Assignment Rubrics	~~Marks~~ split up	Mark score	Total marks
Question-1: Tracking Session Data with Java Servlets	Code Implementation	8	8
	Session Data Accuracy	5	5
	Efficiency and clarity	3	3
	Explanation	4	4
Question-2: Using JSTL for Complex Problem Solving.	Scenario Explanation	6	6
	Function library Explanation	5	5
	custom Functions	5	5
	Clarity and organization	4	4
Question-3: Stock Market Page Auto-Refresh with JavaScript	Script functionality	8	8
	User Interaction Design	5	5
	Code efficiency	4	4
	Explanation	3	3
Question-4	Understanding Of WSDL	6	6
	Client code Generation	6	6
	Error Handling	4	4
	clarity & Depth	4	4

Assignment - 3

① Implementing a feature in a web application that tracks the number of accesses by a client within a single session using Java servlets using HttpSession object to manage and monitor session data.

Servlet Code:

```
import java.io.IOException;
import java.io.PrintWriter;
import java.util.Date;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

@WebServlet("/SessionTracker")
public class SessionTracker extends HttpServlet {
    protected void doGet(HttpServletRequest request,
            HttpServletResponse response) throws
            ServletException, IOException {

        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        HttpSession session = request.getSession(true);
        String sessionId = session.getId();
        long creationTime = session.getCreationTime();
        long lastAccessedTime = session.getLast
```

```
        AccessedTime();
    Integer visitCount = (Integer) session.get
            Attribute("visitCount");
    if (visitCount == null) {
        visitCount = 0;
    }
    visitCount++;
    session.setAttribute("visitCount", visitCount);
    out.println("<html><body>");
    out.println("<h1>Session Tracking Example</h1>");
    out.println("<p>Session ID: " + sessionId +
                                    "</p>");
    out.println("<p>Session Created: " + new Date
                        (creationTime) + "</p>");
    out.println("<p>Last Accessed: " + new Date
                        (lastAccessedTime) + "</p>");
    out.println("<p>No. of accesses in this session: " +
                        visitCount + "</p>");
    out.println("</body></html>");
  }
}
```

Output:

Session Tracking Example

Session ID: 12345ABCDE

Session Created: Mon Sep 09 12:00:00 IST 2024

Last Accessed: Mon Sep 09 12:01:05 IST 2024

No. of accesses in this session: 1

(2) write a JSTL to so how you wer function libr custom fun

JSP code

```
<%@ taglib

<%@ taglib

<html>
<head>
    <title>
</head>
<body>
    <h2>
    <form
```

(2) Write a scenario where you had to use JSTL to solve a complex problem and how you went about it. Also, elaborate the function library in JSTL and how to create custom functions.

JSP code Using JSTL

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/
        core" prefix="c" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/
        functions" prefix="fn" %>

<html>
<head>
    <title>Order Management</title>
</head>
<body>
    <h2>Order List</h2>
    <form method="GET" action="order.jsp">
        <label for="status">Filter by Status:</label>
        <select name="status" id="status">
            <option value="All">All</option>
            <option value="Delivered">Delivered
                </option>
            <option value="Pending">Pending
                </option>
        </select>
        <input type="submit" value="Filter">
    </form>
    <table border="1">
        <thead>
            <tr>
                <th>Order ID</th>
```

```
        <th> Date </th>
        <th> Status </th>
        <th> Amount </th>
    </tr>
</thread>
<tbody>
    <c:for each var="order" items="${orders
        <c:choose>
            <c:when test="${param.status=='All
                ||order.status==param.status}
                <tr>
                    <td>${order.id}</td>
                    <td>${order.date}</td>
                    <td>${order.status}</td>
                    <td>${order.amount}</td>
                </tr>
            </c:when>
        </c:choose>
    </c:forchoose>
    <c:for each>
</tbody>
</table>
</body>
</html>
```

Output:

Order List

Order ID	Date	Status	Amount
1002	2024-09-08	Pending	150.00
1003	2024-09-09	Pending	300.00

JSTL Fun

* creating C

<taglib xmlns

<tlib-v

<short-

<uri>h

<funct

<

<f

<f

</fun

</taglib>

Output:

Custom

Original

Reversed:

(3) To in

for refr

every fi

dialog

refresh.

JSTL Function Library (fn)

(*) Creating Custom Functions in JSTL

```
<taglib xmlns="http://java.sun.com/xml/ns/javaee"
                                    version="2.1">
    <tlib-version>1.0</tlib-version>
    <short-name>custom</short-name>
    <uri>http://example.com/custom</uri>
    <function>
        <name>reverseString</name>
        <function-class>com.example.CustomFunc-
                                tions</function-class>
        <function-signature>java.lang.String
                reverseString(java.lang.String)
                </function-signature>
    </function>
</taglib>
```

Output:

```
Custom Function Example
Original: Hello World
Reversed: dlroW olleH
```

(3) To implement the described functionality for refreshing a stock market quotes page every five minutes, with a confirmation dialog appearing 20 seconds before the refresh.

Javascript Code snippet:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Stock Market Quotes</title>
    <script>
        function refreshPage(){
            location.reload();
        }
        setTimeout(()=>{
            const confirmRefresh= confirm("The Pag
              will refresh in 20 seconds.");
            if(!confirmRefresh){
                refreshPage();
            }else{
                alert("Page refresh canceled.");
            }
        }, 20000);
    </script>
</head>
<body>
    <h1>Stock Market Quotes</h1>
</body>
</html>
```

output:

Page Displa

Sto
- Apple In
- Microsoft
- Alphabe

confirmatio

The Page

Alert

Page

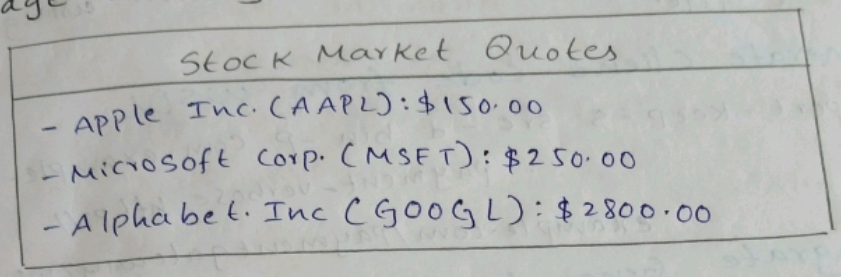
Page Ref

(The Pag

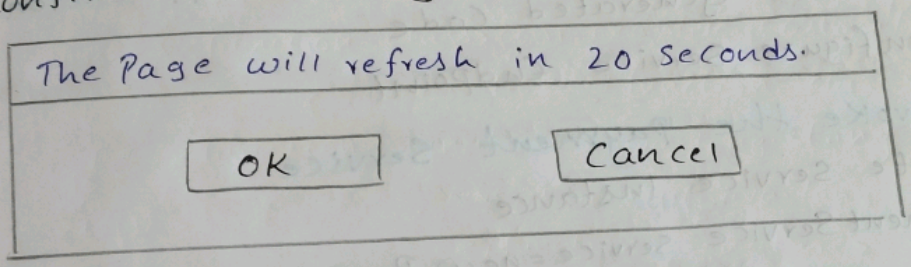
- Apple
- Microso
- Alpha

output:

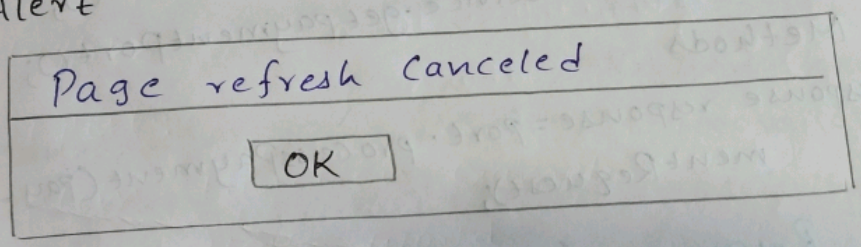
Page Display



confirmation Dialog

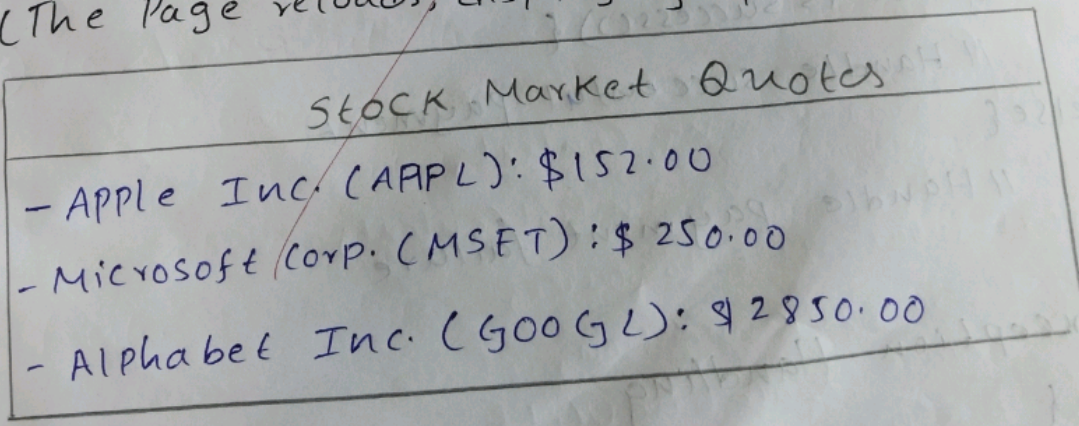


Alert



Page Refresh

(The Page reloads, displaying updated content)



④ To integrate an external payment gateway service into your e-commerce application using a WSDL file.

① Generate Cliend code from WSDL

wsimport -keep -s src -d bin -p com.example.payment -verbose http://example.com/paymentgateway?wsdl

② Integrate Generate Code into Application

- Include Generated Code
- Configure Service Endpoint

③ Invoke the Payment Service

- Create service instance

```
PaymentService service = new PaymentService();
paymentPortType port = service.getpaymentPort();
```

- Invoke Methods

```
paymentResponse response = port.processPayment(PaymentRequest);
```

④ Handle Response and Errors

- check Response

```
if(response.isSuccess()){
    // Handle successful payment
}else{
    // Handle payment failure
}
```

- Exception Handling

```
try{
    PaymentResponse response = port.processPayment
```

```
}catch(SOAP
    // Handl
}catch(webs
    // Handle
}
```

output:

succesful
payme
Payment F
Payme

SOAP fau
paymen

Connectivi
paymen

(PaymentRequest);

} catch (SOAPFaultException e) {
    // Handle SOAP faults (e.g., Invalid request)
} catch (WebServiceException e) {
    // Handle connectivity or configuration errors
}

output:

succesful Payment:

Payment succesful. Transaction ID: 1987653210

Payment Failure (e.g., Invalid Card Details):

Payment failed. Error: Invalid Credit Card Details.

SOAP fault (e.g., Invalid Request)

payment failed due to a SOAP fault: Invalid Request format.

Connectivity issue (eg: Service Unavailable)

payment failed due to a Connectivity issue: Cannot connect to the payment gateway.