

Experiment# 1 Java Networking

Practical 1: Write an application which will retrieve IP address for given website.

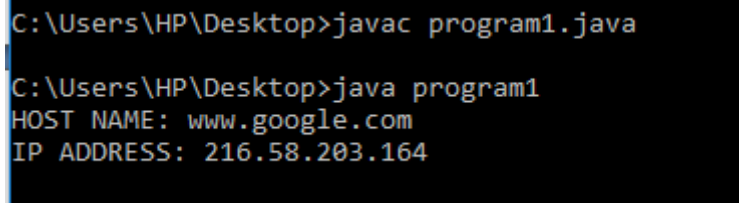
Program:

```
import java.io.*;
import java.net.*;
public class program1
{
    public static void main(String[] args)
    {
        try
        {
            //for given website name address

            InetAddress a1=InetAddress.getByName("www.google.com");
            System.out.println("HOST NAME: " +a1.getHostName());
            System.out.println("IP ADDRESS: " +a1.getHostAddress());

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

Output:



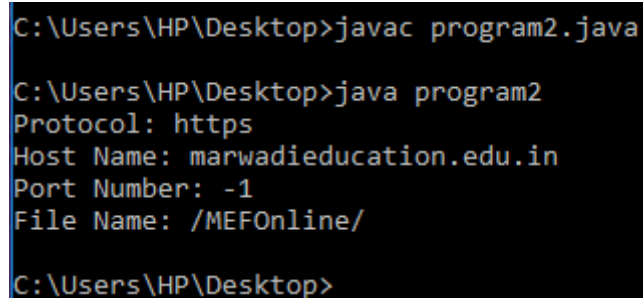
```
C:\Users\HP\Desktop>javac program1.java
C:\Users\HP\Desktop>java program1
HOST NAME: www.google.com
IP ADDRESS: 216.58.203.164
```

Practical 2: Write an application which will retrieve the content of the given URL with different web-page related information

Program:

```
import java.io.*;
import java.net.*;
public class program2
{
    public static void main(String[] args)
    {
        Try
        {
            URL url = new URL("https://marwadieducation.edu.in/MEFOnline/");
            System.out.println("Protocol: "+url.getProtocol());
            System.out.println("Host Name: "+url.getHost());
            System.out.println("Port Number: "+url.getPort());
            System.out.println("File Name: "+url.getFile());
        }
        catch(Exception e){System.out.println(e);}
    }
}
```

Output:



```
C:\Users\HP\Desktop>javac program2.java
C:\Users\HP\Desktop>java program2
Protocol: https
Host Name: marwadieducation.edu.in
Port Number: -1
File Name: /MEFOnline/
C:\Users\HP\Desktop>
```

Practical 3: Write a two – way network based chat application. It will use TCP/IP protocol and it will do communication in serial manner

Program:

Server side:

```
import java.net.*;

import java.io.*;

class MyServer{

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

        ServerSocket ss=new ServerSocket(3333);

        Socket s=ss.accept();

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

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

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

        String str="",str2="";

        while(!str.equals("stop")){

            str=din.readUTF();

            System.out.println("client says: "+str);

            str2=br.readLine();

            dout.writeUTF(str2);

            dout.flush();

        }

        din.close();

        s.close();
```

```
ss.close();  
}}
```

Client side:

```
import java.net.*;  
import java.io.*;  
class MyClient{  
    public static void main(String args[])throws Exception{  
        Socket s=new Socket("localhost",3333);  
        DataInputStream din=new DataInputStream(s.getInputStream());  
        DataOutputStream dout=new DataOutputStream(s.getOutputStream());  
        BufferedReader br=new BufferedReader(new InputStreamReader(System.in));  
  
        String str="",str2="";  
        while(!str.equals("stop")){  
            str=br.readLine();  
            dout.writeUTF(str);  
            dout.flush();  
            str2=din.readUTF();  
            System.out.println("Server says: "+str2);  
        }  
        dout.close();  
        s.close();  
    }  
}
```

Output:

Server side:

```
C:\Users\HP\Desktop>javac MyServer.java

C:\Users\HP\Desktop>java MyServer
client says: Hi im your client
how can i help you
client says: it was just a normal check for working
ok!!
```

Client side:

```
C:\Users\HP\Desktop>javac MyClient.java

C:\Users\HP\Desktop>java MyClient
Hi im your client
Server says: how can i help you
it was just a normal check for working
Server says: ok!!
stop
```

Practical 4: Write an application which will retrieve file from server machine and save that file on client machine. File name will be provided by client.

Program:

Server side:

```
import java.io.*;
import java.net.*;
class FileServer
{
    public static void main(String args[ ]) throws Exception
    {
        ServerSocket ss = new ServerSocket(8888);
        Socket s = ss.accept();
        System.out.println("Connection established");

        BufferedReader in = new BufferedReader(new
            InputStreamReader(s.getInputStream()));
```

```
DataOutputStream out = new  
DataOutputStream(s.getOutputStream());
```

```
String fname = in.readLine();
```

```
FileReader fr = null;
```

```
BufferedReader file = null;
```

```
boolean flag;
```

```
File f = new File(fname);
```

```
if(f.exists()) flag = true;
```

```
else flag = false;
```

```
if(flag == true) out.writeBytes("Yes"+"\\n");
```

```
else out.writeBytes("No"+"\\n");
```

```
if(flag == true)
```

```
{
```

```
    fr = new FileReader(fname);
```

```
    file = new BufferedReader(fr);
```

```
    String str;
```

```
    while((str = file.readLine()) != null)
```

```
    {
```

```
        out.writeBytes(str+"\\n");
```

```
    }
```

```
    file.close();
```

```

        out.close();
        in.close();
        fr.close();
        s.close();
        ss.close();
    }
}

```

Client side:

```

import java.io.*;
import java.net.*;
class FileClient
{
    public static void main(String args[ ]) throws Exception
    {
        Socket s = new Socket("localhost", 8888);

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

        System.out.print("Enter filename: ");
        String fname = kb.readLine();

        DataOutputStream out = new
DataOutputStream(s.getOutputStream());
        out.writeBytes(fname+"\n");
    }
}

```

```

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

        String str;
        str = in.readLine();

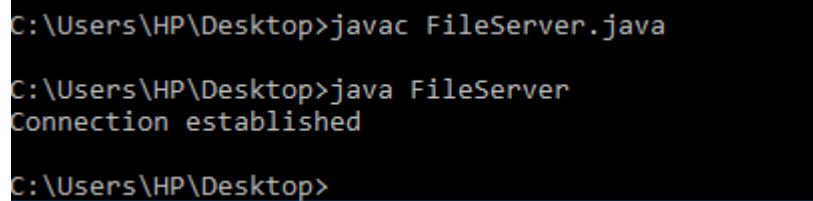
        if(str.equals("Yes"))
        {
            while((str = in.readLine()) != null)
                System.out.println(str);

            kb.close();
            out.close();
            in.close();
            s.close();
        }
        else System.out.println("File not found");
    }
}

```

Output:

Server side:



```

C:\Users\HP\Desktop>javac FileServer.java

C:\Users\HP\Desktop>java FileServer
Connection established

C:\Users\HP\Desktop>

```

Client side:


```
C:\Users\HP\Desktop>javac FileClient.java

C:\Users\HP\Desktop>java FileClient
Enter filename: myfile.txt
This is the file stored at the server and requested by the client.
```

Practical 5: Write a client program to send any string from its standard input to the server program. The server program reads the string, finds number of characters and digits and sends it back to client program. Use connection-oriented communication

Program:

Server side:

```
import java.io.*;
import java.net.*;
class server
{
    public static void main(String args[]) throws Exception
    {
        ServerSocket ss = new ServerSocket(3333);
        Socket s = ss.accept();
        System.out.println("Connection established");
        PrintStream p = new PrintStream(s.getOutputStream());
        BufferedReader br = new BufferedReader(new
InputStreamReader(s.getInputStream()));
        BufferedReader kb = new BufferedReader(new
InputStreamReader(System.in));
        while(true)
        {
            String str,str1;
            while((str = br.readLine()) != null)
            {
                System.out.println("\n Count Of Characters & Digits In
String Displayed ");
                int countCha = 0,countNum = 0;
                for(int i=0;i<str.length();i++)
                {
```

```

        if( (str.charAt(i) >= 'a' && str.charAt(i) <= 'z') ||
(str.charAt(i) >= 'A' && str.charAt(i) <= 'Z'))
            countCha++;
        else if(str.charAt(i) >= '0' && str.charAt(i) <= '9')
            countNum++;
    }
    str1 = " Total Number Of Characters = " + countCha;
    str1 += " Total Number Of Digits = " + countNum;
    p.println(str1);
}
ss.close();
s.close();
}
}
}

```

Client side:

```

import java.io.*;
import java.net.*;
class client
{
    public static void main(String args[]) throws Exception
    {
        Socket s = new Socket("localhost", 3333);
        int stop = 1;
        DataOutputStream dout = new
DataOutputStream(s.getOutputStream());
        BufferedReader br = new BufferedReader(new
InputStreamReader(s.getInputStream()));
        BufferedReader kb = new BufferedReader(new
InputStreamReader(System.in));
        String str="",str1="";
        do
        {
            System.out.print("Enter String: ");
            str = kb.readLine();
            dout.writeBytes(str+"\n");
            str1 = br.readLine();
            System.out.println("\n");
            System.out.println(str1);
        }while(stop != 1);
        dout.close();
        br.close();
    }
}

```

```
        kb.close();  
        s.close();  
    }  
}
```

Output:

Server side:

```
C:\Users\HP\Desktop>javac server.java  
  
C:\Users\HP\Desktop>java server  
Connection established  
  
Count Of Characters & Digits In String Displayed
```

Client side:

```
C:\Users\HP\Desktop>javac client.java  
  
C:\Users\HP\Desktop>java client  
Enter String: priyanka savani 13  
  
Total Number Of Characters = 14 Total Number Of Digits = 2
```

Practical 6: Write a client program to send any string from its standard input to the server program. The server program reads the string, finds number of characters and digits and sends it back to client program. Use connection-less communication

Program:

Server side:

```
import java.net.DatagramPacket;  
import java.net.DatagramSocket;  
import java.net.InetAddress;
```

```

public class NumCharServer
{
    public static void main(String[] args)
    {
        try {
            DatagramSocket ds = new DatagramSocket(3000);
            byte [] buf = new byte[100];

            DatagramPacket dp = new DatagramPacket(buf, 100);

            ds.receive(dp);

            String str = new String(dp.getData(), 0, dp.getLength());

            int carcount =0,digcount = 0;

            for(int i =0;i<str.length();i++)
            {
                if(str.charAt(i)>='a'&&str.charAt(i)<='z')
                    carcount++;
                if(str.charAt(i)>='0'&&str.charAt(i)<='9')
                    digcount++;
            }

            System.out.println("Characters : "+carcount);
            System.out.println("digcount :"+digcount);

            ds.close();

        } catch (Exception e) {
        }
    }
}

```

Client side:

```

import java.net.*;
import java.io.*;
import java.util.Scanner;

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

```

```

{
    DatagramSocket ds = new DatagramSocket();

    Scanner s = new Scanner(System.in);
    System.out.println("Enter any string :");
    String str = s.nextLine();

    InetAddress ip = InetAddress.getByName("localhost");

    DatagramPacket dp1 = new DatagramPacket(str.getBytes(), str.length(), ip,
3000);
    ds.send(dp1);

    ds.close();
}
}

```

Output:

Server side:

```

C:\Users\HP\Desktop>javac NumCharServer.java

C:\Users\HP\Desktop>java NumCharServer
Characters : 8
digcount :11

```

Client side:

```

C:\Users\HP\Desktop>javac NumCharClient.java

C:\Users\HP\Desktop>java NumCharClient
Enter any string :
priyanka 91600104013

```

Experiment# 2 JDBC Programming

Practical 1: Write down Five Basic steps to establish JDBC connection from Java Application. Also mention sample code for each step.

Answer: Five Basic steps to establish JDBC connection

1. Register the driver
2. Establish connection
3. Create the statement
4. Execute query
5. Close the connection

Program:

```
import java.sql.*;
public class DemoJdbc {

    public static void main(String[] args) {
        try {
            //1. Register the driver
            Class.forName("com.mysql.jdbc.Driver");

            //2. establish connection by con object

            Connection
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/mu","ro
ot","");

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

            //3. Create the statement object which is
            //used to execute query in database

            Statement stmt=con.createStatement();

            //step4 execute query

            ResultSet rs=stmt.executeQuery("select * from emp");
```

```

        while(rs.next())
            System.out.println(rs.getInt(1)+" "+rs.getString(2)+"
"+rs.getString(3));
            System.out.println("DATA FETCHED");
            //5.close the connection

        con.close();

    } catch (Exception e) {
    }
}
}

```

Output:

```

run:
Connection established
1 Siddharth IT
2 Jay Computer
3 Pranav IT
DATA FETCHED
BUILD SUCCESSFUL (total time: 3 seconds)

```

Practical 2: Write a JDBC application which will interact with Database and perform the following task.1) Create Student Table with Roll No, Name, and Address field and insert few records.2) Using Statement Object display the content of Record.3) Using Statement Object Insert Two Record.4) Using Statement Object Update One Record5) Using Statement Object Delete One Record.6) Using Statement Object display the content of Record.

Program:

```

import java.net.*;
import java.sql.*;
import java.util.*;
public class JDBC_Application{
    public static void main(String[] args){
        try{

```

```

        Class.forName("com.mysql.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:" +
"mysql://localhost:3306/mu","root","");
        System.out.println("Connection Established");
        Scanner s = new Scanner(System.in);
        Statement st = con.createStatement();
        System.out.println("1. To Insert Data");
        System.out.println("2. To Display Data");
        System.out.println("3. To Update Data");
        System.out.println("4. To Delete a Record");
        System.out.println("Enter Your Chice:");
        int ch = s.nextInt();
        switch(ch){
            case 1: {
                st.executeUpdate("Insert into student values
(5,'jbp','Mehsana')");
                System.out.println("Data Inserted");
                break;
            }
            case 2: {
                ResultSet rs = st.executeQuery("select * from
student");
                while(rs.next()){
                    int rollno = rs.getInt(1);
                    String name = rs.getString(2);
                    String add = rs.getString(3);
                    System.out.println("Roll: " +rollno+" Name: "
+name+ " " +"add: "+add);
                }
                break;
            }
            case 3: {
                st.executeUpdate("UPDATE student" + " SET name =
"+ "'Jinesh' WHERE Roll_no =5");
                System.out.println("Row Updated!!!");
                break;
            }
            case 4: {
                st.executeUpdate("DELETE FROM" + " student
WHERE Roll_no = 5");

```



```

        System.out.println("Row Deleted!!!");
        break;
    }
}
}
catch(Exception e){
    System.out.print(e);
}
}
}
}

```

Output:

Insert:

```

run:
Connection Established
1. To Insert Data
2. To Display Data
3. To Update Data
4. To Delete a Record
Enter Your Chice:
1
Data Inserted

```

Update:

```

run:
Connection Established
1. To Insert Data
2. To Display Data
3. To Update Data
4. To Delete a Record
Enter Your Chice:
3
Row Updated!!!

```

Display:

```

run:
Connection Established
1. To Insert Data
2. To Display Data
3. To Update Data
4. To Delete a Record
Enter Your Chice:
2
Roll: 32 Name: Siddharth add: Upleta
Roll: 5 Name: jbp add: Mehsana
BUILD SUCCESSFUL (total time: 4 seconds)

```

Delete:

```

run:
Connection Established
1. To Insert Data
2. To Display Data
3. To Update Data
4. To Delete a Record
Enter Your Chice:
4
Row Deleted!!!
BUILD SUCCESSFUL (total time: 1 second)

```

Practical 3: Write a JDBC application which will interact with Database and perform the following task.

- 1) Create Student Table with RollNo, Name, and Address field and insert few records.
- 2) Using PreparedStatementObject display the content of Record.
- 3) Using PreparedStatementObject Insert Two Record.

- 4) Using PreparedStatementObject Update One Record.
- 5) Using PreparedStatementObject Delete One Record.
- 6) Using PreparedStatementObject display the content of Record.

Program:

```
import java.sql.*;
import java.util.*;
import java.net.*;

public class JDBC_Application {
    public static void main(String[] args) {
        try{
            Class.forName("com.mysql.jdbc.Driver");

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

            System.out.println("Connection Established");

            Scanner s = new Scanner(System.in);

            Statement st = con.createStatement();
            PreparedStatement pst;

            System.out.println("1. To Insert Data");
            System.out.println("2. To Display Data");
            System.out.println("3. To Update Data");
            System.out.println("4. To Delete a Record");
            System.out.println("Enter Your Chice:");

            int ch = s.nextInt();

            switch(ch){
                case 1: {
```

```

        pst = con.prepareStatement("insert into student
values(?,?,?)");
        pst.setInt(1,1);
        pst.setString(2,"jbp");
        pst.setString(3, "Mehsana");
        pst.executeUpdate();
        System.out.println("Data Inserted");

        break;
    }

    case 2: {
        pst = con.prepareStatement("Select * FROM
student ");
        ResultSet rs = pst.executeQuery();
        while(rs.next()){
            int rollno = rs.getInt(1);
            String name = rs.getString(2);
            String add = rs.getString(3);
            System.out.println("Roll: " +rollno+" Name: "
+name+ " " +"add: "+add);
        }

        break;
    }

    case 3: {
        pst = con.prepareStatement("UPDATE student
SET name = ? WHERE Roll_no = ?");
        pst.setString(1,"Nancy");
        pst.setInt(2,1);
        pst.executeUpdate();
        System.out.println("Row Updated!!!");

        break;
    }

    case 4: {
        pst= con.prepareStatement("DELETE FROM
student Where Roll_no=?");

```

```

        pst.setInt(1,1);
        pst.executeUpdate();
        System.out.println("Row Deleted!!!");

        break;
    }
}

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

```

Output:

Insert:

```

run:
Connection Established
1. To Insert Data
2. To Display Data
3. To Update Data
4. To Delete a Record
Enter Your Chice:
1
Data Inserted

```

Update:

```

run:
Connection Established
1. To Insert Data
2. To Display Data
3. To Update Data
4. To Delete a Record
Enter Your Chice:
3
Row Updated!!!

```

Display:

```

run:
Connection Established
1. To Insert Data
2. To Display Data
3. To Update Data
4. To Delete a Record
Enter Your Chice:
2
Roll: 32 Name: Siddharth add: Upleta
Roll: 5 Name: jbp add: Mehsana
BUILD SUCCESSFUL (total time: 4 seconds)

```

Delete:

```

run:
Connection Established
1. To Insert Data
2. To Display Data
3. To Update Data
4. To Delete a Record
Enter Your Chice:
4
Row Deleted!!!
BUILD SUCCESSFUL (total time: 1 second)

```

Practical 4: Write a JDBC application which will interact with Database and perform the following task.1) Create a store procedure which will insert one record into employee table.2) Create a store procedure which will retrieve salary for given employee id.3) Write a java application which will call the above procedure and display appropriate information on screen.

Program:

```
import java.sql.*;

public class JDBC_Application{

    public static void main(String[] args){

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

            Connection con =
            DriverManager.getConnection("jdbc:mysql://"
            +"localhost:3306/mu","root","");

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

            PreparedStatement pst =
            con.prepareStatement("Insert into employee values(?,?,?)");
            pst.setInt(1,4);
            pst.setString(2, "Mohit");
            pst.setInt(3,60000);

            pst.executeUpdate();
            Statement st = con.createStatement();

            int i = st.executeUpdate("UPDATE employee SET
            salary = 65000 where emp_id = 15");

            if(i==0)
                System.out.println("Table not updated");
```

```

        else
            System.out.println("Table updated");

        ResultSet rs = st.executeQuery("select * from
employee");

        System.out.println("\n\n=====
=====");
        System.out.println("Emp_ID.\t\tName\t\tSalary");
        System.out.println("-----\t\t-----\t\t-----");
        while(rs.next())
            System.out.println("    " + rs.getInt(1) + "\t\t" +
rs.getString(2) + "\t\t" + rs.getInt(3));

        System.out.println("=====
=====");

        con.close();
    }

    catch(Exception e){
    }

}

}

```

Output:

```

run:
Connection established
Table updated

```

```

=====
Emp_ID.      Name      Salary
-----
      8      Sid      30000
     15     Pranav     65000
      4     Mohit     60000
=====

```

Practical 5: Design a JDBC application which will demonstrate Scrollable ResultSet functionality.

Program:

```
import java.net.*;

import java.sql.*;
import java.sql.DriverManager;
import java.util.*;

public class Scroll {

    public static void main(String[] args) {

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

            Connection con = DriverManager.getConnection("jdbc:" +
"mysql://localhost:3306/mu","root","");
            System.out.println("Connection Established");
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery("Select * from student");
            Scanner s = new Scanner(System.in);

            rs.afterLast();
            while(rs.previous())
            {
                System.out.println(rs.getInt(1)+" "+rs.getString(2)+"
"+rs.getString(3));
            }
        }
    }
}
```

```

        rs.absolute(3);
        System.out.println(rs.getInt(1) + " " + rs.getString(2) + " " +
rs.getString(3));

        rs.relative(2);
        System.out.println(rs.getInt(1) + " " + rs.getString(2) + " " +
rs.getString(3));

        int i=rs.getRow();
        System.out.println("Cursor position: "+i);

        rs.close();
        st.close();
        con.close();

    }
    catch(Exception e){
    }
}
}

```

Output:

```

run:
Connection Established
Roll_No.: 2, Name: Jay, Address : Rajkot
Roll_No.: 25, Name: Pranav, Address : Porbandar
Roll_No.: 32, Name: Siddharth , Address : Upleta
Roll_No.: 25, Name: Pranav, Address : Porbandar

```

Practical 6: Design a JDBC application which will demonstrate Transaction management functionality.

Program:

```

import java.sql.*;

class JDBC_Application
{

```



```

public static void main(String args[])throws Exception
{
    Class.forName("com.mysql.jdbc.Driver");

    Connection con=DriverManager.getConnection("jdbc:"
+ "mysql://localhost:3306/mu","root","");
    con.setAutoCommit(false);

    Statement stmt=con.createStatement();
    stmt.executeUpdate("insert into student
values(45,'abhi','Surat')");

    stmt.executeUpdate("insert into student
values(39,'Hardik','Nepal')");
    System.out.println("Data Inserted");
    con.commit();
    con.close();
}
}

```

Output:

```

run:
Data Inserted

```

Experiment# 3 Servlet

Practical 1: Write down the Program for testing the Servlet and study deployment descriptor.

Program:

web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="3.1" xmlns="http://xmlns.jcp.org/xml/ns/javaee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app\_3\_1.xsd">
    <servlet>
        <servlet-name>MyServlet</servlet-name>
        <servlet-class>MyServlet</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>MyServlet</servlet-name>
        <url-pattern>/MyServlet</url-pattern>
    </servlet-mapping>
    <session-config>
        <session-timeout>
            30
        </session-timeout>
    </session-config>
    <welcome-file-list>
        <welcome-file>MyServlet</welcome-file>
    </welcome-file-list>
</web-app>
```

MyServlet.java

```
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.*;

public class MyFirstServlet extends HttpServlet
{
    public void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException
    {
        response.setContentType("text/html");
    }
}
```

```

try (PrintWriter out = response.getWriter())
{
    out.println("<html>");
    out.println("<head>");
    out.println("<title>MyServlet</title>");
    out.println("</head>");
    out.println("<body>");
    out.println("<h2>Hello! This is Servlet </h2>");
    out.println("<h2>Welcome to My Servlet !</h2>");
    out.println("</body>");
    out.println("</html>");
}
catch(Exception e)
{
    System.out.println(e);
}
}
}

```

Output:

Hello! This is Servlet

Welcome to My Servlet !

