Practical File

Advance Object Technology

20MCA22C1

Submitted To:

Dr. Gopal Singh

Submitted By:

DINESH KUMAR

Roll No.: 24135

M.C.A 2nd Sem

Department of Computer Science and Applications

(Maharshi Dayanand University, Rohtak)

INDEX

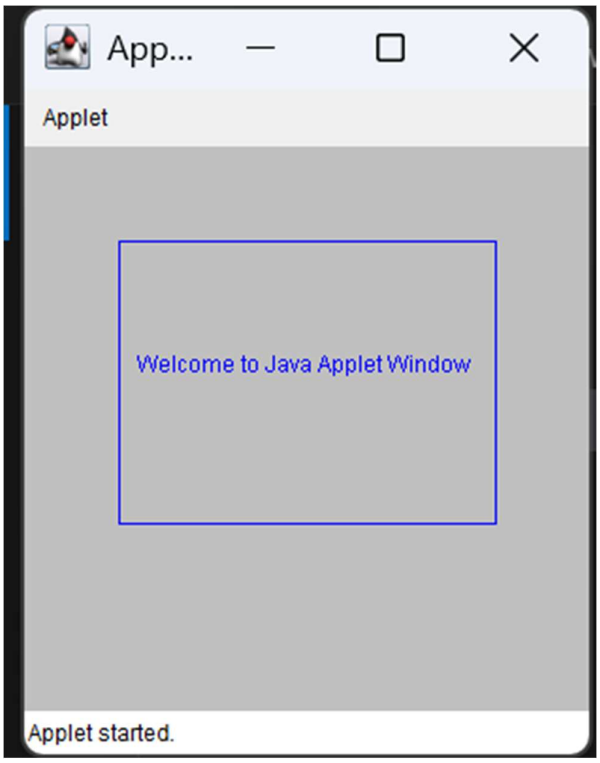
1.Write a program to develop a window using an Applet.

Code:

```
import java.applet.Applet;
import java.awt.*;
public class java1 extends Applet {
   public void init() {
      setBackground(Color.lightGray);
   }
   public void paint(Graphics g) {
      g.setColor(Color.blue);
      g.drawRect(50, 50, 200, 150);
      g.drawString("Welcome to Java Applet Window", 60, 120);
   }
}
```

```
<html>
  <body>
    <applet code="java1.class" width="300" height="300"></applet>
  </body>
</html>
```

Output:

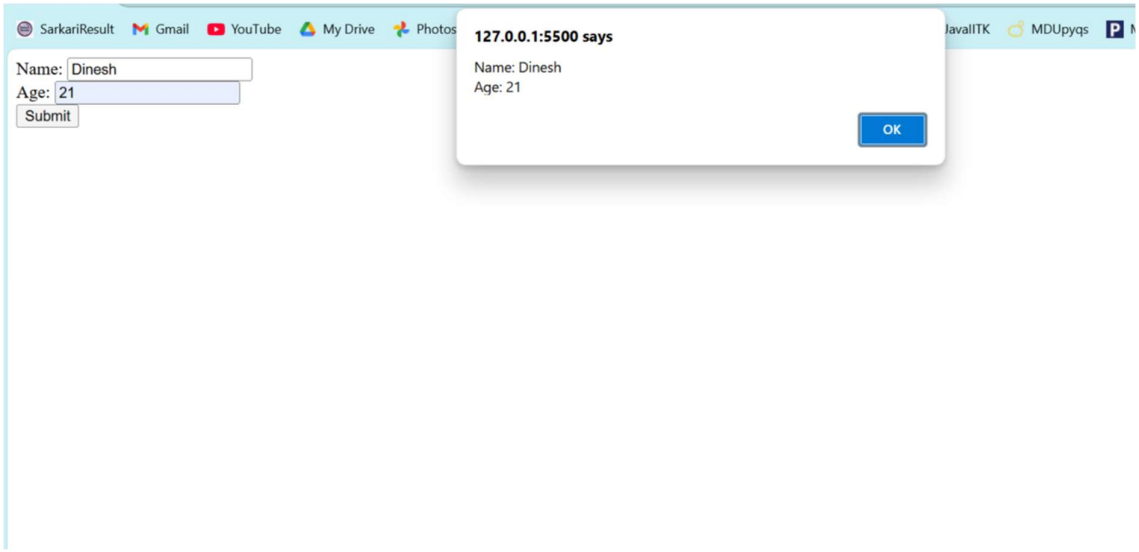


2.Write a program to generate Form using HTML & JAVA SCRIPT.

Code:

```
<!DOCTYPE html>
<html>
<body>
  <form>
    Name: <input type="text" id="name"><br>
    Age: <input type="text" id="age"><br>
    <input type="button" value="Submit" onclick="validateForm()">
  </form>
  <script>
    function validateForm() {
      var name = document.getElementById("name").value;
      var age = document.getElementById("age").value;
      alert("Name: " + name + "\nAge: " + age);
    }
  </script>
</body>
</html>
```

Output:


SarkariResult
Gmail
YouTube
My Drive
Photos
MDUpyqs
Name: Dinesh
Age: 21
Submit
127.0.0.1:5500 says
Name: Dinesh
Age: 21
OK

3.Write a program to implement Event and AWT components.

a. **Button**

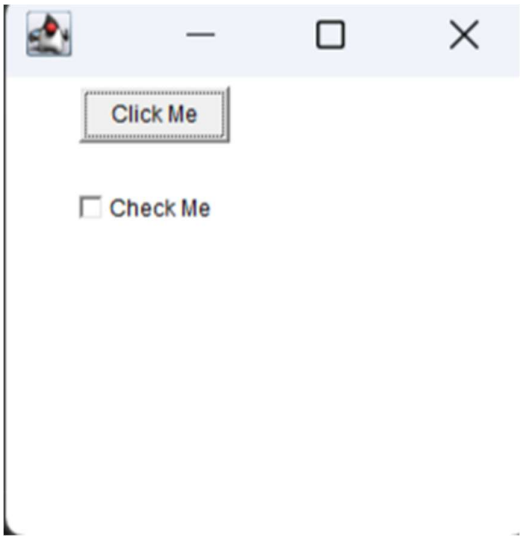
b. **Checkbox**

Code:

```
import java.awt.*;
import java.awt.event.*;
public class java3 extends Frame implements ActionListener {
   Button b;
   Checkbox cb;
   java3() {
      b = new Button("Click Me");
      b.setBounds(50, 50, 80, 30);
      cb = new Checkbox("Check Me");
      cb.setBounds(50, 100, 100, 30);
      b.addActionListener(this);
      add(b);
      add(cb);
      setSize(300, 300);
      setLayout(null);
      setVisible(true);
   }
```

```
    public void actionPerformed(ActionEvent e) {
        System.out.println("Button Clicked");
    }
    public static void main(String args[]) {
        new java3();
    }
}
```

Output:



4.Write a program to implement Swing components.

