Assignment-1

Q-4	Form design	4
	Layout design	4
	Navigation menu	2
Q-5	Form design	5
	css layout	5

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 ensure the layout adjusts gracefully to different screen sizes. For instance, on a desktop, the webpage might display multiple columns with detailed content, while on a smartphone, the same content could stack vertically for easy scrolling. Navigation menus should transform into a hamburger icon on smaller screens, ensuring they remain accessible without taking up too much space. Additionally, touch-friendly elements and optimized images ensure fast loading times and a smooth experience on all devices.	10
5	Given a scenario (e.g., creating a blog post, a product listing), design a webpage using appropriate elements, tables, lists, and images for optimal readability and user experience. Also, describe step-by-step the sequence of HTTP requests and responses that occur when a user accesses a webpage containing multiple resources (HTML, CSS, JavaScript, images).	10

4. To create a webpage that ensures a seamless user experience across devices, responsive design principles must be applied.

Fluid grid: Fluid grid use percentage-based widths rather than fixed pixel values. This allows your layout to adapt to different screen sizes gracefully.

Example: css

```
.container{
      display: grid;
      grid-template-columns: repeat (300px, 1fr));
      gap: 20px;
      padding: 20px;
   }.
```

Flexible images: Images should resize according to the container they're in, maintaining their aspect ratio whi adapting to various screen size.

Example: css

```
   img {
      max-width: 100%;
      height: auto;
      }.
```

Media Queries: media queries are essential for adjusting the layout based on the device's screen size.

Example: css

```
   @media (max-width: 768px){
      .container{
            grid-template-columns: 1fr;
      }
```

4. 4. Responsive Navigation menus: One smaller screens, it's crucial to transform navigation menus into a more space efficient design, like a hamburger menu.

Example: html

```
<nav class= "nav bar">
    <div class= "hamburger" onclick= "toggleMenu()"> &#9776;</
    div>
    <ul class = "menu">
     <li><a href = "#"> Home </a></li>
     </nav>.
```

5. JavaScript for Toggling:

Javascript

```
        function toggleMenu(){
    document.query selector ('.menu'). classlist. toggle ('active');
    }.
```

6. Touch friendly elements; Ensure that button & links are large enough to be easily tapped on touch devices.

Example: css

```
    button. menu a {
      padding: 10 px  20px;
      font - size: 16px;
      }.
```

7. Performance Optimization:

For a smooth experience, prioritize fast loading times by optimizing images minifying CSS and Javascript and using lazy loading for images and other media.

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

Layout & structure:

Header, Main content Area, sidebar, Footer.

HTML structure:

html

```
<!DOCTYPE html>
<html lang="en">
 <head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-
   width, initial-scale=1.0">
   <title> Responsive 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="#">about</a></li>
           <li><a href="#"> Blog</a></li>
           <li><a href="#"> contact</a></li>

              </ul>
           </nav>
           </header>
```

```
<div class="content">
<article class="post">
<h2> understanding css grid layout</h2>
<p> by Jane Doe | August 27, 2024 </p>
<img src="css-grid.png" alt ="css grid eg">
<p> The css grid layout is a two dimensional</p>

<table>
<caption> comparsion of css layout techniques</caption>

<thead>
  <tr>
 <th> feature</th>
 <th> Flexbox </th>
 <th> Grid </th>

 </th>
 </thead>
 <tbody>
    <tr>
     <td> 1D/ 2D layout </td>
    <td> 1D </td>
    <td> 2D </td>

     </tr>
     <tr>
       <td> use

  @media (max-width: 768px){
      .content {
        flex-direction: column;
      }
      .post {
       } margin-right: 0;
```

Css fo

Css

body

f

Sans

CSS for styling:

```
CSS
  body{
     font-family: Arial,
 Sans,-serif;
        line-height: 1.6;
        margin: 0;
        padding: 0;
         color: #333;
     }
     header{
          background-color: #333;
          color: white;
          padding: 20px 0;
           text-align: center;
       }
         nav ul{
                list-style-type; none;
                padding: 0;
             }
               nav ul li{
                     display: inline;
                     margin: 0 10px;
                  }
                   nav ul li a{
                      color: white;
```

```
.content {
    display : flex;
    flex-wrap: wrap;
    padding: 20px;
}
 .post{
     flex: 3;
      margin-right: 20px;
   }.
 .sidebar{
        flex: 1;
 }
 .sidebar h3{
         margin-top: 0;
 }
table {
     width: 100%.
     border-collapse: collapse;
     margin : 20px 0;
   }
  @media (max-width: 768 px){
      .content {
           flex-direction: column
        }
       .post {
            margin-right: 0;
         }
       }
```

Javascript for interactivity:

Javascript

```
//Example: Add interactivity or track events
document. addEventListener ('DOMContentLoaded', () =>
{
    console.log ('page loaded');
});
```

[illegible]