Practical 2: Write down the program for testing the include action for Servlet collaboration.

Program:

index.html

```

<!DOCTYPE >
<html>
  <head>
    <title>Login Page</title>

  </head>
  <body>
    <h1>Login</h1>
    <form action="Home" method="get" >
      <table>
        <tr>
          <td>User Name:</td>

```

```

        <td>
        <input type="text" name="name"/>
        </td>
    </tr>
    <tr>
        <td>Password:</td>
        <td><input type="password" name="pass"/>
        </td>
    </tr>
    <tr>
        <td></td>
        <td><input type="submit" value="Login"/></td>
    </tr>
</table>
</form>
</body>
</html>

```

web.xml

```

<?xml version="1.0" encoding="UTF-8"?>
<web-app      version="3.1"      xmlns="http://xmlns.jcp.org/xml/ns/javaee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app\_3\_1.xsd">
    <servlet>
        <servlet-name>Login</servlet-name>
        <servlet-class>Login</servlet-class>
    </servlet>
    <servlet>
        <servlet-name>Welcome</servlet-name>
        <servlet-class>Welcome</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>Login</servlet-name>
        <url-pattern>/Home</url-pattern>
    </servlet-mapping>
    <servlet-mapping>
        <servlet-name>Welcome</servlet-name>
        <url-pattern>/Welcome</url-pattern>
    </servlet-mapping>
    <session-config>
        <session-timeout>
            30
        </session-timeout>
    </session-config>
    <welcome-file-list>
        <welcome-file>index.html</welcome-file>
    </welcome-file-list>
</web-app>

```

Login.java

```
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.*;

public class Login extends HttpServlet
{
    public void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException
    {
        try
        {
            response.setContentType("text/html");
            PrintWriter out = response.getWriter();

            String p=request.getParameter("pass");
            if(p.equals("Password1234"))
            {
                RequestDispatcher rd=request.getRequestDispatcher("Home");
                rd.forward(request, response);
            }
            else
            {
                out.print("Sorry ! username or password are wrong , Please try
again");
                RequestDispatcher
rd=request.getRequestDispatcher("index.html");
                rd.include(request, response);
            }
        }
        catch(Exception e)
        {
            System.out.println(e);
        }
    }
}
```

Welcome.java

```
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.*;

public class Welcome extends HttpServlet
{
    public void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException
```

```

{
    response.setContentType("text/html");
    PrintWriter out = response.getWriter();
    String n=request.getParameter("name");
    out.print("Welcome "+n);
}
}

```

Output:

Login

User Name:

Password:

Welcome Priyanka

Practical 3: Write down the program for testing the forward action for Servlet collaboration.

Program:

index.html

```

<!DOCTYPE >
<html>
  <head>
    <title>Login Page</title>

  </head>
  <body>
    <h1>Login</h1>
    <form action="Home" method="get" >
      <table>
        <tr>
          <td>User Name:</td>

```

```

        <td>
        <input type="text" name="name"/>
        </td>
    </tr>
    <tr>
        <td>Password:</td>
        <td><input type="password" name="pass"/>
        </td>
    </tr>
    <tr>
        <td></td>
        <td><input type="submit" value="Login"/></td>
    </tr>
</table>
</form>
</body>
</html>

```

web.xml

```

<?xml version="1.0" encoding="UTF-8"?>
<web-app      version="3.1"      xmlns="http://xmlns.jcp.org/xml/ns/javaee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app\_3\_1.xsd">
    <servlet>
        <servlet-name>Login</servlet-name>
        <servlet-class>Login</servlet-class>
    </servlet>
    <servlet>
        <servlet-name>Welcome</servlet-name>
        <servlet-class>Welcome</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>Login</servlet-name>
        <url-pattern>/Home</url-pattern>
    </servlet-mapping>
    <servlet-mapping>
        <servlet-name>Welcome</servlet-name>
        <url-pattern>/Welcome</url-pattern>
    </servlet-mapping>
    <session-config>
        <session-timeout>
            30
        </session-timeout>
    </session-config>
    <welcome-file-list>
        <welcome-file>index.html</welcome-file>
    </welcome-file-list>
</web-app>

```

Login.java

```
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.*;

public class Login extends HttpServlet
{
    public void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException
    {
        try
        {
            response.setContentType("text/html");
            PrintWriter out = response.getWriter();

            String p=request.getParameter("pass");
            if(p.equals("Password1234"))
            {
                RequestDispatcher rd=request.getRequestDispatcher("Home");
                rd.forward(request, response);
            }
            else
            {
                out.print("Sorry ! username or password are wrong , Please try
again");
                RequestDispatcher
rd=request.getRequestDispatcher("index.html");
                rd.include(request, response);
            }
        }
        catch(Exception e)
        {
            System.out.println(e);
        }
    }
}
```

Welcome.java

```
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.*;

public class Welcome extends HttpServlet
{
    public void doGet(HttpServletRequest req,HttpServletResponse res)
    {
```



```

try {
    res.setContentType("text/html");
    PrintWriter out=res.getWriter();

    out.println("<html>");
    out.println("<head>");
    out.println("<title>Home </title>");
    out.println("</head>");
    out.println("<body>");
    out.println("<h3>Welcome to HomePage Servlet!</h3>");
    out.println("</body>");
    out.println("</html>");
}
catch(Exception e)
{
    System.out.println(e);
}
}
}

```

Output:

Login

User Name:	<input type="text" value="Privanka"/>
Password:	<input type="password" value="••••••••••"/>
	<input type="button" value="Login"/>

Welcome to HomePage Servlet!

Practical 4: Create login form and perform state management using Cookies, HttpSession and URL Rewriting.

Program:

web.xml

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

<web-app version="3.1" xmlns="http://xmlns.jcp.org/xml/ns/javaee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd">

    <servlet>

        <servlet-name>s1</servlet-name>

        <servlet-class>s1</servlet-class>

    </servlet>

    <servlet>

        <servlet-name>s2</servlet-name>

        <servlet-class>s2</servlet-class>

    </servlet>

    <servlet>

        <servlet-name>s3</servlet-name>

        <servlet-class>s3</servlet-class>

    </servlet>

    <servlet-mapping>

        <servlet-name>s1</servlet-name>

        <url-pattern>/s1</url-pattern>

    </servlet-mapping>

    <servlet-mapping>

        <servlet-name>s2</servlet-name>

        <url-pattern>/s2</url-pattern>

    </servlet-mapping>
```

```

<servlet-mapping>
  <servlet-name>s3</servlet-name>
  <url-pattern>/s3</url-pattern>
</servlet-mapping>
<session-config>
  <session-timeout>
    30
  </session-timeout>
</session-config>
</web-app>

```

index.html

```

<!DOCTYPE html>
<html>
  <head>
    <title>Login page</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
  </head>
  <body>
    <form action="s1" method="post">
      <br>
      Name:<input type="text" name="name">
      <!--br-->
      <br>
      Password:<input type="password" name="password">
      <!--br-->
      <br>
      <input type="submit" value="login">
      <br>
    </form>
  </body>
</html>

```

```
</body>
</html>
```

s1.java

```
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.Cookie;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

public class s1 extends HttpServlet {

    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();

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

        if(password.equals("admin123")){
            out.print("<a href='s2'>Home Page</a>");
            HttpSession session=request.getSession();
            session.setAttribute("name",name);

            Cookie ck = new Cookie ("name",name);
            response.addCookie(ck);
```

```

    }
    else{
        out.print("Sorry, incorrect username or password");
        request.getRequestDispatcher("index.html").include(request, response);
    }
    out.close();
}
}

```

s2.java

```

import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.Cookie;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

public class s2 extends HttpServlet {

    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();

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

        out.print("Welcome, "+name);
    }
}

```

```

        out.print("<a href='s3'>Logout</a>");
    }
}

```

s3.java

```

import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.Cookie;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

public class s3 extends HttpServlet {

    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();

        HttpSession session=request.getSession();
        session.invalidate();

        Cookie ck=new Cookie("name","");//deleting value of cookie
        ck.setMaxAge(0);//changing the maximum age to 0 seconds
        response.addCookie(ck);
        out.print("Successfully logged out!");
        out.close();
    }
}

```

Output:

The image displays three sequential screenshots of a web application's login interface. The first screenshot shows a login form with a 'Name' field containing 'ns', a 'Password' field with masked characters, and a 'login' button. The second screenshot shows an error message 'Sorry, incorrect username or password' above the login form. The third screenshot shows a 'Welcome ps' message, a 'Logout' link, and a 'Successfully logged out!' message.

Home Page

Welcome ps Logout

Successfully logged out!

Practical 5: Create Servlet file which contains following functions:

1. Connect
2. Create Database
3. Create Table
4. Insert Records into respective table
5. Update records of particular table of database
6. Delete Records from table.
7. Delete table and also database.

Program:

web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app      version="3.1"      xmlns="http://xmlns.jcp.org/xml/ns/javaee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app\_3\_1.xsd">
  <servlet>
    <servlet-name>dbconnect</servlet-name>
    <servlet-class>dbconnect </servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>dbconnect </servlet-name>
```

```

        <url-pattern>/Database</url-pattern>
    </servlet-mapping>
    <session-config>
        <session-timeout>
            30
        </session-timeout>
    </session-config>
</web-app>

```

dbconnect.java

```

import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.Statement;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
public class dbconnect extends HttpServlet {

    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        Statement st,st1,st2,st3;
        PreparedStatement ps,ps1,ps2;
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        out.println("<html><body>");
        try {

            Class.forName("com.mysql.jdbc.Driver");
            Connection
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/student", "root",
"admin");
            st=con.createStatement();
            ResultSet rs=st.executeQuery("select * from stuinfo");
            out.println("<b>Fetching all the records</b>");
            out.println("<table
border='1'><tr><td>ID</td><td>Name</td><td>AJ</td><td>WT</td></tr>");
            while(rs.next()){
                out.println("<tr>");
                out.println("<td>" +rs.getString("id")+"</td>");
                out.println("<td>" +rs.getString("name")+"</td>");
                out.println("<td>" +rs.getString("aj")+"</td>");
                out.println("<td>" +rs.getString("wt")+"</td>");
                out.println("</tr>");
            }
        }
    }
}

```



```

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

//Inserting New Record
out.println("<b>Inserting New Tuple</b>");
ps=con.prepareStatement("insert into stuinfo(id,name,aj,wt)values(?,?,?,?)");
ps.setInt(1,4);
ps.setString(2,"s4");
ps.setInt(3,80);
ps.setInt(4,80);
ps.executeUpdate();

st1=con.createStatement();
ResultSet rs1=st1.executeQuery("select * from stuinfo");
out.println("<b>& Fetching Records after Insertion</b>");
out.println("<table
border='1'><tr><td>ID</td><td>Name</td><td>AJ</td><td>WT</td></tr>");
while(rs1.next()){
    out.println("<tr>");
    out.println("<td>"+rs1.getString("id")+"</td>");
    out.println("<td>"+rs1.getString("name")+"</td>");
    out.println("<td>"+rs1.getString("aj")+"</td>");
    out.println("<td>"+rs1.getString("wt")+"</td>");
    out.println("</tr>");
}
out.println("</table>");
st1.close();

//Updating Record 3
out.println("<b>Updating 3rd Tuple</b>");
ps1=con.prepareStatement("update stuinfo set wt=50 where id=3");
int i=ps1.executeUpdate();
System.out.println(i);

st2=con.createStatement();
ResultSet rs2=st2.executeQuery("select * from stuinfo");
out.println("<b>& Fetching Records after Updation</b>");
out.println("<table
border='1'><tr><td>ID</td><td>Name</td><td>AJ</td><td>WT</td></tr>");
while(rs2.next()){
    out.println("<tr>");
    out.println("<td>"+rs2.getString("id")+"</td>");
    out.println("<td>"+rs2.getString("name")+"</td>");
    out.println("<td>"+rs2.getString("aj")+"</td>");
    out.println("<td>"+rs2.getString("wt")+"</td>");
    out.println("</tr>");
}
out.println("</table>");

```

```

        st2.close();

        //Deleting 1nd record
        out.println("<b>Deleting 2nd Record</b>");
        ps2=con.prepareStatement("delete from stuinfo where id=1");
        ps2.executeUpdate();

        st3=con.createStatement();
        ResultSet rs3=st3.executeQuery("select * from stuinfo");
        out.println("<b>& Fetching Records after Deletion</b>");
        out.println("<table
border='1'><tr><td>ID</td><td>Name</td><td>AJ</td><td>WT</td></tr>");
        while(rs3.next()){
            out.println("<tr>");
            out.println("<td>" +rs3.getString("id")+"</td>");
            out.println("<td>" +rs3.getString("name")+"</td>");
            out.println("<td>" +rs3.getString("aj")+"</td>");
            out.println("<td>" +rs3.getString("wt")+"</td>");
            out.println("</tr>");
        }
        out.println("</table>");
        st3.close();

        out.println("</body></html>");
        rs3.close();
        rs2.close();
        rs1.close();
        rs.close();
        ps2.close();
        ps1.close();
        ps.close();
        st.close();
        con.close();

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

```

Output:

Database created successfully...