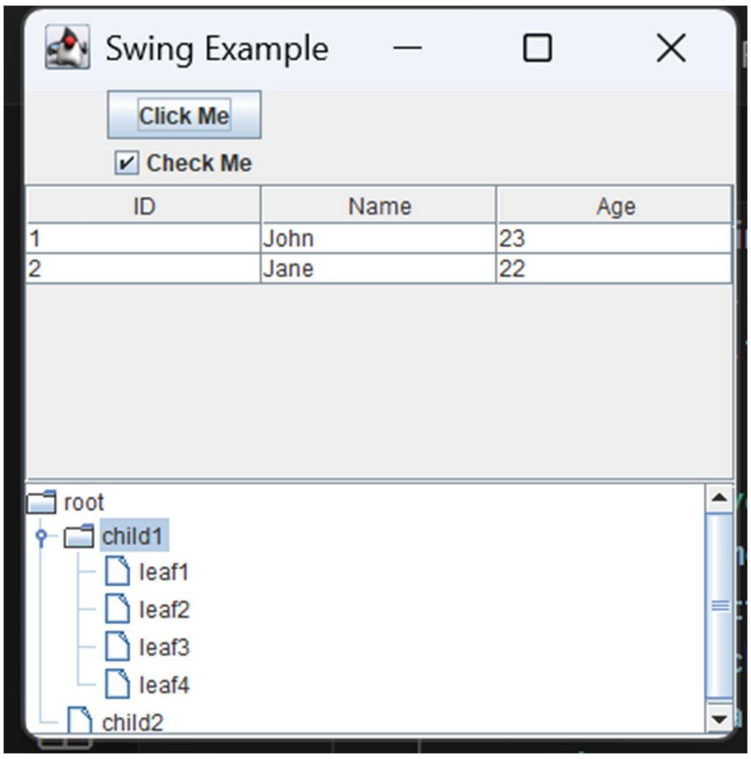
a. **Button**
b. **Table**
c. **Tree**
d. **Checkbox Pane**

Code:

```
import javax.swing.*;
import javax.swing.tree.*;
public class java4 {
   public static void main(String[] args) {
      JFrame frame = new JFrame("Swing Example");
      JButton button = new JButton("Click Me");
      JCheckBox checkBox = new JCheckBox("Check Me");
      String data[][] = { { "1", "John", "23" }, { "2", "Jane", "22" } };
      String column[] = { "ID", "Name", "Age" };
      JTable table = new JTable(data, column);
      DefaultMutableTreeNode root = new DefaultMutableTreeNode("root");
      DefaultMutableTreeNode child1 = new DefaultMutableTreeNode("child1");
      DefaultMutableTreeNode child2 = new DefaultMutableTreeNode("child2");
      root.add(child1);
      root.add(child2);
```

```
        DefaultMutableTreeNode leaf1 = new DefaultMutableTreeNode("leaf1");
        DefaultMutableTreeNode leaf2 = new DefaultMutableTreeNode("leaf2");
        DefaultMutableTreeNode leaf3 = new DefaultMutableTreeNode("leaf3");
        DefaultMutableTreeNode leaf4 = new DefaultMutableTreeNode("leaf4");
        child1.add(leaf1); child1.add(leaf2); child1.add(leaf3); child1.add(leaf4);
        JTree tree = new JTree(root);
        frame.add(button);
        frame.add(checkBox);
        frame.add(new JScrollPane(table));
        frame.add(new JScrollPane(tree));
        frame.setLayout(new BoxLayout(frame.getContentPane(),
BoxLayout.Y_AXIS));
        frame.setSize(400, 400);
        frame.setVisible(true);
    }
}
```

Output:


Swing Example
Click Me
Check Me
ID
Name
Age
1
John
23
2
Jane
22
root
child1
leaf1
leaf2
leaf3
leaf4
child2

5. Write a program to implement Swing components.

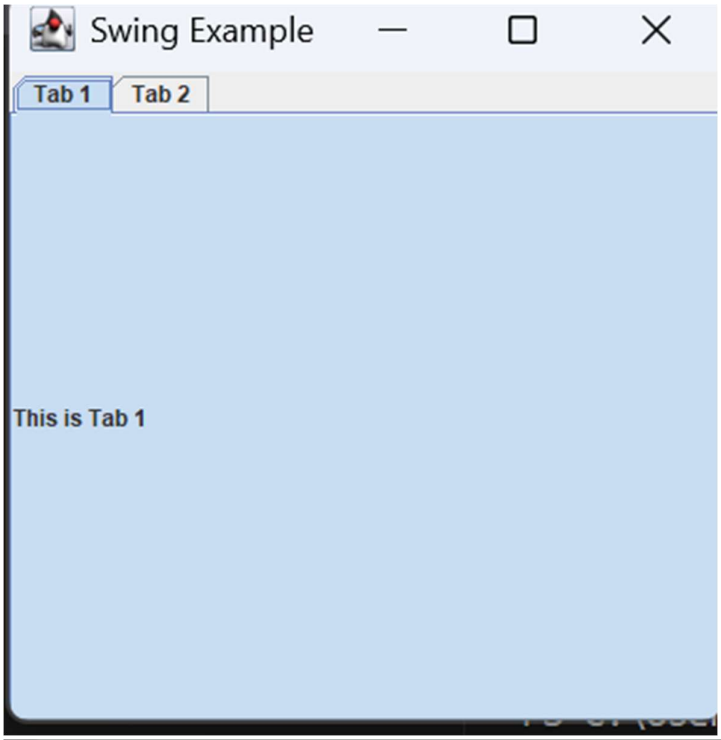
(a) Tabbed Pane

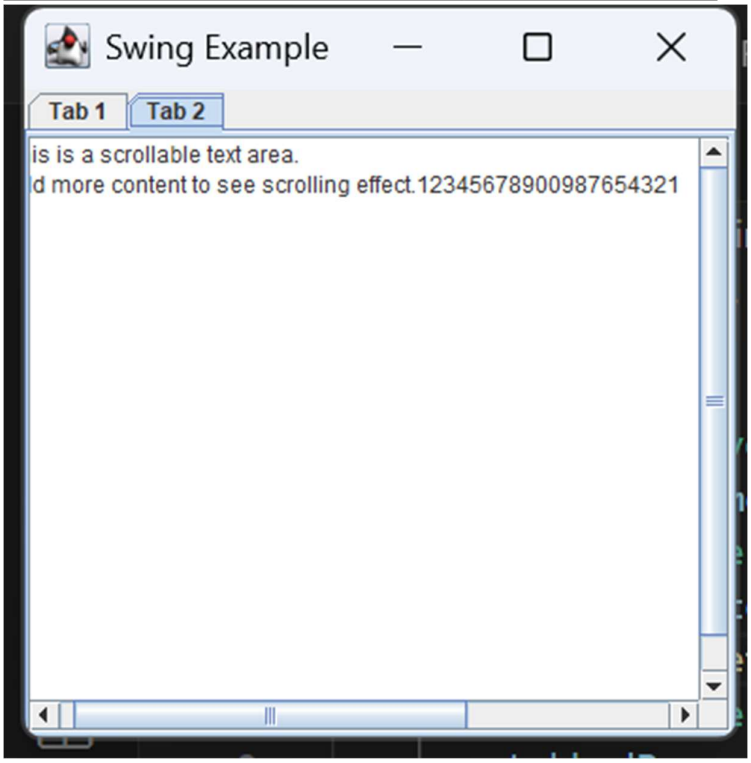
(b) Scroll Pane

Code:

```
import javax.swing.*;

public class java5 {

    public static void main(String[] args) {

        JFrame frame = new JFrame("Swing Example");

        JTabbedPane tabbedPane = new JTabbedPane();

        JTextArea textArea = new JTextArea(20, 50);

        textArea.setText("This is a scrollable text area.\nAdd more content to see
scrolling effect.12345678900987654321");

        JScrollPane scrollPane = new JScrollPane(textArea);

        tabbedPane.add("Tab 1", new JLabel("This is Tab 1"));

        tabbedPane.add("Tab 2", scrollPane);

        frame.add(tabbedPane);

        frame.setSize(400, 400);

        frame.setVisible(true);

    }

}
```

Output:


Swing Example
Tab 1
Tab 2
This is Tab 1


Swing Example
Tab 1
Tab 2
is is a scrollable text area.
d more content to see scrolling effect.12345678900987654321

