**Practical No:-1**

**Aim:-**Write a program to implement the connection oriented echo client server application.

**Source Code:-**

**Server:**

package javaapplication1;

import java.io.\*;

import java.net.\*;

public class Main

{

public static void main(String[] args)

{

try{

ServerSocket ss = new ServerSocket(8080);

Socket s = ss.accept();

InputStream i = s.getInputStream();

DataInputStream dis = new DataInputStream(i);

System.out.println(dis.readUTF());

}

catch(Exception e){

System.out.print(e);

}

}

}

**Client :**

package javaapplication1;

import java.io.\*;

import java.net.\*;

public class client

{

public static void main(String[] args) {

try{

Socket s = new Socket("localhost",8080);

OutputStream o = s.getOutputStream();

DataOutputStream dos = new DataOutputStream(o);

dos.writeUTF("client connected to server");

}

catch(Exception e) {

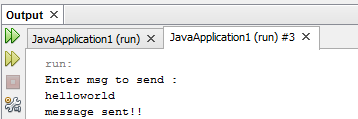
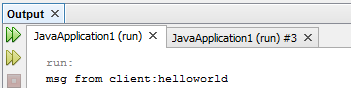
System.out.print(e);

}

}

}

**Output :-**

****

**Practical No:-2**

**Aim:-**Write a program in which the client sends the string to the server and server sends acknowledgement to the client.

**Source Code:-**

**Server :**

package javaapplication1;

import java.io.\*;

import java.net.\*;

public class server

{

public static void main(String args[]) {

try {

String str3;

String str4;

ServerSocket ss = new ServerSocket(8080);

while(true)

{

Socket s = ss.accept();

BufferedReader in\_clint = new BufferedReader(new InputStreamReader(s.getInputStream()));

DataOutputStream out\_clinet = new DataOutputStream(s.getOutputStream());

str3 = in\_clint.readLine();

str4 = str3 + "->Recived" + '\n';

out\_clinet.writeBytes(str4);

System.out.println(str3);

}

}

catch(Exception e)

{

System.out.print(e);

}

}

}

Client :

package javaapplication1;

import java.io.\*;

import java.net.\*;

public class client

{

public static void main(String args[]){

try {

Socket s = new Socket("localhost",8080);

if(s.isConnected())

{

System.out.println("Connceted");

}

String str;

String str1;

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

DataOutputStream dout = new DataOutputStream(s.getOutputStream());

BufferedReaderin\_server = new BufferedReader(new InputStreamReader(s.getInputStream()));

str = br.readLine();

dout.writeBytes(str+"\n");

str1 = in\_server.readLine();

System.out.println("From Server" +str1);

s.close();

}

catch(Exception e)

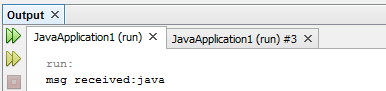
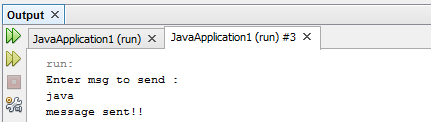
{

System.out.print(e);

}

}

**Output :-**

****

**Practical No:-3**

**Aim:-**Create Chat application using either TCP or UDP protocol.

**Source Code:-**

**Server :**

package javaapplication1;

import java.io.\*;

import java.net.\*;

public class p3server

{

public static void main(String[] args){

try{

ServerSocket ss = new ServerSocket(8080);

Socket s = ss.accept();

BufferedReaderkey Read = new BufferedReader(new InputStreamReader(System.in));

OutputStream ostream = s.getOutputStream();

PrintWriter pwrite = new PrintWriter(ostream,true);

InputStream istream = s.getInputStream();

BufferedReader reciveRead = new BufferedReader(new InputStreamReader(istream));

String reciveMessage,sendMessage;

while(true)

{

if((reciveMessage=reciveRead.readLine())!= null)

{

System.out.println(reciveMessage);

}

sendMessage = keyRead.readLine();

pwrite.println(sendMessage);

pwrite.flush();

}

}

catch(Exception e)

{

System.out.print(e);

}

}

}

**Client :**

Package javaapplication1;

Import java.io.\*;

import java.net.\*;

public class client

{

public static void main(String[] args){

try{

Socket s = new Socket("localhost",8080);

if(s.isConnected())

{

System.out.println("Connected");

}

BufferedReaderkeyRead = new BufferedReader(new InputStreamReader(System.in));

OutputStreamostream = s.getOutputStream();

PrintWriterpwrite = new PrintWriter(ostream,true);

InputStreamistream = s.getInputStream();

BufferedReaderreceiveRead = new BufferedReader(new InputStreamReader(istream));

System.out.println("start chat,type& press enter key");

String receiveMessage ,sendMessage;

while(true)

{

sendMessage = keyRead.readLine();

pwrite.println(sendMessage);

pwrite.flush();

if((receiveMessage=receiveRead.readLine())!=null)

{

System.out.println(receiveMessage);

}

}

}

catch(Exception e)

{

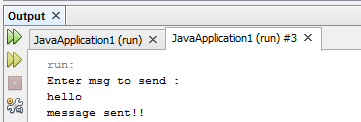
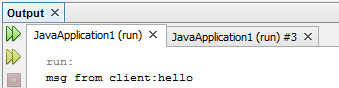
System.out.print(e);

}

}

}

**Output :-**



**Practical No:-4**

**Aim:-**Write a program that implements a simple client/server application. The client sends data to a server the server receives the data, uses it to produce a result and then sends the result back to the client. The client displays the result on the console.

(A) The data send from the client is a number and the result produce by the server is count even number and odd number.

**Source Code:-**

**Server :**

package javaapplication1;

import java.io.\*;

import java.net.\*;

class server

{

public static void main(String args[])throws Exception

{

ServerSocket ss=new ServerSocket(8080);

Socket s=ss.accept();

InputStream in=s.getInputStream();

OutputStream o=s.getOutputStream();

int n11,c=0,d=0;

int a=in.read();

for(int i=0;i<a;i++)

{ n11=in.read();

if(n11%2==0)

{

c=c+1;

}

else

{

d=d+1;

}

}

o.write(c);

o.write(d) ;

o.close();

in.close();

s.close();

}

}

**Client:**

package javaapplication1;

import java.io.\*;

import java.net.\*;

class client

{

public static void main(String args[])throws Exception

{

Socket s=new Socket("localhost",8080);

OutputStream out=s.getOutputStream();

InputStreami=s.getInputStream();

int c=Integer.parseInt(args[0]);

int l=args.length;

out.write(l);

for(int j=0;j<l;j++)

{

int n1=Integer.parseInt(args[j]);

out.write(n1);

}

int f=i.read();

System.out.println("Even no:"+f);

int h=i.read();

System.out.println("odd no:"+h);

i.close();

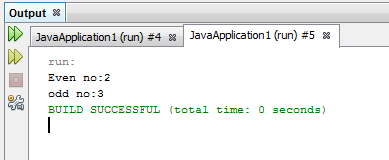
out.close();

s.close();

}

}

**Output :-**



(B)The data send from the client its two numbers and the result produce by the server is addition of two numbers.

**Source Code:-**

**Server:**

import java.io.\*;

import java.net.\*;

import java.lang.\*;

class server

{

public static void main(String args[]) throws Exception

{

int n1,n2,ans;

System.out.println("Server is starting...");

ServerSocket ss=new ServerSocket(8080);

System.out.println("Server is waiting for client Requsting");

Socket s=ss.accept();

OutputStream o=s.getOutputStream();

InputStream i=s.getInputStream();

n1=i.read();

n2=i.read();

ans=n1+n2;

o.write(ans);

System.out.println("Data send successfully");

i.close();

o.close();

ss.close();

System.out.println("client is connected");

} }

**Client**

import java.io.\*;

import java.net.\*;

importjava.lang.\*;

class client

{

public static void main(String args[]) throws Exception

{

String ip="localhost";

int port=8080;

intnum,sum;

Socket s=new Socket(ip,port);

OutputStream out=s.getOutputStream();

InputStream in=s.getInputStream();

sum=Integer.parseInt(args[0]);

num=Integer.parseInt(args[1]);

out.write(num);

out.write(sum);

sum=in.read();

System.out.println("ans is="+sum);

out.close();

in.close();

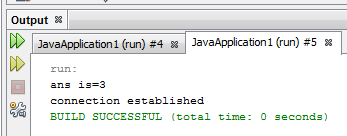
s.close();

System.out.println("connection established");

}

}

**Output :-**



**Practical No:-5**

**Aim:-**Implement any one sorting algorithm using TCP/UDP on Server application and Give Input on Client side and client should sorted output from server and display sorted on input side.

