

PRACTICAL 1

WRITE A TCP CLIENT-SERVER PROGRAM TO GET THE DATE & TIME DETAILS FROM SERVER ON THE CLIENT REQUEST.

Client.java

```
import java.io.*;
import java.net.*;
import java.net.Socket;
import java.util.Date;

public class client {
    public static void main(String[] args) {
        //client side scripting
        try {
            Socket s = new Socket("localhost",1234);
            DataOutputStream dto = new DataOutputStream(s.getOutputStream());
            Date dt = new Date(System.currentTimeMillis());
            dto.writeUTF("Time is : "+dt);
            dto.flush();
            dto.close();
            s.close();

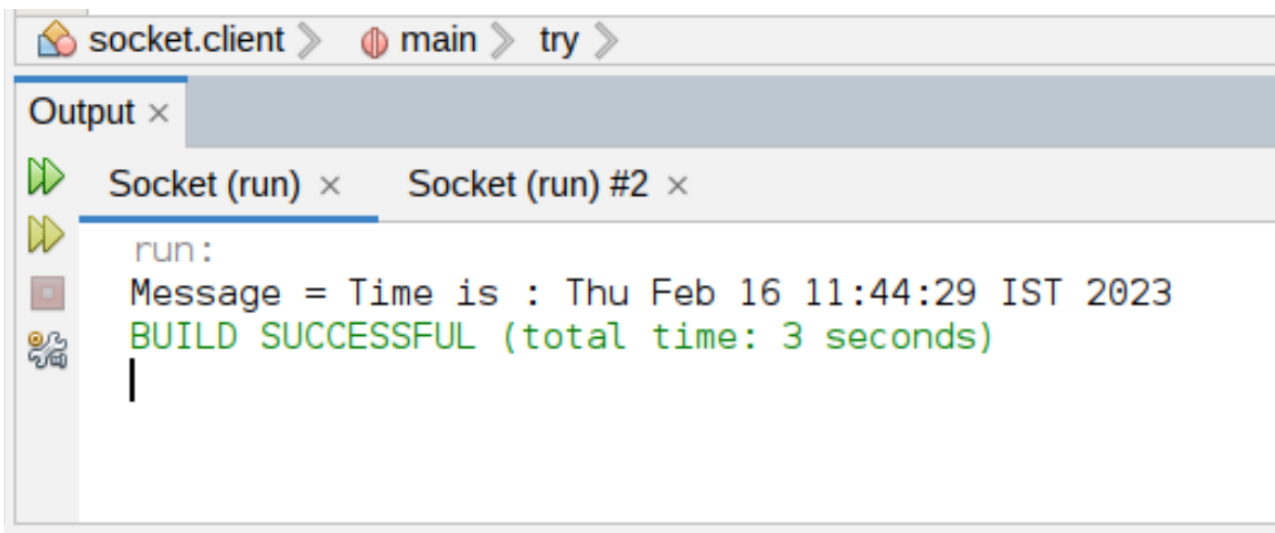
        } catch (Exception e) {
            System.out.println(e);
        }
    }
}
```

Server.java

```
import java.io.*;
import java.net.*;
import java.net.Socket;

public class Server {
    public static void main(String args[]) {
        try {
            ServerSocket ss = new ServerSocket(1234);
            Socket s = ss.accept();
            DataInputStream dis= new DataInputStream(s.getInputStream());
            String str= (String)dis.readUTF();
            System.out.println("Message = "+str);
            ss.close();
        }
        catch(Exception e) {
            System.out.println(e);
        }
    }
}
```

Output



PRACTICAL 2

WRITE A UDP CLIENT-SERVER PROGRAM IN WHICH THE CLIENT SENDS A STRING AND THE SERVER RESPONDS WITH THE REVERSE OF A STRING.

Client.java

```
import java.net.*;

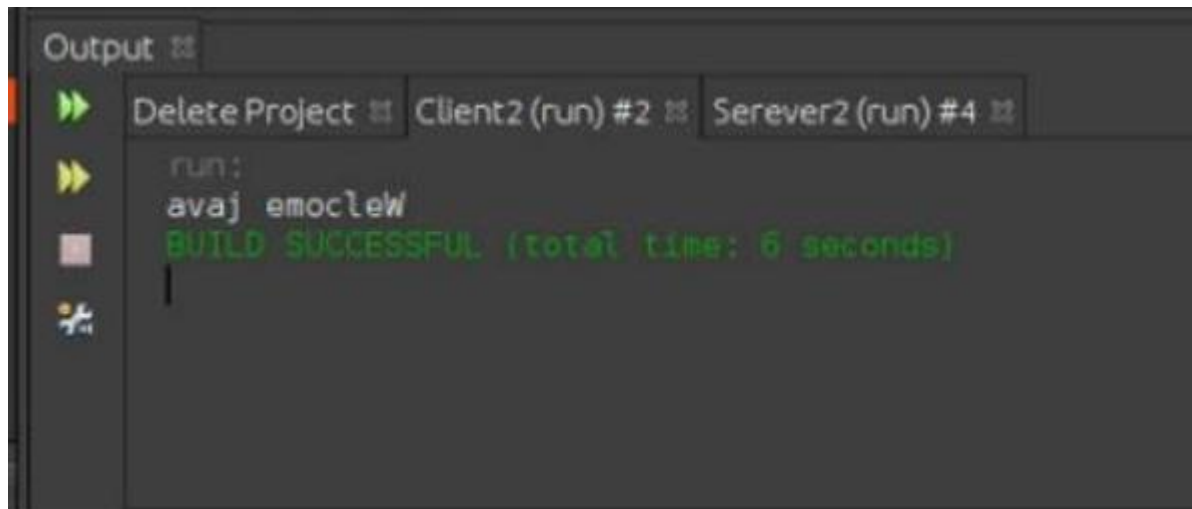
public class Client {
    public static void main(String[] args) throws Exception {
        DatagramSocket ds = new DatagramSocket();
        String str = "Welcome java";
        InetAddress ip = InetAddress.getByName("localhost");
        DatagramPacket dp = new DatagramPacket(str.getBytes(), str.length(), ip, 3000);
        ds.send(dp);
        ds.close();
    }
}
```

Server.java

```
import java.net.*;

public class Server {
    public static void main(String[] args) throws Exception {
        DatagramSocket ds = new DatagramSocket(3000);
        byte[] buf = new byte[1024];
        DatagramPacket dp = new DatagramPacket(buf, 1024);
        ds.receive(dp);
        String str = new String(buf);
        StringBuilder sb = new StringBuilder();
        sb.append(str);
        sb = sb.reverse();
        System.out.println(sb);
        ds.close();
    }
}
```

Output



The screenshot shows an IDE's Output window with a dark theme. At the top, there are three tabs: 'Delete Project', 'Client2 (run) #2', and 'Serever2 (run) #4'. The 'Client2 (run) #2' tab is currently selected. Below the tabs, the output text is as follows:

```
run:
avaj emoclew
BUILD SUCCESSFUL (total time: 6 seconds)
```

On the left side of the output area, there are four icons: a green play button, a yellow play button, a red square, and a blue icon with a white figure.

PRACTICAL 3

WRITE A CLIENT SERVER PROGRAM USING TCP WHERE CLIENT SENDS A STRING AND SERVER CHECKS WHETHER THAT STRING IS PALINDROME OR NOT AND RESPONDS WITH AN APPROPRIATE MESSAGE.

Client.java

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

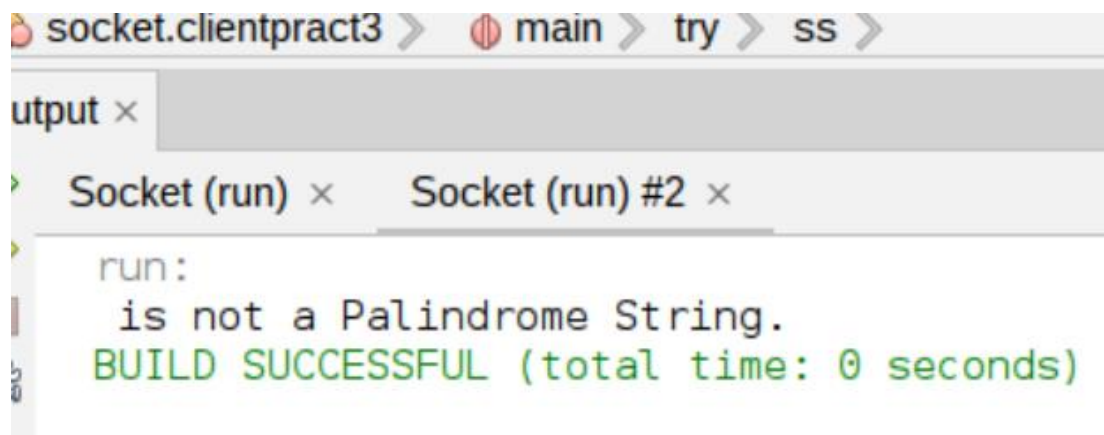
public class Client {
    public static void main(String[] args) {
        try {
            String ss="ababba";
            Socket s = new Socket("localhost", 1234);
            DataOutputStream dout=new DataOutputStream(s.getOutputStream());
            dout.writeUTF(ss);
            DataInputStream din = new DataInputStream(s.getInputStream());
            String str=(String)din.readUTF();
            System.out.println(""+str);
            dout.flush();
            dout.close();
            s.close();
        }
        catch (Exception e) {
            System.out.println(e);
        }
    }
}
```

Server.java

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

public class Server {
    public static void main(String[] args) {
        try {
            ServerSocket sr = new ServerSocket(1234);
            Socket s = sr.accept();//establishing the connection.
            DataInputStream dis=new DataInputStream(s.getInputStream());
            String str= (String)dis.readUTF();
            int strLength = str.length();
            String reverseStr="";
            String strout="";
            for (int i = (strLength - 1); i >=0; --i) {
                reverseStr = reverseStr + str.charAt(i);
            }
            if (str.toLowerCase().equals(reverseStr.toLowerCase())) {
                strout=" is a Palindrome String.";
            }
            else {
                strout = " is not a Palindrome String.";
            }
            DataOutputStream dout = new DataOutputStream(s.getOutputStream());
            dout.writeUTF(strout);
            dout.flush();
            dout.close();
            sr.close();
        }
        catch (Exception e) {
            System.err.println(e);
        }
    }
}
```

Output



PRACTICAL 4

WRITE A CLIENT-SERVER PROGRAM USING UDP SOCKET. CLIENT SENDS A LIST OF N NUMBERS TO SERVER AND SERVER RESPONDS WITH THE SUM OF N NUMBERS.