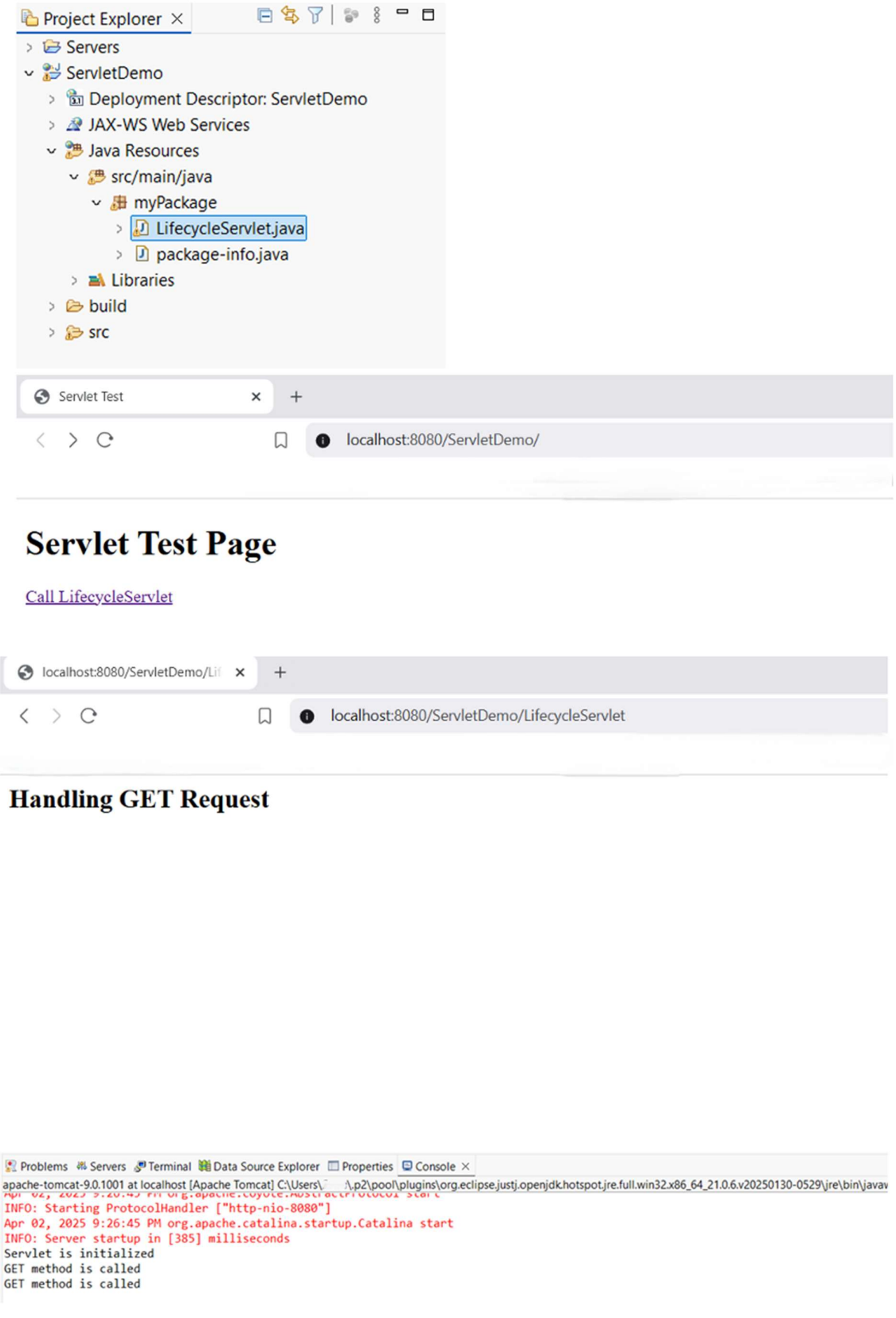
6. Write a program to implement all the phases of life cycle of Servlet.

Code:

```
package myPackage;

import java.io.IOException;

import java.io.PrintWriter;

import javax.servlet.ServletConfig;

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("/LifecycleServlet")

public class LifecycleServlet extends HttpServlet {

    public void init(ServletConfig config) throws ServletException {

       super.init(config);

       System.out.println("Servlet is initialized");

   }

   public void doGet(HttpServletRequest request, HttpServletResponse

response) throws ServletException, IOException {

       response.setContentType("text/html");

       PrintWriter out = response.getWriter();

       out.println("<h2>Handling GET Request</h2>");
```

```
System.out.println("GET method is called");
    }
    public void destroy() {
        System.out.println("Servlet is being destroyed");
    }
}
```

Output:


Project Explorer
Servers
ServletDemo
Deployment Descriptor: ServletDemo
JAX-WS Web Services
Java Resources
src/main/java
myPackage
LifecycleServlet.java
package-info.java
Libraries
build
src
Servlet Test
localhost:8080/ServletDemo/
Servlet Test Page
Call LifecycleServlet
localhost:8080/ServletDemo/LifecycleServlet
Handling GET Request
Problems
Servers
Terminal
Data Source Explorer
Properties
Console
apache-tomcat-9.0.1001 at localhost [Apache Tomcat] C:\Users\
\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.6.v20250130-0529\jre\bin\javav
INFO: Starting ProtocolHandler ["http-nio-8080"]
Apr 02, 2025 9:26:45 PM org.apache.catalina.startup.Catalina start
INFO: Server startup in [385] milliseconds
Servlet is initialized
GET method is called
GET method is called

7. Write a program to show implement DHTML and CSS with java script.

Code:

```
<!DOCTYPE html>
<html>
<head>
<style>
  #text { color: blue; font-size: 20px; }
</style>
<script>
  function changeText() {
    document.getElementById("text").style.color = "red";
  }
</script>
</head>
<body>
  <p id="text">This is DHTML Example</p>
  <button onclick="changeText()">Change Color</button>
</body>
</html>
```

Output:

8. How is role of server side different from client side in a typical website? Clear using an example.

Reason:

The server-side handles backend tasks like processing requests, managing databases, and generating dynamic content (e.g., fetching user data), using languages like Java, Python, or PHP. The client-side focuses on the frontend, rendering the UI with HTML/CSS, and handling user interactions with JavaScript in the browser. They work together: the server provides data, and the client displays and interacts with it.

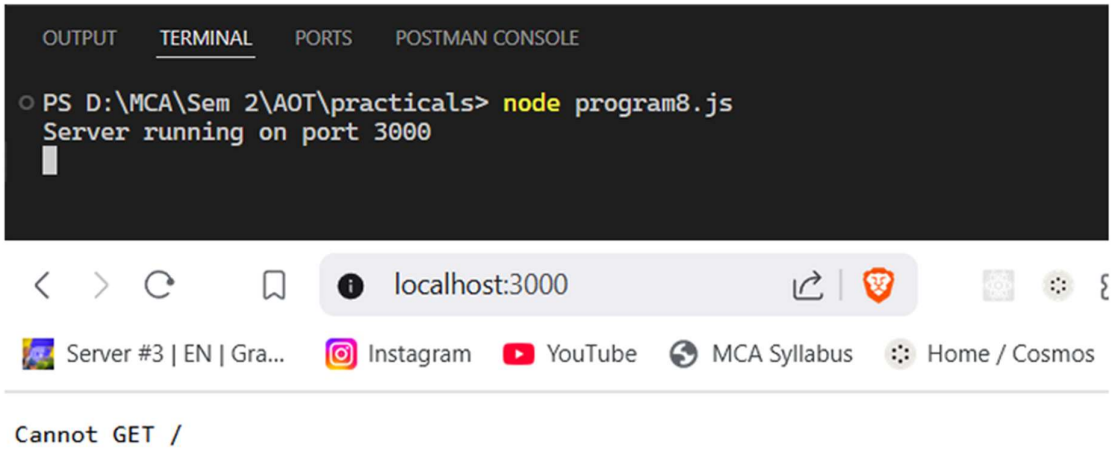
Code:

Server side:

```
const express = require('express');
const bodyParser = require('body-parser');
 const app = express();
app.use(bodyParser.urlencoded({ extended: true }));
 app.post('/login', (req, res) => {
   const { username, password } = req.body;
   if (username === "admin" && password === "1234") {
      res.send("Login Successful!");
   } else {
      res.send("Invalid Credentials!");
```

```
}
});
 app.listen(3000, () => {
    console.log("Server running on port 3000");
});
```

Output:

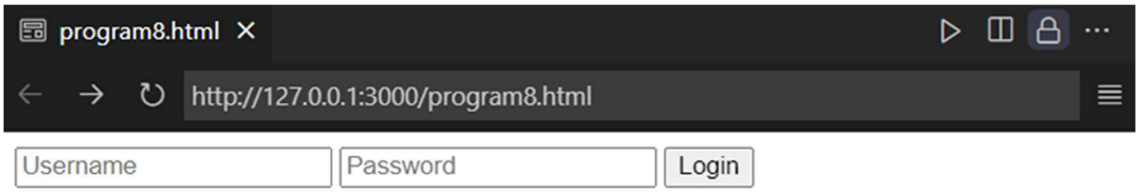


Client side:

```
<form onsubmit="return validateForm()" action="/login" method="POST">
<input type="text" id="username" placeholder="Username">
<input type="password" id="password" placeholder="Password">
<button type="submit">Login</button>
```

```
</form>
 <script>
   function validateForm() {
       let username = document.getElementById("username").value;
       let password = document.getElementById("password").value;
       if (username === "" || password === "") {
           alert("Fields cannot be empty!");
           return false; // Prevents form submission
       }
return true;
}
</script>
```

Output:



9. Write a program in Java using JSP which accept two integer numbers from user and display the result.

Code:

Index.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
   <meta charset="UTF-8">
   <title>Number Calculator - Input Form</title>
</head>
<body>
   <h1>Number Calculator</h1>
   <form action="calculateResult.jsp" method="post">
      <label for="number1">Enter First Number:</label>
      <input type="number" id="number1" name="number1"
required><br><br>

      <label for="number2">Enter Second Number:</label>
      <input type="number" id="number2" name="number2"
required><br><br>
```

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

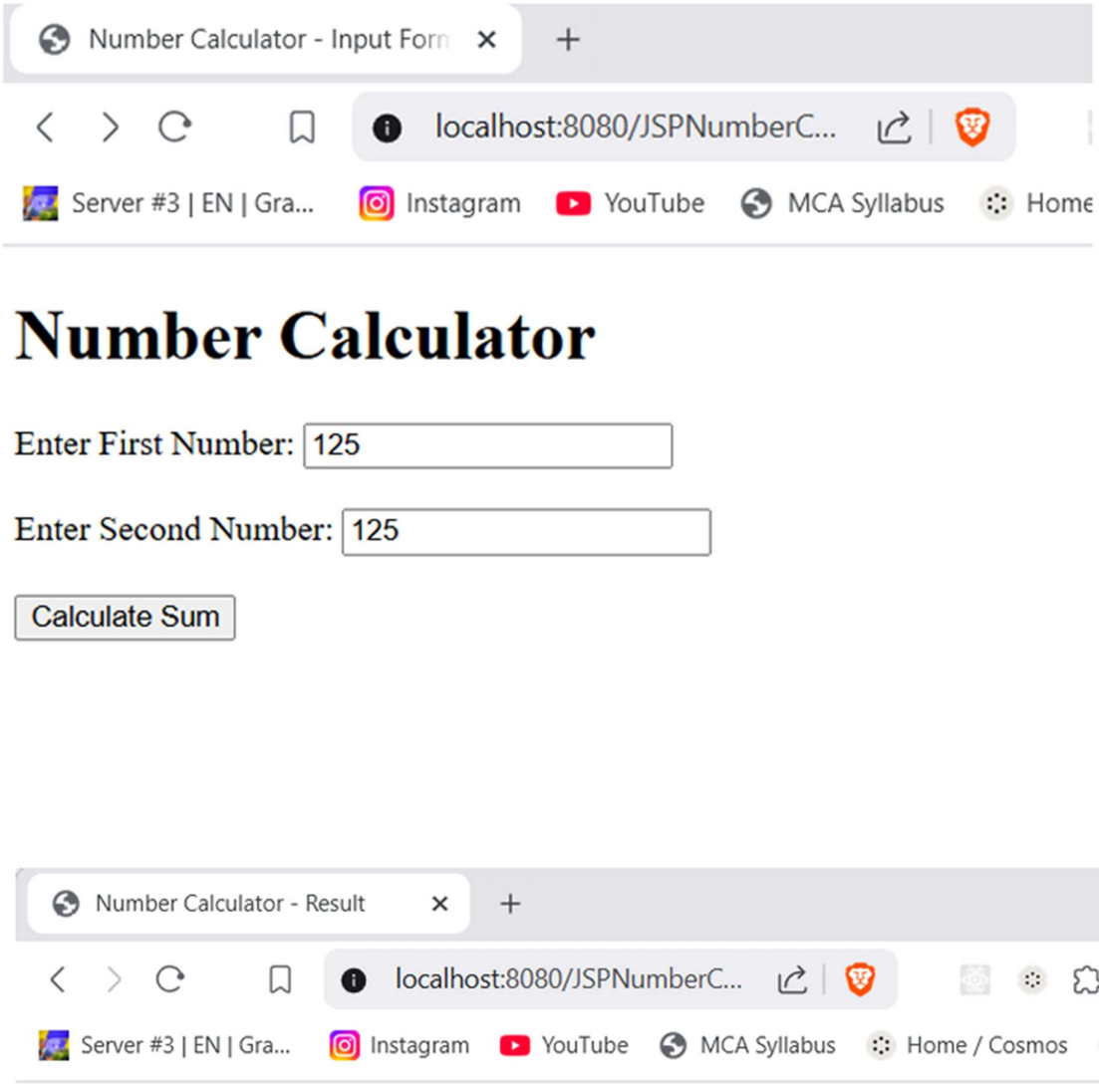
calculateResult.jsp

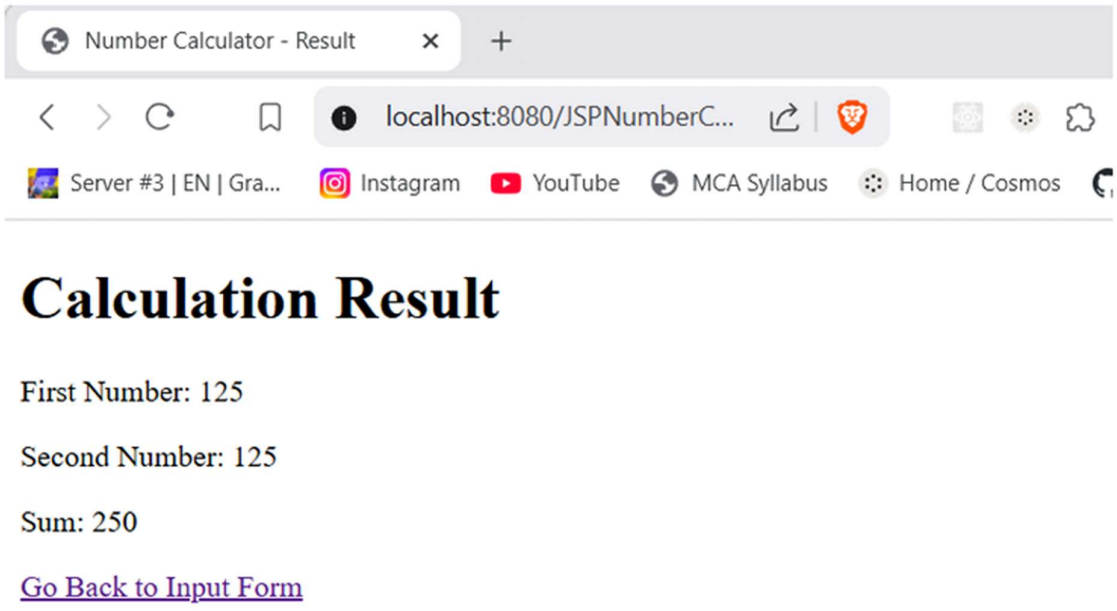
```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>