**Source Code:-**

**Server :**

package javaapplication1;

import java.io.\*;

import java.net.\*;

import java.util.\*;

public class server

{

public static void main(String[] args)

{

try

{

ServerSocket ss = new ServerSocket(3000);

Socket s = ss.accept();

System.out.println("Connected");

DataInputStream din = new DataInputStream(s.getInputStream());

DataOutputStream dout = new DataOutputStream(s.getOutputStream());

int i=0;

int n=din.readInt();

int a[] = new int[n];

System.out.println("data");

System.out.println("receiving data");

for(i=0;i<n;i++)

{

a[i]=din.readInt();

}

System.out.println("data received");

System.out.println("sorting data");

Arrays.sort(a);

System.out.println("data sorted");

System.out.println("sending data");

for(i=0;i<n;i++)

{

dout.writeInt(a[i]);

}

System.out.println("data sent sucessfully");

s.close();

ss.close();

}

catch(Exception e)

{

System.out.println(e);

}

}

}

**Client :**

package javaapplication1;

import java.io.\*;

import java.net.\*;

importjava.util.\*;

public class client

{

public static void main(String[] args)

{

try

{

Socket s = new Socket("localhost",3000);

if(s.isConnected())

{

System.out.println("Connected to server");

}

System.out.println("Enter size of array");

Scanner scan = new Scanner(System.in);

int n = scan.nextInt();

int a[] = new int[n];

System.out.println("Enter element to array");

DataOutputStreamdout = new DataOutputStream(s.getOutputStream());

dout.writeInt(n);

for(int i=0;i<n;i++)

{

int r = scan.nextInt();

dout.writeInt(r);

}

System.out.println("data sent sucessfully");

DataInputStream din = new DataInputStream(s.getInputStream());

int r;

System.out.println("receiving sorted data");

for(int i=0;i<n;i++)

{

r=din.readInt();

System.out.println(r +"");

}

s.close();

}

catch(Exception e)

{

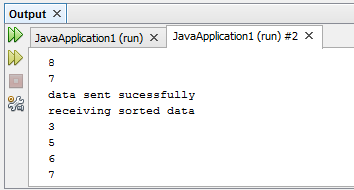
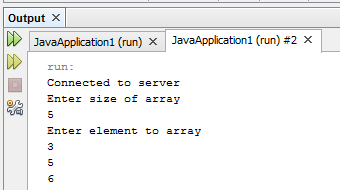
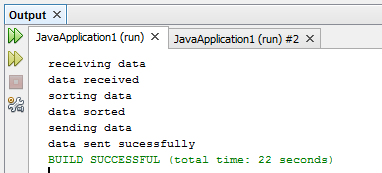
System.out.println(e);

}

}

}

**Output :-**

****

**Practical No:-6**

**Aim:-** Write a program that takes host name and output its IP Address, getting the IP address for corresponding host name and getting all IP Addresses related to the same hostname using getLocalHost, getByName and getAllByName method.

**Source Code:-**

**LocalHost :**

package javaapplication1;

import java.io.\*;

import java.net.\*;

public class p6

{

public static void main(String[] args) throws Exception

{

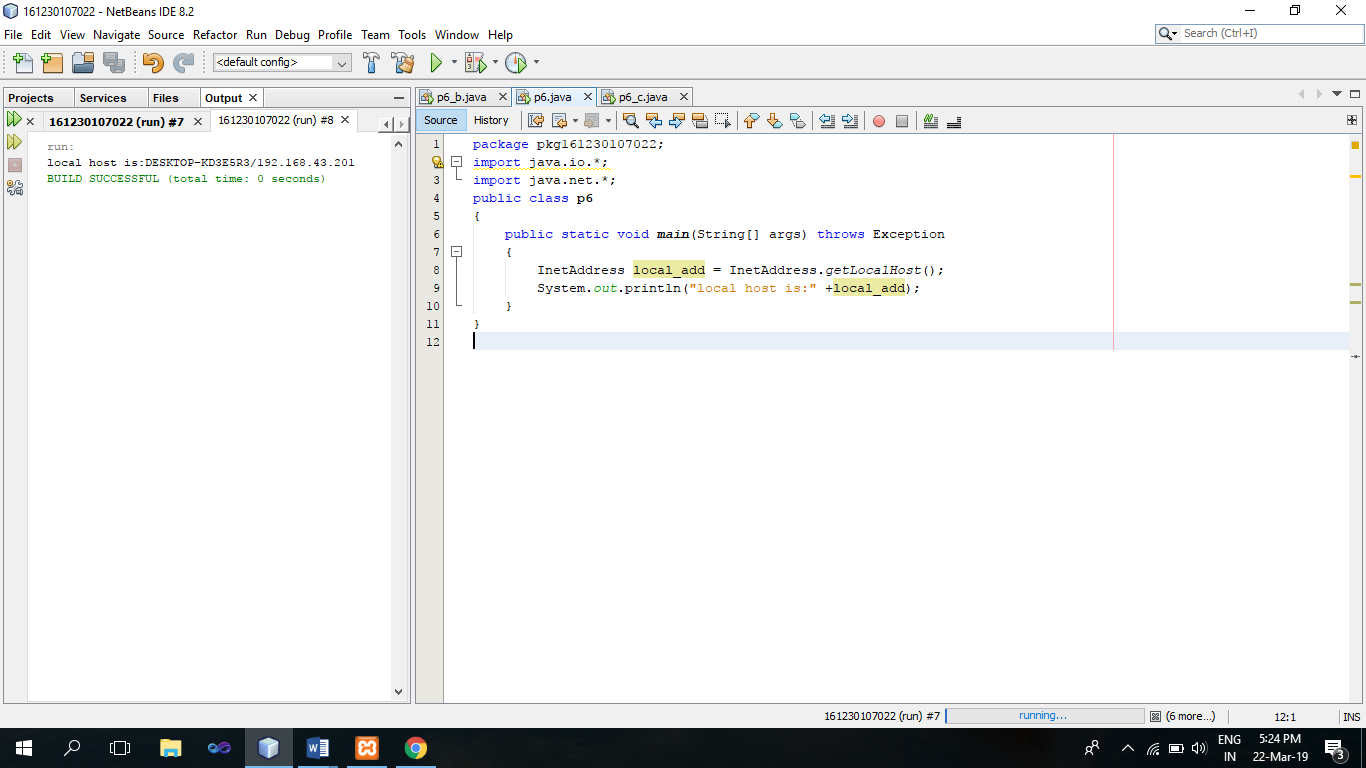
InetAddresslocal\_add = InetAddress.getLocalHost();

System.out.println("local host is:" +local\_add);

}

}

**Output:**



**GetByName :**

package javaapplication1;

import java.io.\*;

import java.net.\*;

public class p6\_b

{

public static void main(String[] args) throws UnknownHostException

{

InetAddress addr = InetAddress.getLocalHost();

System.out.println(addr);

