:ETHA SCHOOL OF ENGINEERING
SIMATS, CHENNAI

Engineer to Excel

SET -2

:y web application to take advantage s. The project involves transitioning version to HTML5. The goal is to features to improve web development ıe functionality of web forms.	10	CO1	2
ation form for a new web application. apture essential information such as nessage from users. It is crucial to validation to ensure that the data complete before the form is submitted.	10	CO1	3
hat needs to function effectively across creen sizes. The design should include outs such as fixed, fluid, and responsive. ıeed to use CSS techniques like Flexbox ıat the layout adapts well to different ıtains a consistent user experience.	10	CO1	2
: that offers a seamless user experience ınsive design principles must be applied. fluid grids, flexible images, and media ıe layout adjusts gracefully to different stance, on a desktop, the webpage might lumns with detailed content, while on a ne content could stack vertically for easy ion menus should transform into a ı smaller screens, ensuring they remain taking up too much space. Additionally, nents and optimized images ensure fast ı smooth experience on all devices.	10	CO1	3
e.g., creating a blog post, a product listing), : using appropriate elements, tables, lists, timal readability and user experience. Also, step the sequence of HTTP requests and ccur when a user accesses a webpage ple resources (HTML, CSS, JavaScript,	10	CO1	3

main aside" footer footer 1fr 2fr

Abend!

1) HTML5 user Registration

```
<form>
  <input type="text"
  name="name" required pattern="[A-Za-z ]+" title="Only letters and spaces allowed"
  placeholder="Name">
  <input type="email" name="email" required placeholder="Email">
  <textarea name="message" required placeholder="Message"
  "></textarea>
  <button type="submit">Register</button>
</form>
```

2) Responsive Web Layout Using CSS Grids

```
<div class="grid">
  <header>Header</header>
  <nav>Nav</nav>
  <main>Main</main>
  <aside>Aside</aside>
  <footer>Footer</footer>
</div>
```

```
<style>
  .grid { display: grid; gap: 10px; grid-template: "header
  header" "nav main" "aside aside" "footer footer"; }
  @media (min-width: 768px) { .grid { grid-template:
  "header header" "nav main" "footer footer" / 1fr 3fr; } }
  @media (min-width: [illegible]px) { .grid { grid-template:
  "header header header" "nav main aside" "footer footer 1fr 2fr
```

3) Responsive Design with Fluid Grids with Media Queries

```
<div class="content">
  <h1>Blog Title</h1>
   <img src="image.jpg" alt="Image">
<p> Blog content ... </p>
</div>
<style> .content { max-width: 100%;
margin: auto; padding: 10px; }
  img { max-width: 100%; height: auto; }
  @ media (max-width: 600px)
  { .content { padding: 5px; } }
  </style>
```

ASSIGNMENT-2

4) Minimal HTTP Request / Response Example

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport"
  content="width=device-width, initial-scale=1.0">
<title> Responsive Page </title>
<link rel="stylesheet" href="styles.css">
</head>
<body>
  <header>
    <nav>
```

Logo

<ul class="nav-links">

Home

About

About

Contact

</nav>

</header>

<main>

<h1>Welcome</h1>

This page is responsive.

<img src="image.jpg" alt="Sample Image">

</main>

<footer>

© 2024 Your site

</footer>

</body>

</html>

5) Blog Post

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport"

content="width=device-width, initial-scale=1.0">

<title>Blog Post</title>

```
<link rel="stylesheet" href="styles.css">
</head>
<body>
<header>
<h1>My Blog</h1>
<nav>
<ul>
<li><a href="#">Home</a></li>
<li><a href="#">Posts</a></li>
<li><a href="#">Contact</a></li>
</ul>
</nav>
</header>
<main>
<h2>Blog Post Title</h2>
<p>Published on August</p>
<img src="image.jpg" alt="Blog Post Image">
<p>This is the content of the Blog post.</p>
</main>
<footer>
<p>&copy; 2024 My Blog</p>
</footer>
</body>
</html>
```

Set-2 output Answers for Assignment 1

1) Html user reg. form with client side validation

Output

* Displays the website title Welcome to our site
* Contains "contact us" form with Name, email, message, and submit button

2) Designing a user's registration form

o/p

* The form captures user's name, email, message
* Uses HTML5 validation, ensures all fields are filled out correctly before submission
* If valid, a success message is shown

3) Designing a Website for Multiple Devices

o/p

* On large screen, Webpage displays a header, a navigation menu, on the left, content area on the right.
* On smaller screen, the menu & content area stack vertically, ensuring readability & usability.

4) Applying Responsive Design Principles

O/P

* Webpage uses a responsive grid layout that adjust based on screen size
* Menu changes from hori. layout to vert. layout on smaller screens.
* Images are flexible, ensuring they fit within containers.

5) Designing Webpage

O/P

* Display:
  - → Header, contains blog title & nav. menu.
  - → Main content, displays post title, text, list of trends & image
  - → Footer, ⊖ fixed footer with copyright info.

Asmt 3

1) Java servlet: Tracking client Accesses in a session