<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Number Calculator - Result</title>
</head>
<body>
    <h1>Calculation Result</h1>

    <%
        // Retrieve the numbers from the form
```

```
    String num1Str = request.getParameter("number1");
    String num2Str = request.getParameter("number2");

    // Initialize variables for the result
    int number1 = 0, number2 = 0, sum = 0;
    String errorMessage = null;

    try {
        // Convert the string inputs to integers
        number1 = Integer.parseInt(num1Str);
        number2 = Integer.parseInt(num2Str);

        // Calculate the sum
        sum = number1 + number2;
    } catch (NumberFormatException e) {
        errorMessage = "Please enter valid integer numbers.";
    }

%>

<!-- Display the result or error message -->
```

```
    <% if (errorMessage != null) { %>
        <p style="color: red;"><%= errorMessage %></p>
    <% } else { %>
        <p>First Number: <%= number1 %></p>
        <p>Second Number: <%= number2 %></p>
        <p>Sum: <%= sum %></p>
    <% } %>

    <p><a href="inputForm.jsp">Go Back to Input Form</a></p>
</body>
</html>
```

Output:


Number Calculator - Input Form
localhost:8080/JSPNumberC...
Server #3 | EN | Gra...
Instagram
YouTube
MCA Syllabus
Home
Number Calculator
Enter First Number: 125
Enter Second Number: 125
Calculate Sum


Number Calculator - Result
localhost:8080/JSPNumberC...
Server #3 | EN | Gra...
Instagram
YouTube
MCA Syllabus
Home / Cosmos
Calculation Result
First Number: 125
Second Number: 125
Sum: 250
Go Back to Input Form

10. Write a program using POST and GET Method in swing.

Code:

```
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.*;
import java.net.HttpURLConnection;
import java.net.URL;
import java.net.URLEncoder;
import javax.swing.*;
public class SwingHttpClient extends JFrame {
   private JTextField nameField;
   private JTextArea responseArea;
   public SwingHttpClient() {
      setTitle("Swing HTTP Client (GET & POST)");
      setSize(500, 400);
      setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      setLayout(new FlowLayout());
      JLabel nameLabel = new JLabel("Enter Name:");
      nameField = new JTextField(20);
      JButton getButton = new JButton("Send GET Request");
      JButton postButton = new JButton("Send POST Request");
```

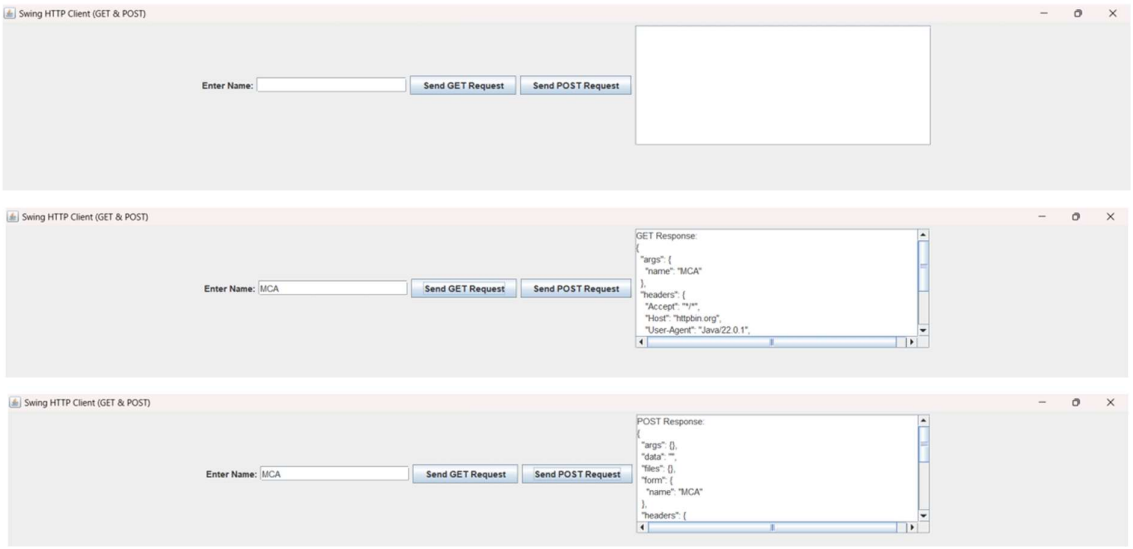
```
    responseArea = new JTextArea(10, 40);
    responseArea.setEditable(false);
    JScrollPane scrollPane = new JScrollPane(responseArea);
    add(nameLabel);
    add(nameField);
    add(getButton);
    add(postButton);
    add(scrollPane);
    getButton.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
            sendGetRequest();
        }
    });
    postButton.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
            sendPostRequest();
        }
    });
    setVisible(true);
}
private void sendGetRequest() {
    try {
```

```
            String name = URLEncoder.encode(nameField.getText(), "UTF-8");
            String urlStr = "https://httpbin.org/get?name=" + name;
            URL url = new URL(urlStr);
            HttpURLConnection conn = (HttpURLConnection)
url.openConnection();
            conn.setRequestMethod("GET");
            BufferedReader in = new BufferedReader(new
InputStreamReader(conn.getInputStream()));
            String inputLine;
            StringBuilder response = new StringBuilder();
            while ((inputLine = in.readLine()) != null) {
               response.append(inputLine).append("\n");
            }
            in.close();
responseArea.setText("GET Response:\n" + response.toString());
        } catch (Exception ex) {
            responseArea.setText("Error in GET request: " + ex.getMessage());
        }
    }
    private void sendPostRequest() {
        try {
            String name = URLEncoder.encode(nameField.getText(), "UTF-8");
```

```
        String urlStr = "https://httpbin.org/post";
        URL url = new URL(urlStr);
        HttpURLConnection conn = (HttpURLConnection)
url.openConnection();
        conn.setRequestMethod("POST");
        conn.setDoOutput(true);
        conn.setRequestProperty("Content-Type", "application/x-www-form
urlencoded");
        OutputStream os = conn.getOutputStream();
        BufferedWriter writer = new BufferedWriter(new
OutputStreamWriter(os, "UTF-8"));
        writer.write("name=" + name);
        writer.flush();
        writer.close();
        os.close();
        BufferedReader in = new BufferedReader(new
InputStreamReader(conn.getInputStream()));
        String inputLine;
        StringBuilder response = new StringBuilder();
        while ((inputLine = in.readLine()) != null) {
           response.append(inputLine).append("\n");
        }
```