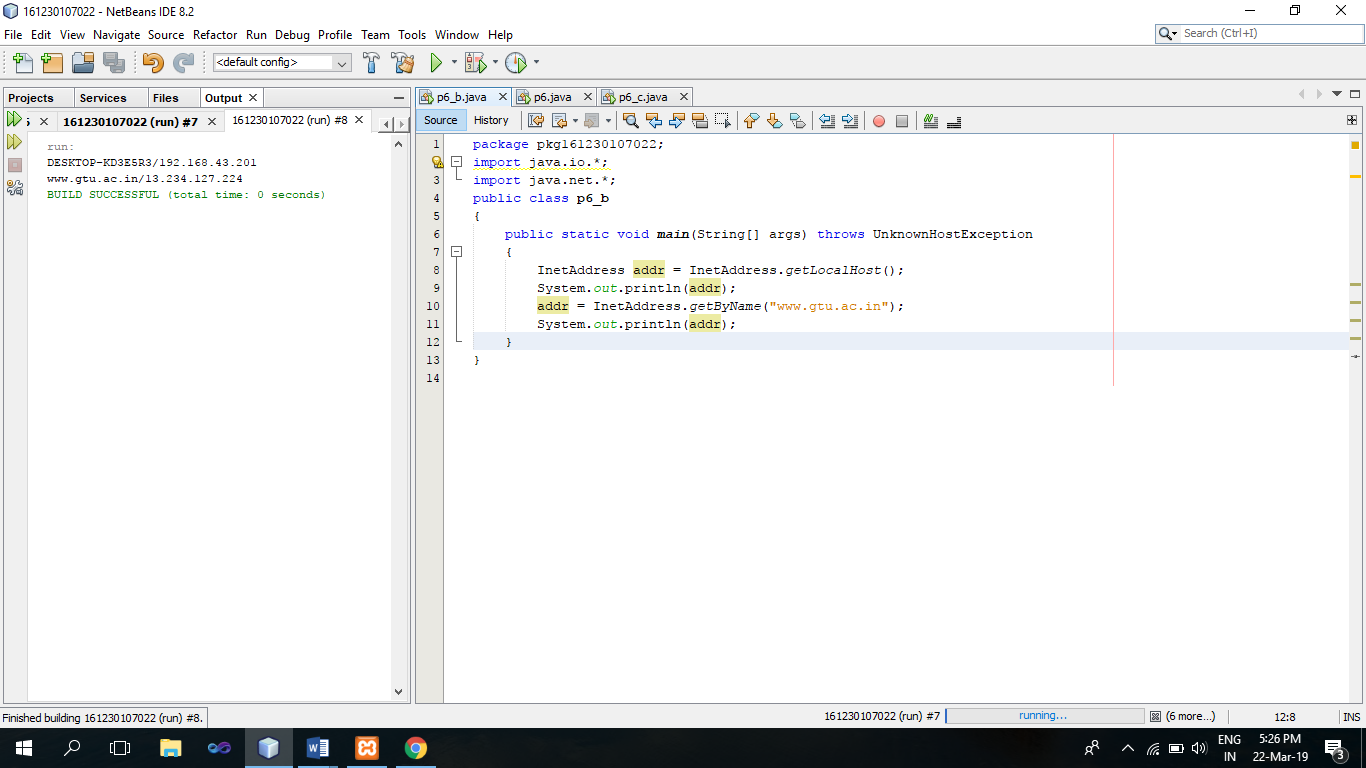
addr = InetAddress.getByName("www.gtu.ac.in");

System.out.println(addr);

}

}

**Output:**



**GetAllByName :**

package javaapplication1;

import java.io.\*;

import java.net.\*;

public class p6\_c

{

public static void main(String[] args) throws UnknownHostException

{

InetAddress[] addr = InetAddress.getAllByName("www.google.com");

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

{

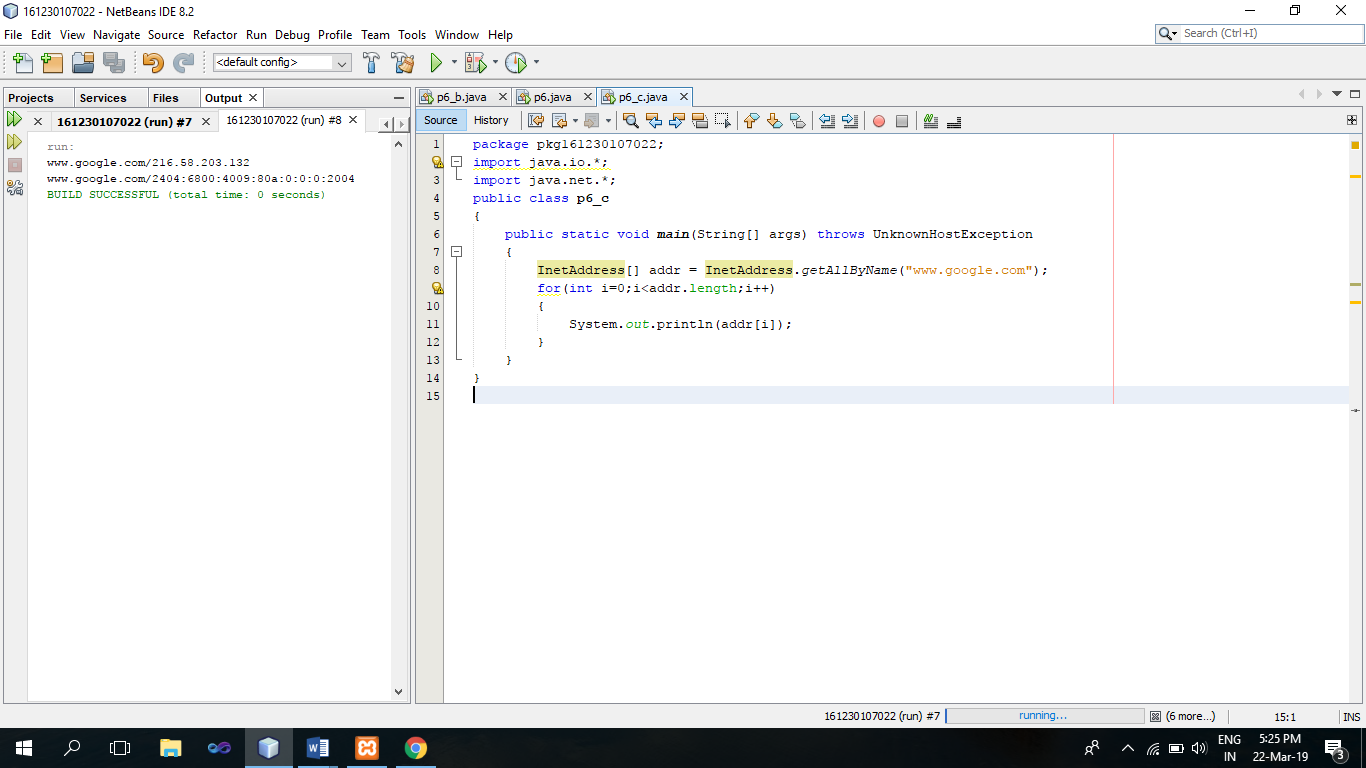
System.out.println(addr[i]);

}

}

}

**Output :**



**Practical No:-7**

**Aim:-**Write a program to Create Simple JDBC Application to insert and retrieve records from Mysql Database (Student Information System).

**Source Code:-**

package javaapplication1;

importjava.sql.\*;

importjava.lang.\*;

importjava.util.\*;

import java.net.\*;

public class pr7

{

public static void main(String[] args)

{

try

{

int c=1;

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

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost/student","root","");

System.out.println("connected to database");

while(c!=0)

{

System.out.println("1.insert\n2.display\n3.enter 0 to exit");

Scanner scan = new Scanner(System.in);

Scanner scan1 = new Scanner(System.in);

c=scan.nextInt();

switch(c)

{

case 1:

String query="Insert Into student\_info(name,branch,year) VALUES(?,?,?)";

PreparedStatementps = conn.prepareStatement(query);

int year1;

String name1,branch1;

System.out.println("enter name");

name1=scan1.nextLine();

System.out.println("enter branch");

branch1=scan1.nextLine();

System.out.println("enter year");

year1=Integer.parseInt(scan1.nextLine());

ps.setString(1, name1);

ps.setString(2,branch1);

ps.setInt(3, year1);

ps.executeUpdate();

System.out.println();

System.out.println("inserted record into table");

break;

case 2:

Statement stmt = conn.createStatement();

String sql ="SELECT \* FROM student\_info";

ResultSetrs = stmt.executeQuery(sql);

while(rs.next())

{

System.out.println("id:" +rs.getInt("id"));

System.out.println("name:" +rs.getString("name"));