Client.java

```
import java.net.*;

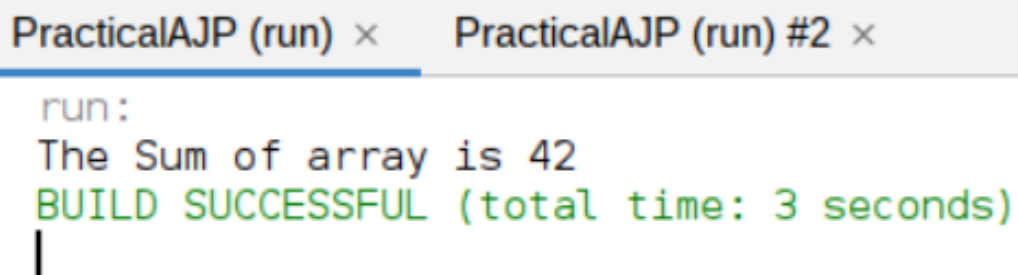
public class pract4udpclient {
    public static void main(String[] args) {
        try {
            DatagramSocket ds = new DatagramSocket();
            byte buffer[] = {10,12,20};
            InetAddress inet = InetAddress.getLocalHost();
            DatagramPacket dp = new DatagramPacket(buffer, buffer.length, inet, 1234);
            ds.send(dp);
            ds.close();
        }
        catch (Exception e) {
            System.out.println(" "+e);
        }
    }
}
```

Server.java

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

public class Server {
    public static void main(String[] args) {
        try {
            byte buf[] = new byte[1024];
            byte sum=0;
            DatagramSocket ds = new DatagramSocket(1234);
            DatagramPacket dp = new DatagramPacket(buf, 1024);
            ds.receive(dp);
            for (int i = 0; i < buf.length; i++) {
                sum+=buf[i];
            }
            System.out.println("The Sum of array is " + sum);
            ds.close();
        }
        catch (Exception e) {
            System.out.println(e);
        }
    }
}
```

Output



```
PracticalAJP (run) × PracticalAJP (run) #2 ×
run:
The Sum of array is 42
BUILD SUCCESSFUL (total time: 3 seconds)
|
```


PRACTICAL 5

WRITE A CLIENT-SERVER PROGRAM USING UDP SOCKET. CLIENT SEND THE LIST OF N STRINGS AND SERVER RESPONDS TO THE CONCATENATION OF THOSE STRINGS.

Client.java

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

public class Client {
    public static void main(String[] args) throws Exception {
        DatagramSocket ds = new DatagramSocket();
        InetAddress ip = InetAddress.getByName("localhost");
        DatagramPacket dp;
        String str="";
        while(!str.equals("exit")) {
            System.out.println("Enter Msg:"); Scanner sc=new Scanner(System.in);
            str=sc.nextLine();
            dp = new DatagramPacket(str.getBytes(), str.length(), ip, 3275);
            ds.send(dp);
        }
    }
}
```

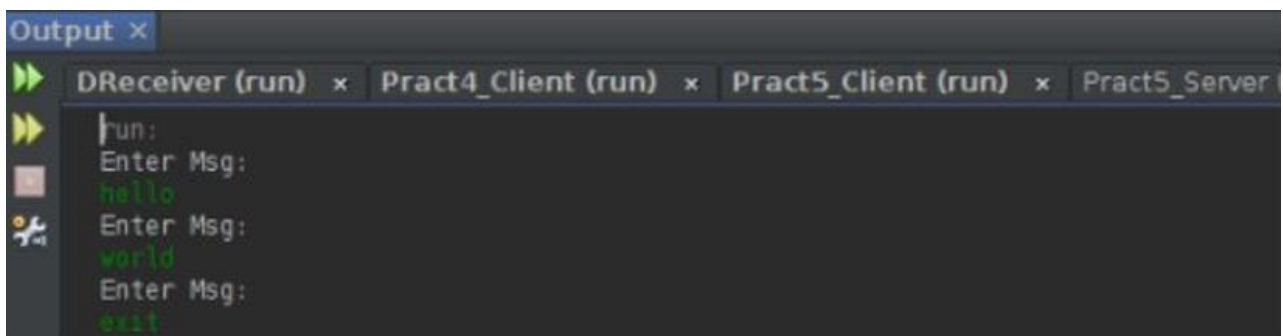
Server.java

```
import java.net.*;

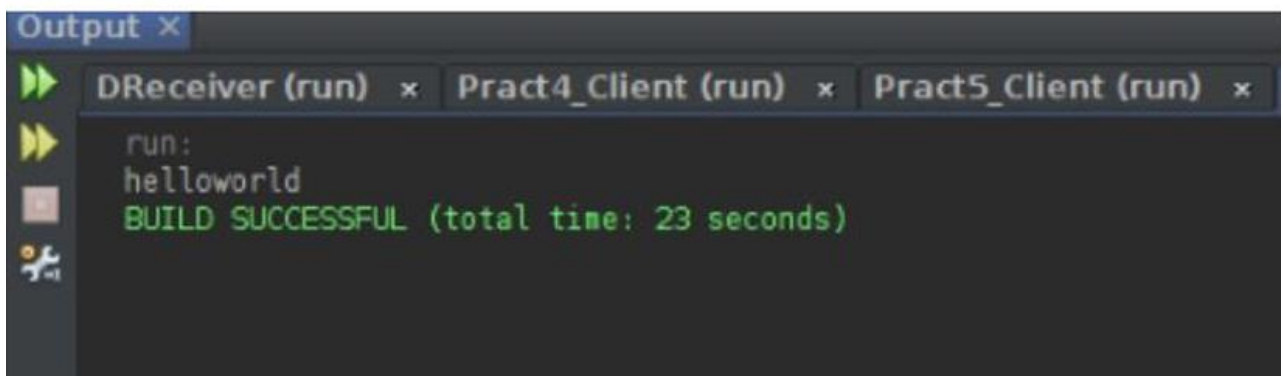
public class Server {

    public static void main(String[] args) throws Exception {
        String str = "", concat = "";
        DatagramSocket ds = new DatagramSocket(3275);
        while(true) {
            byte[] buf= new byte[1024];
            DatagramPacket dp= new DatagramPacket(buf, 1024);
            ds.receive(dp);
            str=new String(dp.getData(),0,dp.getLength());
            if (!str.equals("exit")) {
                concat+=str;
            }
            else {
                break;
            }
        }
        System.out.println(concat);
        ds.close();
    }
}
```

Output



```
Output x
run:
Enter Msg:
hello
Enter Msg:
world
Enter Msg:
exit
```



```
Output x
run:
helloworld
BUILD SUCCESSFUL (total time: 23 seconds)
```

PRACTICAL 7

USER CAN CREATE A NEW DATABASE AND ALSO CREATE NEW TABLES UNDER THAT DATABASE. ONCE DATABASE HAS BEEN CREATED THEN THE USER CAN PERFORM DATABASE CRUD OPERATIONS BY CALLING ABOVE FUNCTIONS. USE FOLLOWING TO IMPLEMENT PROGRAM: A) STATEMENT B) PREPARED STATEMENT C) CALLABLE STATEMENT

A) Statement

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

public class statement {
    public static void main(String[] args) {
        String url = "jdbc:mysql://localhost:3306/jdbc";
        String username = "root";
        String password = "";
        try (Scanner scanner = new Scanner(System.in)) {
            Class.forName("com.mysql.cj.jdbc.Driver");
            Connection connection = DriverManager.getConnection(url, username, password);
            System.out.println("Connection established.....");

            System.out.println("1. Create");
            System.out.println("2. Read");
            System.out.println("3. Update");
            System.out.println("4. Delete");
            System.out.print("Enter case : ");
            int i = scanner.nextInt();

            Statement statement = connection.createStatement();
            switch (i) {
                case 1: // Create
                    String query1 = "INSERT INTO employee(" + "ID,Name,Department,Salaries) VALUES "
                        + "('3','Vatsal','Computer','35000')";
                    statement.executeUpdate(query1);
                    System.out.println("successfully inserted");
                    ResultSet resultSet1 = statement.executeQuery("SELECT * FROM employee");
                    while (resultSet1.next()) {
                        System.out.println(
                            resultSet1.getInt(1) + " " + resultSet1.getString(2) + " " + resultSet1.getString(3)
                                + " " + resultSet1.getInt(4));
                    }
                    break;
```

```

case 2: // Read
    ResultSet resultSet = statement.executeQuery("SELECT * FROM employee");
    while (resultSet.next()) {
        System.out.println(
            resultSet.getInt(1) + " " + resultSet.getString(2) + " " + resultSet.getString(3)
            + " " + resultSet.getInt(4));
    }
    break;

case 3: // Update
    String query2 = "UPDATE employee SET Name='Ranchal' WHERE id='1'";
    statement.executeUpdate(query2);
    System.out.println("Record has been updated in the table successfully.....");
    ResultSet resultSet2 = statement.executeQuery("SELECT * FROM employee");
    while (resultSet2.next()) {
        System.out.println(
            resultSet2.getInt(1) + " " + resultSet2.getString(2) + " " + resultSet2.getString(3)
            + " " + resultSet2.getInt(4));
    }
    break;

case 4: // Delete
    String query3 = "DELETE FROM employee WHERE id='3'";
    statement.executeUpdate(query3);
    System.out.println("Record has been updated in the table successfully.....");
    ResultSet resultSet3 = statement.executeQuery("SELECT * FROM employee");
    while (resultSet3.next()) {
        System.out.println(
            resultSet3.getInt(1) + " " + resultSet3.getString(2) + " " + resultSet3.getString(3)
            + " " + resultSet3.getInt(4));
    }
    break;

default:
    System.out.println("Invalid input");
    break;
}
connection.close();
} catch (Exception e) {
    System.out.println(e);
}
}
}

```

Output

