Html file:

```
<html>
   <head>
      <title>WebSocket</title>
      <meta charset="UTF-8">
      <meta name="viewport" content="width=device-width, initial-scale=1.0">
   </head>
   <body>
      <textarea id="messages" rows="10" cols="45"></textarea>
      <br>
      <input id="msg" type="text" size="50"/>
      <input type="button" value="Send" onclick="send()"/>

      <script>
         var uri = "ws://localhost:8080/WebsocketEncoderDecoder/ChatServerEndpoint";
         var websocket = new WebSocket(uri);
         var textMsg = document.getElementById("messages");
         var inpMessage = document.getElementById("msg");

         function connect(){
            websocket.onopen = function(event){
              console.log("Connection open");
            };

            websocket.onmessage = function processMessage(chatMessage){
              var json = JSON.parse(chatMessage.data);
              textMsg.value += json.name + ': ' + json.message + "\n";
            };

            websocket.onclose = function(event){
              console.log("Connection closed");
            };
         }

         function send(){
            websocket.send(JSON.stringify({'inpMessage' : inpMessage.value}));
            inpMessage.value = "";
         }

         connect();
      </script>
   </body>
</html>
```

HelloWorldEndPoint.java

```
import java.util.logging.Level;
import java.util.logging.Logger;
import jakarta.websocket.*;
import jakarta.websocket.server.ServerEndpoint;

@ServerEndpoint(value = "/hello",
        decoders = {MessageDecoder.class},
        encoders = {MessageEncoder.class}
)
public class HelloWorldEndpoint {

    @OnMessage
    public Person hello(Person person, Session session) {
        if (person.getName().equals("bhavya")) {
            person.setName("Mr. Bhavya");
        }
        try {
            session.getBasicRemote().sendObject(person);
            System.out.println("sent ");
        } catch (Exception ex) {
            Logger.getLogger(HelloWorldEndpoint.class.getName()).log(Level.SEVERE, null,
ex);
        }
        return person;

    }

    @OnOpen
    public void myOnOpen(Session session) {
    }

}
```

MessageEncoder.java

```
import java.io.StringWriter;

import jakarta.websocket.EncodeException;
import jakarta.websocket.Encoder;
import jakarta.websocket.EndpointConfig;
import jakarta.xml.bind.JAXBContext;
import jakarta.xml.bind.Marshaller;

public class MessageEncoder implements Encoder.Text<Person> {

    @Override
    public String encode(Person object) throws EncodeException {
```

```
        JAXBContext jaxbContext = null;
        StringWriter st = null;
        try {
            jaxbContext = JAXBContext.newInstance(Person.class);

            Marshaller marshaller = jaxbContext.createMarshaller();
            st = new StringWriter();
            marshaller.marshal(object, st);
            System.out.println("OutGoing XML " + st.toString());

        } catch (Exception ex) {
            ex.printStackTrace();
        }
        return st.toString();
    }

    @Override
    public void init(EndpointConfig endpointConfig) {
        // do nothing.
    }

    @Override
    public void destroy() {
        // do nothing.
    }
}
```

MessageDecoder.java

```
import java.io.StringReader;

import jakarta.websocket.Decoder;
import jakarta.websocket.EndpointConfig;
import jakarta.xml.bind.*;


public class MessageDecoder implements Decoder.Text<Person> {

    @Override
    public Person decode(String s) {
        System.out.println("Incoming XML " + s);
        Person person = null;
        JAXBContext jaxbContext;
        try {
            jaxbContext = JAXBContext.newInstance(Person.class);

            Unmarshaller unmarshaller = jaxbContext.createUnmarshaller();

            StringReader reader = new StringReader(s);
            person = (Person) unmarshaller.unmarshal(reader);
```

```
        } catch (Exception ex) {
            ex.printStackTrace();
        }
        return person;
    }

    @Override
    public boolean willDecode(String s) {

        return (s != null);
    }

    @Override
    public void init(EndpointConfig endpointConfig) {
        // do nothing.
    }

    @Override
    public void destroy() {
        // do nothing.
    }
}
```

Person.java

```
class Person {
    private String name;

    public String getName() {
        return name;
    }

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

No. of times website visited:

```
<%! int count = 0; %>
<% count++; %>

<%@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>
      <h3>Number of times website visited : <%=count%></h3>
   </body>
</html>
```

Listener code:

NewServlet.java

```
import jakarta.servlet.ServletContext;
import java.io.IOException;
import java.io.PrintWriter;
import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import jakarta.servlet.http.HttpSession;

@WebServlet("/abc")
public class NewServlet extends HttpServlet {

    protected void processRequest(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");

        HttpSession httpSession=request.getSession(true);

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

        ServletContext context=httpSession.getServletContext();

        int count=(Integer)context.getAttribute("sessionCount");

        System.out.print(count);

        try (PrintWriter out = response.getWriter()) {
            /* TODO output your page here. You may use following sample code. */
            out.println("<!DOCTYPE html>");
            out.println("<html>");
            out.println("<head>");
            out.println("<title>Servlet NewServlet</title>");
            out.println("</head>");
            out.println("<body>");
            out.println("<h1>Servlet NewServlet at " + request.getContextPath() + "</h1>");
            out.println("====================================Session Created=="+
count);
            out.println("</body>");
            out.println("</html>");
        }
    }

    @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);
  }
}
```

SessionListenerDemo.java

```
import jakarta.servlet.ServletContext;
import jakarta.servlet.http.HttpSessionEvent;
import jakarta.servlet.http.HttpSessionListener;

public class SessionListenerDemo implements HttpSessionListener{

  public static int count=0;

  ServletContext context=null;

  @Override
  public void sessionCreated(HttpSessionEvent se) {
    count++;
    context=se.getSession().getServletContext();
    context.setAttribute("sessionCount", count);
    System.out.println("=====Session Object
Created============================");
  }

  @Override
  public void sessionDestroyed(HttpSessionEvent se) {
    System.out.println("=====Session Object
Created============================");
    count--;
  }
}
```