Table created successfully...

Data inserted...!!

Table updated....!!

Record Deleted....!!

Table Deleted....!!

SELECT YOUR CHOICE

DELETE DATABASE ▾

Lets try..!!

DB Deleted....!!

Practical 6: Write down the Program in which error is handled by the deployment descriptor file (web.xml).

Program:

web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns="http://java.sun.com/xml/ns/javaee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app\_3\_0.xsd
  version="3.0">

  <display-name>ServletExceptionHandler</display-name>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
  </welcome-file-list>

  <error-page>
    <error-code>404</error-code>
    <location>/ExceptionHandler</location>
  </error-page>

  <error-page>
    <exception-type>javax.servlet.ServletException</exception-type>
    <location>/ExceptionHandler</location>
  </error-page>

</web-app>
```

ExceptionHandlerServlet.java

```
import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.*;

@WebServlet("/ExceptionHandlerServlet")
public class ExceptionServlet extends HttpServlet
{
    private static final long serialVersionUID = 1L;

    protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException
    {
        throw new ServletException("GET method is not supported!!!");
    }
}
```

ExceptionHandler.java

```
import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@WebServlet("/ExceptionHandler")
public class ExceptionHandling extends HttpServlet
{
    private static final long serialVersionUID = 1L;

    protected void doGet(HttpServletRequest request,
        HttpServletResponse response) throws ServletException,
IOException {
        processError(request, response);
    }

    protected void doPost(HttpServletRequest request,
        HttpServletResponse response) throws ServletException,
IOException {
        processError(request, response);
    }

    private void processError(HttpServletRequest request,
        HttpServletResponse response) throws IOException {
        // Analyze the servlet exception
    }
}
```

```

        Throwable throwable = (Throwable) request
            .getAttribute("javax.servlet.error.exception");
        Integer statusCode = (Integer) request
            .getAttribute("javax.servlet.error.status_code");
        String servletName = (String) request
            .getAttribute("javax.servlet.error.servlet_name");
        if (servletName == null) {
            servletName = "Unknown";
        }
        String requestUri = (String) request
            .getAttribute("javax.servlet.error.request_uri");
        if (requestUri == null) {
            requestUri = "Unknown";
        }

        // Set response content type
        response.setContentType("text/html");

        PrintWriter out = response.getWriter();
        out.write("<html><head><title>Exception/Error
Details</title></head><body>");
        if(statusCode != 500){
            out.write("<h3>Error Details</h3>");
            out.write("<strong>Status Code:</strong>"+statusCode+"<br>");
            out.write("<strong>Requested URI</strong>:"+requestUri);
        }else{
            out.write("<h3>Exception Details</h3>");
            out.write("<ul><li>Servlet Name:</li>"+servletName+"</li>");
            out.write("<li>Exception
Name:</li>"+throwable.getClass().getName()+"</li>");
            out.write("<li>Requested URI:</li>"+requestUri+"</li>");
            out.write("<li>Exception
Message:</li>"+throwable.getMessage()+"</li>");
            out.write("</ul>");
        }

        out.write("<br><br>");
        out.write("<a href=\"index.html\">Home Page</a>");
        out.write("</body></html>");
    }
}

```

Output:

Exception Details

- Servlet Name:ExceptionServlet
- Exception Name:javax.servlet.ServletException
- Requested URI:/Experiment_4_6_/ExceptionServlet
- Exception Message:GET method is not supported!!!

[Home Page](#)

Practical 7: Implement Authentication filter using filter API. Program:

index.html

```
<!DOCTYPE>
<html>
  <head>
    <title>Login Page</title>

  </head>
  <body>
    <h1>Login Application Using Filters </h1>
    <form action="Welcome" >
      <table>
        <tr>
          <td>User Name:</td>
          <td>
            <input type="text" name="username"/>
          </td>
        </tr>
        <tr>
          <td>Password:</td>
          <td><input type="password" name="password"/>
          </td>
        </tr>
        <tr>
          <td></td>
          <td><input type="submit" value="Submit"/></td>
        </tr>
      </table>
    </form>
  </body>
</html>
```

web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app      version="3.1"      xmlns="http://xmlns.jcp.org/xml/ns/javaee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app\_3\_1.xsd">

    <filter>
        <filter-name>MyFilter</filter-name>
        <filter-class>MyFilter</filter-class>
    </filter>
    <filter-mapping>
        <filter-name>MyFilter</filter-name>
        <servlet-name>WelcomePage</servlet-name>
    </filter-mapping>

    <servlet>
        <servlet-name>WelcomePage</servlet-name>
        <servlet-class>WelcomePage</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>WelcomePage</servlet-name>
        <url-pattern>/Welcome</url-pattern>
    </servlet-mapping>
    <session-config>
        <session-timeout>
            30
        </session-timeout>
    </session-config>
</web-app>
```

MyFilter.java

```
import java.util.*;
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class MyFilter extends GenericFilter
{
    public void doFilter(ServletRequest req, ServletResponse res, FilterChain
chain) throws IOException, ServletException
    {
        String password = ((HttpServletRequest)
req).getParameter("password");

        if(password.equals("Password1234"))
        {
```

```

        String uri =
        ((HttpServletRequest)req).getRequestURI();
        chain.doFilter(req, res);
    }
    else
    {
        res.setContentType("text/html");
        PrintWriter pw = res.getWriter();
        pw.println("<html>");
        pw.println("<head><title>Wrong
Password</title></head>");
        pw.println("<body>");
        pw.println("<h2>Sorry , Password is wrong</h3>");
        pw.println("<h3>Please try again</h3>");
        pw.println("</body>");
        pw.println("</html>");
    }
}
}

```

GenericFilter.java

```

import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class GenericFilter implements Filter
{
    private FilterConfig filterconf = null;

    public void doFilter(final ServletRequest req, final ServletResponse res,
FilterChain chain) throws IOException, ServletException
    {
        chain.doFilter(req,res);
    }

    public FilterConfig getFilterConfig()
    {
        return filterconf;
    }

    public void setFilterConfig(final FilterConfig filterconf)
    {
        this.filterconf = filterconf;
    }

    public void init(FilterConfig filterconf)
    {
        this.filterconf = filterconf;
    }
}

```



```

    }

    public void destroy()
    {
        this.filterconf = null;
    }
}

```

WelcomePage.java

```

import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.util.*;

public class WelcomePage extends HttpServlet
{
    public void doGet(HttpServletRequest req, HttpServletResponse res) throws
ServletException, IOException
    {
        res.setContentType("text/html");
        PrintWriter out = res.getWriter();
        String username = req.getParameter("username");
        out.println("<html><body><h2><b>Welcome  <b>" + username +
"<br/><br/></h2>");
        out.println(new Date().toString());
        out.println("</b></body></html>");
    }
}

```

Output:

Login Application Using Filters

User Name:	<input type="text" value="Privanka"/>
Password:	<input type="password" value="....."/>
	<input type="button" value="Submit"/>

Welcome Priyanka

Tue Jul 31 18:33:31 IST 2018

Practical 8: Write down the Program for testing the Servlet context interface.

Program:

web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app      version="3.1"      xmlns="http://xmlns.jcp.org/xml/ns/javaee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app\_3\_1.xsd">
    <servlet>
        <servlet-name>ServletContext</servlet-name>
        <servlet-class>ServletContext</servlet-class>
    </servlet>

    <context-param>
        <param-name>dburl</param-name>
        <param-value>com.mysql.jdbc.Driver</param-value>
    </context-param>

    <servlet-mapping>
        <servlet-name>ServletContext</servlet-name>
        <url-pattern>/ServletContext</url-pattern>
    </servlet-mapping>
    <session-config>
        <session-timeout>
            30
        </session-timeout>
    </session-config>

    <welcome-file-list>
        <welcome-file>ServletContext</welcome-file>
    </welcome-file-list>

</web-app>
```

ServletContext.java

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

import java.util.*;

public class ServletContext extends HttpServlet{
```

```

public void doGet(HttpServletRequest req, HttpServletResponse res)
{
    try {
        res.setContentType("text/html");
        PrintWriter out=res.getWriter();

        //creating servletContext
        javax.servlet.ServletContext context=getServletContext();

        //getting value of the initialization parameter and printing it
        String db=context.getInitParameter("dburl");
        out.println("Database URL is = "+db);
    }
    catch(Exception e)
    {
        System.out.println(e);
    }
}
}

```

Output:

Database URL is = com.mysql.jdbc.Driver

Experiment# 4

JSP

Practical 1: Write down the Program which displays the simple JSP file.

Program Code:

myjsp.jsp

```
<%--
  Document   : myjsp
  Created on : Jul 30, 2018, 9:13:02 AM
  Author    : student
--%>

<% @page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>JSP Page</title>
  </head>
  <body>
    <h1>Hello User!</h1>
    <%= "This tag is used to print data"%>
  </body>
</html>
```

Output:

Hello User!

This tag is used to print data

Practical 2: Write down the program in which input the two numbers in an html file and then display the addition in JSP file.

Program Code:

Index.html

```
<!DOCTYPE html>
<!--
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.
-->
<html>
    <head>
        <title>Addition of numbers via JSP</title>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    </head>

    <body>
        <form action="addjsp.jsp">
            Number 1: <input type="text" name="no1">
            <br>
            Number 2: <input type="text" name="no2">
            <br>
            <input type="submit" value="calculate">
            <br>
        </form>
    </body>
</html>
```

addjsp.jsp

```
<%--
Document : addjsp
Created on : Jul 30, 2018, 9:21:35 AM
Author : student
--%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>JSP Page</title>
    </head>
    <body>
        <%
            String n1 = request.getParameter("no1");
            String n2 = request.getParameter("no2");
```

```
int a1= Integer.parseInt(n1);
int a2= Integer.parseInt(n2);

int a3= a1+ a2;

out.print("Addition of "+ a1+ " & "+a2+ " is " + a3);
%>
</body>
</html>
```

Output:

Number 1: 4	Addition of 4 & 6 is 10
Number 2: 6	
<input type="button" value="calculate"/>	

Practical 3: Write down the program in which display the error by common file for all general pages.