```
            in.close();
            responseArea.setText("POST Response:\n" + response.toString());
        } catch (Exception ex) {
            responseArea.setText("Error in POST request: " +
ex.getMessage());
        }
    }
    public static void main(String[] args) {
        new SwingHttpClient();
    }
}
```

Output:



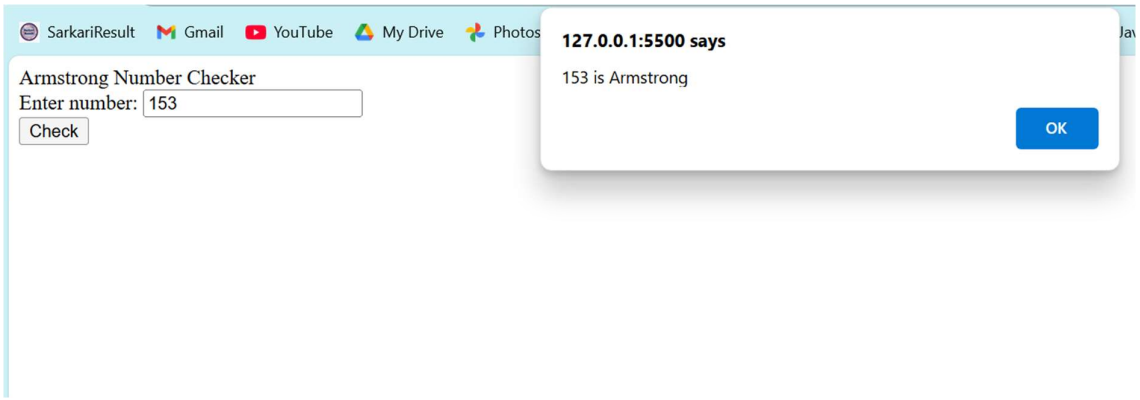
11. Write a JavaScript program to check number entered is an Armstrong number or not.

Code:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Document</title>
</head>
<body>
   Armstrong Number Checker
   <form onsubmit="return isArmstrong(num)">
      Enter number: <input type="integer" id="num"><br>
      <input type="submit" value="Check">
   </form>
   <script>
      function isArmstrong() {
      let num = parseInt(document.getElementById("num").value);
      let sum = 0, temp = num;
      while (temp > 0) {
         let digit = temp % 10;
```

```
        sum += digit ** 3;
        temp = Math.floor(temp / 10);
    }
    if(sum == num){
        alert(num+" is Armstrong");
    } else{
        alert(num+" is not Armstrong");
    }
}
</script>
</body>
</html>
```

Output:



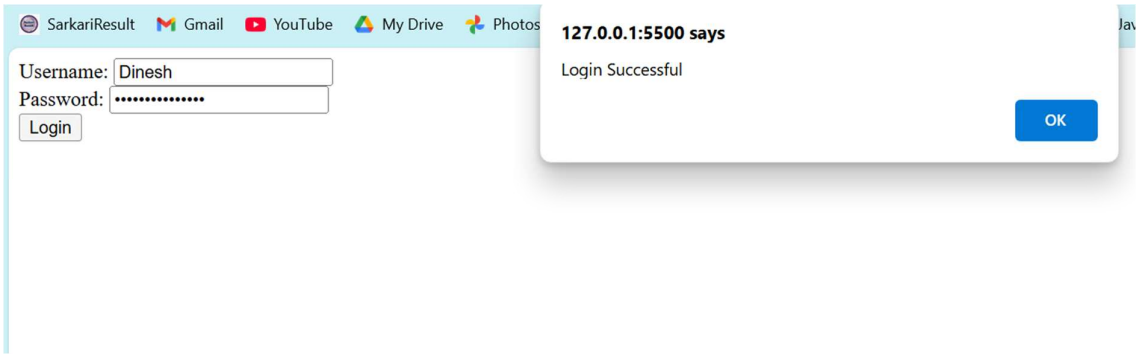
12. Write a JavaScript program to create a Login Form and validate it.

Code:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Document</title>
</head>
<body>
   <form onsubmit="return validateLogin()">
      Username: <input type="text" id="username"><br>
      Password: <input type="password" id="password"><br>
      <input type="submit" value="Login">
   </form>
   <script>
      function validateLogin() {
         let user = document.getElementById("username").value;
         let pass = document.getElementById("password").value;
         if (user == "Dinesh" && pass == "dineshisthebest") {
            alert("Login Successful");
            return true;
```

```
        } else {
            alert("Invalid Credentials");
            return false;
        }
    }
  </script>
</body>
</html>
```

Output:



13. Write a program to implement Event and AWT components.

a. **CANVAS**

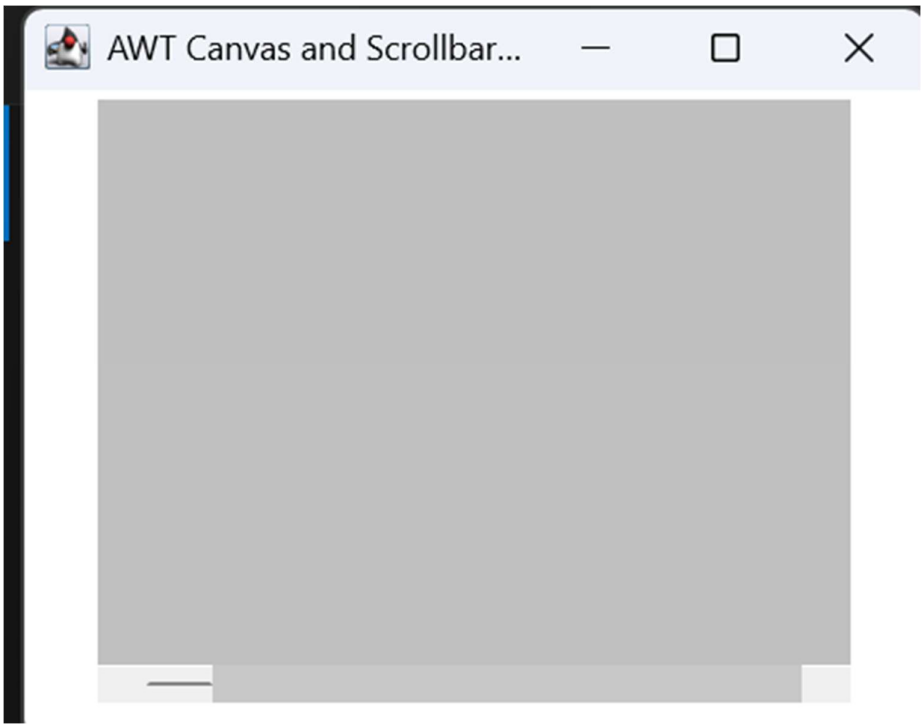
b. **SCROLLBAR**

Code:

```
import java.awt.*;
import java.awt.event.*;
public class java13 extends Frame {
   public java13() {
      setTitle("AWT Canvas and Scrollbar Example");
      Canvas canvas = new Canvas();
      canvas.setSize(400, 300);
      canvas.setBackground(Color.LIGHT_GRAY);
      Scrollbar scrollbar = new Scrollbar();
      scrollbar.setOrientation(Scrollbar.HORIZONTAL);
      scrollbar.setBounds(50, 350, 400, 20);
      add(canvas);
      add(scrollbar);
      setLayout(null);
      canvas.setLocation(50, 50);
      scrollbar.addAdjustmentListener(new AdjustmentListener() {
         public void adjustmentValueChanged(AdjustmentEvent e) {
```