System.out.println("branch:" +rs.getString("branch"));

System.out.println("year:" +rs.getInt("year"));

}

System.out.println("");

break;

}

}

}

catch(Exception e)

{

System.out.println(e);

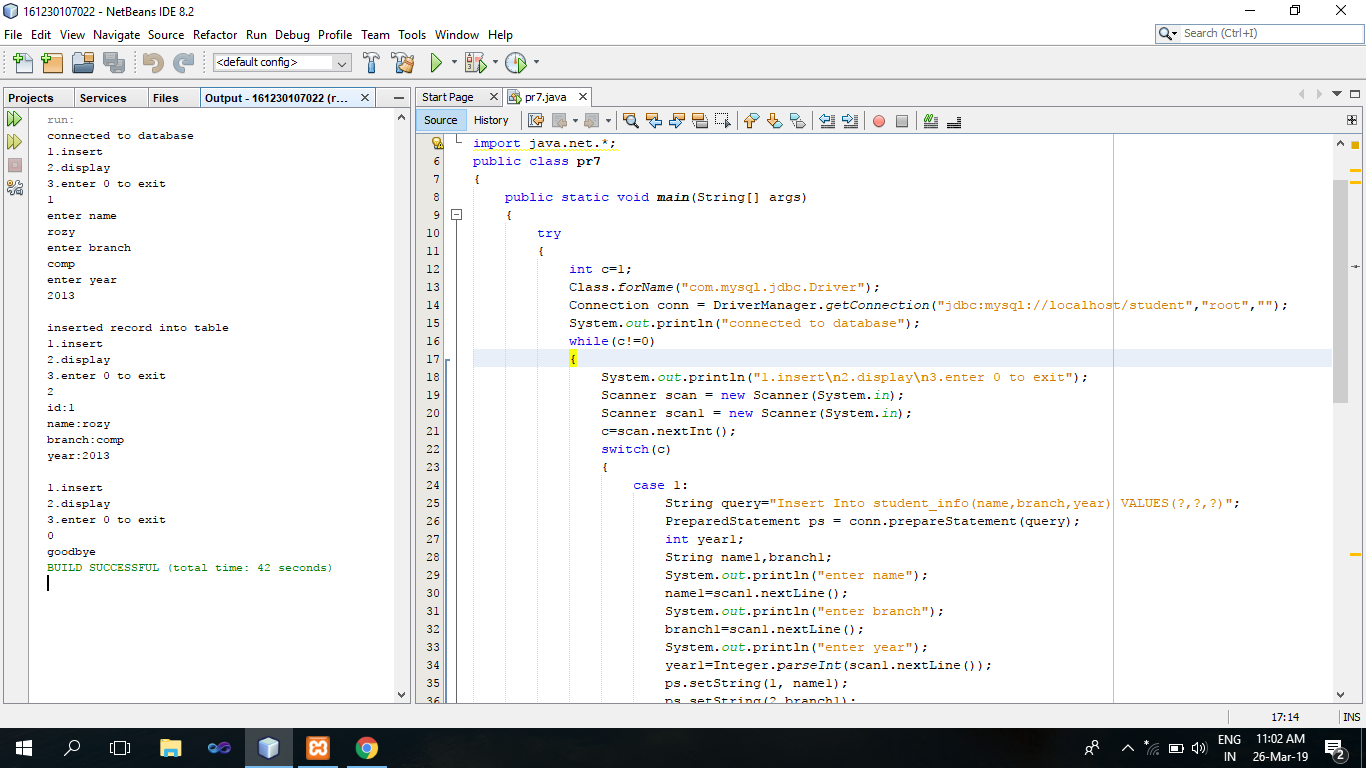
}

System.out.println("goodbye");

}

}

**Output :-**



**Practical No:-8**

**Aim:-**Write an HTML page which takes inputs for below mentioned fields and invokes a java servlet program which enters the fields in the database table. 1. EMP\_ID, 2. EMP\_NAME, 3. DEPT, 4. CITY.

**Source 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>

<h1>Please Enter Employee Details</h1>

<form action="data" method="get" >

<table><tr><td>Emp\_ID:</td><td><input type="text" name="E\_ID"></td></tr>

<tr><td>Emp\_Name:</td><td><input type="text" name="E\_Name"></td></tr>

<tr><td>Dept:</td><td><input type="text" name="Dept"></td></tr>

<tr><td> City:</td><td><input type="text" name="City"></td></tr>

<tr><td></td><td><input type="submit" value="Submit"></td></tr>

</table>

</form>

</body>

</html>

**data.java**

importjava.io.IOException;

importjava.io.PrintWriter;

importjava.sql.Connection;

importjava.sql.DriverManager;

importjava.sql.PreparedStatement;

importjava.sql.SQLException;

importjava.util.logging.Level;

importjava.util.logging.Logger;

importjavax.servlet.ServletException;

importjavax.servlet.http.HttpServlet;

importjavax.servlet.http.HttpServletRequest;

importjavax.servlet.http.HttpServletResponse;

public class data extends HttpServlet {

protected void processRequest(HttpServletRequest request, HttpServletResponse response)

throwsServletException, IOException, ClassNotFoundException, SQLException {

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

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/servlet", "root", "");

response.setContentType("text/html;charset=UTF-8");

PreparedStatementst = conn.prepareStatement("insert into emp (emp\_id,emp\_name,dept,city) values(?, ?, ?, ?)");

try (PrintWriter out = response.getWriter()) {

String eid = request.getParameter("E\_ID");

String e\_name = request.getParameter("E\_Name");

String dept = request.getParameter("Dept");

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

st.setString(1, eid);

st.setString(2, e\_name);

st.setString(3, dept);

st.setString(4, city);

st.executeUpdate();

st.close();

conn.close();

out.println("<!DOCTYPE html>");

out.println("<html>");

out.println("<head>");

out.println("<title>Servlet data</title>");

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

out.println("<body>");

out.println("<h1>Servlet data at " + eid + " " + e\_name + " " + dept + " " + city + "</h1>");

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

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

}

}

@Override

protected void doGet(HttpServletRequest request, HttpServletResponse response)

throwsServletException, IOException {

try {

processRequest(request, response);

} catch (ClassNotFoundException ex) {

Logger.getLogger(data.class.getName()).log(Level.SEVERE, null, ex);

} catch (SQLException ex) {

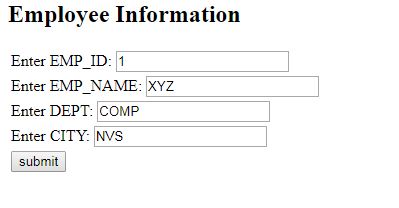
Logger.getLogger(data.class.getName()).log(Level.SEVERE, null, ex);

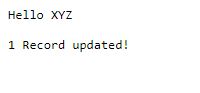
}

}

}

**Output :-**

****

****

**8(3).JPG**

**Practical No:-9**

**Aim:-**Write a HTML page which inputs the below mentioned fields and invoke the java Servlet program which enters the fields in Access Database using Prepared Statement.