Program Code:

Index.html

```
<!DOCTYPE html>
<!--
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.
-->
<html>
    <head>
        <title>ERROR CALCULATION</title>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    </head>
    <body>

        <form action="myjsp.jsp">
        No1:<input type="text" name="n1" />
        <br/><br/>
        No2:<input type="text" name="n2" />
        <br/><br/>
        <input type="submit" value="divide"/>
        </form>

    </body>
</html>
```

myjsp.jsp

```
<%--
Document : myjsp
Created on : Aug 1, 2018, 11:06:27 AM
Author : student
--%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>JSP Page</title>
    </head>
    <body>
        <%@ page errorPage="errorjsp.jsp" %>
        <%
```



```
String num1=request.getParameter("n1");
String num2=request.getParameter("n2");

int a=Integer.parseInt(num1);
int b=Integer.parseInt(num2);
int c=a/b;
out.print("division of numbers is: "+c);

%>
</body>
</html>
```

errorjsp.jsp

```
<%--
Document : errorjsp
Created on : Aug 1, 2018, 11:06:50 AM
Author : student
--%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Page Error</title>
</head>
<body>

<%@ page isErrorPage="true" %>

<h3>Sorry an exception occurred!</h3>

Exception is: <%= exception %>
</body>
</html>
```

Output:

No1:

No2:

Sorry an exception occurred!

Exception is: java.lang.ArithmeticException: / by zero

Practical 4: Perform Database Access through JSP.

Program Code:

Index.html

```
<html>
  <head>
    <title>MU results</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
  </head>
  <body>
    <form action="data.jsp">
      Enrollment no:
      <input type="text" name="no" /><br/>

      <input type="submit" value="Enter"/>
    </form>
  </body>
</html>
```

data.jsp

```
<%--
  Document   : data
  Created on : Aug 8, 2018, 2:23:38 PM
  Author      : HP
--%>

<% @page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title></title>
  </head>
  <body>
    <% @page import="java.sql.*" %>
    <h1>MU RESULT</h1>
    <%
      Class.forName("com.mysql.jdbc.Driver");
      Connection con =
DriverManager.getConnection("jdbc:mysql://localhost:3306/mu", "root", "");
      Statement st = con.createStatement();
      ResultSet rs = st.executeQuery("select * from student where enroll_no = '" +
request.getParameter("no")+"'");
```

```

        out.println("<table border=\"1\"><tr><td>Enrollment
no.</td><td>Name</td><td>Physics</td><td>Chemistry</td><td>Mathematics</td><td>O
verall</td></tr><br>");
        while(rs.next())
        {
            out.println("<tr>");
            out.println("<td> "+rs.getInt(1)+" </td>");
            out.println("<td> "+rs.getString(2)+" </td>");
            out.println("<td> "+rs.getInt(3)+" </td>");
            out.println("<td> "+rs.getInt(4)+" </td>");
            out.println("<td> "+rs.getInt(5)+" </td>");
            out.println("<td> "+rs.getString(6)+" </td>");
            out.println("<tr>");
        }
    %>
</table>
</body>
</html>

```

Output:

Enrollment no:

MU RESULT

Enrollment no.	Name	Physics	Chemistry	Mathematics	Overall
100873	Raj	56	67	49	PASS

Practical 5: Write down the Program for testing the include action tag in JSP.

Program Code:

Index.html

```
<!DOCTYPE html>
<!--
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.
-->
<html>
  <head>
    <title>Including tags</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  </head>
  <body>
    <div>This is your html file</div>
  </body>
</html>
```

jsp1.jsp

```
<%--
Document : jsp1
Created on : Aug 2, 2018, 8:33:48 AM
Author : student
--%>

<% @page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>JSP Page</title>
  </head>

  <br>
  Today's date: <%= new java.util.Date().getDate()%>
  <%= new java.util.Date().getMonth()%>
  <%= new java.util.Date().getYear()%><br><br>
  Time <%= new java.util.Date().getHours()%>
  : <%= new java.util.Date().getMinutes()%>
  : <%= new java.util.Date().getSeconds()%>

</html>
```

jsp2.jsp

```
<%--
  Document   : jsp2
  Created on : Aug 2, 2018, 8:34:40 AM
  Author    : student
--%>

<% @page contentType="text/html" pageEncoding="UTF-8"% >
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Include Page</title>
  </head>
  <body>
    <!--center-->
    <h2>The include action Example</h2>
    <jsp:include page = "index.html" />
    <jsp:include page = "jsp1.jsp" />
    <!--/center-->
  </body>
</html>
```

Output:

The include action Example

This is your html file

Today's date: 2 7 118

Time 8 : 50 : 44

Practical 6: Write down the Program for testing the forward action tag.

Program Code:

j_1.jsp

```
<%--
  Document   : j_1
  Created on : Aug 2, 2018, 9:05:07 AM
  Author      : student
--%>

<% @page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Forwarded page</title>
  </head>
  <body>
    <br>
    Student details
    <br>
    Name: <%= request.getParameter("name")%><br>
    Enroll no: <%= request.getParameter("no")%><br>
    Branch: <%= request.getParameter("no")%><br>
    E-mail: <%= request.getParameter("mail")%><br>
  </body>
</html>
```

j_2.jsp

```
<%--
  Document   : jsp2
  Created on : Aug 2, 2018, 8:34:40 AM
  Author      : student
--%>

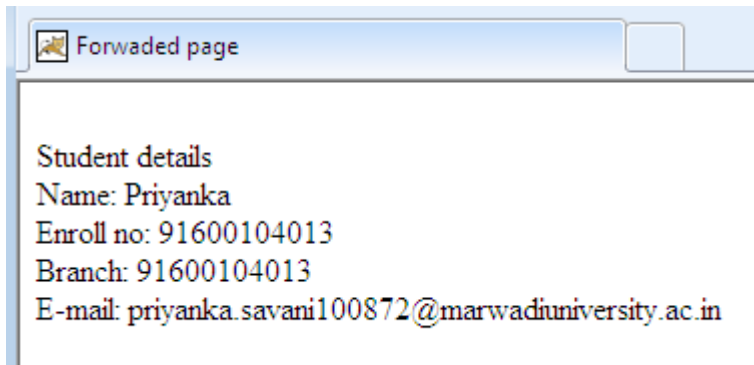
<% @page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>JSP Page</title>
  </head>
  <body>

    <h2>The include action Example</h2>
    <jsp:forward page = "j_1.jsp" >
    <jsp:param name="name" value="Priyanka" />
```

```
<jsp:param name="no" value="91600104013" />
<jsp:param name="branch" value="Information Technology" />
<jsp:param name="mail" value="priyanka.savani100872@marwadiuniversity.ac.in" />
</jsp:forward>

</body>
</html>
```

Output:



Practical 7: Write down a program which demonstrates the core tag of JSTL.

Program Code:

Index.html

```
<!DOCTYPE html>
<!--
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.
-->
<html>
  <head>
    <title>Login using core tag</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  </head>
  <body>
    <form action="core1.jsp" method="Post">
      Name <input type="text" name="uname"><br>
      Password <input type="text" name="pass">
      <input type="submit" value="Login">
    </form>
  </body>
</html>
```

core1.jsp

```
<%--
Document : core1
Created on : Aug 11, 2018, 8:20:20 AM
Author : HP
--%>

<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<%@ page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Login Page</title>
  </head>
  <body>

    <c:out value="List of books"/><br>

    <c:set var="Site" scope="session" value="New Beginning"/> <br>
    <c:set var="author" scope="session" value="Chaitanya"/>
    <c:remove var="author"/>
    <c:import var="mydata" url="core2.jsp"/>
```



```
<c:out value="\${mydata}"/>
</body>
</html>
```

core2.jsp

```
<%--
Document : core2
Created on : Aug 11, 2018, 8:34:42 AM
Author : HP
--%>

<% @page contentType="text/html" pageEncoding="UTF-8"%>
<% @ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>

<c:out value="\${Site}"/>
<c:out value="\${author}" default=" Book has no author"/>
```

Output:

Name

Password

List of books

New Beginning Book has no author

Practical 8: Write down a program which demonstrates the Format tag of JSTL.

Program Code:

format_tag.jsp

```
<%--
    Document   : format_tag
    Created on : Aug 28, 2018, 8:42:55 AM
    Author    : HP
--%>

<% @page contentType="text/html" pageEncoding="UTF-8"%>
<% @ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<% @ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>
<!DOCTYPE html>
<html>
<head>
    <title>Format Tag</title>
</head>
<body>
<h3>Location Details:</h3>
<c:set var="date" value="<%=new java.util.Date()%>" />
<p>
<b>Formatted Time : </b>
<fmt:formatDate type="time" value="${Date}" />
</p>
<p>
<b>Formatted Date :</b>
<fmt:formatDate type="date" value="${Date}" />
</p>
```

```

<b>Parsed Date :</b> 28-11-2016
<c:set var="Date" value="28-11-2016" />
<fmt:parseDate value="\${Date}" var="parsedDate" pattern="dd-MM-yyyy" />
<p><c:out value="\${parsedDate}" /></p>
<b>Date & Time : </b>
<p>Date and Time in Indian Standard Time(IST) Zone:<fmt:formatDate value="\${date}"
    type="both" timeStyle="long" dateStyle="long" /></p>
<fmt:setTimeZone value="GMT-10" />
<p>Date and Time in GMT-10 time Zone: <fmt:formatDate value="\${date}"
    type="both" timeStyle="long" dateStyle="long" /></p>

</body>
</html>

```

Output:

Location Details:

Formatted Time :

Formatted Date :

Parsed Date : 28-11-2016

Mon Nov 28 00:00:00 IST 2016

Date & Time :

Date and Time in Indian Standard Time(IST) Zone: August 28, 2018 8:54:23 AM IST

Date and Time in GMT-10 time Zone: August 27, 2018 5:24:23 PM GMT-10:00

Practical 9: Write down a program which demonstrates the Function tag of JSTL.

Program Code:

functiontag.jsp

```
<%--
    Document : functiontag
    Created on : Aug 28, 2018, 8:11:16 AM
    Author : HP
--%>

<% @page contentType="text/html" pageEncoding="UTF-8"%>
<% @ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<% @ taglib uri="http://java.sun.com/jsp/jstl/functions" prefix="fn" %>
<!DOCTYPE html>

<html>
<head>
<title> JSTL Function </title>
</head>
<body>
<c:set var="string" value="Welcome to My Blog"/>
${fn:toLowerCase("HELLO,")} <br> <br>
${fn:toUpperCase(string)} <br> <br>
${fn:substring(string, 11, 18)} <br> <br>
<c:set var="user" value="Priyanka Savani"/>
This page is developed by ${fn:toUpperCase(user)}<br>
This page is developed by ${fn:replace(user, "Savani", "Patel")}
</body>
</html>
```

Output:

hello,

WELCOME TO MY BLOG

My Blog

This page is developed by PRIYANKA SAVANI

This page is developed by Priyanka Patel

Practical 10: Write down a program which demonstrates the SQL tag of JSTL.

Program Code:

Index.html

```
<!DOCTYPE html>
<!--
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.
-->
<html>
    <head>
        <title>SQL Query</title>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    </head>
    <body>
        <form action="sql.jsp">
            ID: <input type="text" name="no" /><br/>

            <input type="submit" value="Enter"/>
        </form>
    </body>
</html>
```

sql.jsp

```
<%--
Document : sql
Created on : Aug 11, 2018, 8:53:03 AM
Author : HP
--%>

<% @ page import="java.io.*,java.util.*,java.sql.*"%>
<% @ page import="javax.servlet.http.*,javax.servlet.*" %>
<% @ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<% @ taglib uri="http://java.sun.com/jsp/jstl/sql" prefix="sql"%>

<head>
<title>sql:query Tag</title>
</head>

<sql:setDataSource var="db" driver="com.mysql.jdbc.Driver"
url="jdbc:mysql://localhost:3306/mu"
user="root" password=""/>
```

```
<sql:query dataSource="${db}" var="rs">
SELECT * from student;
</sql:query>
```

```
<table border="2" width="100%">
<tr>
<th>Student ID</th>
<th>Name</th>
<th>Physics</th>
<th>Chemistry</th>
<th>Mathematics</th>
<th>Overall</th>

</tr>
<c:forEach var="table" items="${rs.rows}">
<tr>
<td><c:out value="${table.enroll_no}"/></td>
<td><c:out value="${table.name}"/></td>
<td><c:out value="${table.sub1}"/></td>
<td><c:out value="${table.sub2}"/></td>
<td><c:out value="${table.sub3}"/></td>
<td><c:out value="${table.result}"/></td>
</tr>
</c:forEach>
</table>
```

Output:

ID:

Student ID	Name	Physics	Chemistry	Mathematics	Overall
100871	Mili	65	42	31	FAIL
100872	Anjali	34	61	75	FAIL
100873	Raj	56	67	49	PASS
100874	Harsh	75	89	62	PASS
100875	Riya	66	63	71	PASS

Practical 11: Write down a program which demonstrates the XML tag of JSTL.