```
mysql> select * from employee;
+----+-----+-----+-----+
| ID | Name   | Department | Salaries |
+----+-----+-----+-----+
| 1  | Ranchal | Computer   | 20000    |
| 2  | Jalaram | Civil      | 40000    |
| 3  | Vatsal  | Computer   | 35000    |
+----+-----+-----+-----+
3 rows in set (0.00 sec)
```

B) Prepared statement

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

public class Prepared_Stmtement {
    public static void main(String[] args) {
        try {
            boolean flag = true;
            Scanner sc = new Scanner(System.in);
            String host = "jdbc:mysql://localhost:3306/jdbc";
            String username = "root";
            String password = "";
            // Create Statement
            try (// Create Connection
                Connection con = DriverManager.getConnection(host, username, password)) {
                // Create Statement
                PreparedStatement st;
                // Execute Query
                while (flag) {
                    System.out.println("To Insert Press 1 \nTo Print Press 2 \nTo Update Press 3 \nTo Delete Press
                        4 \nTo Exit press 5");
                    int choice = sc.nextInt();
                    switch (choice) {
                        case 1: { // insert
                            sc.nextLine();
                            System.out.println("Enter the employee ID:");
                            String e_ID = sc.nextLine();
                            int ID = Integer.parseInt(e_ID);
                            System.out.println("Enter the employee name:");
                            String Name = sc.nextLine();
                            System.out.println("Enter the employee's department:");
                            String Department = sc.nextLine();
                            System.out.println("Enter the employee's Salary:");
                            String salary = sc.nextLine();
                            int Salaries = Integer.parseInt(salary);

                            st = con.prepareStatement("insert into employee values(?,?,?,?)");
                            st.setInt(1, ID);
                            st.setString(2, Name);
                            st.setString(3, Department);
                            st.setInt(4, Salaries);
                            // Execute Query
                            st.execute();
```

```

case 2: {      // Print
    String getingdata = "select * from employee";

    st = con.prepareStatement(getingdata);
    ResultSet resultset = st.executeQuery();
    while (resultset.next()) {

        System.out.println("emp_id: " + resultset.getInt(1));
        System.out.println("emp_name: " + resultset.getString(2));
        System.out.println("emp_dept: " + resultset.getString(3));
        System.out.println("emp_salary: " + resultset.getInt(4));
        System.out.println("\n");
    }
    break;
}

case 3: {      // update
    System.out.println("Enter the employee ID whose details are to be modified");
    sc.nextLine();
    String eid = sc.nextLine();
    int ID = Integer.parseInt(eid);
    System.out.println("Enter the employee's name:");
    String Name = sc.nextLine();
    System.out.println("Enter the employee's dept:");
    String Department = sc.nextLine();
    System.out.println("Enter the employee's salary:");
    String sal = sc.nextLine();
    int Salaries = Integer.parseInt(sal);
    st = con.prepareStatement("update employee set Name=?, Department=?, Salaries=?
        where eid = ?");
    st.setString(1, Name);
    st.setString(2, Department);
    st.setInt(3, Salaries);
    st.setInt(4, ID);
    st.execute();
    System.out.println("\n Updated \n");
    break;
}

case 4: {      // delete
    System.out.println("Enter the emp_id to be deleted from emp table");
    sc.nextLine();
    String eid = sc.nextLine();
    int ID = Integer.parseInt(eid);
    st = con.prepareStatement("delete from employee where ID =?");
    st.setInt(1, ID);

```

```

        case 5:// exit
            flag = false;
            break;
        default:
            System.out.println("WRONG TURN");
            break;
    }
}
// close connection
}
} catch (Exception e) {
    System.out.println(e);
}
}
}

```

Output

```

PS D:\College Study Material\6th SEM\AJP\Program> & 'C:\Program Files\Java\jdk-18.0.2\bin\java.exe' '@C:\Users\VATSAL\AppData\Local\Temp\cp_262d2a0iy2fk4ezxx5q34be8m.argfile' 'Practical_6.Prepared_Stmt'
To Insert Press 1
To Print Press 2
To Update Press 3
To Delete Press 4
To Exit press 5
1
Enter the employee ID:
6
Enter the employee name:
parth
Enter the employee's department:
Civil
Enter the employee's Salary:
30000

Inserted

To Insert Press 1
To Print Press 2
To Update Press 3
To Delete Press 4
To Exit press 5

```

```

mysql> select * from employee;
+----+-----+-----+-----+
| ID | Name   | Department | Salaries |
+----+-----+-----+-----+
| 1  | Ranchal | Computer  | 20000   |
| 2  | Jalaram | Civil     | 40000   |
| 3  | Vatsal  | Computer  | 35000   |
| 4  | jash    | farmacy   | 20000   |
| 5  | Dev     | Computer  | 50000   |
| 6  | parth   | Civil     | 30000   |
+----+-----+-----+-----+
6 rows in set (0.00 sec)

```


C) Callable statement

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

public class Callable_Stmtement {
    static final String DB_URL = "jdbc:mysql://localhost/jdbc";
    static final String USER = "root";
    static final String PASS = "";

    public static void main(String[] args) {
        Connection conn = null;
        CallableStatement stmt = null;

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

            System.out.println("Connecting to database...");
            conn = DriverManager.getConnection(DB_URL, USER, PASS);

            System.out.println("Choose Operation: ");
            System.out.println("1. Create Employee");
            System.out.println("2. Update Employee");
            System.out.println("3. Delete Employee");
            System.out.println("4. Get Employee Details");
            Scanner sc = new Scanner(System.in);
            int choice = sc.nextInt();
            switch (choice) {
                case 1:
                    System.out.println("Enter Employee ID: ");
                    int empId = sc.nextInt();
                    System.out.println("Enter Employee Name: ");
                    String empName = sc.next();
                    System.out.println("Enter Employee Department: ");
                    String empDept = sc.next();
                    System.out.println("Enter Employee Salary: ");
                    int empSalary = sc.nextInt();

                    stmt = conn.prepareCall("{call createEmployee(?,?,?,?)}");
                    stmt.setInt(1, empId);
                    stmt.setString(2, empName);
                    stmt.setString(3, empDept);
                    stmt.setInt(4, empSalary);
                    stmt.executeUpdate();
```

```

case 2:
    System.out.println("Enter Employee ID: ");
    int empldToUpdate = sc.nextInt();
    System.out.println("Enter Employee Salary: ");
    int empSalaryToUpdate = sc.nextInt();
    stmt = conn.prepareStatement("call updateEmployeeSalary(?,?)");
    stmt.setInt(1, empldToUpdate);
    stmt.setInt(2, empSalaryToUpdate);
    stmt.executeUpdate();
    System.out.println("Employee salary updated successfully!");
    break;
case 3:
    System.out.println("Enter Employee ID: ");
    int empldToDelete = sc.nextInt();
    stmt = conn.prepareStatement("call deleteEmployee(?)");
    stmt.setInt(1, empldToDelete);
    stmt.executeUpdate();
    System.out.println("Employee deleted successfully!");
    break;
case 4:
    System.out.println("Enter Employee ID: ");
    int empldToGet = sc.nextInt();
    stmt = conn.prepareStatement("call getEmployeeDetails(?)");
    stmt.setInt(1, empldToGet);
    ResultSet rs = stmt.executeQuery();
    while (rs.next()) {
        System.out.println("Employee ID: " + rs.getInt("emp_id"));
        System.out.println("Employee Name: " + rs.getString("emp_name"));
        System.out.println("Employee Department: " + rs.getString("emp_dept"));
        System.out.println("Employee Salary: " + rs.getInt("emp_salary"));
    }
    break;
default:
    System.out.println("Invalid choice!");
}
} catch (SQLException se) {
    se.printStackTrace();
} catch (Exception e) {
    e.printStackTrace();
} finally {
    try {
        if (stmt != null) stmt.close();
    } catch (SQLException se2) {
    }
}

```

```

        se.printStackTrace();
    }
}
}
}

```

PRACTICAL 8

CREATE SERVLET FILE AND STUDY WEB DESCRIPTOR FILE.

Java web applications use a deployment descriptor file to determine how URLs map to servlets, which URLs require authentication, and other information. This file is named web.xml, and resides in the app's WAR under the WEB-INF/ directory. web.xml is part of the servlet standard for web applications.

A web application's deployment descriptor describes the classes, resources and configuration of the application and how the web server uses them to serve web requests. When the web server receives a request for the application, it uses the deployment descriptor to map the URL of the request to the code that ought to handle the request.

The deployment descriptor is a file named web.xml. It resides in the app's WAR under the WEB-INF/ directory. The file is an XML file whose root element is <web-app>.

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app " xmlns="http://xmlns.jcp.org/xml/ns/javaee">
<servlet>
<servlet-name>WebServlet</servlet-name>
<servlet-class>WebServlet</servlet-class>
</servlet>
<welcome-file-list>
<welcome-file>MainPage.html</welcome-file>
<welcome-file>index.html</welcome-file>
</welcome-file-list>
<servlet-mapping>
<servlet-name>WebServlet</servlet-name>
<url-pattern>/WebServlet</url-pattern>
</servlet-mapping>
. . .
```

web.xml

Here, above example show from where the server url request redirect. here the Servlet class and name is.WebServlet and and <url-pattern> tag map the /Servlet URL in Servlet and fetch result about URL.

<url-pattern> parent tag is <servlet-mapping> and always start with <webapp> and end with</web-app>.

PRACTICAL 9

CREATE A SERVLET THAT PRINTS TODAY'S DATE.