```
            int value = scrollbar.getValue();
            System.out.println("Scrollbar Value: " + value);
        }
    });
    setSize(500, 400);
    setVisible(true);
    addWindowListener(new WindowAdapter() {
        public void windowClosing(WindowEvent we) {
            System.exit(0);
        }
    });
  }
  public static void main(String[] args) {
    new java13();
  }
}
```

Output:



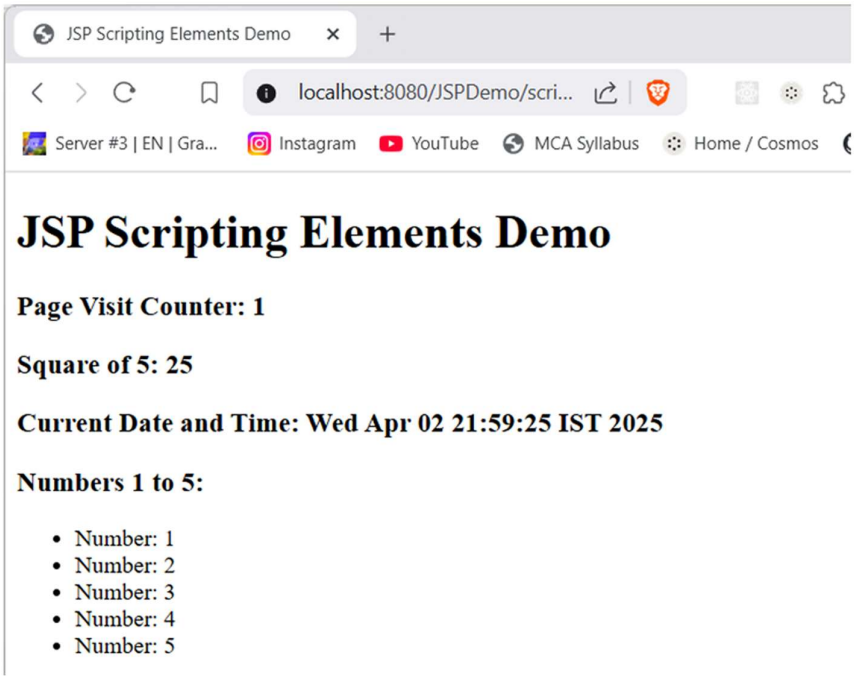
14. Write a program using JSP to implement the Scripting Elements.

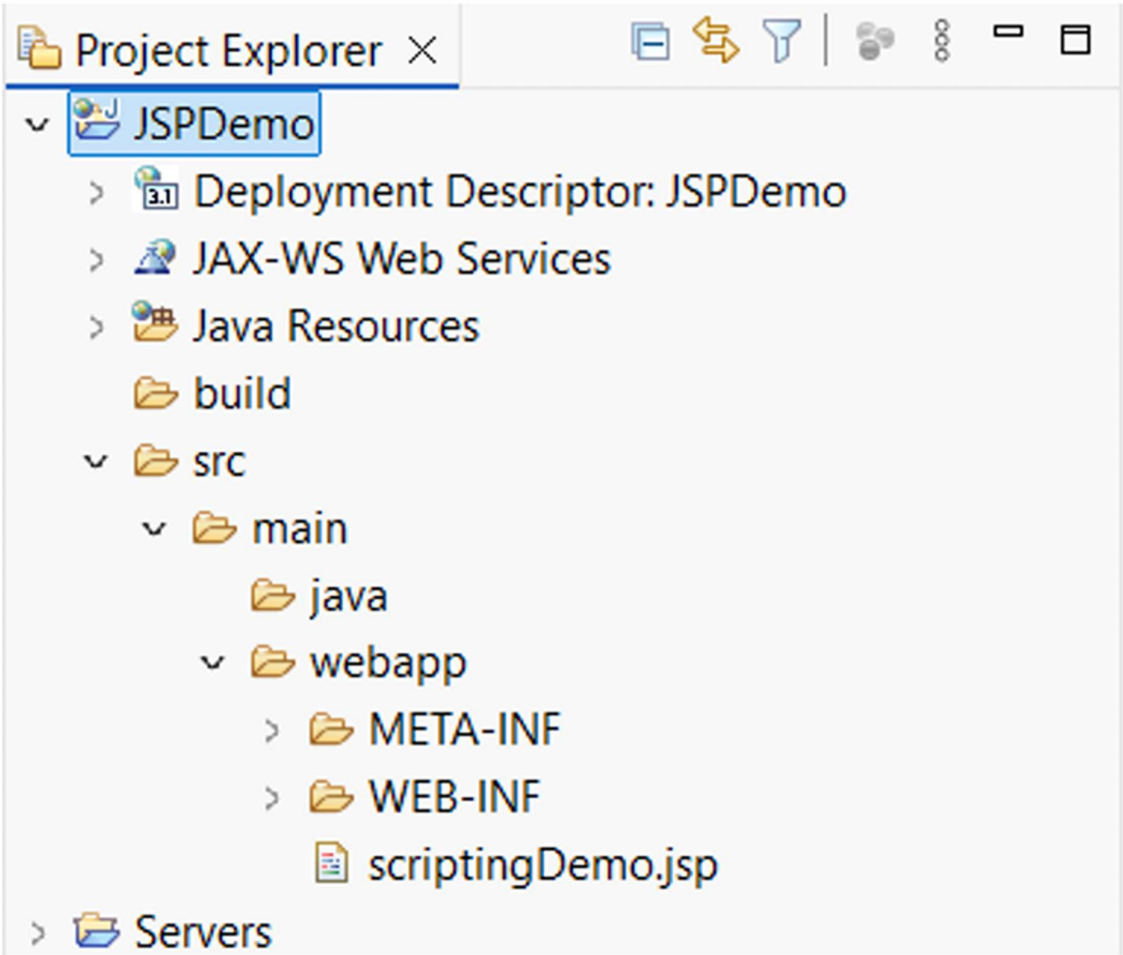
Code:

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>JSP Scripting Elements Demo</title>
</head>
<body>
<h1>JSP Scripting Elements Demo</h1>
<%!
    int counter = 0;
    public int square(int number) {
        return number * number;
    }
  %>
  <%
    counter++;
    java.util.Date currentDate = new java.util.Date();
  %>
```

```
    <h3>Page Visit Counter: <%= counter %></h3>
    <h3>Square of 5: <%= square(5) %></h3>
    <h3>Current Date and Time: <%= currentDate %></h3>
    <h3>Numbers 1 to 5:</h3>
    <ul>
        <%
            for (int i = 1; i <= 5; i++) {
        %>
            <li>Number: <%= i %></li>
        <%
            }
        %>
    </ul>
</body>
</html>
```

Output:


JSP Scripting Elements Demo
localhost:8080/JSPDemo/scri...
Server #3 | EN | Gra...
Instagram
YouTube
MCA Syllabus
Home / Cosmos
JSP Scripting Elements Demo
Page Visit Counter: 1
Square of 5: 25
Current Date and Time: Wed Apr 02 21:59:25 IST 2025
Numbers 1 to 5:
Number: 1
Number: 2
Number: 3
Number: 4
Number: 5


Project Explorer
JSPDemo
Deployment Descriptor: JSPDemo
JAX-WS Web Services
Java Resources
build
src
main
java
webapp
META-INF
WEB-INF
scriptingDemo.jsp
Servers