Program Code:

sql.jsp

```
<%--
    Document : xml
    Created on : Aug 16, 2018, 11:23:07 AM
    Author : HP
--%>

<% @page contentType="text/html" pageEncoding="UTF-8"%>
    <% @ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
    <% @ taglib prefix="x" uri="http://java.sun.com/jsp/jstl/xml" %>
    <html>
    <head>
        <title>x:set Tag</title>
    </head>
    <body>
        <h3>Books Information:</h3>
        <c:set var="book">
        <books>
        <book>
            <name>Three mistakes of my life</name>
            <author>Chetan Bhagat</author>
            <price>200</price>
        </book>
        <book>
            <name>Tomorrow land</name>
            <author>Brad Bird</author>
```



```
<price>2000</price>
</book>
</books>
</c:set>
<x:parse xml="{book}" var="output"/>
<x:set var="fragment" select="$output/books/book[2]/price"/>
<b>The price of the Tomorrow land book</b>:
<x:out select="$fragment" />
</body>
</html>
```

Output:

Books Information:

The price of the Tomorrow land book: 2000

Practical 12: Write down a program which demonstrates the Tag Handler with appropriate output.

Program Code:

MyTagHandler.java

```
package com;

import java.io.IOException;
import java.util.Calendar;
import javax.servlet.jsp.JspException;
import javax.servlet.jsp.JspWriter;
import javax.servlet.jsp.tagext.SimpleTagSupport;

public class MyTagHandler extends SimpleTagSupport {
    public void doTag() throws IOException, JspException {
        getJspContext().getOut().append("Hello from a simple tag handler!!");
    }
}
```

mytags.tld

```
<?xml version="1.0" encoding="UTF-8"?>
<taglib version="2.1" xmlns="http://java.sun.com/xml/ns/javaee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-jsptaglibrary_2_1.xsd">
    <tlib-version>1.0</tlib-version>
    <short-name>mytags</short-name>
    <uri>/WEB-INF/tlds/mytags</uri>

    <tag>
        <name>simple</name>
        <tag-class>com.MyTagHandler</tag-class>
        <body-content>empty</body-content>
    </tag>
</taglib>
```

web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="2.5" xmlns="http://java.sun.com/xml/ns/javaee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd">
  <session-config>
    <session-timeout>
      30
    </session-timeout>
  </session-config>
  <welcome-file-list>
    <welcome-file>index.jsp</welcome-file>
  </welcome-file-list>
</web-app>
```

content.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<Context antiJARLocking="true" path="/TagDemo"/>
```

Output:

```
Hello from a simple tag handler!!
```

Practical 13: Create database of student subject-wise data and retrieve all data using JSP and generate xml structure along with DTD and XML Schema definition

Program Code:

genXML.jsp

```
<?xml version="1.0" encoding="UTF-8"?>
<% @page contentType="text/xml" pageEncoding="UTF-8"%>
<% @page import="stocks.*" %>
<jsp:useBean id="portfolio" class="stocks.PortfolioBean" />
<%
java.util.Iterator folio = portfolio.getPortfolio();
Stock stock = null;
%>
<portfolio>
<% while (folio.hasNext()) { %>
<% stock = (Stock)folio.next(); %>
<stock>
<symbol><%=
    stock.getSymbol() %></symbol>
<name><%=
    stock.getName() %></name>
<price><%=
    stock.getPrice() %></price>
</stock>
<% } %>
</portfolio>
```

PortfolioBean.java

```
package stocks;

import java.util.*;

public class PortfolioBean implements
java.io.Serializable {
    private Vector portfolio = new Vector();

    public PortfolioBean() {
```

```

        portfolio.addElement(new Stock("SUNW",
"Sun Microsystems", (float) 17.1));
        portfolio.addElement(new Stock("AOL",
"America Online", (float) 51.05));
        portfolio.addElement(new Stock("IBM",
"International Business Machines",
(float) 116.10));
        portfolio.addElement(new Stock("MOT",
"MOTOROLA", (float) 15.20));
    }

    public Iterator getPortfolio() {
        return portfolio.iterator();
    }
}

```

Stock.java

```

package stocks;

public class Stock implements java.io.Serializable {
    private String symbol;
    private String name;
    private float price;

    public Stock(String symbol, String name,
float price) {
        this.symbol = symbol;
        this.name = name;
        this.price = price;
    }

    public String getSymbol() {
        return symbol;
    }

    public String getName() {
        return name;
    }

    public float getPrice() {
        return price;
    }
}

```

Output:

```
<?xml version="1.0" encoding="UTF-8"?>
- <portfolio>
  - <stock>
    <symbol>SUNW</symbol>
    <name>Sun Microsystems</name>
    <price>17.1</price>
  </stock>
  - <stock>
    <symbol>AOL</symbol>
    <name>America Online</name>
    <price>51.05</price>
  </stock>
  - <stock>
    <symbol>IBM</symbol>
    <name>International Business Machines</name>
    <price>116.1</price>
  </stock>
  - <stock>
    <symbol>MOT</symbol>
    <name>MOTOROLA</name>
    <price>15.2</price>
  </stock>
</portfolio>
```

Experiment# 5 Hibernate Framework

Practical 1: Study and implement Hibernate.

Program:

hibernet.java

```
package hibernet;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;
public class Hibernet {
    public static void main(String[] args){
        Configuration con = new Configuration();
        con.configure("Recources/Student.cfg.xml");
        SessionFactory sf = con.buildSessionFactory();
        sf.close();
    }
}
```

Studentbeans.java

```
package Beans;
public class Studentbeans{
    private String name;
    private int id;
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
}
```

```
}  
}
```

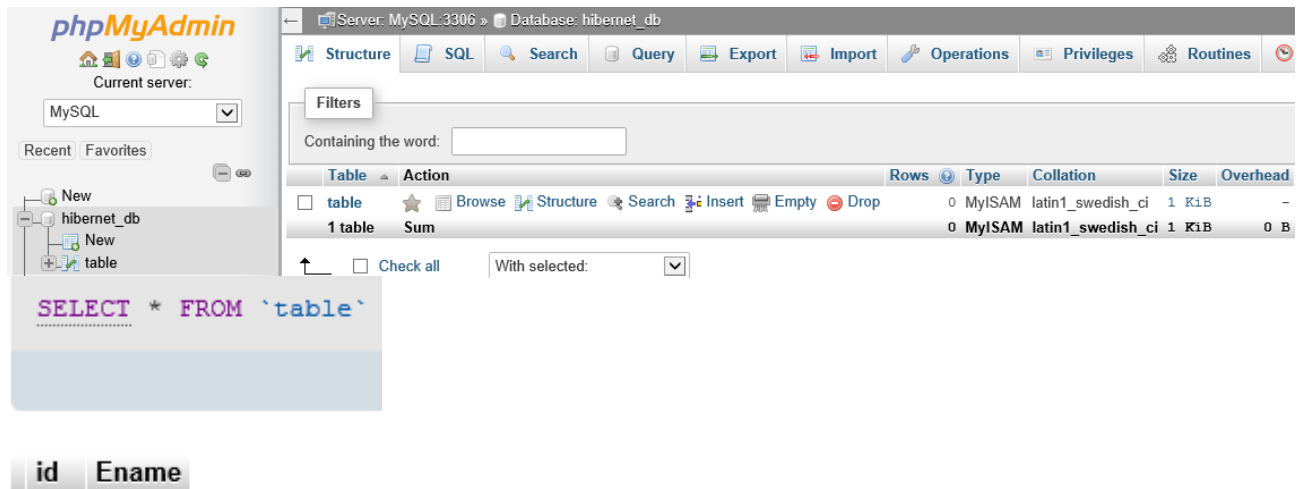
Student.cfg.xml

```
<?xml version="1.0" encoding="UTF-8"?>  
<!DOCTYPE hibernate-configuration PUBLIC "-//Hibernate/Hibernate  
Configuration DTD 3.0//EN" "http://hibernate.sourceforge.net/hibernate-  
configuration-3.0.dtd">  
<hibernate-configuration>  
  <session-factory>  
    <property  
      name="hibernate.dialect">org.hibernate.dialect.MariaDBDialect</property>  
    <property  
      name="hibernet.connection.driver_class">com.mysql.jdbc.Driver</property>  
    <property  
      name="hibernet.connection.url">jdbc:mysql://localhost:3306/hibernet_db</pro  
      perty>  
    <property name="hibernet.connection.username">root</property>  
    <property name="hibernet.connection.password"></property>  
    <property name="hibernet.show_sql">true</property>  
    <property name="hibernet.hbm2ddl.auto">create</property>  
    <mapping resource="Recources/Student.hbm.xml"/>  
  </session-factory>  
</hibernate-configuration>
```

Student.hbm.xml

```
<?xml version="1.0" encoding="UTF-8"?>  
<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping  
DTD 3.0//EN" "http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">  
<hibernate-mapping>  
  <class name="Beans.Studentbeans" schema="hibernet_db" table="table">  
    <id name="id" ></id>  
    <property name="name" column="Ename"></property>  
  </class>  
</hibernate-mapping>
```


Output:



Practical 2: Study and Implement Hibernate Annotations.

Program:

Employee.java

```
package com;
import javax.persistence.*;
public class Employee {
    private int id;
    private String firstName;
    private String lastName;
    private int salary;
    public Employee() {}
    public int getId() {
        return id;
    }
    public void setId( int id ) {
        this.id = id;
    }
    public String getFirstName() {
        return firstName;
    }
}
```

```

    }
    public void setFirstName( Stringfirst_name ) {
this.firstName = first_name;
    }
    public String getLastName() {
        return lastName;
    }
    public void setLastName( Stringlast_name ) {
this.lastName = last_name;
    }
    public int getSalary() {
        return salary;
    }
    public void setSalary( int salary ) {
this.salary = salary;
    }
}

```

ManageEmployee.java

```

package com;
import java.util.Iterator;
import java.util.List;
import org.hibernate.HibernateException;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;
import org.hibernate.cfg.AnnotationConfiguration;
public class ManageEmployee {
    private static SessionFactory factory;
    public static void main(String[] args) {
        try {
            factory = new AnnotationConfiguration().
configure().
addPackage("com"). //add package if used.
addAnnotatedClass(Employee.class).
buildSessionFactory();
        } catch (Throwable ex) {
System.err.println("Failed to create sessionFactory object." + ex);
            throw new ExceptionInInitializerError(ex);

```

```

    }
    ManageEmployee ME = new ManageEmployee();

    Integer empID1 = ME.addEmployee("Siddharth", "Patel", 50000);
    Integer empID2 = ME.addEmployee("Pranav", "Joshi", 5500);
    Integer empID3 = ME.addEmployee("Neel", "Gondalia", 100000);
    Integer empID4 = ME.addEmployee("Priyanka", "Savani", 45000);
    /* List down all the employees */
    ME.listEmployees();
    /* Update employee's records */
    // ME.updateEmployee(empID1, 5000);
    /* Delete an employee from the database */
    // ME.deleteEmployee(empID2);
    /* List down new list of the employees */
    // ME.listEmployees();
    }
    /* Method to CREATE an employee in the database */
    public Integer addEmployee(String fname, String lname, int salary){
        Session session = factory.openSession();
        Transaction tx = null;
        Integer employeeID = null;
        try {
            tx = session.beginTransaction();
            Employee employee = new Employee();
            employee.setFirstName(fname);
            employee.setLastName(lname);
            employee.setSalary(salary);
            employeeID = (Integer) session.save(employee);
            tx.commit();
        } catch (HibernateException e) {
            if (tx!=null) {
                tx.rollback();
            }
            e.printStackTrace();
        } finally {
            session.close();
        }
        return employeeID;
    }
    /* Method to READ all the employees */