DateServlet.java

```
import java.io.IOException;
import java.io.PrintWriter;
import java.time.LocalDate;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

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

        // Set the content type to plain text
        response.setContentType("text/plain");

        // Get the current date
        LocalDate today = LocalDate.now();
        // Write the date to the response output stream
        PrintWriter out = response.getWriter();
        out.println("Today's date is " + today.toString());
    }
}
```

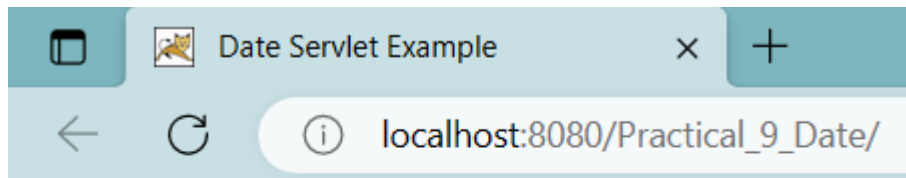
Index.html

```
<!DOCTYPE html>
<html>
<head>
    <title>Date Servlet Example</title>
</head>
<body>
    <h1>Today's date is:</h1>
    <p id="date"></p>
    <script>
        // Make a GET request to the servlet URL and display the response in the 'date' paragraph
        fetch('date')
            .then(response => response.text())
            .then(text => document.getElementById('date').textContent = text);
    </script>
</body>
```

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_4_0.xsd"
  version="4.0">
  <display-name>Date Servlet Example</display-name>
  <servlet>
    <servlet-name>DateServlet</servlet-name>
    <servlet-class>DateServlet</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>DateServlet</servlet-name>
    <url-pattern>/date</url-pattern>
  </servlet-mapping>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
  </welcome-file-list>
```

Output



Today's date is:

Today's date is 2023-05-13

PRACTICAL 10

CREATE A SERVLET THAT DISPLAYS SOME HEADER INFORMATION FROM YOUR REQUEST AS WELL AS ANY FORM DATA.

HeaderAndFormDataServlet.java

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

public class HeaderAndFormDataServlet extends HttpServlet {

    @Override
    public void doGet(HttpServletRequest request, HttpServletResponse response) throws
        ServletException, IOException {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        out.println("<html>");
        out.println("<head><title>Header and Form Data Servlet</title></head>");
        out.println("<body>");
        out.println("<h1>Header Information:</h1>");

        Enumeration<String> headerNames = request.getHeaderNames();
        while (headerNames.hasMoreElements()) {
            String headerName = headerNames.nextElement();
            String headerValue = request.getHeader(headerName);
            out.println("<p>" + headerName + ": " + headerValue + "</p>");
        }
        out.println("<h1>Form Data:</h1>");
        Enumeration<String> parameterNames = request.getParameterNames();
        while (parameterNames.hasMoreElements()) {
            String paramName = parameterNames.nextElement();
            String paramValue = request.getParameter(paramName);
            out.println("<p>" + paramName + ": " + paramValue + "</p>");
        }
        out.println("</body></html>");
    }

    @Override
    public void doPost(HttpServletRequest request, HttpServletResponse response) throws
        ServletException, IOException {
        doGet(request, response);
    }
}
```

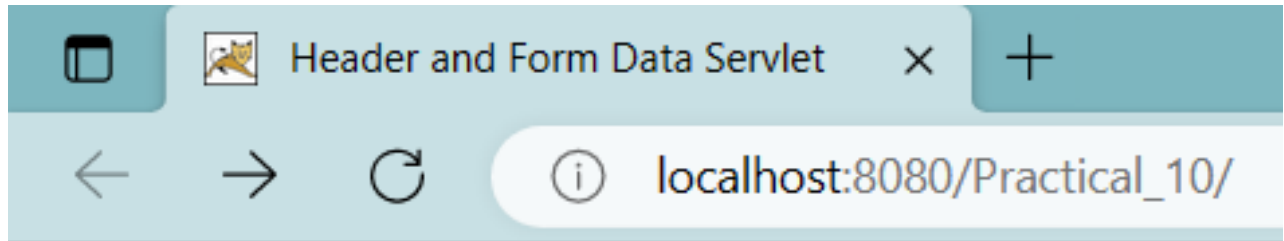
Index.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>Header and Form Data Servlet</title>
  </head>
  <body>
    <h1>Header and Form Data Servlet</h1>
    <form action="/HeaderAndFormDataServiceServlet" method="post">
      <label for="name">Name:</label>
      <input type="text" id="name" name="name"><br>
      <label for="email">Email:</label>
      <input type="email" id="email" name="email"><br>
      <input type="submit" value="Submit">
    </form>
  </body>
```

Web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app>
  <display-name>Header and Form Data Servlet</display-name>
  <servlet>
    <servlet-name>HeaderAndFormDataServiceServlet</servlet-name>
    <servlet-class>HeaderAndFormDataServiceServlet</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>HeaderAndFormDataServiceServlet</servlet-name>
    <url-pattern>/HeaderAndFormDataServiceServlet</url-pattern>
  </servlet-mapping>
```

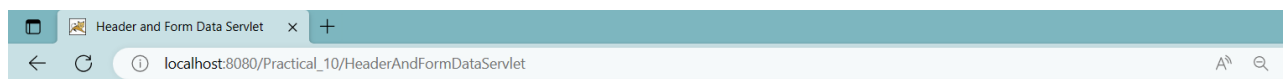
Output



Header and Form Data Servlet

Name:

Email:

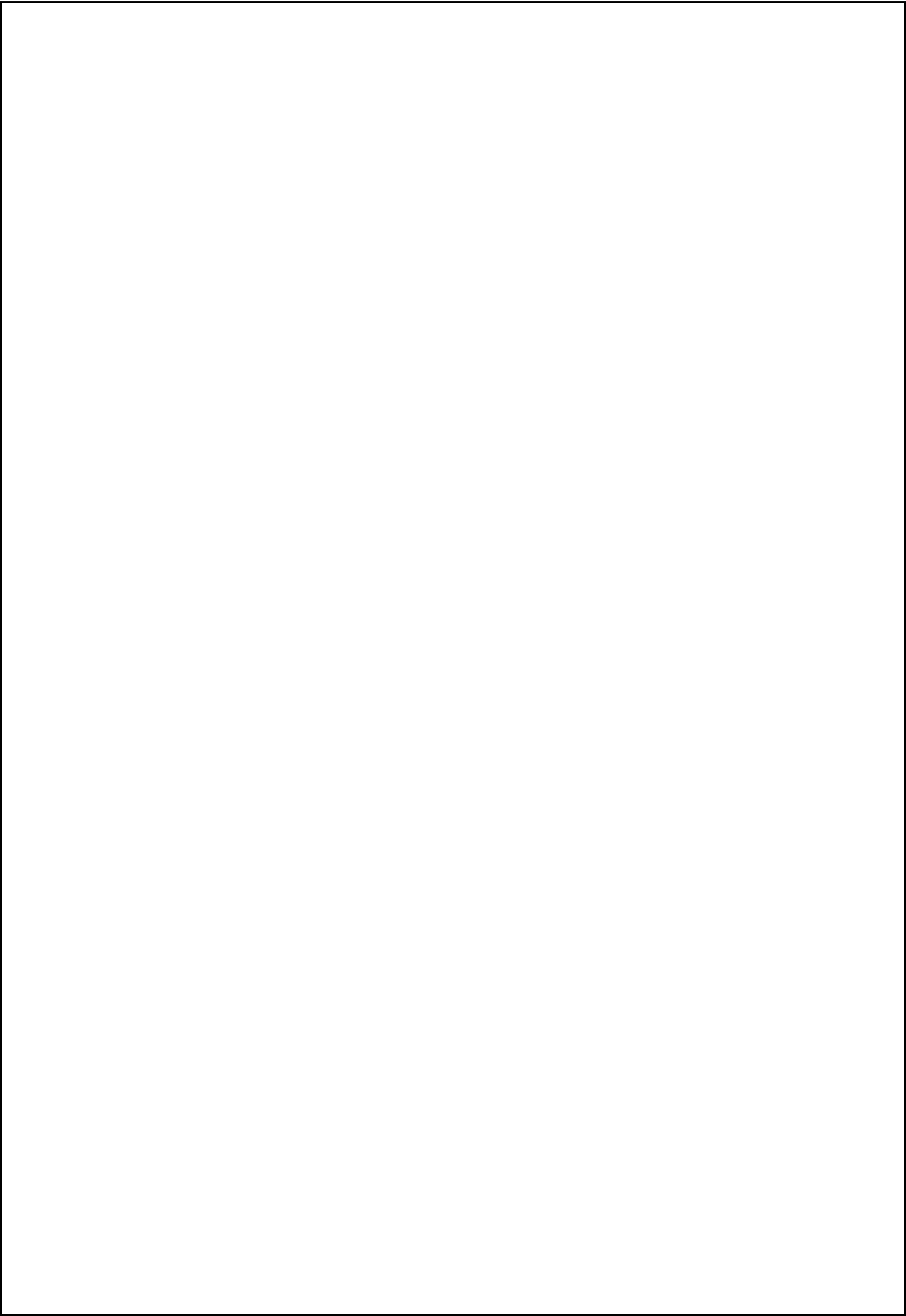


Header Information:

host: localhost:8080
connection: keep-alive
content-length: 34
cache-control: max-age=0
sec-ch-ua: "Microsoft Edge";v="113", "Chromium";v="113", "Not-A.Brand";v="24"
sec-ch-ua-mobile: ?0
sec-ch-ua-platform: "Windows"
upgrade-insecure-requests: 1
origin: http://localhost:8080
content-type: application/x-www-form-urlencoded
user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/113.0.0.0 Safari/537.36 Edg/113.0.1774.35
accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
sec-fetch-site: same-origin
sec-fetch-mode: navigate
sec-fetch-user: ?1
sec-fetch-dest: document
referrer: http://localhost:8080/Practical_10/
accept-encoding: gzip, deflate, br
accept-language: en-US,en;q=0.9
cookie: username=localhost-8889="2|1:0|10:1683138862|23:username=localhost-8889|44:YThlMGYzMjkyODMxNGE4OWE2ZjJlYjg2NGE4MmE4ZjE=|84d5ea476b8fc8ab60b4f3064cd1048eb6ee7bfc6e0dd464139d6b0baa88bc2e";_xsrf=2|f3386a5|a6d018e4db081dc28b24ff3d7787629cd|1683210671; username=localhost-8888="2|1:0|10:1683863325|23:username=localhost-8888|44:Y2VmOGVhY2VjMzE2NGNmZjklNGVhZjY5NjU0OTUyZDM=|66e45c3461b7f133ef02e4cb49edd7737c2465530a3584f2ca4cb02d6d89188f"

Form Data:

name: admin
email: admin@gmail.com



PRACTICAL 12

CREATE LOGIN FORM AND PERFORM STATE MANAGEMENT USING COOKIES, HTTPSESSION AND URL REWRITING.