1. ENROLMENT, 2. STUDENT\_NAME, 3. BRANCH, 4. YEAR.

**Source Code:-**

**index.html**

<html>

<head>

<title>prepare\_stm</title>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width">

</head>

<body>

<h1>Please Enter Student Details</h1>

<form action="StdDB" method="get" >

<table>

<tr>

<td>En\_NO</td><td><input type="text" name="en\_no"></td>

</tr>

<tr>

<td>Student\_Name:</td><td><input type="text" name="st\_Name"></td>

</tr>

<tr>

<td> Branch:</td><td><input type="text" name="branch"></td>

</tr>

<tr>

<td> Year:</td><td><input type="text" name="year"></td>

</tr>

<tr>

<td colspan="3"><input type="submit" value="Insert">

</tr>

</table>

</form>

</body>

</html>

**StdDB.java**

importjava.io.IOException;

importjava.io.PrintWriter;

importjava.sql.Connection;

importjava.sql.DriverManager;

importjava.sql.PreparedStatement;

importjava.sql.SQLException;

importjava.util.logging.Level;

importjava.util.logging.Logger;

importjavax.servlet.ServletException;

importjavax.servlet.http.HttpServlet;

importjavax.servlet.http.HttpServletRequest;

importjavax.servlet.http.HttpServletResponse;

public class data extends HttpServlet {

protected void processRequest(HttpServletRequest request, HttpServletResponse response)

throwsServletException, IOException, ClassNotFoundException, SQLException {

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

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/servlet", "root", "");

response.setContentType("text/html;charset=UTF-8");

PreparedStatementst = conn.prepareStatement("insert into student (en\_no,st\_name,branch,year) values(?, ?, ?, ?)");

try (PrintWriter out = response.getWriter()) {

String en\_no = request.getParameter("en\_no");

String st\_name = request.getParameter("st\_name");

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

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

st.setString(1, en\_no);

st.setString(2, st\_name);

st.setString(3, branch);

st.setString(4, year);

st.executeUpdate();

st.close();

conn.close();

out.println("<!DOCTYPE html>");

out.println("<html>");

out.println("<head>");

out.println("<title>Servlet data</title>");

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

out.println("<body>");

out.println("<h1>Servlet data at " + en\_no + " " + st\_name + " " + branch + " " + year + "</h1>");

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

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

}

}

protected void doGet(HttpServletRequest request, HttpServletResponse response)

throwsServletException, IOException {

try {

processRequest(request, response);

} catch (ClassNotFoundException ex) {

Logger.getLogger(data.class.getName()).log(Level.SEVERE, null, ex);

} catch (SQLException ex) {

Logger.getLogger(data.class.getName()).log(Level.SEVERE, null, ex);

}

}

@Override

protected void doPost(HttpServletRequest request, HttpServletResponse response)

throwsServletException, IOException {

try {

processRequest(request, response);

} catch (ClassNotFoundException ex) {

Logger.getLogger(data.class.getName()).log(Level.SEVERE, null, ex);

} catch (SQLException ex) {

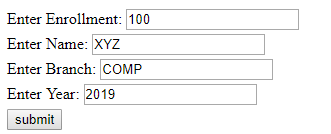
Logger.getLogger(data.class.getName()).log(Level.SEVERE, null, ex);

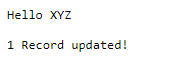
}

}

}

**Output :-**

****

****

**9(3).PNG**

**Practical No:-10**

**Aim:-**Write a Servlet that take First Name & Last Name from the user. Generate login-id such that three characters are from First Name & three characters are from Last Name. Also Display the generated login-id.

**Source 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>

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

First Name: <input type="text" name="fn"><br/><br/>

Last Name: <input type="text" name="ln"></br><br/>

<input type="submit" value="Generated login ID"/>

</form>

</body>

</html>

**id.java**

importjava.io.IOException;

importjava.io.PrintWriter;

importjavax.servlet.ServletException;

importjavax.servlet.annotation.WebServlet;

importjavax.servlet.http.HttpServlet;

importjavax.servlet.http.HttpServletRequest;

importjavax.servlet.http.HttpServletResponse;

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

public class login extends HttpServlet

{

protected void processRequest(HttpServletRequest request, HttpServletResponse response)

throwsServletException, IOException

{

response.setContentType("text/html;charset=UTF-8");

}

@Override

protected void doGet(HttpServletRequest request, HttpServletResponse response)

throwsServletException, IOException {

processRequest(request, response);

}

@Override

protected void doPost(HttpServletRequest request, HttpServletResponse response)

throwsServletException, IOException {

processRequest(request, response);

PrintWriter out = response.getWriter();

String fn,ln,loginid;

Integer len;

fn=request.getParameter("fn");

ln=request.getParameter("ln");

fn=fn.substring(0,3);

len=ln.length();

ln=ln.substring(len-3,len);

loginid=fn+ln;

out.println("Your Login ID:"+loginid);

}

@Override

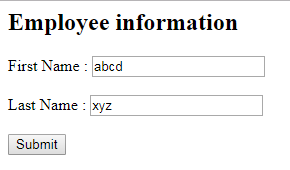
public String getServletInfo() {

return "Short description";

}

}

**Output :-**

****

**Capture1.PNG**

**Practical No:-11**

**Aim:-** Implement a Servlet which counts the number of Hits. (I.e. Website Hit-counter).

**Source Code:-**

importjava.io.IOException;

importjava.io.PrintWriter;

importjavax.servlet.ServletException;

importjavax.servlet.annotation.WebServlet;

importjavax.servlet.http.HttpServlet;

importjavax.servlet.http.HttpServletRequest;

importjavax.servlet.http.HttpServletResponse;

importjavax.servlet.http.HttpSession;

@WebServlet(urlPatterns = {"/Hitcounter"})

public class Hitcounter extends HttpServlet {

int counter=0;

protected void processRequest(HttpServletRequest request, HttpServletResponse response)

throwsServletException, IOException

{

response.setContentType("text/html;charset=UTF-8");

// PrintWriter out = response.getWriter() ;

}

@Override

protected void doGet(HttpServletRequest request, HttpServletResponse response)

throwsServletException, IOException {

PrintWriter out = response.getWriter() ;

HttpSession session=request.getSession();

String msg;

Integer counter=(Integer) session.getAttribute("counter");

if(counter==null)

{

counter=new Integer(0);

msg="Welcome you are accesing the page of first time.";

}

else

{

counter=new Integer(counter.intValue()+1);

msg="Welcome once again.";

}

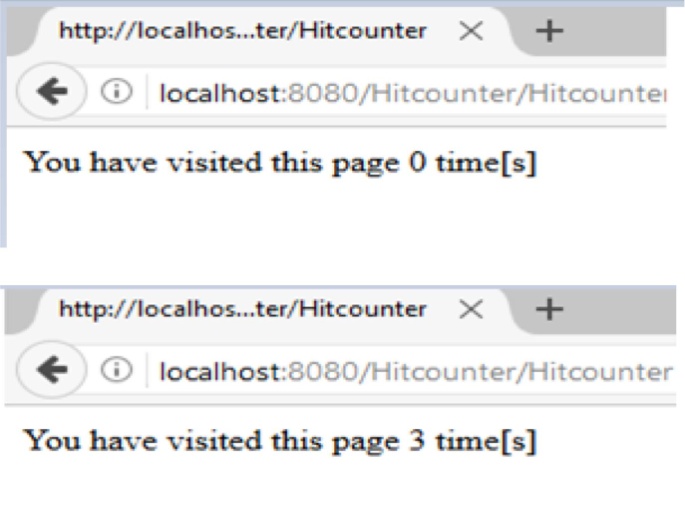
session.setAttribute("counter",counter);

out.println("<h1>" + msg );

out.println("<br><BODY>You have visited this page " + counter + " time[s]");

}

**Output :-**



**Practical No:-12**

**Aim:-**Write a JSP page that displays the user from containing the two fields-user name and city.

**Source Code:-**

**Index.html**

<html>

<body bgcolor="pink">

<form action="http://localhost:8080/user/MyFromProcess.jsp" method="GET">

Name:<input type="text" name="name"></br>

City:<input type="text" name="city"/>

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

</form>

</body>

</html>

**MyFromProcess.jsp**

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

<!DOCTYPE html>

<html>

<body>

<h2>Displying from data</h2>

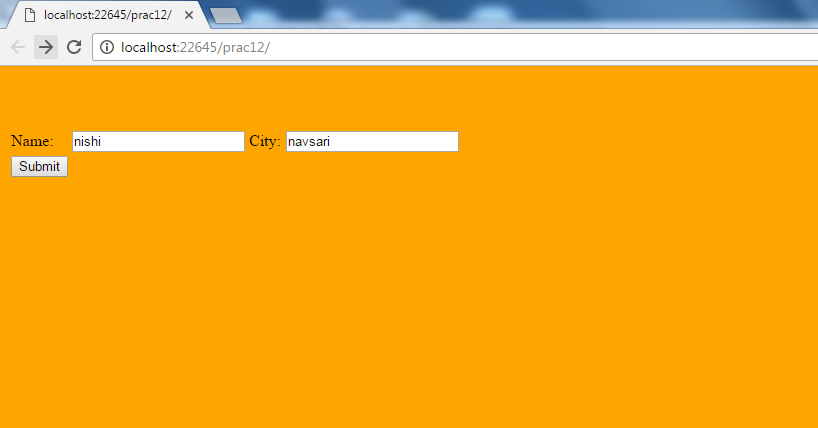
<b>Name:</b><%= request.getParameter("name")%><br/>