```

```

    public void listEmployees( ){
        Session session = factory.openSession();
        Transaction tx = null;
        try {
tx = session.beginTransaction();
            List employees = session.createQuery("FROM Employee").list();
            for (Iterator iterator = employees.iterator(); iterator.hasNext();){
                Employee employee = (Employee) iterator.next();
System.out.print("First Name: " + employee.getFirstName());
System.out.print(" Last Name: " + employee.getLastName());
System.out.println(" Salary: " + employee.getSalary());
            }
tx.commit();
        }catch (HibernateException e) {
            if (tx!=null) {
tx.rollback();
            }
e.printStackTrace();
        } finally {
session.close();
        }
    }
}

```

hibernet.cfg.xml

```

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-configuration PUBLIC "-//Hibernate/Hibernate
Configuration DTD 3.0//EN" "http://hibernate.sourceforge.net/hibernate-
configuration-3.0.dtd">
<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/hibernet_db?zer
oDateTimeBehavior=convertToNull</property>

```

```
<property name="hibernate.connection.username">root</property>
<property name="hibernate.connection.password"></property>
</session-factory>
</hibernate-configuration>
```

Output:

id	first_name	last_name	salary
10	Priyanka	Savani	45000
9	Neel	Gondalia	100000
8	Pranav	Joshi	5500
7	Siddharth	Patel	50000

Practical 3: Use Hibernate Query Language to insert, update and delete records in database.

Program:

ManageEmployee.java

```
package com;
import java.util.Iterator;
import java.util.List;
import org.hibernate.HibernateException;
import org.hibernate.Query;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;
import org.hibernate.cfg.Configuration;

public class ManageEmployee {
    private static SessionFactory factory;
    public static void main(String[] args) {
        try {
            factory = new Configuration().configure().buildSessionFactory();
        } catch (Throwable ex) {
            System.err.println("Failed to create sessionFactory object." + ex);
            throw new ExceptionInInitializerError(ex);
        }
    }
}
```

```
}
```

```
ManageEmployee ME = new ManageEmployee();
```

```
/* Add few employee records in database */
```

```
Integer empID1 = ME.addEmployee("Siddharth", "Patel", 15000);
```

```
Integer empID2 = ME.addEmployee("Priyanka", "Savani",55000);
```

```
Integer empID3 = ME.addEmployee("ABC", "XYZ",5000);
```

```
System.out.println("Record inserted..!!!");
```

```
/* List down all the employees */
```

```
System.out.println("List of Employees..!!!");
```

```
ME.listEmployees();
```

```
/* Update employee's records */
```

```
ME.updateEmployee(empID1, 5000);
```

```
System.out.println("Record of emp_id1 Updated..!!!");
```

```
/* Delete an employee from the database */
```

```
ME.deleteEmployee(empID3);
```

```
System.out.println("Record of emp_id3 Deleted..!!!");
```

```
/* List down new list of the employees */
```

```
System.out.println("Updated List of Employees..!!!");
```

```
ME.listEmployees();
```

```
}
```

```
/* Method to add an employee in the database */
```

```
public Integer addEmployee(String fname, String lname, int salary){
```

```
    Session session = factory.openSession();
```

```
    Transaction tx = null;
```

```
    Integer employeeID = null;
```

```
    try {
```

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

```
    Employee employee = new Employee(fname, lname, salary);
```

```
employeeID = (Integer) session.save(employee);
```

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

```
    } catch (HibernateException e) {
```

```
        if (tx!=null) tx.rollback();
```

```
e.printStackTrace();
```

```
    } finally {
```

```

session.close();
    }
    return employeeID;
}

/* Method to READ all the employees */
public void listEmployees( ){
    Session session = factory.openSession();
    Transaction tx = null;

    try {
tx = session.beginTransaction();
        String hql="FROM Employee";
        Query q=session.createQuery(hql);
        List employees=q.list();
        //List employees = session.createQuery("FROM Employee").list();
        for (Iterator iterator = employees.iterator(); iterator.hasNext();){
            Employee employee = (Employee) iterator.next();
System.out.print("First Name: " + employee.getFirstName());
System.out.print(" Last Name: " + employee.getLastName());
System.out.println(" Salary: " + employee.getSalary());
        }
tx.commit();
    } catch (HibernateException e) {
        if (tx!=null) tx.rollback();
e.printStackTrace();
    } finally {
session.close();
    }
}

/* Method to UPDATE salary for an employee */
public void updateEmployee(Integer EmployeeID, int salary ){
    Session session = factory.openSession();
    Transaction tx = null;

    try {
tx = session.beginTransaction();
        Employee employee = (Employee)session.get(Employee.class,
EmployeeID);
employee.setSalary( salary );

```

```

session.update(employee);
tx.commit();
    } catch (HibernateException e) {
        if (tx!=null) tx.rollback();
e.printStackTrace();
    } finally {
session.close();
    }
}

/* Method to DELETE an employee from the records */
public void deleteEmployee(Integer EmployeeID){
    Session session = factory.openSession();
    Transaction tx = null;

    try {
tx = session.beginTransaction();
        Employee employee = (Employee)session.get(Employee.class,
EmployeeID);
session.delete(employee);
tx.commit();
    } catch (HibernateException e) {
        if (tx!=null) tx.rollback();
e.printStackTrace();
    } finally {
session.close();
    }
}
}

```

Employee.java

```

package com;
public class Employee {
    private int id;
    private String firstName;
    private String lastName;
    private int salary;

```



```

    public Employee() { }
    public Employee(String fname, String lname, int salary) {
this.firstName = fname;
this.lastName = lname;
this.salary = salary;
    }

    public int getId() {
        return id;
    }

    public void setId( int id ) {
        this.id = id;
    }

    public String getFirstName() {
        return firstName;
    }

    public void setFirstName( Stringfirst_name ) {
this.firstName = first_name;
    }

    public String getLastName() {
        return lastName;
    }

    public void setLastName( Stringlast_name ) {
this.lastName = last_name;
    }

    public int getSalary() {
        return salary;
    }

    public void setSalary( int salary ) {
this.salary = salary;
    }
}

```

hibernet.cfg.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-configuration PUBLIC "-//Hibernate/Hibernate
Configuration DTD 3.0//EN" "http://hibernate.sourceforge.net/hibernate-
configuration-3.0.dtd">
<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/hibernet_db?zero
DateTimeBehavior=convertToNull</property>
<property name="hibernate.connection.username">root</property>
<property name="hibernate.connection.password"></property>
<mapping resource="Employee.hbm.xml"/>
</session-factory>
</hibernate-configuration>
```

Employee.hbm.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping
DTD 3.0//EN" "http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<hibernate-mapping>
<class name="com.Employee" table="temployee">
<meta attribute="class-description">
    This class contains the employee detail.
</meta>
<id name="id" type="int" column="id">
<generator class="native"/>
</id>
<property name="firstName" column="first_name" type="string"/>
<property name="lastName" column="last_name" type="string"/>
<property name="salary" column="salary" type="int"/>
</class>
</hibernate-mapping>
```

Output:

```
Record inserted..!!!  
List of Employees..!!!  
First Name: ABC   Last Name: XYZ   Salary: 5000  
First Name: Priyanka   Last Name: Savani   Salary: 55000  
First Name: Siddharth   Last Name: Patel   Salary: 15000  
Record of emp_id1 Updated..!!!  
Record of emp_id3 Deleted..!!!  
Updated List of Employees..!!!  
First Name: Priyanka   Last Name: Savani   Salary: 55000  
First Name: Siddharth   Last Name: Patel   Salary: 5000
```

id	first_name	last_name	salary
23	Priyanka	Savani	55000
22	Siddharth	Patel	5000

Experiment# 7JSF

Question 1: Use JSF Standard Components and Facelets Tags.

Standard Components:

Index.xhtml:

```
<html xmlns="http://www.w3.org/1999/xhtml"
xmlns:h="http://xmlns.jcp.org/jsf/html"
xmlns:f="http://xmlns.jcp.org/jsf/core">
<h:head><title>User Registration Form</title>
</h:head>
<h:body>
<h:form id="form">
<table>
<tr>
<td><h:outputLabel for="username">User Name</h:outputLabel></td>
<td><h:inputText id="name-id" value="#{user.name}"/></td>
</tr>
<tr>
<td><h:outputLabel for="email">Your Email</h:outputLabel></td>
<td><h:inputText id="email-id" value="#{user.email}"/></td>
</tr>
<tr>
<td><h:outputLabel for="password">Password</h:outputLabel></td>
<td><h:inputSecret id="password-
id" value="#{user.password}"/></td>
</tr>
<tr>
<td><h:outputLabel for="gender">Gender</h:outputLabel></td>
<td><h:selectOneRadio value="#{user.gender}">
<f:selectItem itemValue="Male" itemLabel="Male" />
<f:selectItem itemValue="Female" itemLabel="Female" />
</h:selectOneRadio></td>
</tr>
<tr><td><h:outputLabel for="address">Address</h:outputLabel></td>
<td><h:inputTextarea value="#{user.address}" cols="50" rows="5"/></t
d></tr>
</table>
<h:commandButton value="Submit" action="response.xhtml"></h:commandB
utton>
</h:form>
```

```
</h:body>
</html>
```

User.java:

```
import javax.faces.bean.ManagedBean;
import javax.faces.bean.RequestScoped;
@ManagedBean
@RequestScoped
public class User{
    String name;
    String email;
    String password;
    String gender;
    String address;
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public String getEmail() {
        return email;
    }
    public void setEmail(String email) {
        this.email = email;
    }
    public String getPassword() {
        return password;
    }
    public void setPassword(String password) {
        this.password = password;
    }
    public String getGender() {
        return gender;
    }
    public void setGender(String gender) {
        this.gender = gender;
    }
    public String getAddress() {
        return address;
    }
    public void setAddress(String address) {
        this.address = address;
    }
}
```

response.xhtml:

```
<html xmlns="http://www.w3.org/1999/xhtml"
xmlns:h="http://xmlns.jcp.org/jsf/html"
```

```
xmlns:f="http://xmlns.jcp.org/jsf/core">
<h:head>
<title>User Details</title>
</h:head>
<h:body>
<h2><h:outputText value="Hello #{user.name}"/></h2>
<h4><h:outputText value="You have Registered with us Successfully, Your D
etails are The Following."/></h4>
<table>
<tr>
<td><b>Email: </b></td>
<td><h:outputText value="#{user.email}"/><br/></td>
</tr>
<tr>
<td><b>Password:</b></td>
<td><h:outputText value="#{user.password}"/><br/></td>
</tr>
<tr>
<td><b>Gender:</b></td>
<td><h:outputText value="#{user.gender}"/><br/></td>
</tr>
<tr>
<td><b>Address: </b></td>
<td><h:outputText value="#{user.address}"/></td>
</tr>
</table>
</h:body>
</html>
```

Final Outcome:

User Name

Your Email

Password

Gender ☐ Male ☐ Female

Address

After giving input and clicking on submit button.



Hello Tarannum Bloch

You have Registered with us Successfully, Your Details are The Following.

