27/8/24

SET-2

ASSIGNMENT-1

1. You are updating a legacy web application to take advantage of modern web standards. The project involves transitioning from an earlier HTML version to HTML5. To goal is to leverage HTML5's new features to improve web development practices and enhance the functionality of web forms.

* **Update the Doctype:** Ensure your HTML documents start with the HTML 5 doctype to leverage HTML5 features:

  <!DOCTYPE html>

* **Use New Semantic Elements:** Replace generic and tags with HTML5 semantic elements to improve readability & accessibility.
  * <header> for headers
  * <nav> for navigation skills
  * <section> for sections
  * <article> for self-contained content
  * <aside> for side content
  * <footer> for footers
  * <main> for primary content

* **update Forms with New Input Types:** HTML5 introduces new input types that enhance user experience and validation:
  * <input type = "email"> for email address
  * <input type = "url"> for URLs
  * <input type = "date"> for dates
  * <input type = "number"> for numeric input

* **Add Form Attributes:** utilize new attributes to improve form functionality:
  * required to make fields mandatory
  * placeholder to provide hint text
  * pattern for custom validation with regex
  * auto complete to suggest or remember inputs.
* **Implement New Form Elements:** Take advantage of new form elements:
  * <datalist> to provide options for an <input>
* **Improve Accessibility:**
  * ARIA Roles and Attributes: Use ARIA roles and attributes to improve accessibility

Example for Transitioned Form:

```
<form action="/submit"
    method="post">
    <div>
        <label
for="username">Username:</label>
        <input type="text"
id="username" name="username"
placeholder="Enter your username"
required>
</div>
<div>
<input type="submit"
value="submit">
</div>
</form>
```

Output:

welcome to our website

Name	Birthday Date

2. Designing a user registration form for a new web application. The form needs to capture essential information such as name, email, and a message from users. It is crucial to implement client-side validation to ergure that the data entered is accurate and complete before the form is submitted.

HTML structure:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport"
content="width=device-width, initial-
    scale=1.0">
    <title>user Registration From </title>
    <style>
        body {
            font-family: Arial,
sans-serif;
            max-width: 600px;
            margin: auto;
        }
        form{
            display: flex;
            flex-direction:
column;
        }
        label{
            margin: 10px 0 5px;
```

```
}
input, textarea {
    padding: 8px;
    margin-bottom: 10px;
    border: 1px solid #ccc;
}
button {
    padding: 10px;
    border: none;
    border-radius: 4px;
    background-color:
#4CAF50;
    color: white;
    cursor: pointer;
}
#45a049;
}
</style>
<h1> User Registration </h1>
<form id="registrationForm">
    <label for="name"> Name: </label>
document.getElementById('registrationForm').addEventListener('submit', function(event){
    var name=document.getElementById
if (!name||!email||!message){
    alert('All fields are required.');
}
</script>
</body>
</html>
```

output

User Registration		
Name	Mail id	MSG

3. Designing a website that needs to function effectively across various devices and screen sizes. The design should include different types of layouts such as fixed, fluid and responsive

To achieve this, you need to use CSS techniques like flexbox and maintains a consistent user experience

* Fixed Layout:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport"
content="width=device-width, initial-scale=1.0">
    <style>
        body {
            margin:0;
            Padding:0
    <div class="container">
        <div class="header">Fixed Layout
            Header</div>
        <div class="main">Fixed width
            Content</div>
    </body>
    </html>
```

* Fluid Layout:

```
<!DocTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport">
}
.main{
    padding:20px;
}
```

```
</style>
</head>
<body>
    <div class="container">
        <div class="head">Fluid layout header
    </div>
</body>
</html>
```

Output:

Responsive web design

This is the main content area.

By incorporating these CSS techniques, we can create a website with fixed, fluid, and responsive layouts that adapt to various devices & screen sizes, user-friendly experience

Assignment-2:

4. To Create a webpage that offers a seamless user experience across devices, responsive design principles must be applied. This involves using fluid grids, flexible images and media queries to engure the layout adjusts gracefully to different screensizes. Additonally, touch-friendly elements and optimized images engure fast loading times and a smooth experience on all devices.

Fluid Grids: percentage-based widths

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name = "viewport"
content="width= device - width,
```

```
initial-scale=1.0">
<title> Responsive layout with Fluid Grids
</title>
<style>
    body {
        margin:0;
        font-family: Arial;
box-sizing: border-box;
    }
    @media (max-width: 768px) {
        .container {
grid-template-columns: 1fr;
        }
    }
</div> class="container">
    <div class="item"> content 1 </div>
    <div class="item"> content 4 </div>
</div>
</body>
</html>
```

Flexible Images:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport"
content="width=device-width, initial-scale
=1.0">
</style>
</head>
```

```
<body>
    <img src = "sample.jpg"
    alt = "Example Image">
    </body>
    </html>
```

output:

Responsive web design

Screen size for optimal experience.

Use fluid grids for flexibility, media queries for responsive design, and techniques like flexible images and touch-friendly elements to enhance usability and performance across devices.

5. Given a scenario (e.g., creating a blog post, a product listing), design a webpage using appropriate elements, table, lists, and images for optimal readability and user experience (HTML, CSS, JavaScript, Images).

Webpage Design: product listing:

Sample HTML layout that includes product images, descriptions, and pricing, laid out with tables and lists where appropriate.

HTML Structure:

```
<!DOCTYPE Html>
<Html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name = "Viewport"
content = "width=device-width, initial-scale
    <title> Product listening </title>
```

Assignment split up:

		Marks obtained	Total Marks
1.	Form Design	4	4
	Button Design	4	4
	Features of HTML	2	2
2.	Form Design	4	4
	Validation	4	4
	Layout	2	2
3.	Layout design	4	4
	CSS Background	6	6
4.	Form design	4	4
	Layout design	4	4
	Navigation Menue	2	2
5.	Form design	4	4
	CSS Layout	4	4
	Java Script Validation	2	2

ASSIGNMENT-3

1. Implementing a feature in a web application that tracks the number of access by a client within a single session. Their session and retrieve information about the session, such as the session ID, creation time, and last accessed time

```
import
javax.servlet.ServletException;
import
javax.servlet.annotation.WebServlet;
import
javax.servlet.http.HttpServlet;
import
javax.servlet.http.HttpServletResponse;
import
java.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpSession;
import java.io.IOException;
import java.io.PrintWriter;
Public class TrackSessionServlet extends
                HttpServlet{
@Override
protected void
doGet(HttpServletRequest request, Http
ServletResponse response)
throws ServletException,
IOException{
```

```
response.setContentType("text/html");
HttpSession session= request.getSession(true);
Integer accessCount=(Integer)
Session.getAttribute("accessCount");
if(access count==null){
    accesscount=0;
}
accessCount++;
String sessionId = session.getId();
long creationTime= long lastCreationTime();
Session.getAccessedTime();
Printwriter out = response.getwriter();
out.println("<html><body>");
   out.println("<h2>session Tracking</h2>"
ID:" + sessionId+ "</p>");
Time:"+new
Accesses:"+accessCount+"</p>");
out.println("</body></html>");
}
```

Output:

session ID: 1234567890abc

 creation Time: Mon Sep 10 15:20:30 2024

 last accessed time: Mon Sep 10 15:25:10 2024

 Number of Accesses: 5

2. Write a Scenario where you had to use JSTL to solve a complex problem and how you went about it. How to create custom functions.

Scenario Using JSTL:

In a web application, you have a requirement to display a list of products with varying categories.

The product list needs to be dynamically categorized into seperate sessions on a web page.

Solution Using JSTL:

```
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import java.io.IOException;
import java.util.List;
import java.util.Map;
import java.util.HashMap;
Public class
ProductListservlet extends HttpServlet {
    protected void
doGet (HttpServletRequest request, HTTPServlet)
product ("Laptop", "Electronics", 1000, true),
        new
product ("shirt", "clothing", 30, true),
        new
```

```
Product ("washing Machine", "Home Appliances", 500,
                                    false
Product ("shirt", "clothing", 30, true),
);
ProductList.jsp").forward (request, response);
}
}
```

Output:

Laptop - $1000 - Available

<clothing> shirt - $30 - Available

<Homeappliances> washing machene - $500 - out of stock

3. A page of stock market quotes uses script to refresh the page every five minutes in order to ensure the latest statistics remain available.

To achieve this setup the HTML page, Add Javascript for Automatic Refresh and Confirm Dialog.

HTML and Javascript code:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
```

```
<meta name = "viewport"
Content = " width = device-width, initial-
                scale = 1.0">
<title> Stock Market Quotes </title>
<script>
    Var user Response = confirm("The
page will refresh in 20 seconds. Do you
need more time?");

    if (! user Response) {

    window.location.reload();
    } else {
      setTimeout (refresh page, 30000);
    }
  }, 20000);
window. location. reload();
  }, refresh Interval);
<h1> stock Market Quotes </h1>
<p> Here you can display real-time
Stock market quotes. </p>
</body>
</html>
```

Output:

Page Display:

The page will refresh in 20 seconds.
Do you need more time?

[cancel] [OK]

4. You are developing an e-commerce application that needs to integrate with an external payment gateway service. Describe the steps involved in generating the client code, invoking the service and handling any potential errors.

Steps to integrate a payment Gateway using WSDL:

For Java:

```
import your.package.name.paymentService;
import.package.name.paymentService port
                                    type;
Public class paymentclient {
    Public static void main (String[] args) {
PaymentService service = new payment
                        Service();
String response = Port.Process payment
("amount", currency", "payment Details");
System.out.println (" Response : "+response);
    }
}
```

Output:

Payment Response; payment successful for amount 100.00 USD.

ASSIGNMENT-4

1. From a developer's perspective, discuss why JDBC is essential in building database-driven applications. Provide examples of executing SQL queries using JDBC statements.

Configure Data source in context.xml :