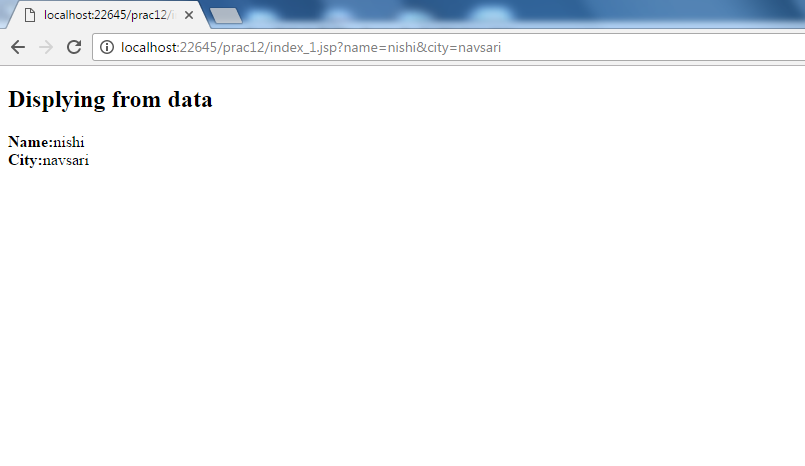
<b>City:</b><%=request.getParameter("city")%>