Email: tarannum.bloch@marwadieducation.edu.in

Password: 123456

Gender: Female

Address:

Code Snippet:

Standard Components:

Code Snippet:

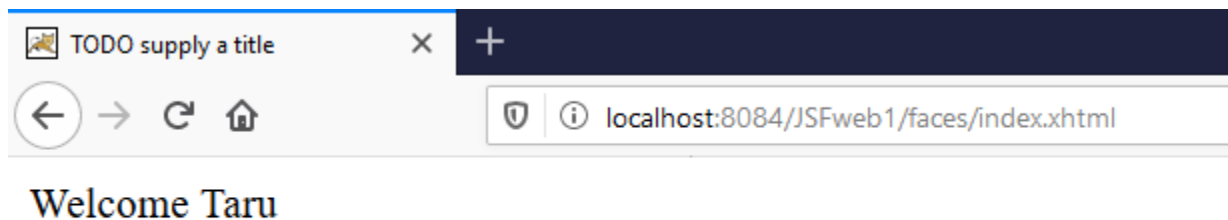
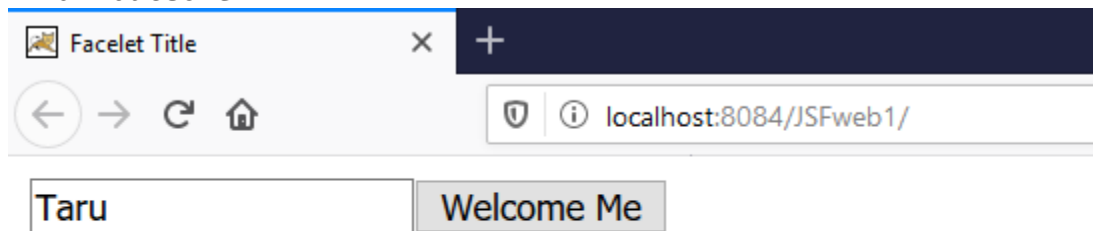
Index.xhtml:

```
<html xmlns="http://www.w3.org/1999/xhtml"
xmlns:h="http://xmlns.jcp.org/jsf/html"
xmlns:f="http://xmlns.jcp.org/jsf/core">
<h:head>
<title>Jsf Form</title>
</h:head>
<h:body>
<h:form id="form">
<h:outputLabel for="username">User Name</h:outputLabel>
<h:inputText id="name" value="#{user.name}" required="true">
<f:validateRequired for="name" />
</h:inputText><br/>
<h:commandButton value="OK" action="response.xhtml"></h:commandButton>
</h:form>
</h:body>
</html>
```

response.xhtml:

```
<html xmlns="http://www.w3.org/1999/xhtml"
xmlns:h="http://xmlns.jcp.org/jsf/html">
<h:head>
<title>Response Page</title>
</h:head>
<h:body>
<h1>
Hello #{user.name}
</h1>
</h:body>
</html>
```

Final Outcome:



Question 2: Implement JSF Converter Tag and Validation Tags

Converter Tag:

Code Snippet:

index.xhtml:

```
<h:form>
<h:outputLabel for="username">User Name</h:outputLabel>
<h:inputText id="user-id" value="#{user.name}"/><br/>
```



```
<h:outputLabel for="shirtPrice">Shirt Price</h:outputLabel>
<h:inputText id="shirtPrice-
id" value="#{user.shirtCost}" autocomplete="off">
</h:inputText>
<br/>
<h:commandButton action="response.xhtml" value="Submit"/>
</h:form>
```

User.java:

```
import javax.faces.bean.ManagedBean;
import javax.faces.bean.RequestScoped;
@ManagedBean
@RequestScoped
public class User {
    String name;
    int shirtPrice;
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public int getShirtCost() {
        return shirtPrice;
    }
    public void setShirtCost(int shirtPrice) {
        this.shirtPrice = shirtPrice;    }    }
```

response.xhtml:

```
<h:body>
<h1> Hello,
<h:outputText value="#{user.name}"/>
</h1>
<h:outputLabel value="Shirt's Price is: "></h:outputLabel>
<h:outputText value="#{user.shirtCost}">
<f:convertNumber pattern="$###" />
</h:outputText>
</h:body>
```

Final Outcome:

User Name

utsav

Shirt Price

99

Submit

Hello, utsav

Shirt's Price is: \$99

Validation Tags:

Snippet:

index.xhtml:

```
<h:form id="form">
<h:outputLabel for="username">User Name</h:outputLabel>
<h:inputText id="name-id" value="#{user.name}"/>
<h:message for="name-id" style="color: red"/>
<br/>
<h:outputLabel for="age">Enter Age</h:outputLabel>
<h:inputText id="age-id" value="#{user.age}"/>
<h:message for="age-id" style="color: red"/>
<br/>
<h:outputLabel for="mobile">Mobile No.</h:outputLabel>
<h:inputText id="mobile-id" value="#{user.mobile}"/>
<h:message for="mobile-id" style="color: red"/>
<br/>
<h:commandButton value="OK" action="response.xhtml"></h:commandButton>
</h:form>
```

User.java:

```
import javax.faces.bean.ManagedBean;
import javax.faces.bean.RequestScoped;
import javax.validation.constraints.NotNull;
import javax.validation.constraints.Min;
import javax.validation.constraints.Size;
@ManagedBean
@RequestScoped
public class User{
    @NotNull(message = "Name can't be empty")
    String name;
    @Min(18)
```

```
int age;
@NotNull(message = "Mobile can't be empty")
@Size(min = 10, max = 10, message = "Mobile must have 10 digits")
String mobile;
public String getName() {
    return name;
}
public void setName(String name) {
    this.name = name;
}
public int getAge() {
    return age;
}
public void setAge(int age) {
    this.age = age; }
public String getMobile() {
    return mobile; }
public void setMobile(String mobile) {
    this.mobile = mobile;
}
}
```

Final Outcome:

User Name	<input type="text"/>	Name can't be empty
Enter Age	<input type="text" value="17"/>	must be greater than or equal to 18
Mobile No.	<input type="text" value="321321654654"/>	Mobile must have 10 digits
<input type="button" value="OK"/>		

Experiment# 6 Spring Framework

Practical 1: Study and Implement MVC using Spring Framework

Program:

index.jsp

```
<% @page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Welcome to Spring Web MVC project</title>
</head>

<body>
<h2>${ message }</h2>
</body>
</html>
```

web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="3.1" xmlns="http://xmlns.jcp.org/xml/ns/javaee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd">

<servlet>
<servlet-name>HelloWeb</servlet-name>
<servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-
class>

</servlet>
<servlet-mapping>
<servlet-name>HelloWeb</servlet-name>
```

```
<url-pattern>*.htm</url-pattern>
</servlet-mapping>

<welcome-file-list>
<welcome-file>redirect.jsp</welcome-file>
</welcome-file-list>
</web-app>
```

HelloController.java

```
package com;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.ui.ModelMap;

@Controller

public class HelloController {
    @RequestMapping(value = "/helloworld.htm", method =
RequestMethod.GET)

    public String printHello(ModelMap model) {
        System.out.println("on method");
        model.addAttribute("message", "Hello Spring MVC Framework!");
        return "index";
    }
}
```

Output:



Practical 2: Inject Service using Aspect Oriented Programming.

Program:

applicationContext.xml

```
import org.springframework.beans.factory.BeanFactory;
import org.springframework.beans.factory.xml.XmlBeanFactory;
import org.springframework.core.io.ClassPathResource;
import org.springframework.core.io.Resource;

public class Test {
    public static void main(String[] args) {
        Resource r=new ClassPathResource("applicationContext.xml");
        BeanFactory factory=new XmlBeanFactory(r);

        A a=(A)factory.getBean("proxy",A.class);
        a.m();
    }
}
```

Test.java

```
import org.springframework.beans.factory.BeanFactory;
import org.springframework.beans.factory.xml.XmlBeanFactory;
import org.springframework.core.io.ClassPathResource;
import org.springframework.core.io.Resource;

public class Test {
    public static void main(String[] args) {
        Resource r=new ClassPathResource("applicationContext.xml");
        BeanFactory factory=new XmlBeanFactory(r);

        A a=(A)factory.getBean("proxy",A.class);
        a.m();
    }
}
```

BeforeAdvisor.java

```
import java.lang.reflect.Method;
import org.springframework.aop.MethodBeforeAdvice;
```

```

public class BeforeAdvisor implements MethodBeforeAdvice{

    @Override
    public void before(Method method, Object[] args, Object target)throws
    Throwable {
        System.out.println("additional concern before actual logic");
    }
}

```

A.java

```

public class A {
    public void m(){
        System.out.println("actual business logic");
    }
}

```

Output:

```

run:
log4j:WARN No appenders could be found for logger (org.springframework.beans.factory.xml.XmlBeanDefinitionReader).
log4j:WARN Please initialize the log4j system properly.
additional concern before actual logic
actual business logic
BUILD SUCCESSFUL (total time: 0 seconds)

```

Practical 3: Using Spring Template manages Database and Transaction.

Program:

index.jsp

```

<% @page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">

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

```

```
<title>Welcome to Spring Web MVC project</title>
</head>
```

```
<body>
<a href="two.htm">click me</a> to fetch data form database..!!
</body>
</html>
```

one.jsp

```
<% @page import="POJO.userinfo"%>
<% @page import="java.util.Iterator"%>
<% @page import="java.util.List"%>
<% @page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>JSP Page</title>
</head>
<body>
<h1>Data Fetched Form Db..!!</h1>
<%
    List n=(List)session.getAttribute("result");
    Iterator it = n.iterator();

    while(it.hasNext())
    {
        userinfo ui=(userinfo)it.next();
        out.println(ui.getName());
        out.println(ui.getPassword());
    }
%>
</body>
</html>
```

web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="3.1" xmlns="http://xmlns.jcp.org/xml/ns/javaee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
    http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd">
```



```

<servlet>
<servlet-name>dispatcher</servlet-name>
<servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-
class>
</servlet>

```

```

<servlet-mapping>
<servlet-name>dispatcher</servlet-name>
<url-pattern>*.htm</url-pattern>
</servlet-mapping>
<welcome-file-list>
<welcome-file>redirect.jsp</welcome-file>
</welcome-file-list>
</web-app>

```

userinfo.java

```

package POJO;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.Id;
import javax.persistence.Table;

```

```

@Entity
@Table(name="userinfo")

```

```

public class userinfo {

    @Id@GeneratedValue
    @Column(name="id")
    Integer id;
    @Column(name="name")
    String name;
    @Column(name="password")
    String password;

    public Integer getId() {
        return id;
    }

    public void setId(Integer id) {
        this.id = id;
    }
}

```

```

    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }
}

```

IndexoneController.java

```

package controller;
import POJO.userinfo;
import java.util.Iterator;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpSession;
import org.hibernate.Query;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.servlet.ModelAndView;

@Controller
public class IndexoneController {
    @RequestMapping(value="/index.htm",method = RequestMethod.GET)
    protected ModelAndView getindex()
    {
        ModelAndView model = new ModelAndView("index");
    }
}

```

```

        return model;
    }

    @RequestMapping(value="/two.htm",method = RequestMethod.GET)
    protected ModelAndView submitindex(HttpServletRequest request)
    {
        ModelAndView model =null;
        model=new ModelAndView("one");
        try {
            Configuration con = new Configuration();
            con.configure("hibernate.cfg.xml");

            SessionFactory sf =
con.addAnnotatedClass(userinfo.class).buildSessionFactory();

            //to perform transaction in db ussing hibernate
            Session session = sf.openSession();

            session.beginTransaction();

            String hql="from userinfo";
            Query query=session.createQuery(hql);
            List result = query.list();

            Iterator it = result.iterator();

            while(it.hasNext())
            {
                userinfo ui=(userinfo)it.next();
                System.out.println(ui.getName());
                System.out.println(ui.getPassword());
            }

            //httpSession to send data

            HttpSession session1=request.getSession();
            session1.setAttribute("result",result);

            session.close();

        }
        catch(Exception e){ }
    }

```

```
        return model;  
    }  
}
```

Output:

