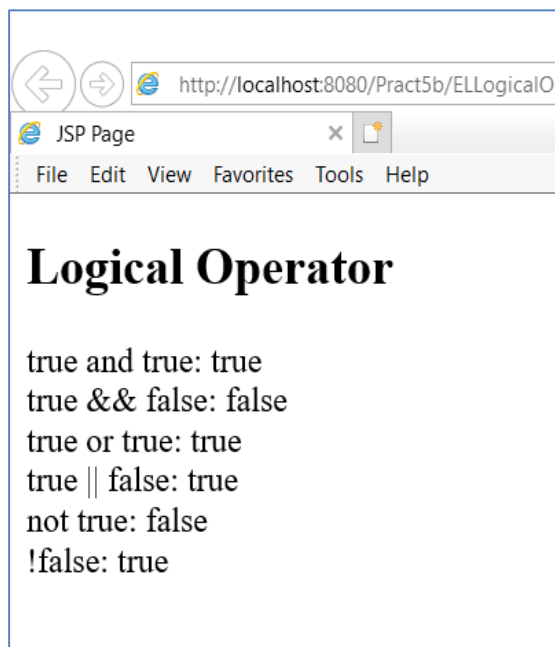
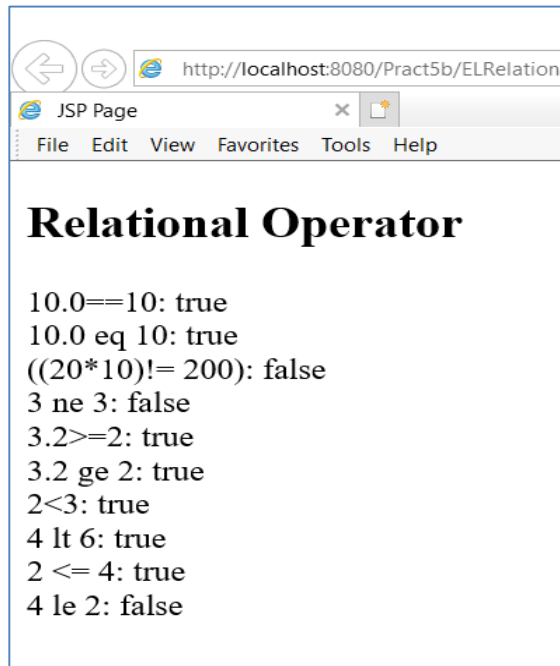
The result of 10>2 is: "${(10>1)?'greater':'lesser'}"
</body>
```

g. Empty Operator

```
<H1>Empty Operator Example</H1>

The Value for the Empty operator is:: ${empty "txt"}
```

OUTPUT:



Q.5 c) Create a JSP application to demonstrate the use of JSTL.**CODE:****index.html**

```
<html><body>

<a href="setDemo.jsp"> SetDemo</a>

<a href="Maxif.html"> MaxIF</a>

<a href="forEachDemo.jsp"> ForEachDemo</a>

<a href="outDemo.jsp"> OutDemo</a>

<a href="URLDemo.jsp"> URLDemo</a>

<a href="choose_when_otherwise.jsp"> choose_when_otherwise</a>

</body></html>
```

setDemo.jsp

```
<% @taglib prefix="c" uri="http://java.sun.com/jstl/core" %>

<c:set var="pageTitle" scope="application"

value="Dukes Soccer League: Registration" />

${pageTitle}
```

Maxif.html

```
<form action ="IFDemo.jsp">

    x=<input type="text" name="x" /><br>

    y=<input type="text" name="y" /><br>

    <input type="submit" value="Check Max" />

</form>
```

IFDemo.jsp

```
<% @taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>

<c:set var="x" value="{param.x}"/>

<c:set var="y" value="{param.y}"/>

<c:if test="{x>y}">

    <font color="blue"><h2>The Ans is:</h2></font>

    <c:out value="{x} is greater than {y}"/>

</c:if>
```

ForeachDemo.jsp

```
<% @taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>

<c:forEach begin="1" end="10" var="i">

    The Square of <c:out value="{i}={i*i}"/><br>

</c:forEach>
```

outDemo.jsp

```
<% @taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>

<c:set var="name" value="John"/>

My name is: <c:out value="{name}"/>
```

URLDemo.jsp

```
<% @taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>

<c:url value="/index.html"/>
```

choose_when_otherwise.jsp

```
<% @taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>

<c:set var="income" value="{4000*4}"/>

Your Income is: <c:out value="{income}"/>

<c:choose>

    <c:when test="{income <=1000}">
```

Income is not good

</c:when>

<c:when test="{ income > 10000}">

Income is Very Good

</c:when>

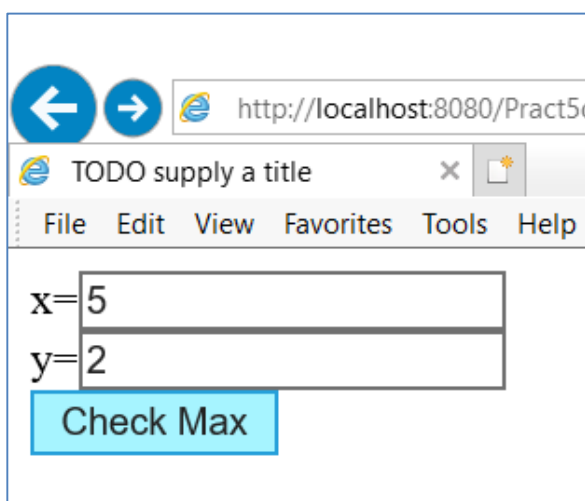
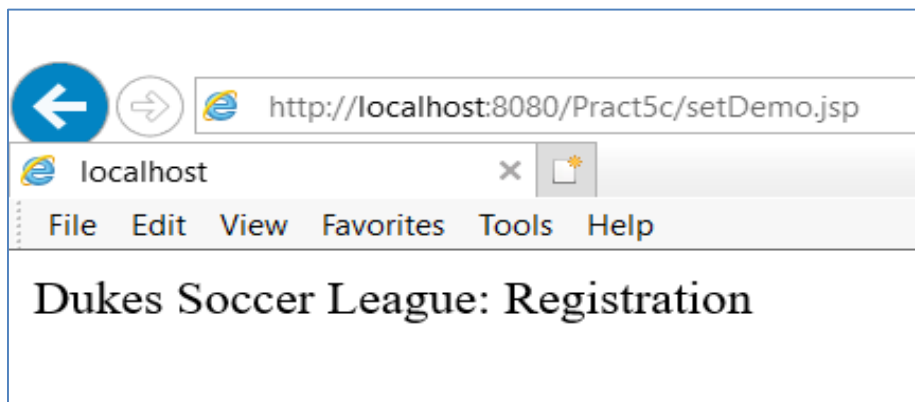
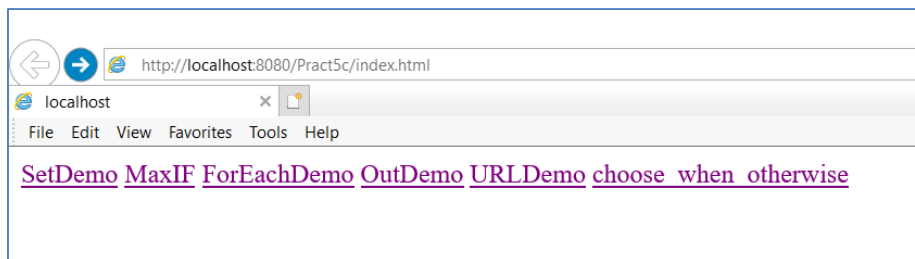
<c:otherwise>

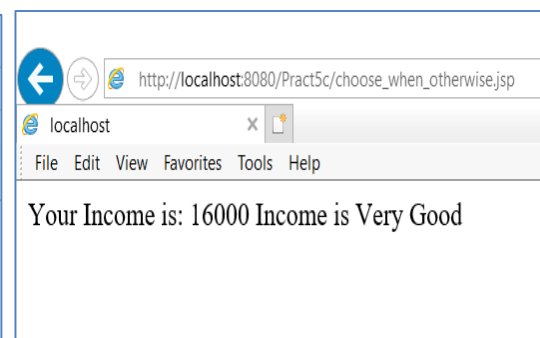
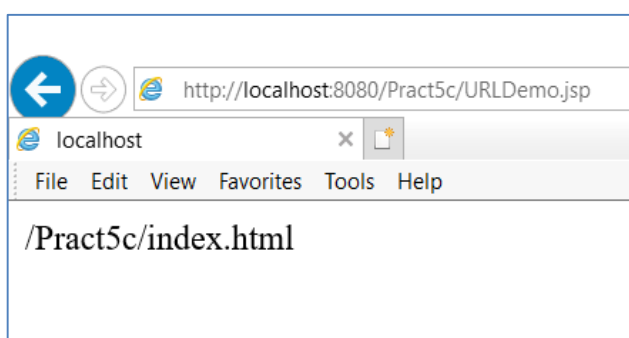
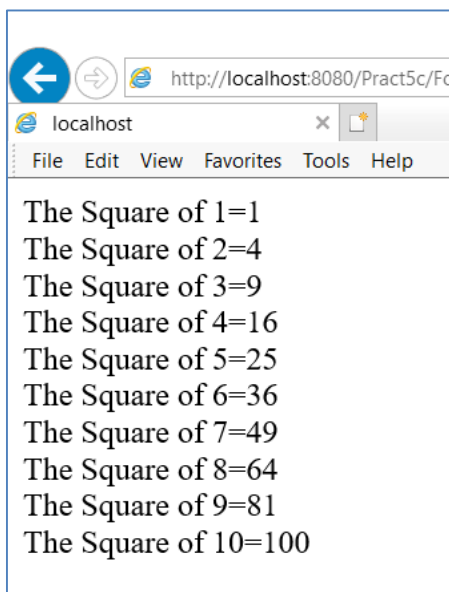
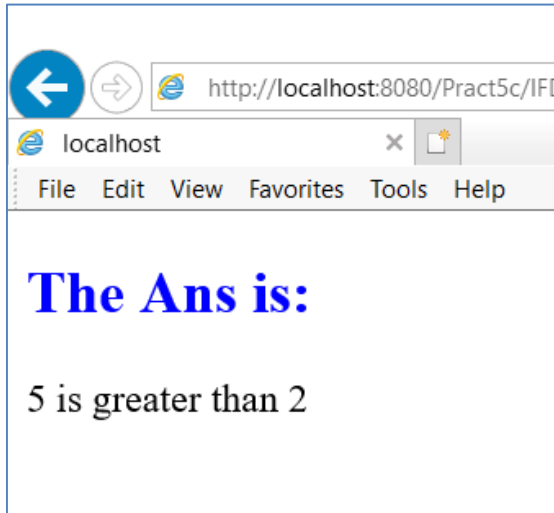
Income is undetermined

</c:otherwise>

</c:choose>

OUTPUT:





PRACTICAL 6**Q.6 a) Create a Currency Converter application using EJB.****CODE:****Index.html**

```
<html><head><title>Currency Converter</title></head>

<body>

    <form action="CCServlet" >

        Enter Amount <input type="text" name="amt"><br>

        Select Conversion Type

        <input type="radio" name="type" value="r2d" checked>Rupees to Dollar

        <input type="radio" name="type" value="d2r" >Dollar to Rupees<br>

        <input type="reset" ><input type="submit" value="CONVERT" >

    </form>

</body>

</html>
```

Step 2 : Create a session bean named as CCBean in the package named mybeans. Select the option Stateless and click on Local Interface.

Here you will find two files created in the mybeans package named as CCBean.java and CCBeanLocal.java

CCBeanLocal.java

```
package mybeans;

import javax.ejb.Stateless;

@Stateless

public interface CCBeanLocal {

    //default constructor

    public double r2Dollar(double r);

    public double d2Rupees(double d); }
```

CCBean.java

```
package mybeans;

import javax.ejb.Stateless;

@Stateless

public class CCBean implements CCBean1Local

{

    public double r2Dollar(double r)

    {

        return r/65.65;

    }

    public double d2Rupees(double d)

    {

        return d*65.65;

    }

}
```

Step 3: Create a Servlet file name CCServlet in the package mypack.

```
package mypack;

import java.io.*;

import javax.servlet.*;

import javax.servlet.http.*;

import javax.ejb.EJB;

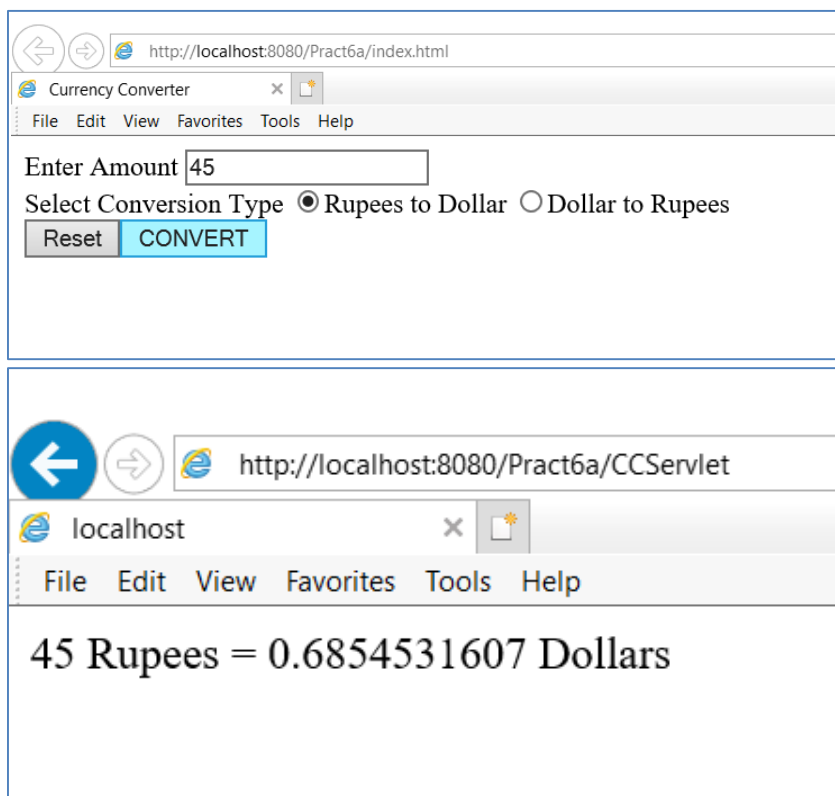
import mybeans.CCBeanLocal;

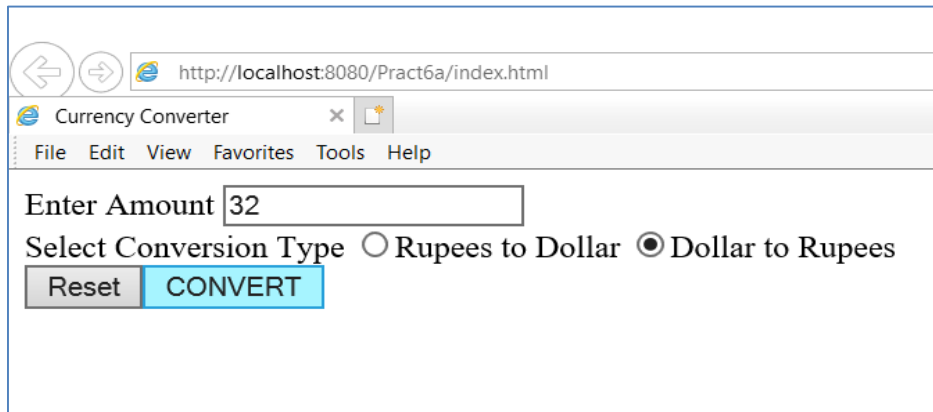
public class CCServlet extends HttpServlet {

    @EJB CCBeanLocal obj;

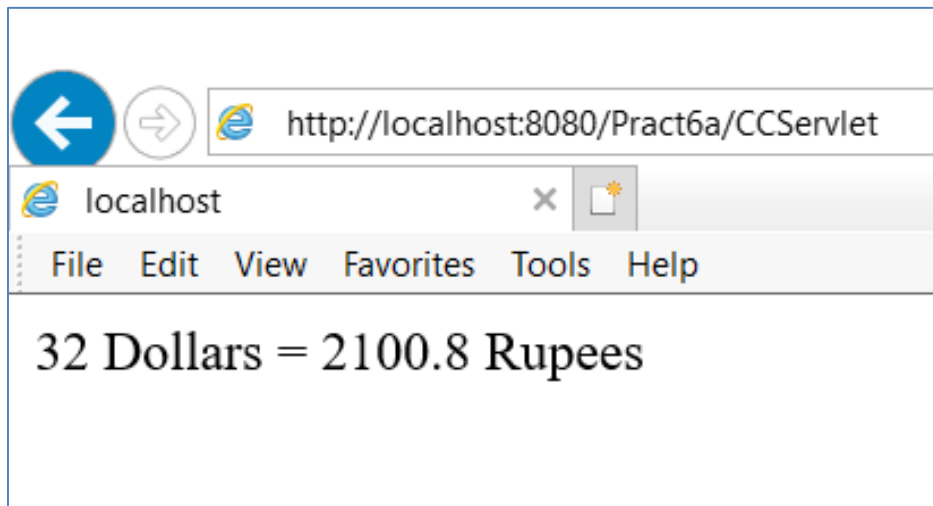
    public void doGet(HttpServletRequest request, HttpServletResponse response)throws
    ServletException, IOException
```

```
{  
  
response.setContentType("text/html;charset=UTF-8");  
  
PrintWriter out = response.getWriter();  
  
double amt = Double.parseDouble(request.getParameter("amt"));  
  
if(request.getParameter("type").equals("r2d"))  
{  
  
out.println("<h1>" + amt + " Rupees = " + obj.r2Dollar(amt) + " Dollars</h1>");  
  
}  
  
if(request.getParameter("type").equals("d2r"))  
{  
  
out.println("<h1>" + amt + " Dollars = " + obj.d2Rupees(amt) + " Rupees</h1>");  
  
}  
  
}  
  
}
```

OUTPUT:



A screenshot of a web browser window. The address bar shows the URL `http://localhost:8080/Pract6a/index.html`. The browser has a menu bar with 'File', 'Edit', 'View', 'Favorites', 'Tools', and 'Help'. The page content includes a text input field labeled 'Enter Amount' with the value '32'. Below it, there is a label 'Select Conversion Type' followed by two radio buttons: 'Rupees to Dollar' (unselected) and 'Dollar to Rupees' (selected). At the bottom, there are two buttons: 'Reset' and 'CONVERT'.



A screenshot of a web browser window. The address bar shows the URL `http://localhost:8080/Pract6a/CCServlet`. The browser has a menu bar with 'File', 'Edit', 'View', 'Favorites', 'Tools', and 'Help'. The page content displays the result of the conversion: '32 Dollars = 2100.8 Rupees'.

Q.6 b) Develop a Simple Room Reservation System Application Using EJB.**CODE:****Index.html**

```
<html>

<head>

<title>Room Reservation</title>

</head>

<body>

<form method="post" action="RoomClient">

<br> No of Rooms <input type="text" name="t1">

<br> <input type="submit" name="btn" value="CheckIN">

<br> <input type="submit" name="btn" value="CheckOUT">

</form>

</body>

</html>
```

Step2: Create a session bean named as RoomBean in the package named ejb. Select the option Stateless and click on Local Interface.

Here you will find two files created in the ejb package named as RoomBean.java and RoomBeanLocal.java

RoomBeanLocal.java

```
package ejb;

import javax.ejb.Local;

@Local

public interface RoomBeanLocal {

    public int checkin(int no);

    public int checkout(int no);

}
```

RoomBean.java

```
package ejb;

import javax.ejb.Stateless;

import java.sql.*;

@Stateless

public class RoomBean implements RoomBeanLocal {

    public int checkin(int no) {

        try

        {

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

            Connection

            con=DriverManager.getConnection("jdbc:mysql://localhost/roomdb","root","tiger");

            String sql1 = "select * from room";

            Statement st=con.createStatement();

            ResultSet rs=st.executeQuery(sql1);

            rs.next();

            int total=rs.getInt(1);

            int occ=rs.getInt(2);

            int free=total-occ;

            System.out.println(total);

            System.out.println(free);

            if (free>=no)

            {

                String sql2="update room set occ=?";

                PreparedStatement ps=con.prepareStatement(sql2);

                ps.setInt(1, occ+no);
```

```
int res=ps.executeUpdate();

return res;

}

else return 0;

}

catch(Exception e)

{

    return 0;

}

}

public int checkout(int no) {

try

{

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

    Connection

con=DriverManager.getConnection("jdbc:mysql://localhost/roomdb","root","tiger");

    String sql1 = "select * from room";

    Statement st=con.createStatement();

    ResultSet rs=st.executeQuery(sql1);

    rs.next();

    int total=rs.getInt(1);

    int occ=rs.getInt(2);

    if (occ>=no)

    {

        String sql2="update room set occ=?";

        PreparedStatement ps=con.prepareStatement(sql2);

        ps.setInt(1, occ-no);
```

```
int res=ps.executeUpdate();

return res;

}

else return 0;

}

catch(Exception e)

{

return 0;

}

}

}
```

Step 3: Create a Servlet file named as RoomClient. Do not click on web.xml (Deployment Descriptor)

```
package servlet;

import ejb.RoomBeanLocal;

import java.io.*;

import javax.ejb.EJB;

import javax.servlet.*;

import javax.servlet.http.*;

import javax.servlet.annotation.*;

@WebServlet(name = "roomclient", urlPatterns = {"/roomclient"})

public class roomclient extends HttpServlet {

    @EJB RoomBeanLocal obj;

    protected void doPost(HttpServletRequest request, HttpServletResponse response)

        throws ServletException, IOException {

        response.setContentType("text/html");

        PrintWriter out = response.getWriter();

        try {

            int no=Integer.parseInt(request.getParameter("t1"));
```

```
String b=request.getParameter("btn");

int res=0;

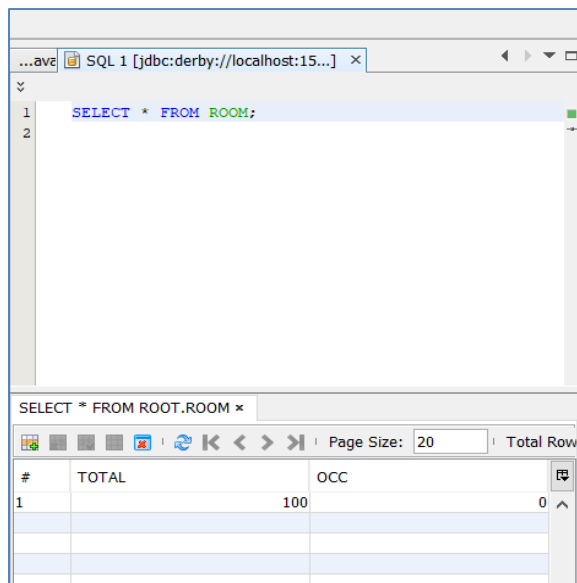
if(b.equals("CheckIN"))
{
    res=obj.checkin(no);
    if (res==1)
        out.println(no + " rooms check-in");
}

if(b.equals("CheckOUT"))
{
    res=obj.checkout(no);
    if (res==1)
        out.println(no + " rooms check-out");
}

if(res==0)
    out.println("Not possible to do Check IN / OUT");

    out.println("<br><br><a href=index.html> Back </a>");
}

finally {
    out.close();
}
}
```

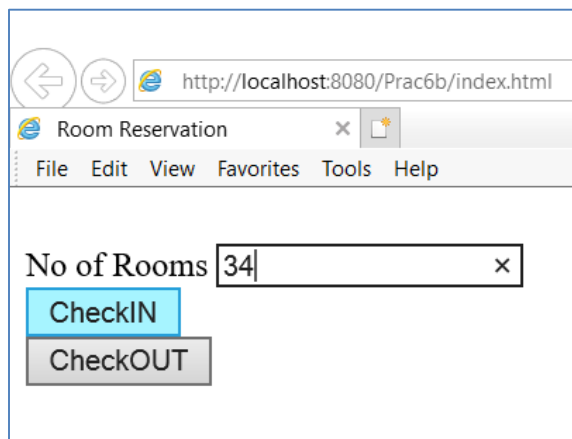
OUTPUT:

SQL 1 [jdbc:derby://localhost:15...]

```
1 SELECT * FROM ROOM;
```

SELECT * FROM ROOT.ROOM

#	TOTAL	OCC
1	100	0

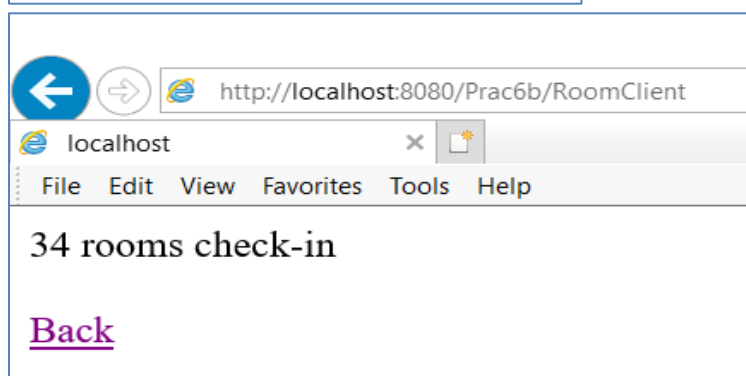


http://localhost:8080/Prac6b/index.html

Room Reservation

File Edit View Favorites Tools Help

No of Rooms



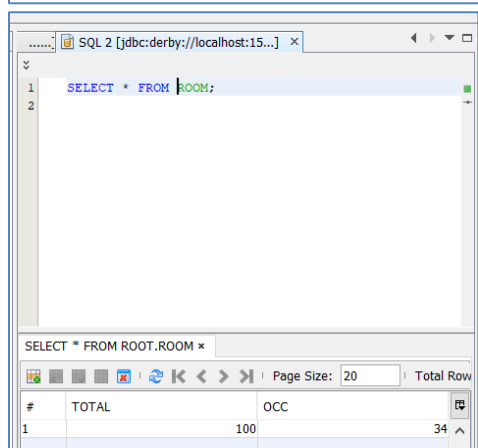
http://localhost:8080/Prac6b/RoomClient

localhost

File Edit View Favorites Tools Help

34 rooms check-in

[Back](#)

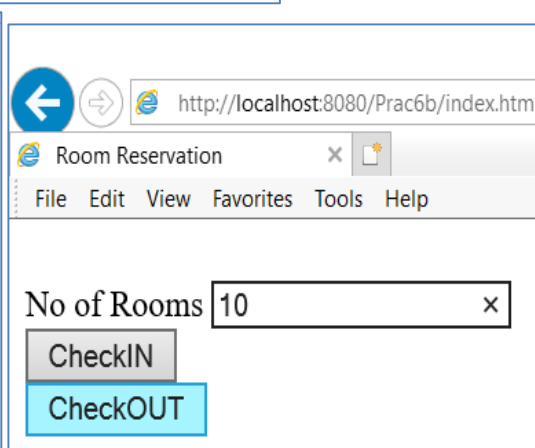


SQL 2 [jdbc:derby://localhost:15...]

```
1 SELECT * FROM ROOM;
```

SELECT * FROM ROOT.ROOM

#	TOTAL	OCC
1	100	34

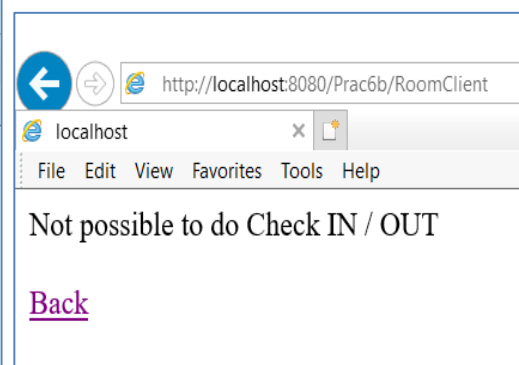
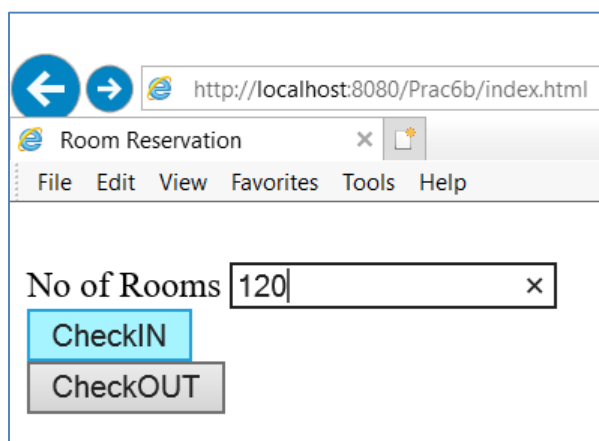
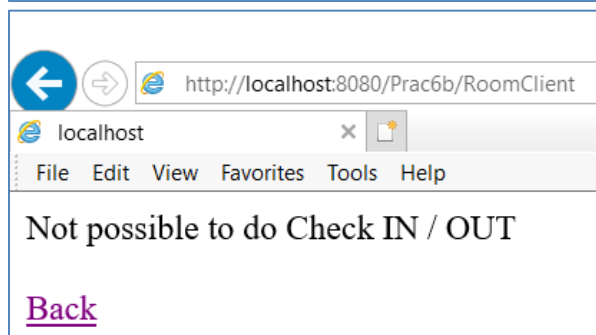
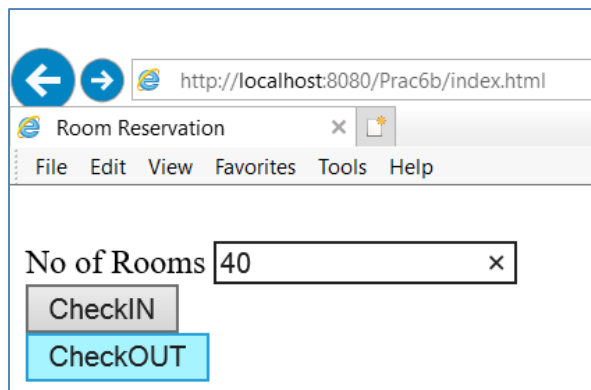
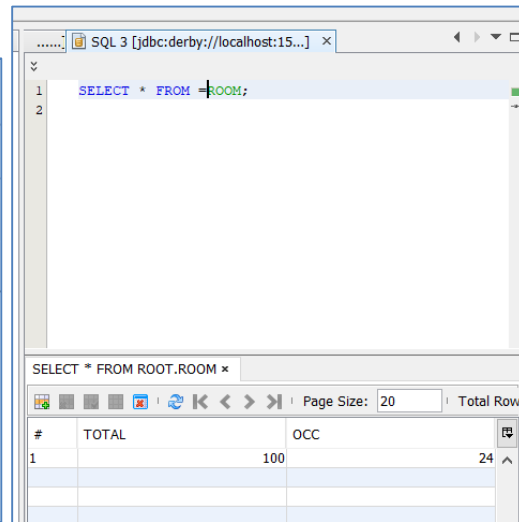
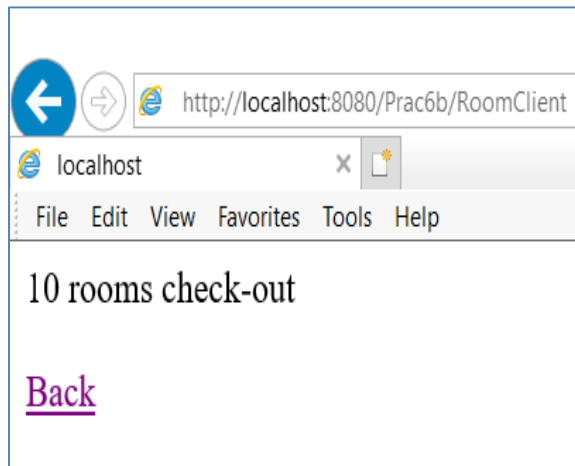


http://localhost:8080/Prac6b/index.html

Room Reservation

File Edit View Favorites Tools Help

No of Rooms



Q.6 c) Develop simple shopping cart application using EJB [Stateful Session Bean].**CODE:****Step 1 creating application**

File -> new project-> java web->web application -> Prac6CShoppingCartApp -> select Use dedicated folder for storing libraries -> finish

Step 2: Creating a stateful session bean

Source package -> new -> other -> enterprise java beans -> session bean -> next -> new session bean -> ejb name: ->ShoppingCart -> package: -> ejb -> session type option -> Stateful -> finish.

ShoppingCart.java

```
package ejb;

import java.sql.*;

import java.util.*;

import javax.ejb.*;

@Stateful

public class ShoppingCart

{   List<String> contents;

    String customerName;

    private Connection conn = null;

    private ResultSet rs;

    private Statement stmt = null;

    private String query = null;

    public void initialize(String person)

    {   if (person != null) {

        customerName = person;

        try {

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

```
conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/cartdb", "root",
"tiger");

    } catch(ClassNotFoundException | IllegalAccessException | InstantiationException |
SQLException e) {

        System.err.println("Sorry failed to connect to the Database." + e.getMessage());

    }

}

contents = new ArrayList<>();

}

public void addBook(String title) {

    try {

        stmt = conn.createStatement();

        query = "INSERT INTO cart VALUES('' + customerName + '' , '' + title + '')";

        stmt.executeUpdate(query);

    } catch(SQLException e) {

        System.err.println("Sorry failed to insert values from the database table. " + e.getMessage());

    }

}

public void removeBook(String title) {

    try {

        stmt = conn.createStatement();

        query = "DELETE FROM cart WHERE UserName='' + customerName + '' AND
ItemName='' + title + ''";

        stmt.executeUpdate(query);

    } catch(SQLException e) {

        System.err.println("Sorry failed to delete values from the database table. " + e.getMessage());

    }

}
```

```
public List<String> getContents() {  
  
    try {  
  
        stmt = conn.createStatement();  
  
        query = "SELECT * FROM cart WHERE UserName='" + customerName + "'";  
  
        rs = stmt.executeQuery(query);  
  
        while(rs.next()) {  
  
            contents.add(rs.getString("ItemName"));  
  
        }  
  
    } catch(SQLException e) {  
  
        System.err.println("Sorry failed to select values from the database table. " + e.getMessage());  
  
    }  
  
    return contents;  
  
}  
  
@Remove()  
  
public void remove() {  
  
    contents = null;  
  
}  
  
}
```

Step 3: creating a web client using index.jsp

Right click on web pages -> new -> JSP -> filename -> index -> finish.

```
<% @page import="java.util.Iterator, java.util.List, javax.naming.InitialContext,  
ejb.ShoppingCart"%>
```

```
<% @page contentType="text/html" pageEncoding="UTF-8"%>
```

```
<!DOCTYPE html>
```

```
<%!
```

```
private static ShoppingCart cart;
```

```
public void jspInit() {

    try {

        InitialContext ic = new InitialContext();

        cart = (ShoppingCart) ic.lookup("java:global/Prac6CShoppingCartApp/ShoppingCart");

    } catch (Exception ex) {

        System.out.println("Could not create cart bean." + ex.getMessage());

    }

}

%>

<%

if(request.getParameter("txtCustomerName") != null) {

    cart.initialize(request.getParameter("txtCustomerName"));

} else {

    cart.initialize("Guest");

}

if (request.getParameter("btnRmvBook") != null) {

    String books[] = request.getParameterValues("chkBook");

    if (books != null) {

        for (int i=0; i<books.length; i++) {

            cart.removeBook(books[i]);

        }

    }

}

if (request.getParameter("btnAddBook") != null) {

    String books[] = request.getParameterValues("chkBook");

    if (books != null) {
```

```
        for (int i=0; i<books.length; i++) {

            cart.addBook(books[i]);

        }

    }

}

%>

<html>

<head>

    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

    <title>Shopping Cart</title>

</head>

<body style="background-color: pink;">

    <h1 style="text-align: center;">Books For Sale</h1><br>

    <form method="post">

        Customer Name: <input type="text" name="txtCustomerName" value=<%=
request.getParameter("txtCustomerName")%> /><br>

        <b>Book Titles</b><br>

        <input type="checkbox" name="chkBook" value="Struts 2.0 For Beginners">Struts
2.0 For Beginners<br>

        <input type="checkbox" name="chkBook" value="Oracle 11g For
Professionals">Oracle 11g For Professionals<br>

        <input type="checkbox" name="chkBook" value="Hibernate 3 For
Beginners">Hibernate 3 For Beginners<br>

        <input type="checkbox" name="chkBook" value="Java Persistence API In EJB 3
For Beginners">Java Persistence API In EJB 3 For Beginners<br>

        <br>

        <input type='submit' value='Add To My Basket' name='btnAddBook'>
```

`<input type='submit' value='Remove From My Basket'`

`name='btnRmvBook'>

`

`<%`

`if(cart!=null)`

`{`

`out.print("Basket
");`

`List<String> bookList = cart.getContents();`

`Iterator iterator = bookList.iterator();`

`while (iterator.hasNext())`

`{`

`String title = (String) iterator.next();`

`%>`

`<%= title %>
`

`<%`

`}`

`}`

`%>`

`</form>`

`</body>`

`</html>`

Step 4:

Create database and database table

Services -> create database -> cartdb ->select cartdb - > right click -> create table ->
cart -> UserName varchar 35

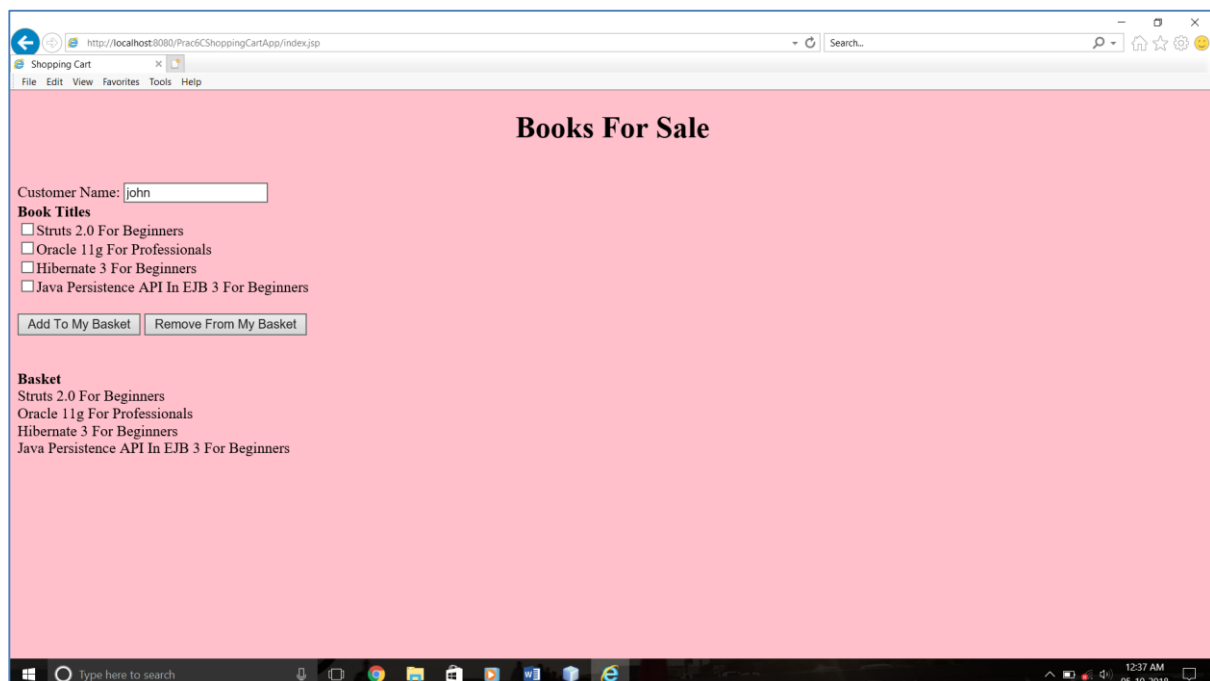
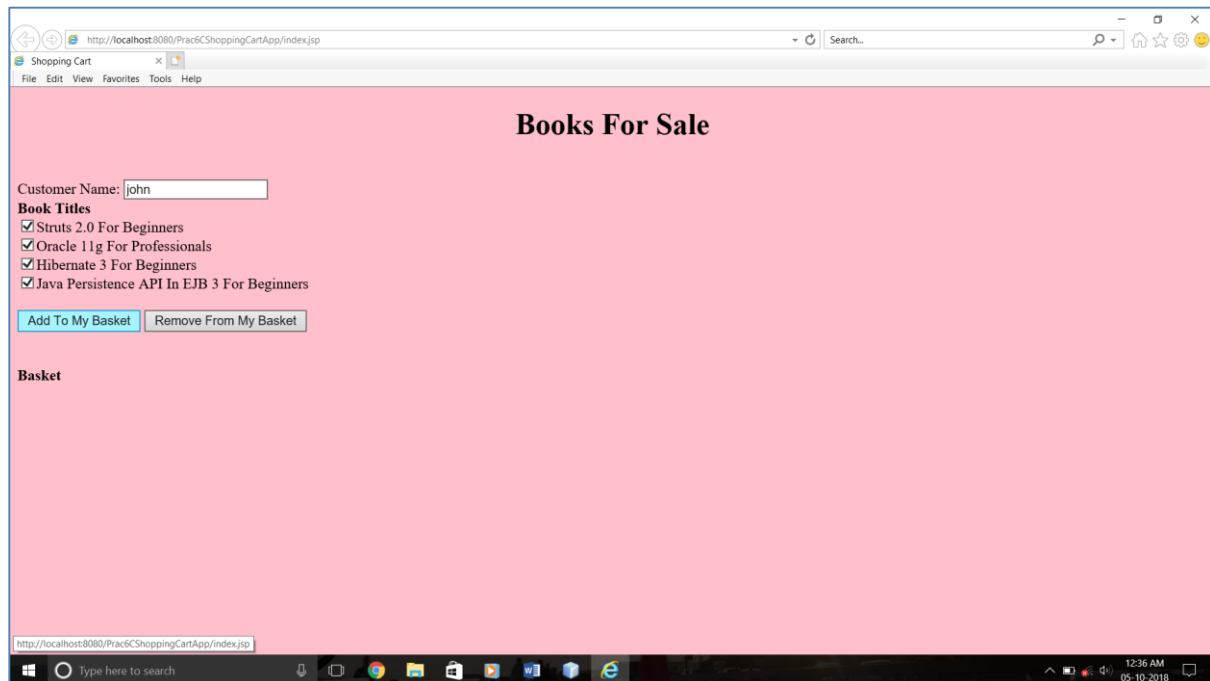
ItemName varchar 50

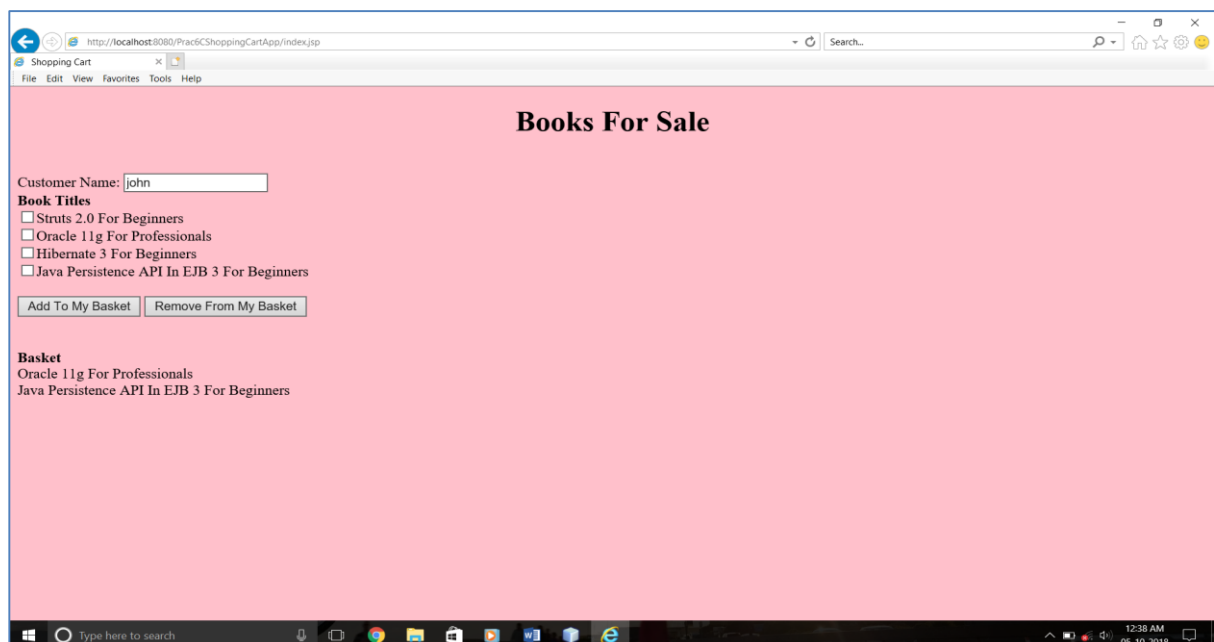
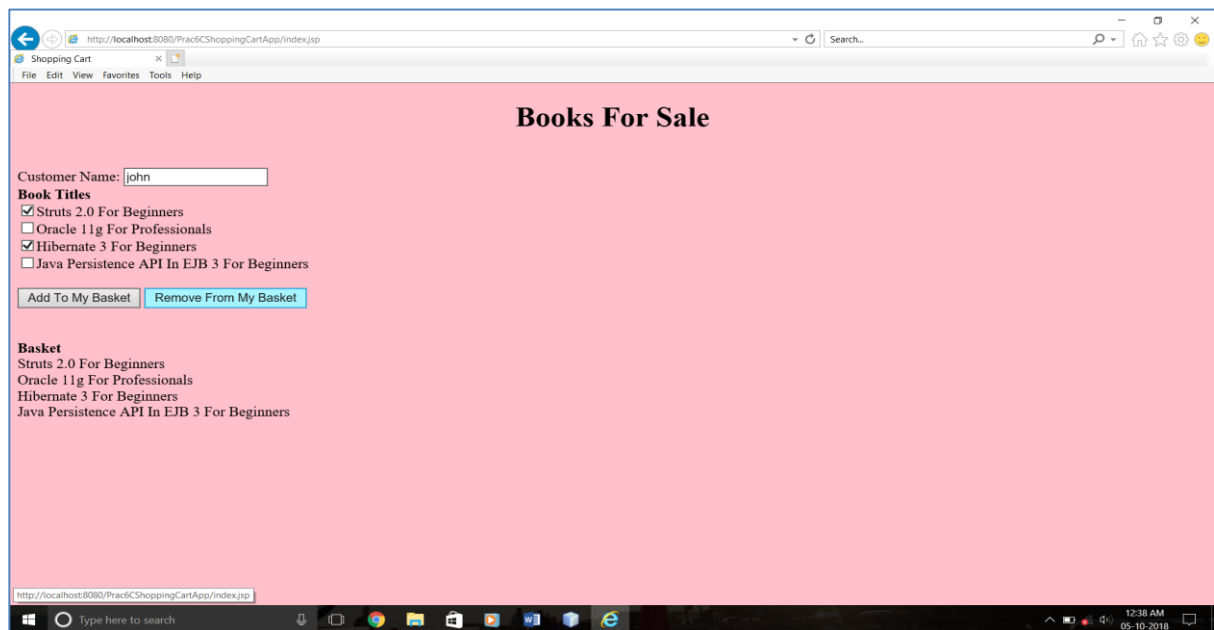
Finish.

Step 5.

Add mysql connector to the library under project tab.

Step 6: build and run the application.

OUTPUT:



PRACTICAL 7

Q.7 a) Develop simple EJB application to demonstrate Servlet Hit count using Singleton Session Beans.

CODE:

Java Web-> web application -> Pract7AServletHitsSingletonApp -> finish.

Step 1: Index.html

```
<html>
  <head>
    <title>TODO supply a title</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="Refresh" content="0; URL=ServletClient">
  </head>
  <body>
    <div>TODO write content</div>
  </body>
</html>
```

Step2: Create a Session Bean named as CountServletHitsBean → Select Singleton → package name as ejb (do not select Local or Remote)

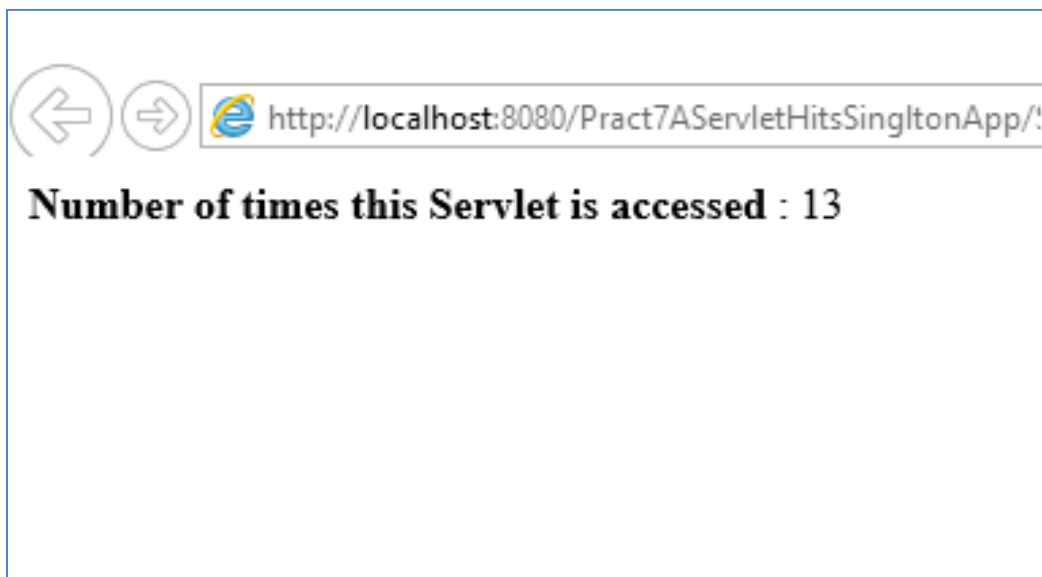
```
package ejb;
import javax.ejb.Singleton;
@Singleton
public class CountServletHitsBean {
    private int hitCount;
    public synchronized int getCount()
    {
        return hitCount++;
    }
}
```

**Step 3: Create a Servlet File name ServletClient in the package name as servlet.
Do not select the Deployment Descriptor file.**

```
package servlet;
import ejb.CountServletHitsBean;
import java.io.*;
import javax.ejb.EJB;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.*;

@WebServlet(name = "ServletClient", urlPatterns = {"/ServletClient"})
public class ServletClient extends HttpServlet {
    @EJB CountServletHitsBean obj;
    @Override
    protected void service (HttpServletRequest req, HttpServletResponse res) throws ServletException,
    IOException
    {
        res.setContentType("text/html");
        PrintWriter out=res.getWriter();
        out.print("<b>Number of times this Servlet is accessed </b>: "+obj.getCount());
    }
}
```

OUTPUT:



Q.7 b) Develop simple visitor Statistics application using Message Driven Bean [Stateless Session Bean].**CODE:**

Web-> web application -> Pract7BVisitorStatisticsMDBApp -> select dedicated folders for storing libraries -> finish.

Step 1:**index.jsp**

```
<% @page import="javax.jms.JMSEException"%>
<% @page import="javax.naming.InitialContext"%>
<% @page import="javax.jms.Connection"%>
<% @page import="javax.jms.TextMessage"%>
<% @page import="javax.jms.MessageProducer"%>
<% @page import="javax.jms.Session"%>
<% @page import="javax.jms.Queue"%>
<% @page import="javax.jms.ConnectionFactory"%>
<% @page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<%!
private static ConnectionFactory connectionFactory;
private static Queue queue;
Connection connection=null;
Session mySession=null;
MessageProducer messageProducer=null;
TextMessage message=null;
%>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>JSP Page</title>
  </head>
  <body>
    Welcome to My Home Page
    <%
    try{
      InitialContext ic= new InitialContext();
      queue= (Queue)ic.lookup("jms/Queue");
      connectionFactory=(ConnectionFactory)ic.lookup("jms/QueueFactory");
```

```
connection= connectionFactory.createConnection();
mySession=connection.createSession(false, Session.AUTO_ACKNOWLEDGE);
messageProducer=mySession.createProducer(queue);
message=mySession.createTextMessage();
message.setText(request.getRemoteAddr());
messageProducer.send(message);
}
catch(JMSEException e)
{
System.out.println("Exception Occoured "+e.toString());
}
%>
</body>
</html>
```

Step2: Create a Database name visitorstat → Create table name → userstat → column names

Firstvisitdt – timestamp

Hostname – varchar 30 Primary Key

Visits – int

Step3: Create a Session Bean named as VisitorStatBean → Select Stateless → package name as ejb, do not select Local / Remote

```
package ejb;
import java.sql.*;
import javax.annotation.PostConstruct;
import javax.annotation.PreDestroy;
import javax.ejb.Stateless;
@Stateless
public class VisitorStatBean {
private Connection conn=null;
private ResultSet rs;
private Statement st=null;
private String query =null;
@PostConstruct
public void connect()
{
try {
Class.forName("com.mysql.jdbc.Driver").newInstance();
conn=DriverManager.getConnection("jdbc:mysql://localhost/visitorstat", "root", "tiger");
}
}
```

```
catch (Exception e) {  
    System.err.println(e.getMessage());  
}  
}  
  
@PreDestroy  
public void disconnect()  
{  
    try {  
        conn.close();  
    } catch (Exception e) {  
        System.err.println(e.getMessage());  
    }  
}  
  
public void addVisitor(String host)  
{  
    try {  
        st= conn.createStatement();  
        query="insert into userstat (hostname,visits) values ('"+host+"',1)";  
        st.executeUpdate(query);  
    }  
    catch (SQLException e)  
    {  
        try {  
            st=conn.createStatement();  
            query="update userstat set visits=visits+1 where hostname='"+host+"' ";  
            st.executeUpdate(query);  
        }  
        catch (SQLException ex) {  
            System.err.println("Cannot Update"+e.getMessage());  
        }  
    }  
}  
}
```

Step 4: Right click on Source Packages → Select New → Other → Enterprise Java Bean → MessageDrivenBean → EJB Name: BasicMessageBean → Package: ejb → Select Project Destination → Click on Add Button → Destination Name: jms/Queue → Destination Type select the option Queue → click on OK → Click on Next → Activation Configuration Properties should be as it is. → Click on Finish

```
package ejb;

import javax.annotation.Resource;
import javax.ejb.ActivationConfigProperty;
import javax.ejb.EJB;
import javax.ejb.MessageDriven;
import javax.ejb.MessageDrivenContext;
import javax.jms.JMSException;
import javax.jms.Message;
import javax.jms.MessageListener;
import javax.jms.TextMessage;

@MessageDriven(activationConfig = {
    @ActivationConfigProperty(propertyName = "destinationLookup", propertyValue = "jms/Queue"),
    @ActivationConfigProperty(propertyName = "destinationType", propertyValue =
"javax.jms.Queue")
})
public class BasicMessageBean implements MessageListener {
    @EJB VisitorStatBean vs;

    @Resource
    private MessageDrivenContext mdc;

    public BasicMessageBean() {
    }

    @Override
    public void onMessage(Message message) {
        try {
            if(message instanceof TextMessage){
                TextMessage msg= (TextMessage) message;
                vs.addVisitor(msg.getText());
            }
        }
        catch (JMSException e) {
            mdc.setRollbackOnly();
        }
    }
}
```

}

Step 5:

Before deploying and running the application, Glassfish Server setting is required.

Browse the path:

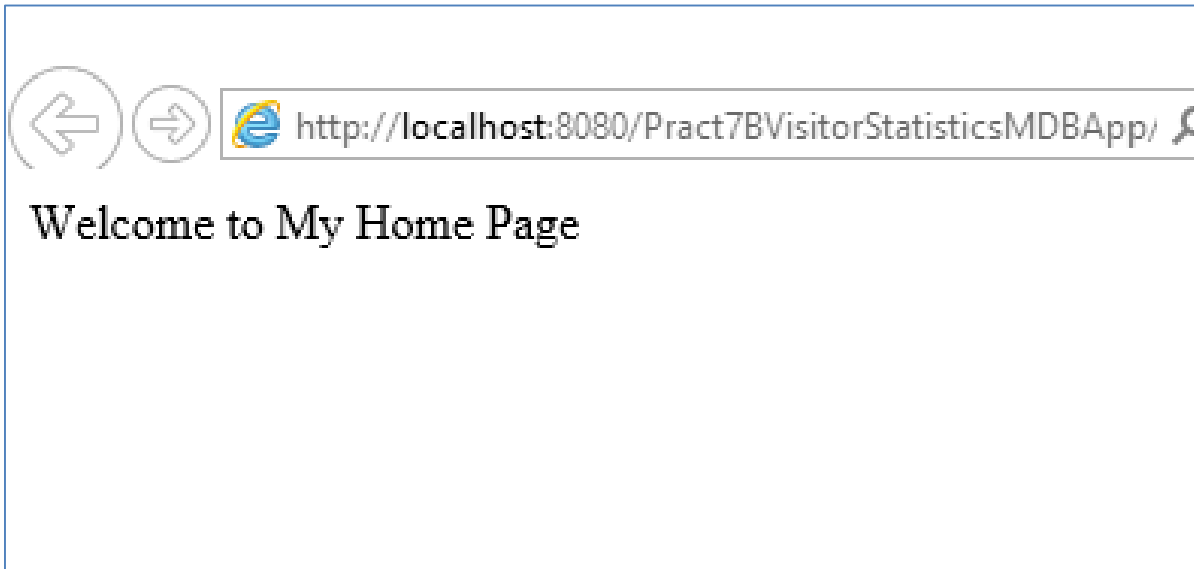
Localhost:4848 on any browser.

Find Resources -> connectors -> Connector Resources double click on Connector Resources -> click on 'New' Button -> write JNDI name as -> jms/QueryFactory.

Find Admin Object Resources and double click on that -> click on 'New' Button -> write JNDI name as -> jms/Queue.

Now run index.jsp file.

OUTPUT:



Q.7 c)_ Develop simple Marks Entry Application to demonstrate accessing Database using EJB.

CODE:

Step 1:

Create web application as pract7CMarksApp.

Step 2:

Create database marksdb

Step 3:

Create tables marks in marksdb database as:

create table marks (id int primary key auto_increment, sname varchar(35), marks1 int, marks2 int, marks3 int);

step 4:

index.jsp

```
<% @page import="ejb.MarksEntryBean"%>
```

```
<% @page import="javax.naming.InitialContext"%>
```

```
<% @page contentType="text/html" pageEncoding="UTF-8"%>
```

```
<!DOCTYPE html>
```

```
<%!
```

```
private static MarksEntryBean obj;
```

```
public void jspInit()
```

```
{
```

```
    try
```

```
    {
```

```
        InitialContext ic=new InitialContext();
```

```
        obj=(MarksEntryBean)ic.lookup("java:global/Pract7CMarksApp/MarksEntryBean");
```

```
    }
```

```
    catch(Exception e)
```

```
    {
```



```
System.out.println(e);

}

}

%>

<%

if(request.getParameter("InsertMarks")!=null)

{

String sname;

int marks1, marks2, marks3;

sname = request.getParameter("sname");

marks1=Integer.parseInt(request.getParameter("m1"));

marks2=Integer.parseInt(request.getParameter("m2"));

marks3=Integer.parseInt(request.getParameter("m3"));

obj.addMarks(sname,marks1,marks2,marks3);

out.print("Marks entered successfully..!!!!");

}

%>

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>JSP Page</title>

</head>

<body>

<h2>Enter Details</h2>

<form name="result" method="post">

Enter student's name: <input type='text' name="sname" /><br>
```

Enter subject 1 marks: <input type='text' name="m1" />

Enter subject 2 marks: <input type='text' name="m2" />

Enter subject 3 marks: <input type='text' name="m3" />

<input type='submit' name="InsertMarks" />

</form>

</body>

</html>

Step 4:

create stateful java bean as select source package -> session bean -> class name -> MarksEntryBean -> package -> ejb -> bean type-> stateful -> don't select Local / Remote -> finish.

```
package ejb;
```

```
import java.sql.*;
```

```
import javax.ejb.Stateful;
```

```
@Stateful
```

```
public class MarksEntryBean {
```

```
    String sname;
```

```
    int m1,m2,m3;
```

```
    Connection con=null;
```

```
    Statement st=null;
```

```
    String query="";
```

```
    public void addMarks(String sname,int m1,int m2,int m3)
```

```
    {
```

```
        try
```

```
        {
```

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

```
            con=DriverManager.getConnection("jdbc:mysql://localhost:3306/marksdbs", "root","tiger");
```

```
st=con.createStatement();

query="insert into marks (sname,marks1,marks2,marks3) values
('"+sname+"','"+m1+"','"+m2+"','"+m3+"')";

st.executeUpdate(query);

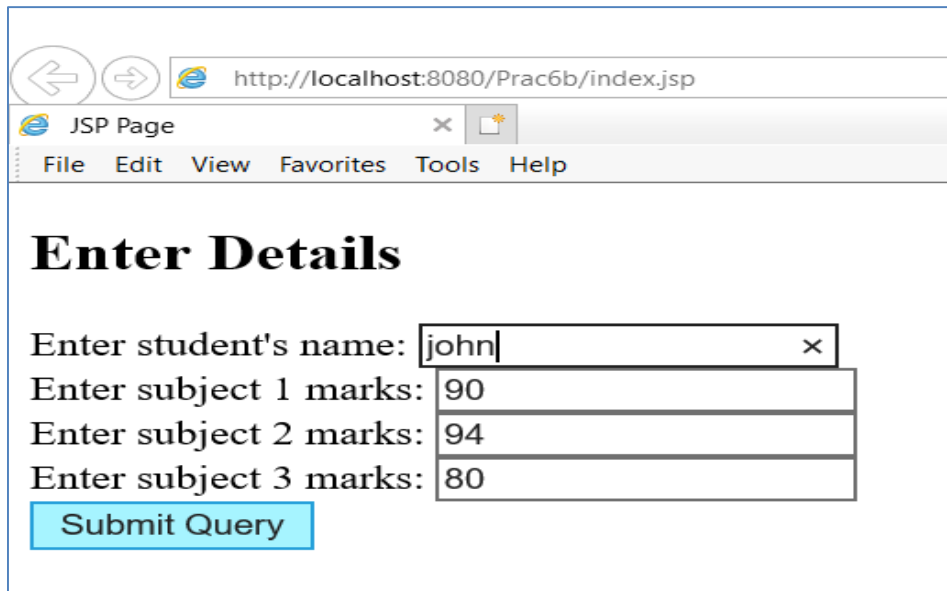
System.out.print("Marks entered sucessfully!!");

}

catch(Exception e){System.out.println(e);}

}

}
```

OUTPUT:

http://localhost:8080/Prac6b/index.jsp

JSP Page

File Edit View Favorites Tools Help

Enter Details

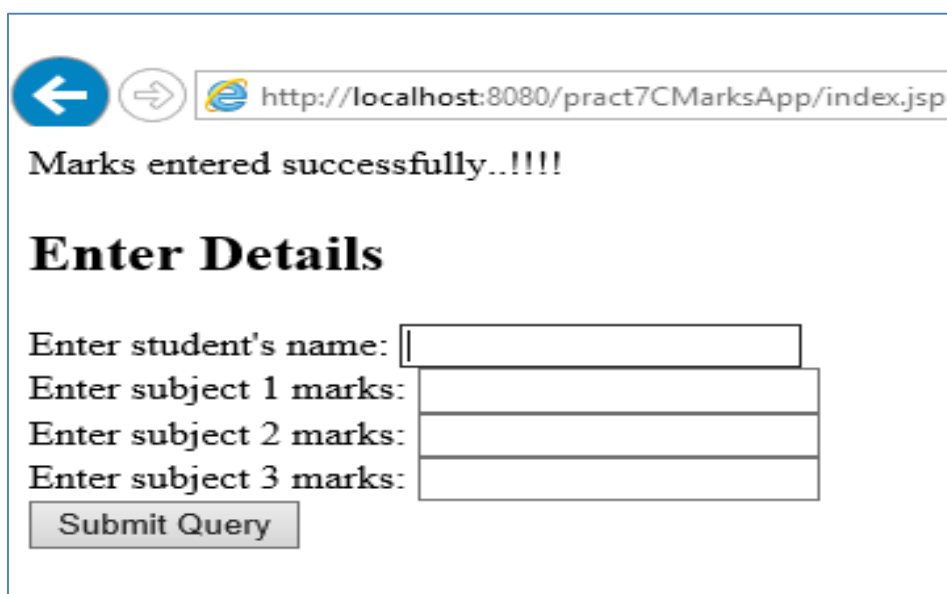
Enter student's name: john

Enter subject 1 marks: 90

Enter subject 2 marks: 94

Enter subject 3 marks: 80

Submit Query



http://localhost:8080/pract7CMarksApp/index.jsp

Marks entered successfully..!!!!

Enter Details

Enter student's name:

Enter subject 1 marks:

Enter subject 2 marks:

Enter subject 3 marks:

Submit Query

PRACTICAL 9**Q.9 a) Develop a JPA Application to demonstrate use of ORM associations.****CODE:****index.html**

```
<html>
  <head>
    <title>TODO supply a title</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
  </head>
  <body>
    User Details <hr><br><br>
    <form action="userview.jsp" >
      Name <input type="text" name="uname" maxlength="20"><br>
      User Type <input type="text" name="utype" maxlength="20">
    <br><input type="submit" value="submit">
  </form>

  </body>
</html>
```

userview.jsp

```
<% @page import="java.util.List"%>
<% @page import="java.util.Iterator"%>
<% @page import="hibernate.User"%>
<% !
  SessionFactory sf;
  org.hibernate.Session ss;
  List<hibernate.User> User;
%>
<%
  sf = new Configuration().configure().buildSessionFactory();
  ss= sf.openSession();
  Transaction tx=null;
  User ur=new User();
  try
  {
```

```
tx=ss.beginTransaction();

String uname=request.getParameter("uname");
String utype=request.getParameter("utype");
ur.setUname(uname);

ur.setUtype(utype);
ss.save(ur);
tx.commit();

}

catch(Exception e){ out.println("Error"+e.getMessage());
}

try

{
ss.beginTransaction();

User=ss.createQuery("from User").list();
}

catch(Exception e){ }

%>
<html>
<head>
<title>Guest View</title>
</head>
<body>
Guest View
Click here to go <a href="index.html"> BACK </a>
<br><br>

<%
Iterator it=User.iterator();

while(it.hasNext())

{
```

```
User eachrecord=(User)it.next();
out.print(eachrecord.getUid()+" ");
out.print(eachrecord.getUname()+"<br>");
out.print(eachrecord.getUtype()+"<br><hr>");
}
%>
</body>
</html>
```

hibernate.revenge.xml

```
<hibernate-reverse-engineering>
<schema-selection match-catalog="userdb"/>
<table-filter match-name="user"/>
</hibernate-reverse-engineering>
```

hibernate.cfg.xml

```
<hibernate-configuration>
<session-factory>
<property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
<property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
<property
name="hibernate.connection.url">jdbc:mysql://localhost:3306/userdb?zeroDateTimeBehavior=conve
rtToNull</property>
<property name="hibernate.connection.username">root</property>
<property name="hibernate.connection.password">tiger</property>
</session-factory>
</hibernate-configuration>
```

User.hbm.xml

```
<hibernate-mapping>
<class optimistic-lock="version" catalog="userdb" table="user" name="hibernate.User">
<id name="uid" type="java.lang.Integer">
<column name="uid"/>
<generator class="identity"/>
</id>
<property name="uname" type="string">
```

```
<column name="uname" length="20"/>
</property>
<property name="utype" type="string">
<column name="utype" length="100"/>
</property>
</class>
</hibernate-mapping>
```

User.java

```
package hibernate;

public class User implements java.io.Serializable {
    private Integer uid;
    private String uname;
    private String utype;
    public User() { }
    public User(String uname, String utype) {
        this.uname = uname;
        this.utype = utype;
    }
    public Integer getUid() {

        return this.uid;

    }
    public void setUid(Integer uid) {

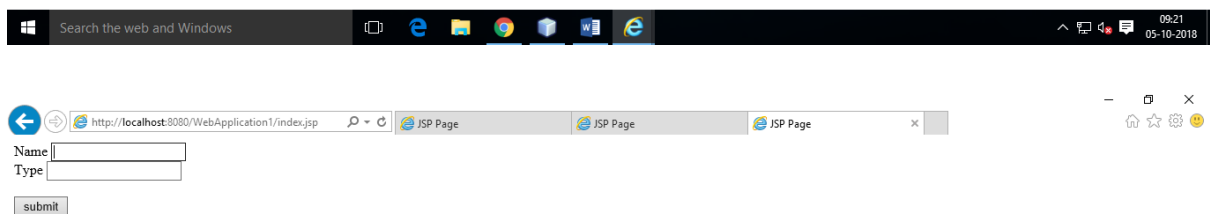
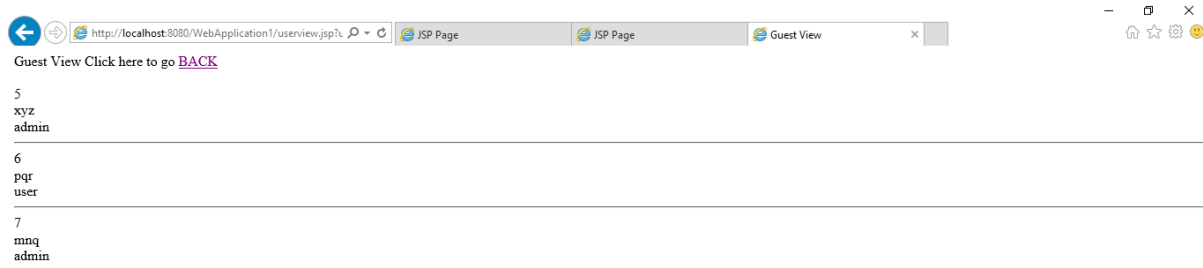
        this.uid = uid;

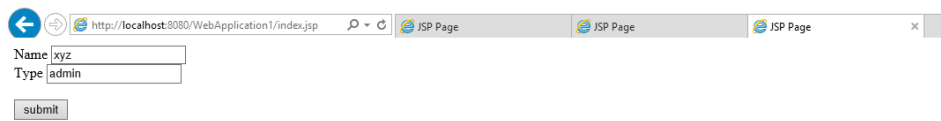
    }
    public String getUname()
    {
        return this.uname;
    }
    public void setUname(String uname)
    {
        this.uname = uname;
    }
    public String getUtype()
    {
```

```
return this.utype;  
}
```

```
public void setUtype(String utype)  
{  
this.utype = utype;  
}  
}
```

OUTPUT:





The screenshot shows a web browser window with the address bar displaying `http://localhost:8080/WebApplication1/index.jsp`. The browser has three tabs, all titled "JSP Page". Below the address bar, there is a login form with two input fields: "Name" containing the text "xyz" and "Type" containing the text "admin". A "submit" button is located below these fields.



Q.9 b) Develop a Hibernate application to store Feedback of Website Visitor in MySQL Database.**Hibernate – Feedback of Website Visitor (on index paper)****Step 1: MySql Command:-**

Select Services -> right click on database -> connect -> password -> ok -> again right click on database -> create database -> db -> ok.

Expand db -> Select and right click table -> click on Execute command ->

Create table guestbook (no int primary key auto_increment, name varchar(20), msg varchar(100), dt varchar(40));

Step 2: Create a Hibernate Project :-

File -> New Project -> Java Web -> Web application -> Next -> give the project name -> browse the location as required -> select the checkbox – “dedicated folder for storing libraries” -> Next

Select glassfish server -> next

Select frame work - hibernate -> select the respective database connection -> finish.

Step 3: Adding Reverse Engineering File :-

Right click on Project -> new -> other -> select Hibernate -> Hibernate Reverse Engineering wizard file type -> next -> file name (hibernate.reveng) , folder -> click on browse and select src->java -> next -> select guestbook table name from the available tables option -> click add (select the checkbox – include related files) -> finish.

Step 4: Adding Hibernate mapping files and POJOs from Database file type:-

Right click on Project -> new -> other -> select Hibernate -> Hibernate mapping files and POJOs from Database file type) -> next -> keep the default configuration file name file name (hibernate.cfg) and Hibernate Reverse Engineering File (hibernate.reveng) -> type the package name (hibernate) -> finish.

Step 5: Creating JSP File :-

Right click on project -> new -> JSP -> filename -> guestbookview -> select radiobutton -> JSP file (Standard syntax) -> Finish.

CODE:**File name - Guestbook.java**

```
package hibernate;

public class Guestbook implements java.io.Serializable {

    private Integer no;

    private String name;

    private String msg;

    private String dt;

    public Guestbook() {

    }

    public Guestbook(String name, String msg, String dt) {

        this.name = name;

        this.msg = msg;

        this.dt = dt;

    }

    public Integer getNo() {

        return this.no;

    }

    public void setNo(Integer no) {

        this.no = no;

    }

    public String getName() {

        return this.name;

    }

    public void setName(String name) {
```

```
this.name = name;

}

public String getMsg() {

    return this.msg;

}

public void setMsg(String msg) {

    this.msg = msg;

}

public String getDt() {

    return this.dt;

}

public void setDt(String dt) {

    this.dt = dt;

}

}
```

File name - hibernate.cfg.xml

```
<hibernate-configuration>

<session-factory>

<property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>

<property name="hibernate.connection.driver_class">com.mysql.jdbc.Driver</property>

<property name="hibernate.connection.url">jdbc:mysql://localhost:3306/db</property>

<property name="hibernate.connection.username">root</property>

<property name="hibernate.connection.password">tiger</property>

<mapping resource="hibernate/Guestbook.hbm.xml"/>

</session-factory>
```

</hibernate-configuration>

File name - Guestbook.hbm.xml

```
<hibernate-mapping>

<class name="hibernate.Guestbook" table="guestbook" catalog="db">

<id name="no" type="java.lang.Integer">

<column name="no" />

<generator class="identity" />

</id>

<property name="name" type="string">

<column name="name" length="20" />

</property>

<property name="msg" type="string">

<column name="msg" length="100" />

</property>

<property name="dt" type="string">

<column name="dt" length="40" />

</property>

</class>

</hibernate-mapping>
```

File name - index.jsp

```
<html>

<head>

<title>Guest Book</title>

</head>

<body>
```

Guest Book <hr>

<form action="guestbookview.jsp" >

Name <input type="text" name="name" maxlength="20">

Message <textarea rows="5" cols="40" maxlength="100" name="msg"></textarea>

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

</form>

</body>

</html>

File name - guestbookview.jsp

<% @page import="org.hibernate.SessionFactory"%>

<% @page import="org.hibernate.Session"%>

<% @page import="org.hibernate.cfg.Configuration"%>

<% @page import="org.hibernate.Transaction"%>

<% @page import="java.util.List"%>

<% @page import="java.util.Iterator"%>

<% @page import="hibernate.Guestbook"%>

<%!

SessionFactory sf;

org.hibernate.Session ss;

List<hibernate.Guestbook> gbook;

%>

<%

sf = new Configuration().configure().buildSessionFactory();

ss= sf.openSession();

Transaction tx=null;

```
Guestbook gb=new Guestbook();

try

{

    tx=ss.beginTransaction();

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

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

    String dt=new java.util.Date().toString();

    gb.setName(name);

    gb.setMsg(msg);

    gb.setDt(dt);

    ss.save(gb);

    tx.commit();

}

catch(Exception e){ out.println("Error"+e.getMessage()); }

try

{ ss.beginTransaction();

    gbook=ss.createQuery("from Guestbook").list();

}

catch(Exception e){ }

%>

<html>

<head>

<title>Guest View</title>

</head>

<body>

    Guest View
```

Click here to go [BACK](index.jsp)

```
<br><br>

<%   Iterator it=gbook.iterator();

    while(it.hasNext())

    {

        Guestbook eachrecord=(Guestbook)it.next();

        out.print(eachrecord.getDt()+"  ");

        out.print(eachrecord.getName()+"<br>");

        out.print(eachrecord.getMsg()+"<br><hr>");

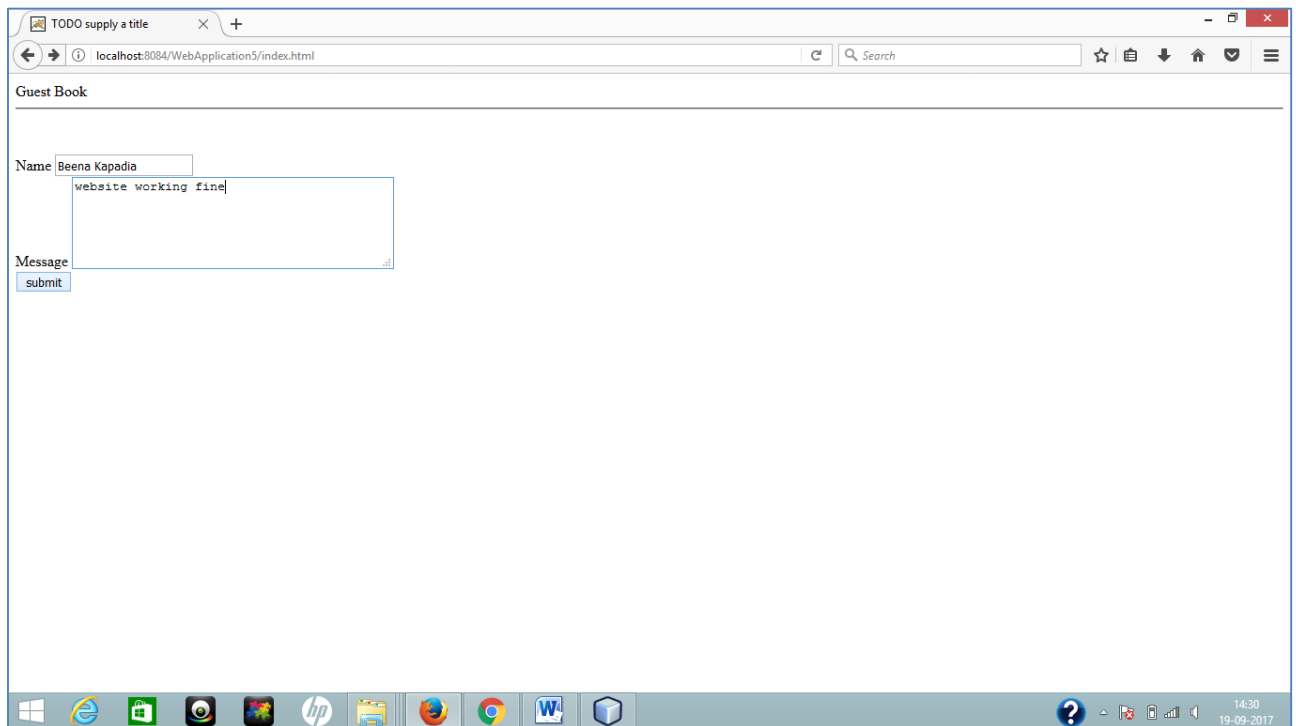
    }

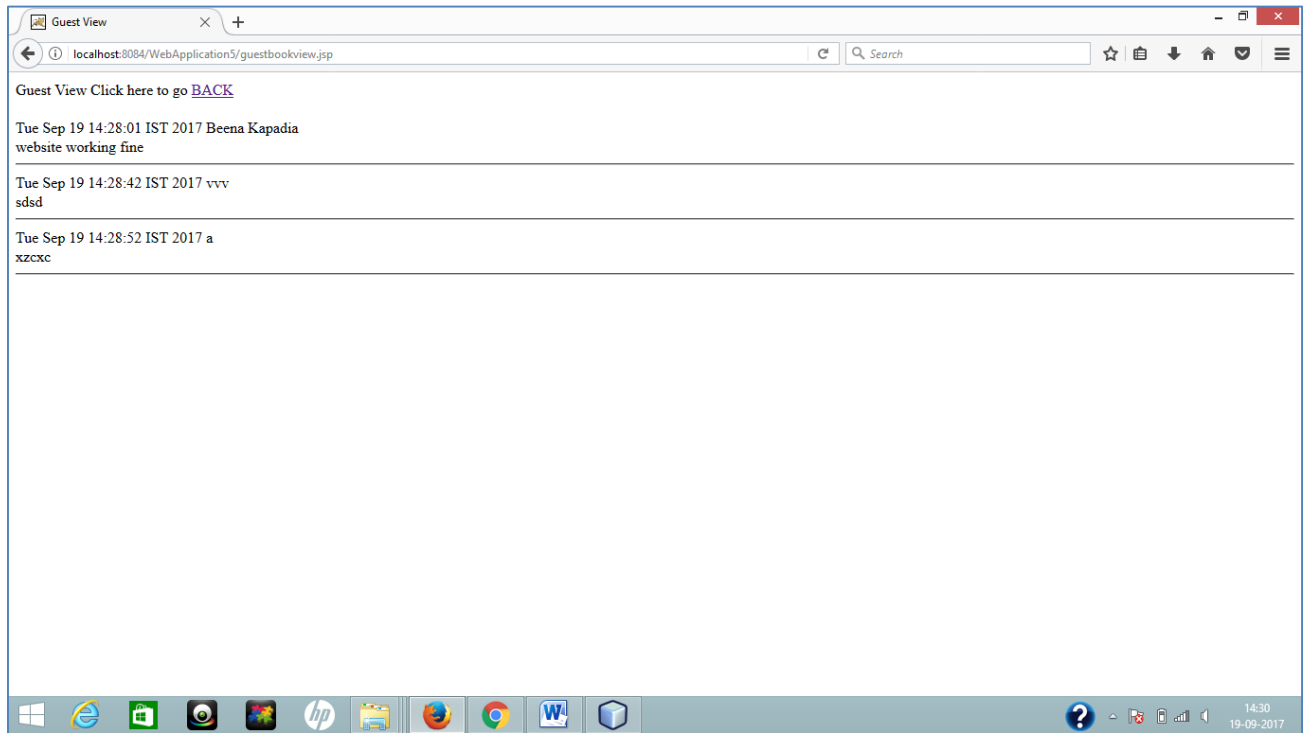
%>

</body>

</html>
```

OUTPUT:





Q.9 c) Develop a Hibernate application to store and retrieve employee details in MySQL Database.

CODE:

index.html

```
<html>
<head>
<title>Employee Details</title>
</head>
<body>
Employee Details <hr><br><br>
<form action="empview.jsp" >
    Name <input type="text" name="name" maxlength="20"><br>
    Salary <input type="text" name="salary" maxlength="20"><br>
    Designation <input type="text" name="designation" maxlength="20">
<br><input type="submit" value="submit">
</form>
</body>
</html>
```

empview.jsp

```
<% @page import="org.hibernate.SessionFactory"%>
<% @page import="org.hibernate.Session"%>
<% @page import="org.hibernate.cfg.Configuration"%>
<% @page import="org.hibernate.Transaction"%>
<% @page import="java.util.List"%>
<% @page import="java.util.Iterator"%>
<% @page import="hibernate.Emp"%>

<% !
SessionFactory sf;
org.hibernate.Session ss;
List<hibernate.Emp> Emplist;
%>

<%
sf = new Configuration().configure().buildSessionFactory();
```

```
ss= sf.openSession();
```

```
Transaction tx=null;
```

```
Emp em=new Emp();
```

```
try
```

```
{
```

```
tx=ss.beginTransaction();
```

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

```
String Salary=request.getParameter("salary");
```

```
String Designation=request.getParameter("designation");
```

```
System.out.print("Name....."+Name+" "+Salary+" "+Designation);
```

```
em.setName(Name);
```

```
em.setSalary(Salary);
```

```
em.setDesignation(Designation);
```

```
System.out.print("set done.....");
```

```
ss.save(em);
```

```
System.out.print("save done.....");
```

```
tx.commit();
```

```
System.out.print("commit done.....");
```

```
}
```

```
catch(Exception e){ out.println("Error"+e.getMessage()); }
```

```
try
```

```
{
```

```
ss.beginTransaction();
```

```
Emplist=ss.createQuery("from Emp").list();
```

```
}
```

```
catch(Exception e){ }
```

```
%>
```

```
<html>
```

```
<head>
```

```
<title>Employee View</title>
```

```
</head>
```

```
<body>
```

Employee View

Click here to go BACK

<%

Iterator it=Emplist.iterator();

while(it.hasNext())

{

Emp eachrecord=(Emp)it.next();

out.print(eachrecord.getName()+"
");

out.print(eachrecord.getSalary()+"
<hr>");

out.print(eachrecord.getDesignation()+"
<hr>");

}

%>

hibernate.revenge.xml

<hibernate-reverse-engineering>

<schema-selection match-catalog="employeedb"/>

<table-filter match-name="emp"/>

</hibernate-reverse-engineering>

hibernate1.cfg.xml

<hibernate-configuration>

<session-factory>

<property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>

<property name="hibernate.connection.driver_class">com.mysql.jdbc.Driver</property>

<property

name="hibernate.connection.url">jdbc:mysql://localhost:3306/employeedb?zeroDateTimeBehavior=c

onvertToNull</property>

<property name="hibernate.connection.username">root</property>

<property name="hibernate.connection.password">tiger</property>

</session-factory>

</hibernate-configuration>

Emp.hbm.xml

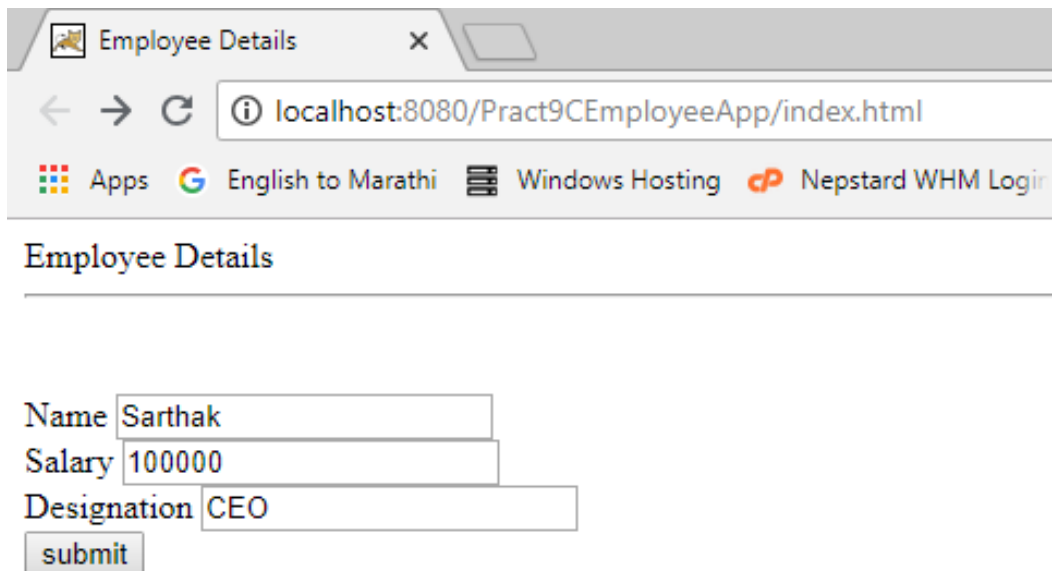
```
<hibernate-mapping>
<class optimistic-lock="version" catalog="employeeedb" table="emp" name="hibernate.Emp">
<id name="id" type="java.lang.Integer">
<column name="id"/>
<generator class="identity"/>
</id>
<property name="name" type="string">
<column name="name" length="20"/>
</property>
<property name="salary" type="string">
<column name="salary" length="20"/>
</property>
<property name="designation" type="string">
<column name="designation" length="20"/>
</property>
</class>
</hibernate-mapping>
```

Emp.java

```
package hibernate;

public class Emp implements java.io.Serializable {
private Integer id;
private String name;
private String salary;
private String designation;
public Emp() { }
public Emp(String name, String salary, String designation) {
this.name = name;
this.salary = salary;
this.designation = designation;
}
public Integer getId() {
return this.id;
}
public void setId(Integer id) {
this.id = id;
}
```

```
public String getName() {  
    return this.name;  
}  
  
public void setName(String name) {  
    this.name = name;  
}  
  
public String getSalary() {  
    return this.salary;  
}  
  
public void setSalary(String salary) {  
    this.salary = salary;  
}  
  
public String getDesignation() {  
    return this.designation;  
}  
  
public void setDesignation(String designation) {  
    this.designation = designation;  
}  
}
```

OUTPUT:

Employee Details

Name

Salary

Designation