```
<context>
    <Resource name = "jdbc/myDataSource"
auth = "container"
type = "javax.sql.DataSource"
                maxTotal = "20"
                maxIdle = "10"
maxWaitMillis = "10000"
Username = "dbuser"
Password = "dbpassword"
driverclassName = "com.mysql.cj.jdbc.Driver"
url = "jdbc:mysql://localhost:3306/mydatabase"
/>
</context>
```

Collable statement :

```
import java.sql.CollableStatement;
import java.sql.connection;
import java.sql.SQLException;
import java.sql.Types;
public class collablestatementExample{
Public void callstoredprocedure(int employeeId){
    String sql = "{call getEmployeeName
                    (?, ?)}";
```

```
try(Connection conn = DatabaseUtility.
        getConnection();
    employeeName = cstmt.getString(2);
    System.out.println("Employee Name:" +
                        employeeName);
} catch (SQLException e){
    e.printStackTrace();
}
}
```

Output:

Employee ID : 1, Name: John
Employee ID : 2, Name: Jane
Employee ID : 3, Name: Emily
Rows updated: 1
Employee Name : John

2. Describe the life cycle of phases of a JSP page. Explain the significance of each phase in the JSP execution process. Discuss the different ways to embed Java code within a Java JSP pages.

Life cycle of phases of a JSP page:

1. Translation phase:

Description: The JSP page is translated into a Java servlet by the JSP engine (e.g., HTML mixed with JSP tags)

1. Significance: This phase ensures that the JSP content is converted int a form that the Java Servlet container can execute.

2. Compilation phase:

Description: The Java source code generated from the translation phase

Significance: Compilation ensures that the JSP is converted into executable.

3. Initialization phase:

Description: The servlet container initializes the servlet instance.

Significance: Initialization sets up any resources the JSP might need such db.

4. Requesting processing phase:

Description: The servlet process incoming client requests by calling the service().

Significance: This phase is where the dynamic content generation occurs.

5. Destroy phase:

Description: The servlet container destroys the servlet instance the destroy()

Significance: This phase ensures that resources are properly released.

3. You need to develop a php program that generates a chessboard using HTML tables. The table should have a total width of 30 px. provide the code for this programs.

PHP code:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport"
content="width=device-width, initial-scale
<title>chessboard</title>
<style>
    table{
        border-collapse: collapse;
        width:400px;
        height:400px;
    }
td{
    width:30px;
    height:30px;
for($row=0; $row<8;$row++){
    echo "<tr>";
    for($col=0;$col<8;$col++){
        echo "</tr>";
    }
?>
</table>
</body>
</html>
```

Output:

[W][B][W][B][W][B][W][B]
[B][W][B][W][B][W][B][W]
[W][B][W][B][W][B][W][B]

[B] [W] [B] [W] [B] [W] [B] [W]
[W] [B] [W] [B] [W] [B] [W] [B]
[B] [W] [B] [W] [B] [W] [B] [W]
[W] [B] [W] [B] [W] [B] [W] [B]
[B] [W] [B] [W] [B] [W] [B] [W]

* W represents a white cell
* B represents a black cell

4: You are developing a PHP application that reads content from a text file and uses regular expressions to extract specific patterns, such as email address and phone numbers. Describe the steps & provide code for the application.

Php code for the Application:

```
<?php
$textFilepath = 'input.txt';
$xmlFile path = 'output.xml';
$ textcontent = file-get_contents($textFile path);
$ email pattern = '/ [a-zA-Z0-9.-/+-]+@[a-zA-Z
0-9.-]+.[a-zA-Z]{2,4}/';
$ phone pattern = '/\b\d{10}\b/';
Preg_match_all ($email pattern, $textcontent,
                $emails);
Preg-match-all ($ phone pattern, $ text content,
                $phones);
$xml = new
simpleXMLElement ('<Data/>');
$ EmailsElement = $xml -> add child('Emails');
```

```
$xml = new SimpleXMLElement('<?xml version="1.0"?><data></data>');
$phones = $xml->addChild('phoneNumbers');
```

```
echo "Data extracted and saved to XML file successfully.";
?>
```

Output:

Email : support@example.com

Phone Numbers: 1234567890.

	Assignment-3	Split up	Total marks
1.	code implementation	8	20
	Session data accuracy	5	
	Efficiency & clarity	3	
	Explanation	4	
2.	Scenario explanation	6	20
	Function	5	
	custom	5	
	clarity & organization	4	
3.	Script	8	20
	Interaction Design	5	
	code efficiency	4	
	Explanation	3	
4.	WSDL	6	20
	code	6	
	Error	4	
	Depth	4	

Assignment-4

		Split up	Total
1	Explanation of JDBC	5	20
	Connection Pooling	6	
	SQL Queries	5	
	Statement type	4	
2.	Life cycle	6	20
	Java code	5	
	ADV & DS	5	
	Depth	4	
3	Code	8	20
3.	HTML Table	5	
	Logic cells	4	
	Explanation	3	
4.	Code	8	20
	Pattern extraction	5	
	XML File	4	
	DTD VS XML	3	