1) Cookies

LoginServlet.java

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

@WebServlet("/LoginServlet")
public class LoginServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;
    private final String userID = "admin";
    private final String password = "password";
    @Override
    protected void doPost(HttpServletRequest request,
        HttpServletResponse response) throws ServletException, IOException {
        // get request parameters for userID and password
        String user = request.getParameter("user");
        String pwd = request.getParameter("pwd");
        if(userID.equals(user) && password.equals(pwd)){
            Cookie loginCookie = new Cookie("user",user);
            //setting cookie to expiry in 30 mins
            loginCookie.setMaxAge(30*60);
            response.addCookie(loginCookie);
            response.sendRedirect("LoginSuccess.jsp");
        }else{
            RequestDispatcher rd =
                getServletContext().getRequestDispatcher("/login.html");
            PrintWriter out= response.getWriter();
            out.println("<font color=red>Either user name or password is wrong.</font>");
            rd.include(request, response);
        }
    }
}
```

LogoutServlet.java

```
import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.Cookie;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
@WebServlet("/LogoutServlet")
public class LogoutServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
        ServletException, IOException {
        response.setContentType("text/html");
        Cookie loginCookie = null;
        Cookie[] cookies = request.getCookies();
        if(cookies != null){
            for(Cookie cookie : cookies){
                if(cookie.getName().equals("user")){
                    loginCookie = cookie;
                    break;
                }
            }
        }
        if(loginCookie != null) {
            loginCookie.setMaxAge(0);
            response.addCookie(loginCookie);
        }
        response.sendRedirect("login.html");
    }
}
```

Web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="http://java.sun.com/xml/ns/javaee"
    xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-
        app_3_0.xsd" id="WebApp_ID" version="3.0">
    <display-name>ServletCookieExample</display-name>
    <welcome-file-list>
        <welcome-file>login.html</welcome-file>
    </welcome-file-list>
</web-app>
```

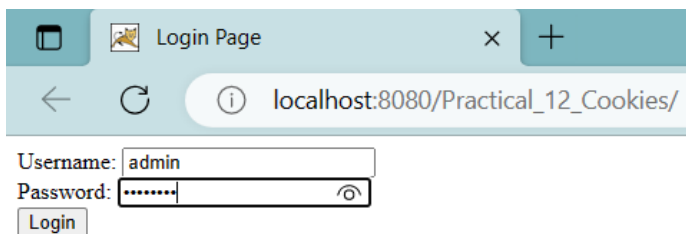
login.html

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="US-ASCII">
        <title>Login Page</title>
    </head>
    <body>
        <form action="LoginServlet" method="post">
            Username: <input type="text" name="user">
            <br>
            Password: <input type="password" name="pwd">
            <br>
            <input type="submit" value="Login">
        </form>
    </body>
```

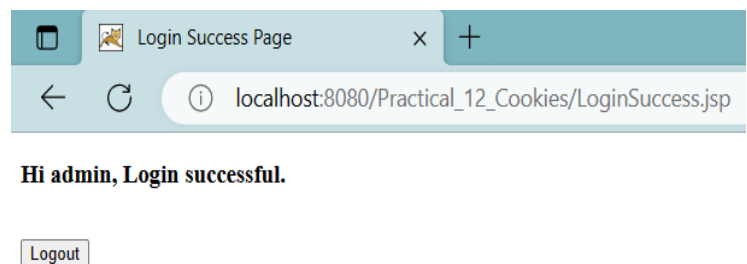
LoginSuccess.jsp

```
<%@ page language="java" contentType="text/html; charset=US-ASCII"
    pageEncoding="US-ASCII"%>
<!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=US-ASCII">
    <title>Login Success Page</title>
  </head>
  <body>
    <%
      String userName = null;
      Cookie[] cookies = request.getCookies();
      if (cookies != null) {
        for (Cookie cookie : cookies) {
          if (cookie.getName().equals("user")) {
            userName = cookie.getValue();
          }
        }
      }
      if (userName == null)
        response.sendRedirect("login.html");
    %>
    <h3>Hi <%=userName%>, Login successful.</h3>
    <br>
    <form action="LogoutServlet" method="post">
      <input type="submit" value="Logout" >
    </form>
```

Output



A screenshot of a web browser window titled "Login Page". The address bar shows "localhost:8080/Practical_12_Cookies/". The page contains a login form with two input fields: "Username:" with the value "admin" and "Password:" with masked characters "*****". Below the password field is a "Login" button.



A screenshot of a web browser window titled "Login Success Page". The address bar shows "localhost:8080/Practical_12_Cookies/LoginSuccess.jsp". The page displays the message "Hi admin, Login successful." in bold. Below the message is a "Logout" button.

2) HttpSession and 3) URL Rewriting

LoginServlet.java

```
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
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;
@WebServlet("/LoginServlet")
public class LoginServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;
    private final String userID = "admin";
    private final String password = "password";

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

        // get request parameters for userID and password
        String user = request.getParameter("user");
        String pwd = request.getParameter("pwd");

        if(userID.equals(user) && password.equals(pwd)){
            HttpSession session = request.getSession();
            session.setAttribute("user", "Vatsal");
            //setting session to expiry in 30 mins
            session.setMaxInactiveInterval(30*60);
            Cookie userName = new Cookie("user", user);
            userName.setMaxAge(30*60);
            response.addCookie(userName);
            response.sendRedirect("LoginSuccess.jsp");
        }else{
            RequestDispatcher rd =
                getServletContext().getRequestDispatcher("/login.html");
            PrintWriter out= response.getWriter();
            out.println("<font color=red>Either user name or password is wrong.</font>");
            rd.include(request, response);
        }
    }
}
```

LogoutServlet.java

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

@WebServlet("/LogoutServlet")
public class LogoutServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
        ServletException, IOException {
        response.setContentType("text/html");
        Cookie[] cookies = request.getCookies();
        if (cookies != null) {
            for (Cookie cookie : cookies) {
                if (cookie.getName().equals("JSESSIONID")) {
                    System.out.println("JSESSIONID=" + cookie.getValue());
                    break;
                }
            }
        }
        //invalidate the session if exists
        HttpSession session = request.getSession(false);
        System.out.println("User=" + session.getAttribute("user"));
        if (session != null) {
            session.invalidate();
        }
        response.sendRedirect("login.html");
    }
}
```

Web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app
  xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
  xmlns="http://java.sun.com/xml/ns/javaee"
  xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-
    app_3_0.xsd" id="WebApp_ID" version="3.0">
  <display-name>ServletHttpSessionExample</display-name>
  <welcome-file-list>
    <welcome-file>login.html</welcome-file>
  </welcome-file-list>
  .
  .
```

login.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>Login Page</title>
  </head>
  <body>
    <form action="LoginServlet" method="post">
      Username: <input type="text" name="user">
      <br>
      Password: <input type="password" name="pwd">
      <br>
      <input type="submit" value="Login">
    </form>
  </body>
```

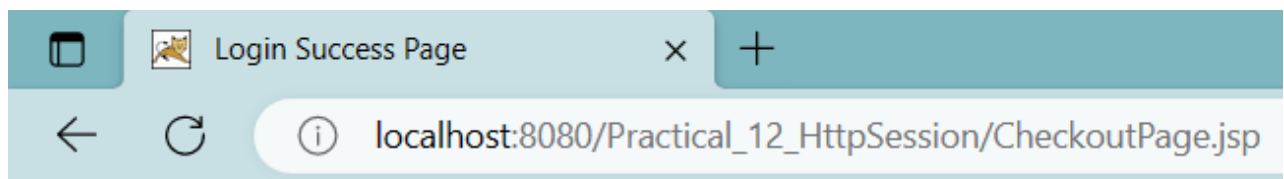
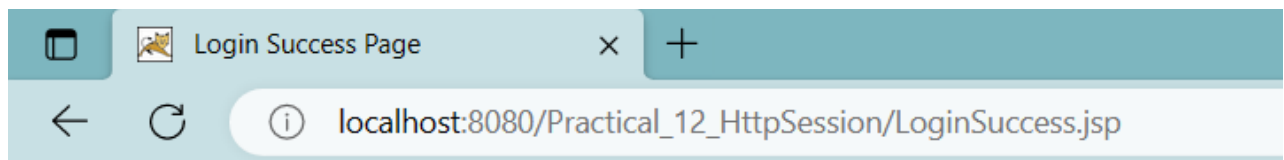
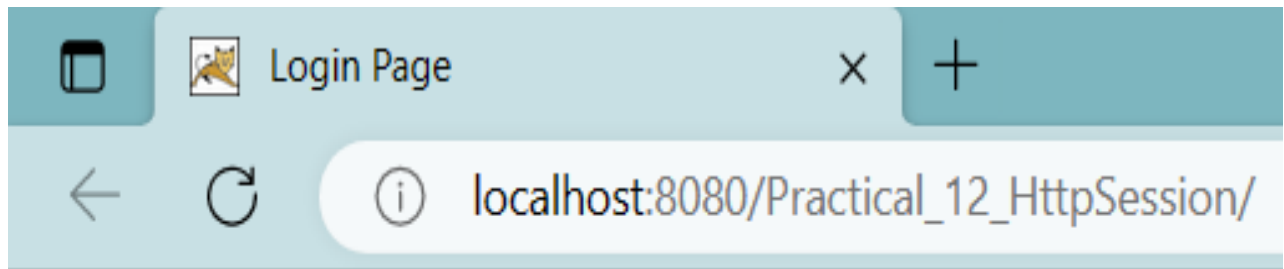

LoginSuccess.jsp

```
<%@ page language="java" contentType="text/html; charset=US-ASCII"
    pageEncoding="US-ASCII"%>
<!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=US-ASCII">
        <title>Login Success Page</title>
    </head>
    <body>
        <%
            //allow access only if session exists
            String user = null;
            if (session.getAttribute("user") == null) {
                response.sendRedirect("login.html");
            } else {
                user = (String) session.getAttribute("user");
            }
            String userName = null;
            String sessionID = null;
            Cookie[] cookies = request.getCookies();
            if (cookies != null) {
                for (Cookie cookie : cookies) {
                    if (cookie.getName().equals("user")) {
                        userName = cookie.getValue();
                    }
                    if (cookie.getName().equals("JSESSIONID")) {
                        sessionID = cookie.getValue();
                    }
                }
            }
        %>
        <h3>Hi <%=userName%>, Login successful. Your Session ID=<%=sessionID%></h3>
        <br>
        User=<%=user%>
        <br>
        <a href="CheckoutPage.jsp">Checkout Page</a>
        <form action="LogoutServlet" method="post">
            <input type="submit" value="Logout" >
```

CheckoutPage.jsp

```
<%@ page language="java" contentType="text/html; charset=US-ASCII"
    pageEncoding="US-ASCII"%>
<!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=US-ASCII">
        <title>Login Success Page</title>
    </head>
    <body>
        <%
            //allow access only if session exists
            if (session.getAttribute("user") == null) {
                response.sendRedirect("login.html");
            }
            String userName = null;
            String sessionID = null;
            Cookie[] cookies = request.getCookies();
            if (cookies != null) {
                for (Cookie cookie : cookies) {
                    if (cookie.getName().equals("user")) {
                        userName = cookie.getValue();
                    }
                }
            }
        %>
        <h3>Hi <%=userName%>, do the checkout.</h3>
        <br>
        <form action="LogoutServlet" method="post">
            <input type="submit" value="Logout" >
        </form>
```

Output



PRACTICAL 13

IMPLEMENT AUTHENTICATION FILTER USING FILTER API.

MyFilter.java

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

public class MyFilter implements Filter {
    private final String password = "password";
    @Override
    public void init(FilterConfig fc) throws ServletException {
    }
    @Override
    public void doFilter(ServletRequest request, ServletResponse response,
        FilterChain chain) throws IOException, ServletException {
        PrintWriter out = response.getWriter();
        String pass = request.getParameter("pass");
        if (password.equals(pass)) {
            chain.doFilter(request, response);
        } else {
            out.println("You have enter a wrong password");
            RequestDispatcher rs = request.getRequestDispatcher("index.html");
            rs.include(request, response);
        }
    }
    @Override
    public void destroy() {
    }
}
```

first.java

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

public class first extends HttpServlet {
    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        String user = request.getParameter("user");
        out.println("Wellcome " + user);
    }
}
```

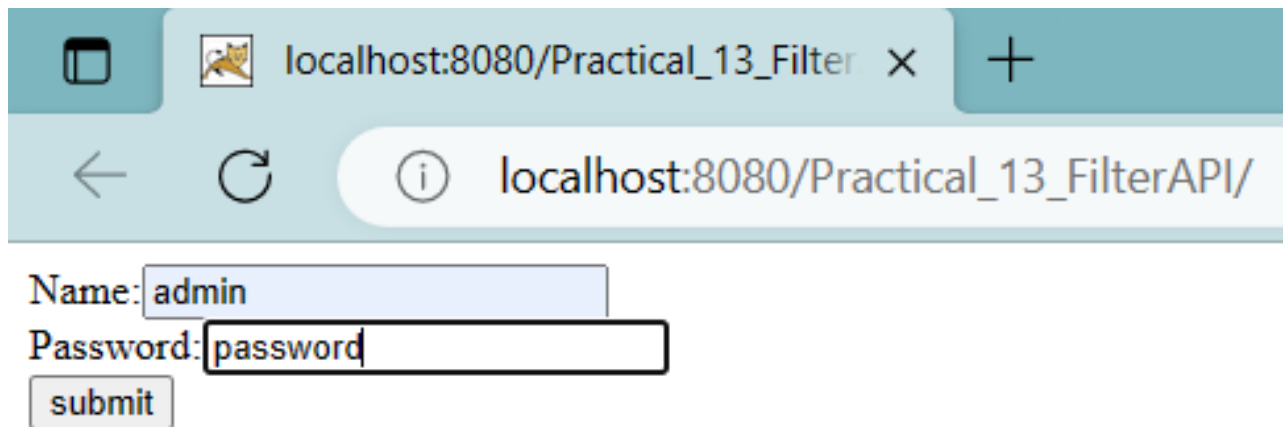
index.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>Login Page</title>
  </head>
  <body>
    <form action="first" method="post">
      Username: <input type="text" name="user">
      <br>
      Password: <input type="password" name="pass">
      <br>
      <input type="submit" value="Login">
    </form>
  </body>
```

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>first</servlet-name>
  </filter-mapping>
  <servlet>
    <servlet-name>first</servlet-name>
    <servlet-class>first</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>first</servlet-name>
    <url-pattern>/first</url-pattern>
  </servlet-mapping>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
  </welcome-file-list>
```

Output



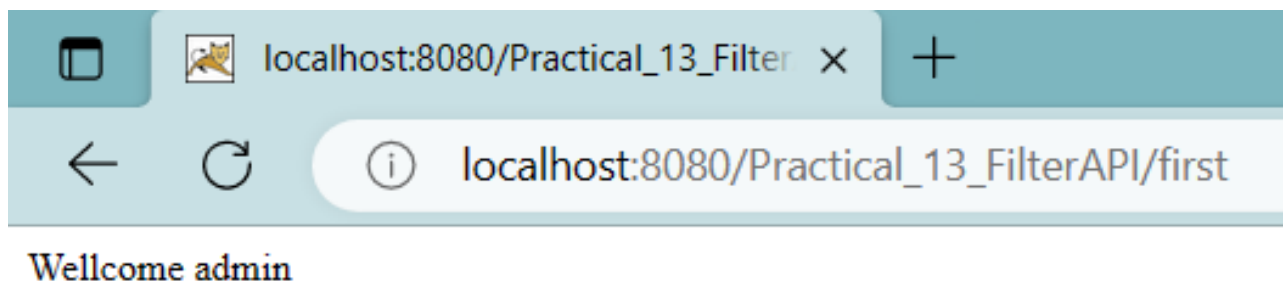
A screenshot of a web browser window. The address bar shows the URL `localhost:8080/Practical_13_FilterAPI/`. Below the address bar, there is a login form with two input fields: "Name:" containing the text "admin" and "Password:" containing the text "password". A "submit" button is located below the password field.

localhost:8080/Practical_13_FilterAPI/

Name: admin

Password: password

submit



A screenshot of a web browser window. The address bar shows the URL `localhost:8080/Practical_13_FilterAPI/first`. Below the address bar, the text "Wellcome admin" is displayed.

localhost:8080/Practical_13_FilterAPI/first

Wellcome admin

PRACTICAL 14

CREATE A JSP (JAVA SERVER PAGE) THAT ADDS AND SUBTRACTS TWO NUMBERS.

Calculate.jsp

```
<!DOCTYPE html>
<html>
  <head>
    <title>Adding and Subtracting Numbers</title>
  </head>
  <body>
    <h2>Adding and Subtracting Numbers</h2>

    <%
      // Get the input parameters
      String param1 = request.getParameter("num1");
      String param2 = request.getParameter("num2");

      // Check if the parameters are not null or empty
      if (param1 != null && param2 != null && !param1.isEmpty() && !param2.isEmpty()) {
        // Parse the input parameters to numbers
        int num1 = Integer.parseInt(param1);
        int num2 = Integer.parseInt(param2);

        // Calculate the results
        int sum = num1 + num2;
        int difference = num1 - num2;

        // Print the results
        out.println("Sum of " + num1 + " and " + num2 + " is " + sum + "<br>");
        out.println("Difference of " + num1 + " and " + num2 + " is " + difference);
      } else {
        out.println("Please provide two numbers to add and subtract.");
      }
    %>
```

Index.html

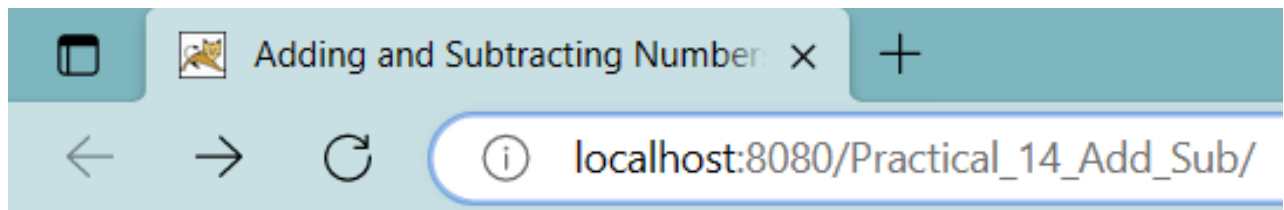
```
<!DOCTYPE html>
<html>
  <head>
    <title>Adding and Subtracting Numbers</title>
  </head>
  <body>
    <h2>Adding and Subtracting Numbers</h2>

    <form method="post" action="calculate.jsp">
      <label for="num1">Number 1:</label>
      <input type="text" name="num1"><br>

      <label for="num2">Number 2:</label>
      <input type="text" name="num2"><br>

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

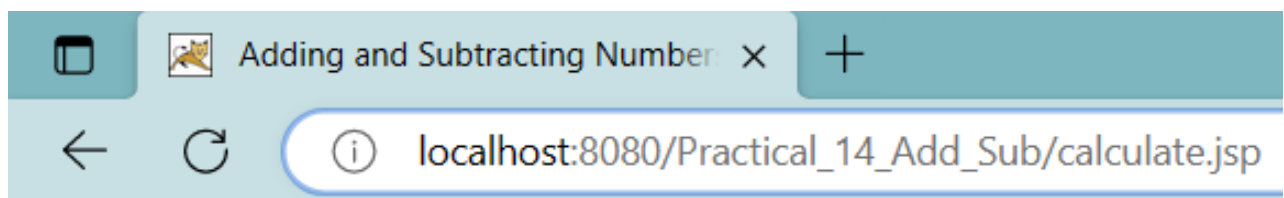
Output



Adding and Subtracting Numbers

Number 1:

Number 2:



Adding and Subtracting Numbers

Sum of 50 and 40 is 90
Difference of 50 and 40 is 10

PRACTICAL 15

DEVELOP A JSP PAGE TO DISPLAY STUDENT INFORMATION WITH SUBJECTS FOR PARTICULAR SEMESTER FROM DATABASE.

studentInfo.jsp

```
<%@ page import="java.sql.*" %>
<%@ page language="java" %>
<%@ page contentType="text/html; charset=UTF-8" %>
<html>
  <head>
    <title>Student Information for Semester</title>
  </head>
  <body>
    <h1>Student Information for Semester</h1>
    <form method="get" action="studentInfo.jsp">
      <label for="semester">Enter Semester:</label>
      <input type="text" id="semester" name="semester" required>
      <input type="submit" value="Submit">
    </form>
    <table border="1">
      <tr>
        <th>Student ID</th>
        <th>Name</th>
        <th>Semester</th>
        <th>Subject</th>
        <th>Grade</th>
      </tr>

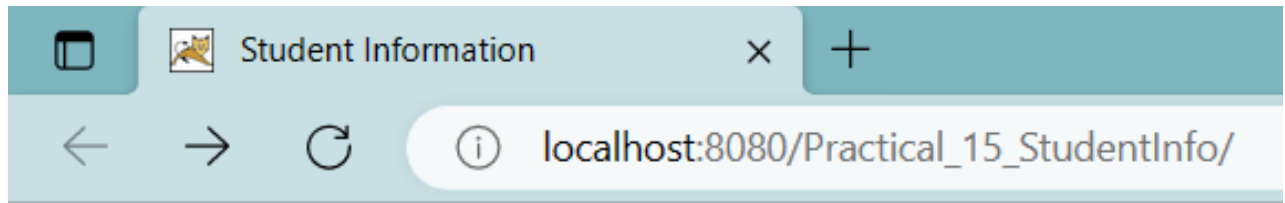
      <%
        // Retrieve the semester value from the request
        String semester = request.getParameter("semester");
        // Connect to the database using JDBC
        String url = "jdbc:mysql://localhost:3306/jdbc";
        String user = "root";
        String password = "";
        Class.forName("com.mysql.jdbc.Driver");
        Connection conn = DriverManager.getConnection(url, user, password);
        // Query the database to retrieve student information
        String sql = "SELECT * FROM student WHERE semester = ?";
        PreparedStatement pstmt = conn.prepareStatement(sql);
        pstmt.setString(1, semester);
        ResultSet rs = pstmt.executeQuery();
```

```
// Iterate through the result set and display the data on the JSP page
while (rs.next()) {
    out.println("<tr>");
    out.println("<td>" + rs.getInt("student_id") + "</td>");
    out.println("<td>" + rs.getString("name") + "</td>");
    out.println("<td>" + rs.getInt("semester") + "</td>");
    out.println("<td>" + rs.getString("subject") + "</td>");
    out.println("<td>" + rs.getString("grade") + "</td>");
    out.println("</tr>");
}

// Close the database connection
conn.close();
%>
</table>
</body>
```

Index.html

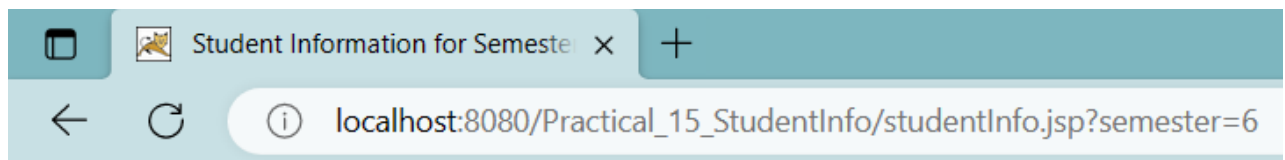
```
<!DOCTYPE html>
<html>
  <head>
    <title>Student Information for Semester</title>
  </head>
  <body>
    <h1>Student Information for Semester</h1>
    <form method="get" action="studentInfo.jsp">
      <label for="semester">Enter Semester:</label>
      <input type="text" id="semester" name="semester" required>
      <input type="submit" value="Submit">
    </form>
  </body>
```



Student Information

Semester:

Output



Student Information for Semester

Enter Semester:

Student ID	Name	Semester	Subject	Grade
7053	Vatsal	6	AJP	A
7051	Dev	6	AJP	A

PRACTICAL 16

IMPLEMENT THE LOGIN APPLICATION WITH USE OF JSF.

Index.xhtml

```
<?xml version='1.0' encoding='UTF-8' ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
    xmlns:h="http://xmlns.jcp.org/jsf/html">
    <h:head>
        <title>Login Form</title>
    </h:head>
    <h:body>
        <h:form>
            <h:outputText value="Login Form"/><br/>
            <h:outputText value="Username"/>
            <h:inputText value="#{login.username}"/><br/>
            <h:outputText value="Password"/>
            <h:inputSecret value="#{login.password}"/><br/>
            <h:commandButton value="Login" action="result.xhtml"/>
        </h:form>
    </h:body>
```

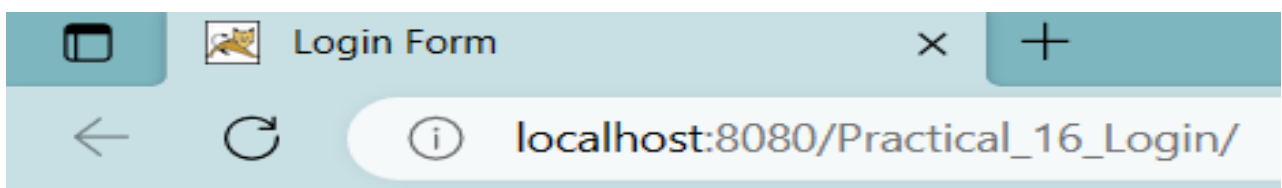
Result.xhtml

```
<?xml version='1.0' encoding='UTF-8' ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
    xmlns:h="http://xmlns.jcp.org/jsf/html">
    <h:head>
        <title>Welcome Page</title>
    </h:head>
    <h:body>
        <h:outputText value="Welcome admin to result.xhtml"/><br/>
        <h:outputText value="#{login.login()}"/>
    </h:body>
</html>
```

Login.java


```
import javax.faces.bean.ManagedBean;
@ManagedBean(name = "login")
public class login {
    String username;
    String password;
    public String getUsername() {
        return username;
    }
    public void setUsername(String username) {
        this.username = username;
    }
    public String getPassword() {
        return password;
    }
    public void setPassword(String password) {
        this.password = password;
    }
    public String login() {
        if (username.equals("admin") && password.equals("password")) {
            return "login successful";
        } else {
            return "login failed";
        }
    }
}
```

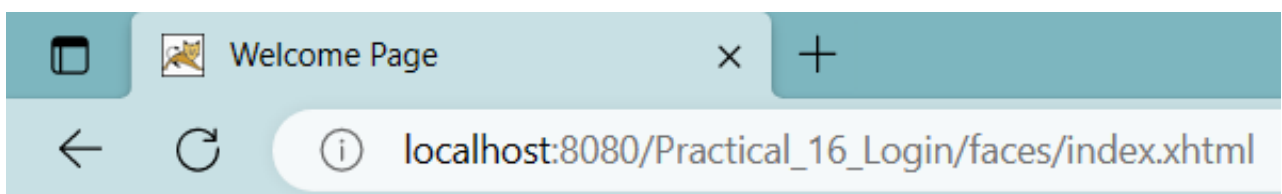
Output



Login Form

Username

Password 



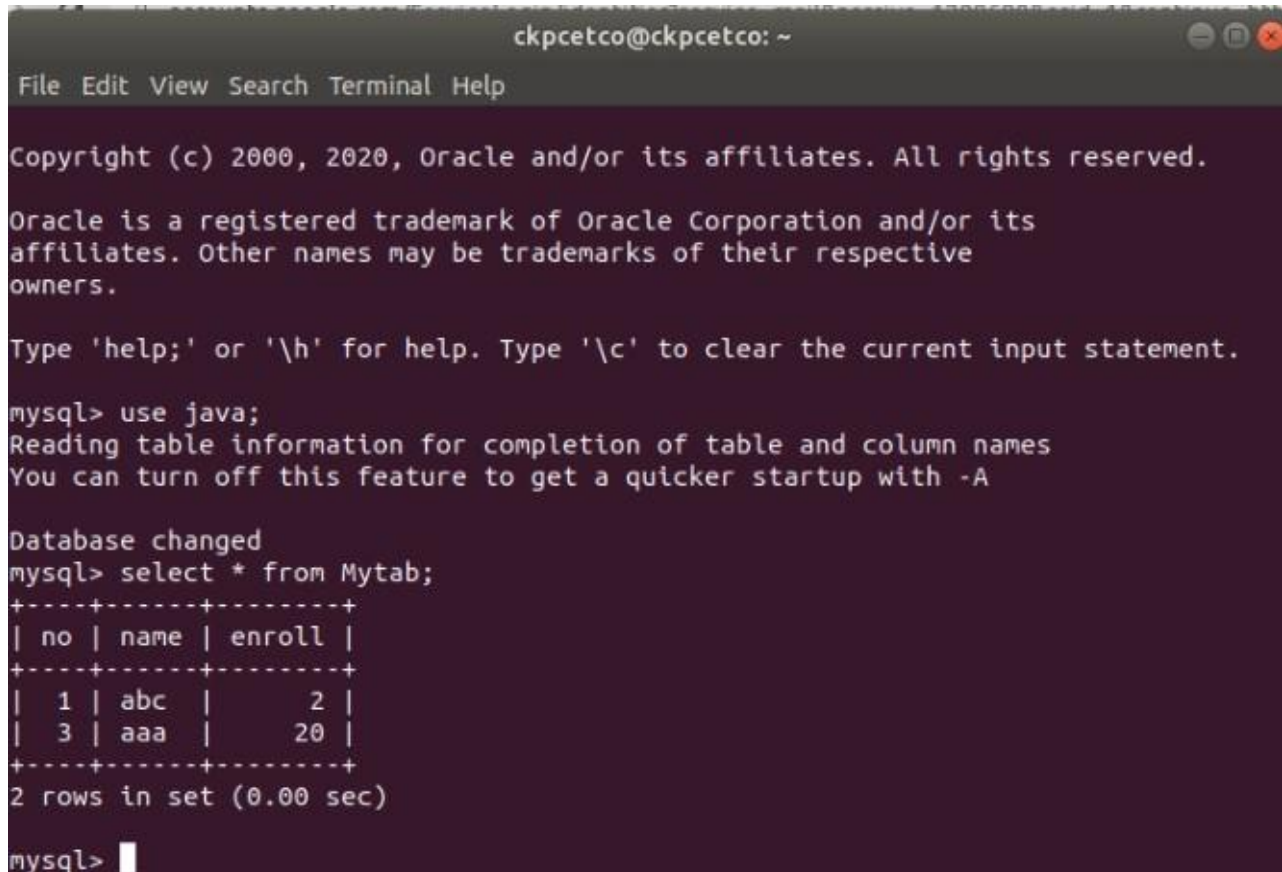
Welcome admin to result.xhtml
login successful

PRACTICAL 17

STUDY AND IMPLEMENT HIBERNATE.

Create an appropriate MYSQL database table in which you want to insert a record using hibernate.

Here, in my case, database:java & table: Mytab



A screenshot of a MySQL terminal window. The window title is 'ckpcetco@ckpcetco: ~'. The menu bar includes 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'. The terminal text shows the MySQL prompt 'mysql>' followed by 'use java;', which changes the database to 'java'. Then, 'select * from Mytab;' is executed, displaying a table with 2 rows: (1, abc, 2) and (3, aaa, 20). The prompt 'mysql>' is visible at the bottom.

```
ckpcetco@ckpcetco: ~
File Edit View Search Terminal Help

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use java;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> select * from Mytab;
+----+-----+-----+
| no | name | enroll |
+----+-----+-----+
|  1 | abc  |      2 |
|  3 | aaa  |     20 |
+----+-----+-----+
2 rows in set (0.00 sec)

mysql>
```

hibernate.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/java?zeroDateTimeB
ehavior=convertToNull</property>
        <property name="hibernate.connection.username">root</property>
        <property name="hibernate.connection.password">mysql</property>
        <mapping resource="mypack/Mytab.hbm.xml"/>
    </session-factory>
</hibernate-configuration>
```

hibernate.reveng.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-reverse-engineering PUBLIC "-//Hibernate/Hibernate Reverse
Engineering DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-reverse-engineering-3.0.dtd">
<hibernate-reverse-engineering>
<schema-selection match-catalog="java"/>
<table-filter match-name="Mytab"/>
</hibernate-reverse-engineering>
```

HibernateUtil.java

```
package mypack;
import org.hibernate.cfg.AnnotationConfiguration;
import org.hibernate.SessionFactory;
public class HibernateUtil {
    private static final SessionFactory sessionFactory;
    static {
        try {
            // Create the SessionFactory from standard (hibernate.cfg.xml) // config file.
            sessionFactory = new
                AnnotationConfiguration().configure().buildSessionFactory();
        } catch (Throwable ex) { // Log the exception.
            System.err.println("Initial SessionFactory creation failed." + ex);
            throw new ExceptionInInitializerError(ex);
        }
    }
    public static SessionFactory getSessionFactory() {
        return sessionFactory;
    }
}
```

Mytab.hbm.xml

```
<?xml version="1.0"?>
<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
"http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
<!-- Generated 13 Apr, 2022 12:30:42 PM by Hibernate Tools 4.3.1 -->
<hibernate-mapping>
<class name="mypack.Mytab" table="Mytab" catalog="java" optimistic-lock="version">
<id name="no" type="int">
<column name="no" />
<generator class="assigned" />
</id>
<property name="name" type="string">
<column name="name" length="20" />
</property>
<property name="enroll" type="java.lang.Integer">
<column name="enroll" />
</property>
</class>
```

Mytab.java

```
package mypack;

public class Mytab implements java.io.Serializable {
    private int no; private String name; private Integer enroll;
    public Mytab() {
    }
    public Mytab(int no) {
        this.no = no;
    }
    public Mytab(int no, String name, Integer enroll) {
        this.no = no; this.name = name;
        this.enroll = enroll;
    }
    public int getNo() {
        return this.no;
    }
    public void setNo(int no) {
        this.no = no;
    }
    public String getName() {
        return this.name;
    }
    public void setName(String name) {
        this.name = name;
    }
}
```



```

    public Integer getEnroll() {
        return this.enroll;
    }

    public void setEnroll(Integer enroll) {
        this.enroll = enroll;
    }
}

```

HibernateServlet.java

```

package mypack;
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;

public class HibernateServlet extends HttpServlet {
    protected void processRequest(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        try (PrintWriter out = response.getWriter()) {
            out.println("<!DOCTYPE html>");
            out.println("<html>");
            out.println("<head>");
            out.println("<title>Servlet HibernateServlet</title>");
            out.println("</head>");
            out.println("<body>");
            out.println("<h1>Servlet HibernateServlet at " + request.getContextPath() +
"</h1>");
            SessionFactory sf;
            Session s;
            Transaction t;
            sf=HibernateUtil.getSessionFactory();
            s=sf.openSession();
            t=s.getTransaction();
            t.begin();
            mypack.Mytab mb=new mypack.Mytab(4,"abc",25);
            s.save(mb);
            t.commit();
            s.close();
            out.println("</body>");
        }
    }
}

```

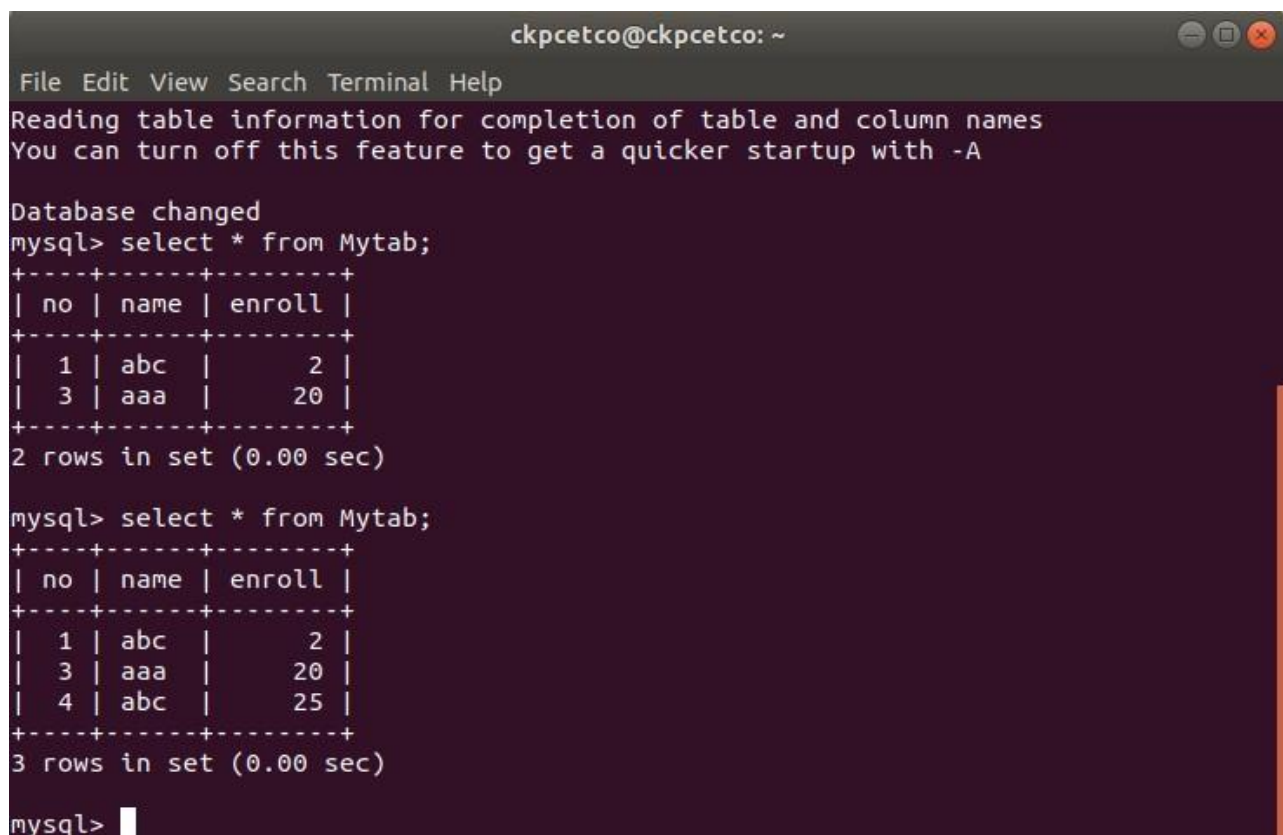
```
// <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the + sign on the left to edit the code.">
```

```
    @Override protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }

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

    @Override
    public String getServletInfo() {
        return "Short description";
    }
}
```

Output

A terminal window titled 'ckpcetco@ckpcetco: ~' with a menu bar (File, Edit, View, Search, Terminal, Help). The terminal shows the output of a MySQL command. It starts with a message about reading table information, followed by 'Database changed'. Then, the command 'mysql> select * from Mytab;' is executed, resulting in a table with 2 rows. The table has columns 'no', 'name', and 'enroll'. The first row has values 1, abc, and 2. The second row has values 3, aaa, and 20. The output is formatted with a header and a separator line. The command is executed again, resulting in a table with 3 rows. The third row has values 4, abc, and 25. The output is also formatted with a header and a separator line. The prompt 'mysql>' is visible at the bottom.

```
ckpcetco@ckpcetco: ~
File Edit View Search Terminal Help
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> select * from Mytab;
+----+-----+-----+
| no | name | enroll |
+----+-----+-----+
| 1  | abc  | 2      |
| 3  | aaa  | 20     |
+----+-----+-----+
2 rows in set (0.00 sec)

mysql> select * from Mytab;
+----+-----+-----+
| no | name | enroll |
+----+-----+-----+
| 1  | abc  | 2      |
| 3  | aaa  | 20     |
| 4  | abc  | 25     |
+----+-----+-----+
3 rows in set (0.00 sec)

mysql>
```

PRACTICAL 18

STUDY AND IMPLEMENT MVC USING SPRING FRAMEWORK.

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

DemoController.java //Abstract Controller Class

demopage.jsp

```
<%@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>Welcome</h1>
        <h2>${obj1}</h2>
    </body>
</html>
```

redirect.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<% response.sendRedirect("demopage.htm"); %>
```

dispatcher-servlet.xml

```
<?xml version='1.0' encoding='UTF-8' ?>
<!-- was: <?xml version="1.0" encoding="UTF-8"?> -->
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:p="http://www.springframework.org/schema/p"
       xmlns:aop="http://www.springframework.org/schema/aop"
       xmlns:tx="http://www.springframework.org/schema/tx"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-4.0.xsd
http://www.springframework.org/schema/aop
http://www.springframework.org/schema/aop/spring-aop-4.0.xsd
http://www.springframework.org/schema/tx
http://www.springframework.org/schema/tx/spring-tx-4.0.xsd">

<bean
    class="org.springframework.web.servlet.mvc.support.ControllerClassNameHandlerMapping"/>
<!-- Most controllers will use the ControllerClassNameHandlerMapping above, but
for the index controller we are using ParameterizableViewController, so we must
define an explicit mapping for it.-->

<bean id="urlMapping"
    class="org.springframework.web.servlet.handler.SimpleUrlHandlerMapping">
    <property name="mappings">
        <props>
            <prop key="index.htm">indexController</prop>
        </props>
    </property>
</bean>

<bean name="/demopage.htm" class="mypack.DemoController"/>
    <bean id="viewResolver"
        class="org.springframework.web.servlet.view.InternalResourceViewResolver"
        p:prefix="/WEB-INF/jsp/"
        p:suffix=".jsp" />
    <!-- The index controller.-->
<bean name="indexController"
    class="org.springframework.web.servlet.mvc.ParameterizableViewController"
    p:viewName="index" />
</beans>
```

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">
    <context-param>
        <param-name>contextConfigLocation</param-name>
        <param-value>/WEB-INF/applicationContext.xml</param-value>
    </context-param>
    <listener>
        <listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>
    </listener>
    <servlet>
        <servlet-name>dispatcher</servlet-name>
        <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
        <load-on-startup>2</load-on-startup>
    </servlet>
    <servlet-mapping>
        <servlet-name>dispatcher</servlet-name>
        <url-pattern>*.htm</url-pattern>
    </servlet-mapping>
    <session-config>
        <session-timeout>
            30
        </session-timeout>
    </session-config>
    <welcome-file-list>
        <welcome-file>redirect.jsp</welcome-file>
    </welcome-file-list>
```

Output