</body>

</html>

**Output :-**



****

**Practical No:-13**

**Aim:-**Write a JSP page that prints your resume in well formatted form.

**Source Code:-**

**Index.html**

<html>

<head>

<title>Resume</title>

</head>

<body>

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

<table>

<tr>

<td> Name:</td>

<td><input type="text" name="user"></td>

</tr>

<tr>

<td> Address:</td>

<td><input type="text" name="addr"></td>

</tr>

<tr>

<td> Email:</td>

<td><input type="text" name="email"></td>

</tr>

<tr>

<td> Phone:</td>

<td><input type="text" name="ph"></td>

</tr>

<tr>

<td>Qualifiacation:</td>

<td><input type="text" name="qualify"></td>

</tr>

<tr>

<td> Experience:</td>

<td><input type="text" name="expr"></td>

</tr>

</table>

<tr>

<input type="submit" name="submit">

</tr>

</form>

</body>

</html>

**resume.jsp**

<%@page language="java" contentType="text/html"%>

<html>

<body>

<h3>RESUME</h3>

<table border=""1">

<tr>

<td> Name:</td><td><b> ${param.user} </b></td>

</tr>

<tr>

<td> Address:</td><td><b> ${param.addr} </b></td>

</t><tr>

<td> Email:</td><td><b> ${param.email} </b></td>

</tr><tr>

<td> Phone:</td><td><b> ${param.ph} </b></td>

</tr>

<tr>

<td> Qualification:</td><td><b> ${param.qualify} </b></td>

</tr>

<tr>

<td> Experience:</td><td><b> ${param.expr} </b></td>

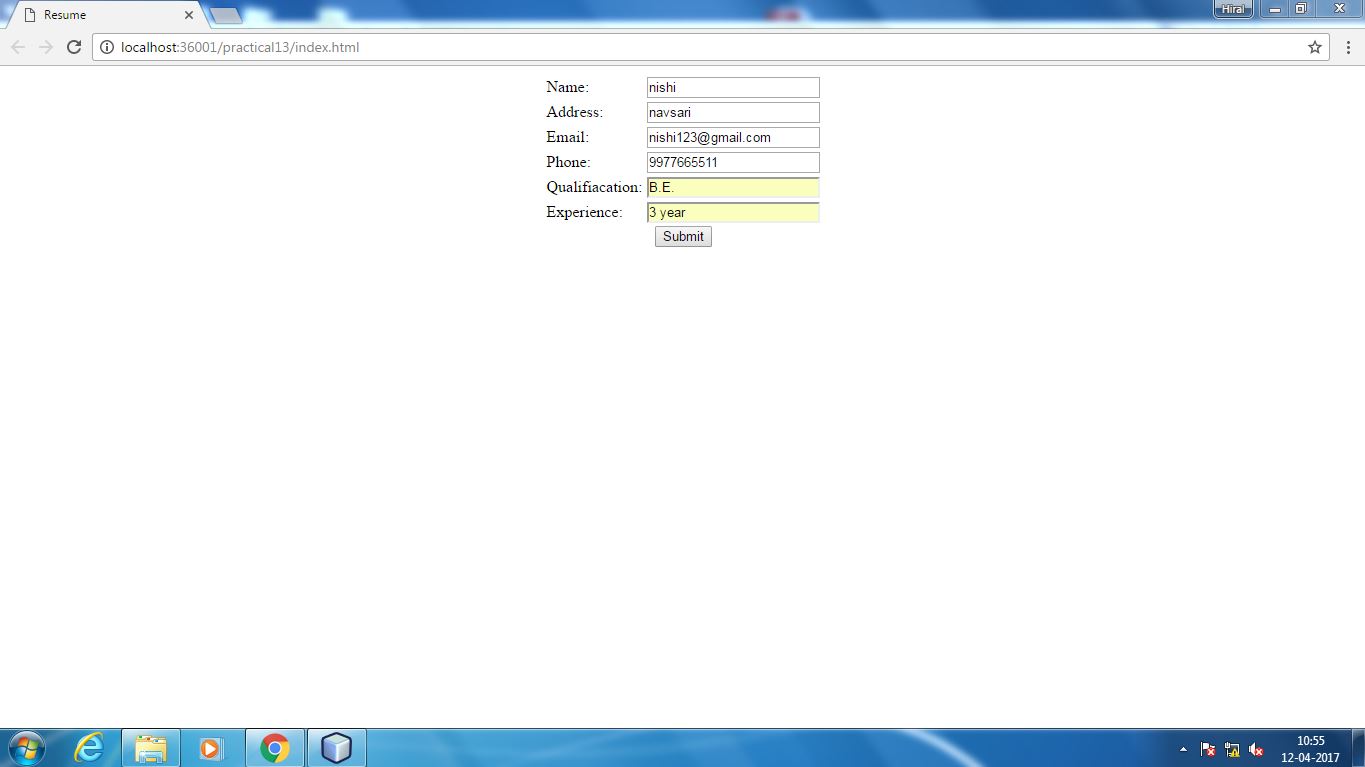
</tr>

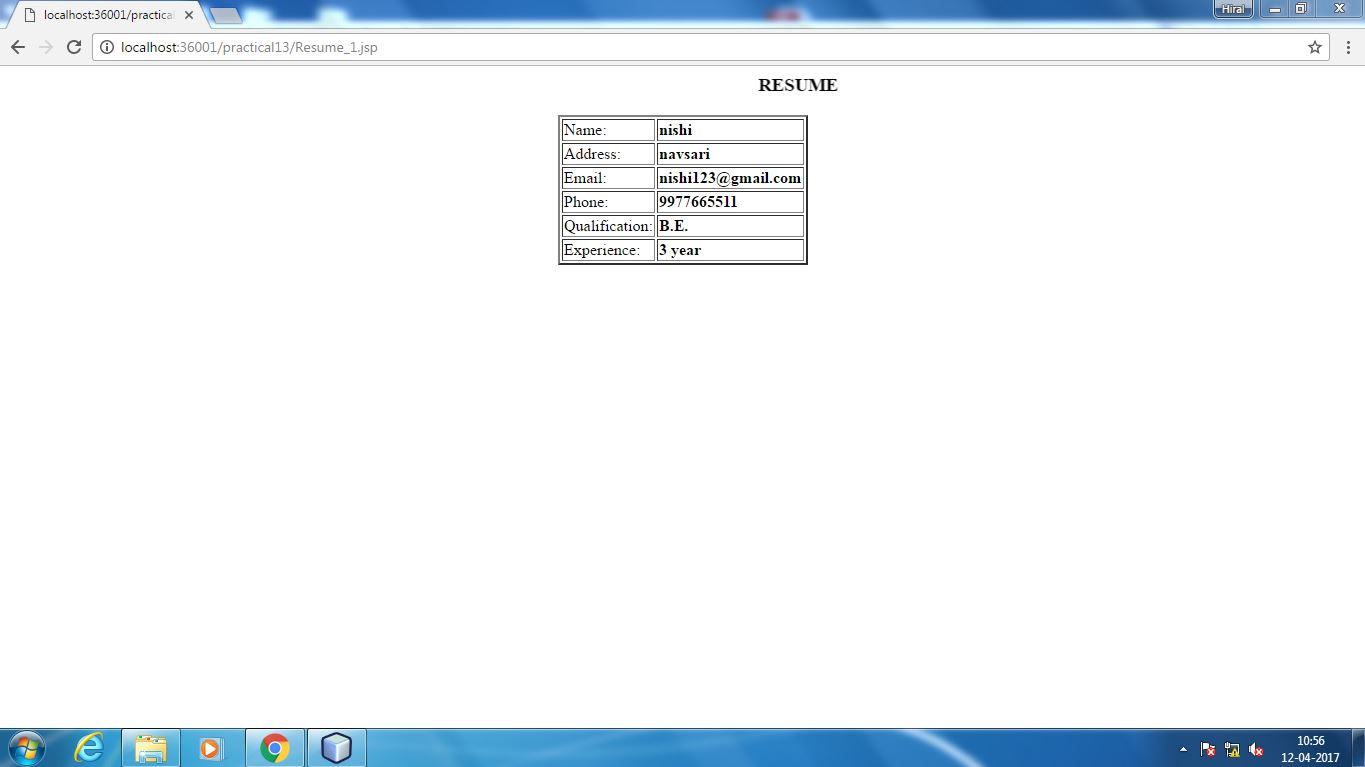
</table>

</body>

</html>

**Output :-**

****

****

**Practical No:-14**

**Aim:-**Write a JSP page to implement Cookie. Create an html file to read the values from the user. Create the cookie, Read the created cookie and delete the cookie using JSP.

**Source Code:-**

**Index.html**

<html>

<head>

<title>cookie</title>

</head>

<body>

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

Username<input type="text" name="name">

City<input type="text" name="city">

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

</form>

</body>

</html>

**Createcookie.jsp**

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

<%

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

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

Cookie namecookie= new Cookie("name",name);

Cookie citycookie= new Cookie("city",city);

response.addCookie(namecookie);

response.addCookie(citycookie);

namecookie.setMaxAge(60\*60\*24);

citycookie.setMaxAge(60\*60\*24);

%>

<h3>

<a href="Readcookie.jsp">Click here to continue..</a>

</h3>

**Readcookie.jsp**

<h3> Reading the cookie</h3>

<%

Cookie[] cookies=request.getCookies();

for(inti=0;i<cookies.length;i++)

{

out.println("Cookie\_Name:"+cookies[i].getName()+"<br>");

out.println("Cookie\_Value:"+cookies[i].getValue()+"<br>");

}

%>

<a href="Deletecookies.jsp">click here to Delete the cookie..</a>

**Deletecookies.jsp**

<%

Cookie namecookie= new Cookie("name","");

namecookie.setMaxAge(0);

namecookie.setValue("");

response.addCookie(namecookie);

Cookie citycookie=new Cookie("city","");

citycookie.setMaxAge(0);

citycookie.setValue("");

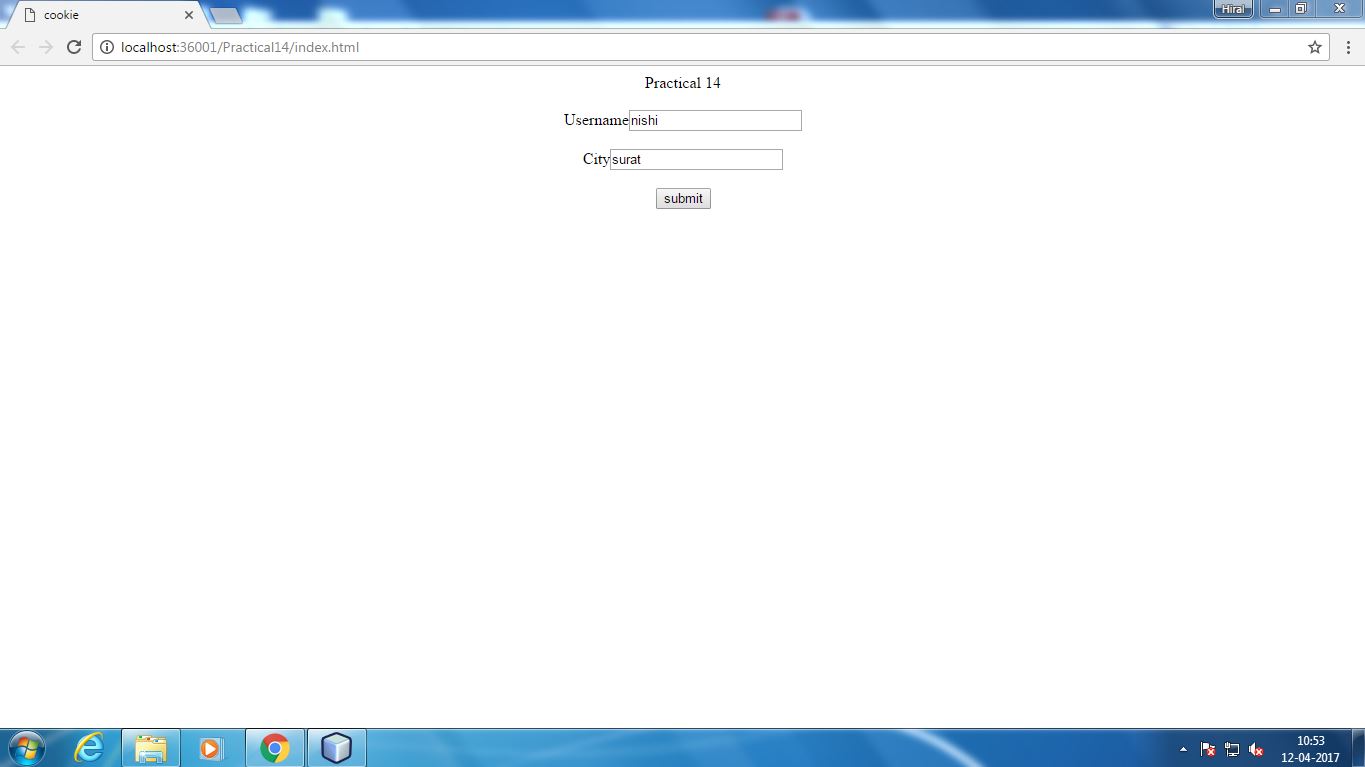
response.addCookie(citycookie);

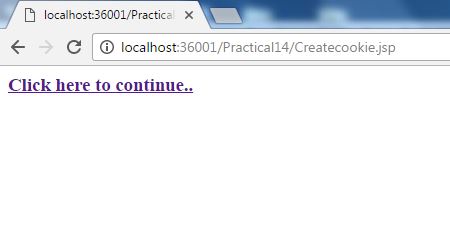
%>

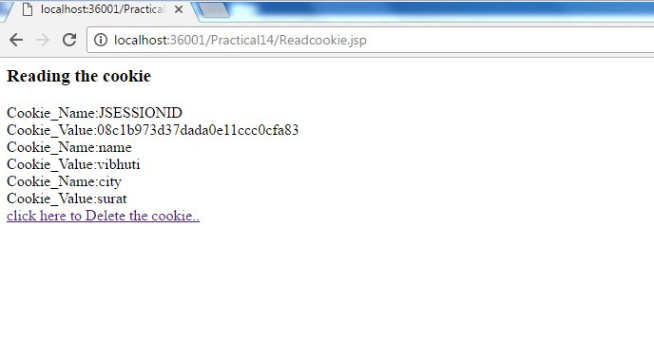
<h3>cookie is deleted</h3>

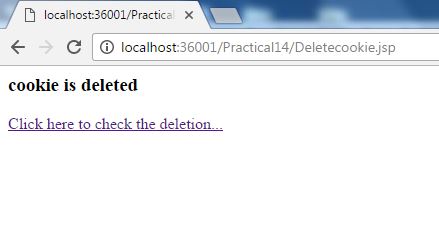
<a href="Readcookie.jsp"> Click here to check the deletion...</a>

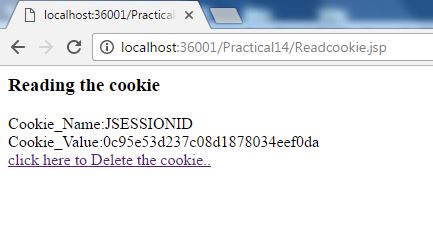
**Output :-**

****

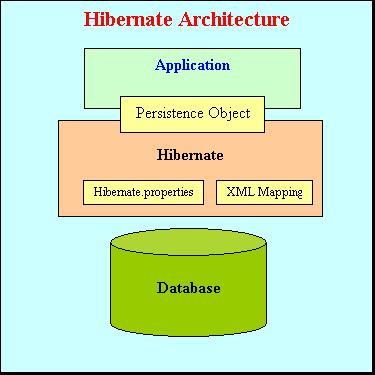
****

****

****



**Practical No:-15**

**Aim:-**Explain the Hibernate architecture.

* In order to persist data to a database, Hibernate create an class mapped with database table.
* This object is called Transient object as they are not yet associated with the session or

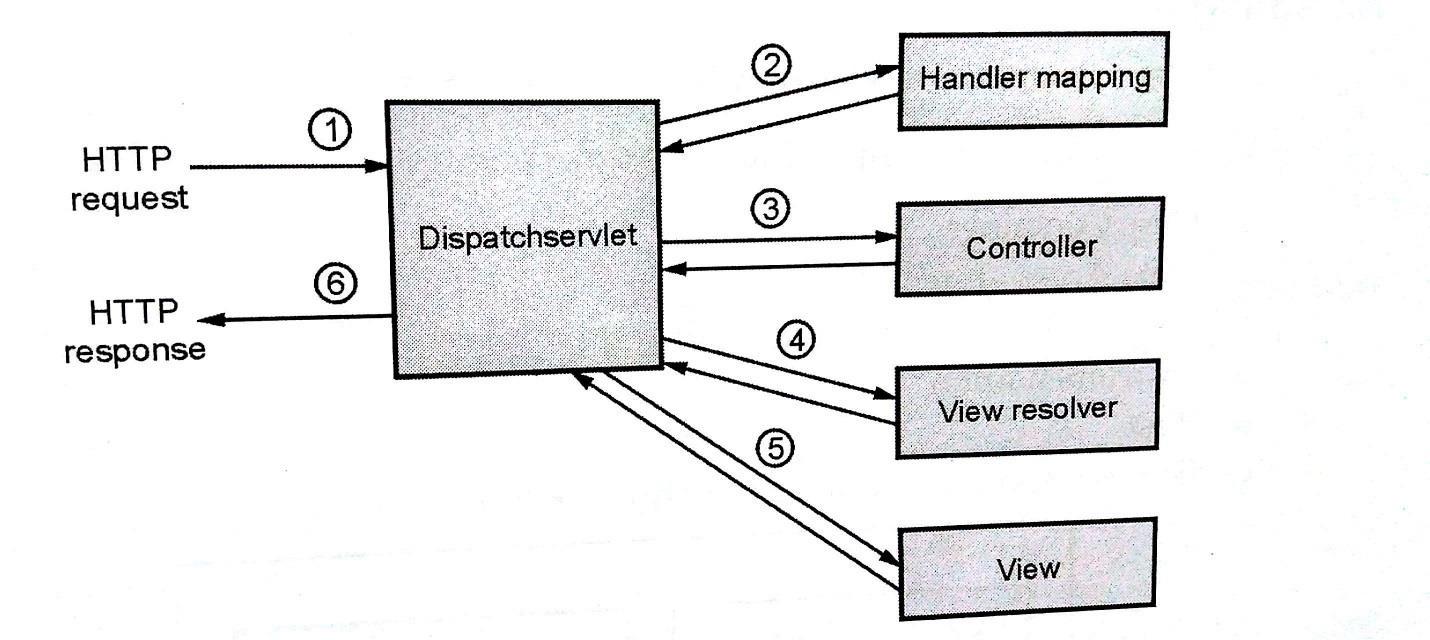
not yet persisted to a database.

* + To persist the object to database, the instance of **SessionFactory** is a singleton instance which implements Factory design pattern.SessionFactory loads hibernate.cfg.xml file and with the help of **ConnectionProvider** implements all the configuration settings on a database.
  + Each database connection in Hibernate is created by creating an instance of Session interface.
  + Session represents a single connection with database.
  + Session objects are created from **SessionFactory** object.
  + Each transaction represents a single atomic unit of the work One Session can span through multiple transactions.
  + Here are some of the part of the hibernate architecture

1. **SessionFactory (org.hibernate.SessionFactory)**
   * + A thread-safe, immutable cache of compiled mappings for a single database. A factory for org.hibernate.Session instances.
     + A client of org.hibernate.connection.ConnectionProvider.
     + Optionally maintains a second level cache of data that is reusable between transactions at a process or cluster level.
2. **Session (org.hibernate.Session)**
   * A single-threaded, short lived object representing a conversation between the application and the persistent store.
   * Wraps a JDBC java.sql.Connection.
   * Factory for org.hibernate.Transaction.
   * Maintains a first level cache of persistent the application’s persistent objects and collections; this cache is used when navigating the object graph or looking up objects by identifier.
3. **Persistent objects and collections**
   * Short-lived, single threaded objects containing persistent state and business function.
   * These can be ordinary JavaBeans/POJOs.
   * They are associated with exactly one org.hibernate.Session.
   * Once the org.hibernate.Session is closed, they will be detached and free to use in any application layer.
4. **Transient and detached objects and collections**
   * Instances of persistent classes that are not currently associated with aorg.hibernate.Session.
   * They may have been instantiated by the application and not yet persisted, or they may have been instantiated by a closed org.hibernate.Session.
5. **Transaction (org.hibernate.Transaction)**
   * A single-threaded, short lived object used by the application to specify atomic units of work.
   * It abstracts the application from the underlying JDBC, JTA or CORBA transaction.
6. **ConnectionProvider (org.hibernate.connection.ConnectionProvider)**
   * A factory for, and pool of, JDBC connections.
   * It abstracts the application from underlying javax.sql.DataSource or java.sql.DriverManager.
7. **TransactionFactory (org.hibernate.TransactionFactory)**
   * A factory for org.hibernate.Transaction instances. It is not exposed to the application, but it can be extended and/or implemented by the dev

**Practical No:-16**

**Aim:-**To Study MVC using Spring Framework.



Three important elements are:

* **Model:** It is normally an application data containing plain object java classes.
* **View:** It is responsible for showing the result. This element generates the HTML output which can be displayed on client’s machine.
* **Controller:** This element is responsible for processing user requests.

**Working:**

* The DispatcherServlet receives the HTTP request.
* The DispatcherServlet consults HandlerMapping to call appropriate Controller.
* The Controller takes the request and finds the appropriate servicer, sends GET/POST method for calling that service. The service on execution defines the business logic and

returns view name to DipatchServlet,

* Using ViewResolver, the DispatchServlet will choose the appropriate View.
* The View will then be returned to DispatchServlet to be displayed on the client’s browser window.

**Benefits:**

* It provides clear separation of roles such as – controller, validator, model object, handler mapping, view resolver, view and so on.
* It provides powerful and straightforward configuration of both framework and application classes as java beans.
* It allows reusability of business code.
* It is also supports for customization and application level validation.
* It supports for simple yet powerful tag library.
* It supports easy integration with any type of view.