15. Write a program using JSP to implement any five Implicit Objects.

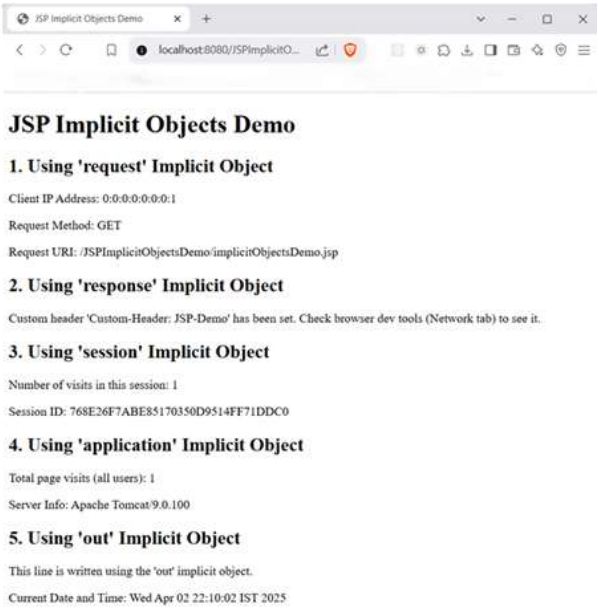
Code:

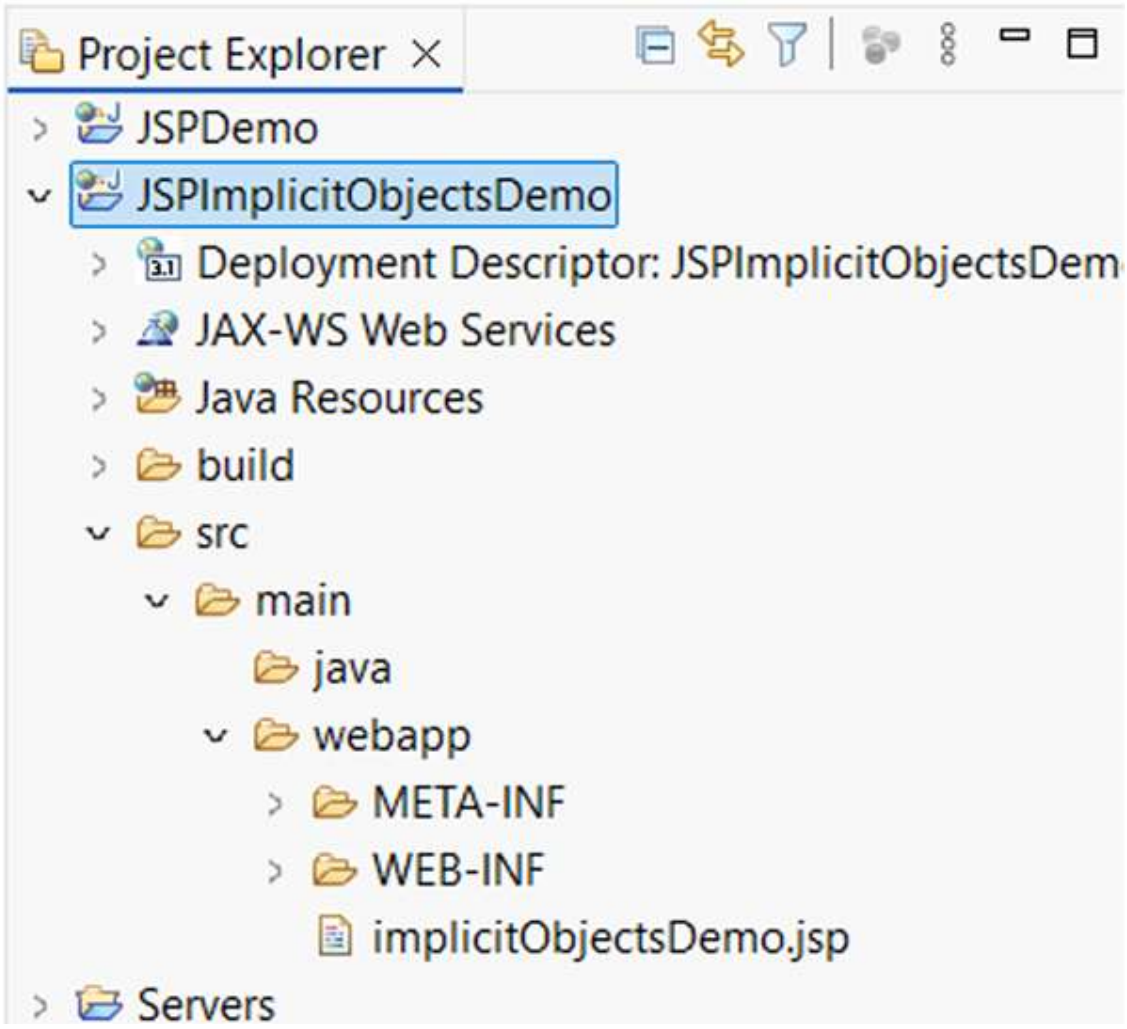
```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>JSP Implicit Objects Demo</title>
 </head>
<body>
   <h1>JSP Implicit Objects Demo</h1>
   <h2>1. Using 'request' Implicit Object</h2>
   <p>Client IP Address: <%= request.getRemoteAddr() %></p>
   <p>Request Method: <%= request.getMethod() %></p>
   <p>Request URI: <%= request.getRequestURI() %></p>
   <h2>2. Using 'response' Implicit Object</h2>
   <%
      response.setHeader("Custom-Header", "JSP-Demo");
      out.println("<p>Custom header 'Custom-Header: JSP-Demo' has been set.
Check browser dev tools (Network tab) to see it.</p>");
   %>
```

```
    <h2>3. Using 'session' Implicit Object</h2>
    <%
        Integer visitCount = (Integer) session.getAttribute("visitCount");
        if (visitCount == null) {
            visitCount = 0;
        }
        visitCount++;
        session.setAttribute("visitCount", visitCount);
    %>
    <p>Number of visits in this session: <%= visitCount %></p>
    <p>Session ID: <%= session.getId() %></p>
    <h2>4. Using 'application' Implicit Object</h2>
    <%
        synchronized (application) {
            Integer totalVisits = (Integer)
application.getAttribute("totalVisits");
            if (totalVisits == null) {
                totalVisits = 0;
            }
            totalVisits++;
            application.setAttribute("totalVisits", totalVisits);
        }
```

```
    %>
    <p>Total page visits (all users): <%=
application.getAttribute("totalVisits") %></p>
    <p>Server Info: <%= application.getServerInfo() %></p>
    <h2>5. Using 'out' Implicit Object</h2>
    <%
        out.println("<p>This line is written using the 'out' implicit
object.</p>");
        out.println("<p>Current Date and Time: " + new java.util.Date() +
"</p>");
    %>
</body>
</html>
```

Output:


JSP Implicit Objects Demo
localhost:8080/JSPImplicitO...
JSP Implicit Objects Demo
1. Using 'request' Implicit Object
Client IP Address: 0:0:0:0:0:0:0:1
Request Method: GET
Request URI: /JSPImplicitObjectsDemo/implicitObjectsDemo.jsp
2. Using 'response' Implicit Object
Custom header 'Custom-Header: JSP-Demo' has been set. Check browser dev tools (Network tab) to see it.
3. Using 'session' Implicit Object
Number of visits in this session: 1
Session ID: 768E26F7ABE85170350D9514FF71DDC0
4. Using 'application' Implicit Object
Total page visits (all users): 1
Server Info: Apache Tomcat/9.0.100
5. Using 'out' Implicit Object
This line is written using the 'out' implicit object.
Current Date and Time: Wed Apr 02 22:10:02 IST 2025


Project Explorer
JSPDemo
JSPImplicitObjectsDemo
Deployment Descriptor: JSPImplicitObjectsDem
JAX-WS Web Services
Java Resources
build
src
main
java
webapp
META-INF
WEB-INF
implicitObjectsDemo.jsp
Servers