```
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
                                    Response
                                    HttpSession;
import java.io.IOException;
public class AccessTrackingServlet extends HttpServlet{
  @override protected void
doGet(HttpServletRequest, request, HttpServlet
throws ServletException, IOException {          Respons
HttpSession session = request.getSession(true);
String sessionId = session.getId();
long creationTime = session.getCreationTime();
response.setContentType("text/html");
response.getWriter().println("Creation Time: "+Time+(
}
}
```

O/P

Session Info
Session ID : <session-id>
Creation Time: <creation-time>
Last Accessed Time:
<Last-accessed-time>
Access Count:
<number-of-accesses>

ex Session info
ID : 123ABC
Time: 16443
Last Acc. Time: 1642
Access Count: 5

2) JSTL: Steps: & o/p

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/cor
prefix="c" %>
<%@ page import "java.util.List" %>
<html>
<body>
  <h2> Product List </h2>
  <c:forEach var="product" items="${products}">
  <p> ${product.name} - ${product.price} </p>
  </c:forEach>
  </body>
  </html>
```

o/p

P1 : Name: "Laptop" Price: 100000

P2 Name = "Smartphone" Price = 10000

3)

```
<!DOCTYPE html>
<html>
<head>
    <title>Stock Quotes</title>
    <script>
    let countdownTimer;
    let refreshTime= 300000;
function startCountdown() {
  countdownTimer: setTimeout(function() {
if (confirm("refresh")) {
start Countdown();
setTimeout(refresh);
</script>
</head>
<body
<h1>
</body> </html>
```

o/p → payer will display stock Market header
* Page will refresh in every 5 mins.
* 20 sec before, a confirmation dialog will appear
* if clicks ok, countdown continues, else cancel

4) WSDL

import com.example.payment.PaymentService;
import com.example.payment.PaymentServicePortType;
public class PaymentClient {
    public static void
main(String[] args) {
    PaymentService service = new PaymentService();
try {
String response = port.processPayment("12345", 100.0);
System.out.println("Response: " + response);
} catch (Exception e) {
e.printStackTrace();
}}}

o/p

Response: Payment processed successfully.

incase of error:

java.lang.Exception: Error message from service
at
com.example.payment.PaymentClient.main
(PaymentClient.java:10)

Asmt 4

1) JDBC Connection Pooling, Executing sql using JDBC

```
import java.sql.Connection;
import java.sql.DriverManager;
   "      "    ResultSet;
   "      "    SQLException
   "      "    Statement;

public class Statement {
   public static void main(String args[]).
   String url: "jbc:mysql://localhost:3306/mydb";
   String user: "dbuser";
   String password: dbpass
  try( Connection: DriverManager)
  System.out.println("ID: " + rs.getInt("id"));
  } catch (SQLException) {
  e.printStackTrace();
     }
     }
```

o/p

ID: 1 : Username: alice

ID : 2 Username: bob

ID: 3, Username: Charlie

2) lifecycle phase

```
<% String message = "Hello";
   out.println(message);
%>
```

o/p

Hello

Expressions:

```
<%= "Hello" %>
```

o/p Hello

3) Chessboard

```
<!DOCTYPE html>
<html>
<head>
<style>
table {
    width: 400px;
    height: 400px;
    border-collapse: collapse; }
td {
    width: 30px;
    height: 400px;
}
.black {
    background-color: black;
}
.white { background-color: white;
</style>
</head
<body>
```

```
<?php
 echo '<table>';
 for ($row=0; $row<8; $row++){
  echo '<tr>';
  class='$color'></td>"; }
    echo '</tr>';
  } echo '</table>';

 ?>
</body>
</html>
```

o/p

Html table representing a chessboard with black & white cells.

4) Application to Extract Data & Create XML

```
<?php
$text = file_get_contents ('data.txt');
$emailPattern = '/[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}/';
$phonePattern = '/\+?[0-9]{10,15}/';
preg_match_all ($emailPattern, $text, $emails);
preg_match_all ($phonePattern, $text, $phones);
$xml = new
SimpleElement ('<root/>');
```

```
$emailsElement =
$xml->addChild('emails');
foreach($emails[0] as $email)

$xml->asXML('output.xml');
?>
```

O/P An XML file called "output.xml" with struct.

```
<root>
  <emails>
<email>example@example.com</email>
 <email>test@test.com</email>
    </emails>
    <phones>
      <phone>+1234</phone>
      <phone>9876</phone>
    </phones>
  </root>
```

Rubrics Asmt 3.

1)	Code implementation (8)	____
	Data Accuracy (5)	____
	Efficiency & clarity (3)	____
	Explanation (4)	____

2)	Explanation (6)	____
	Function exp (5)	____
	Custom fun. (5)	____
	Clarity & org (4)	____

3)	Script func. (8)	___
	Interaction Design (5)	___
	Code Efficiency (4)	___
	Exp (3)	___

4)	Understanding (6)	___
	Code Generation (6)	___
	Error Handle (4)	___
	Clarity (4)	___

Batch Rubrics

1.	Explanation (5)	___
	Pooling (5)	___
	Sql Queri (5)	___
	Statement Type (4)	___

4) Code (8)	___
Extraction (5)	___
File Generation (4)	___
Comparison Schema (3)	___

2.	Explanation (6)	___
	Java Code (5)	___
	Adv & Disad (5)	___
	Clarity (4)	___

3.	Code (8)	___
	Structure (5)	___
	Logic (4)	___
	Exp (3)	___