# Module 3 - CSS

## Ans (1) :-

## What is a CSS selector:-

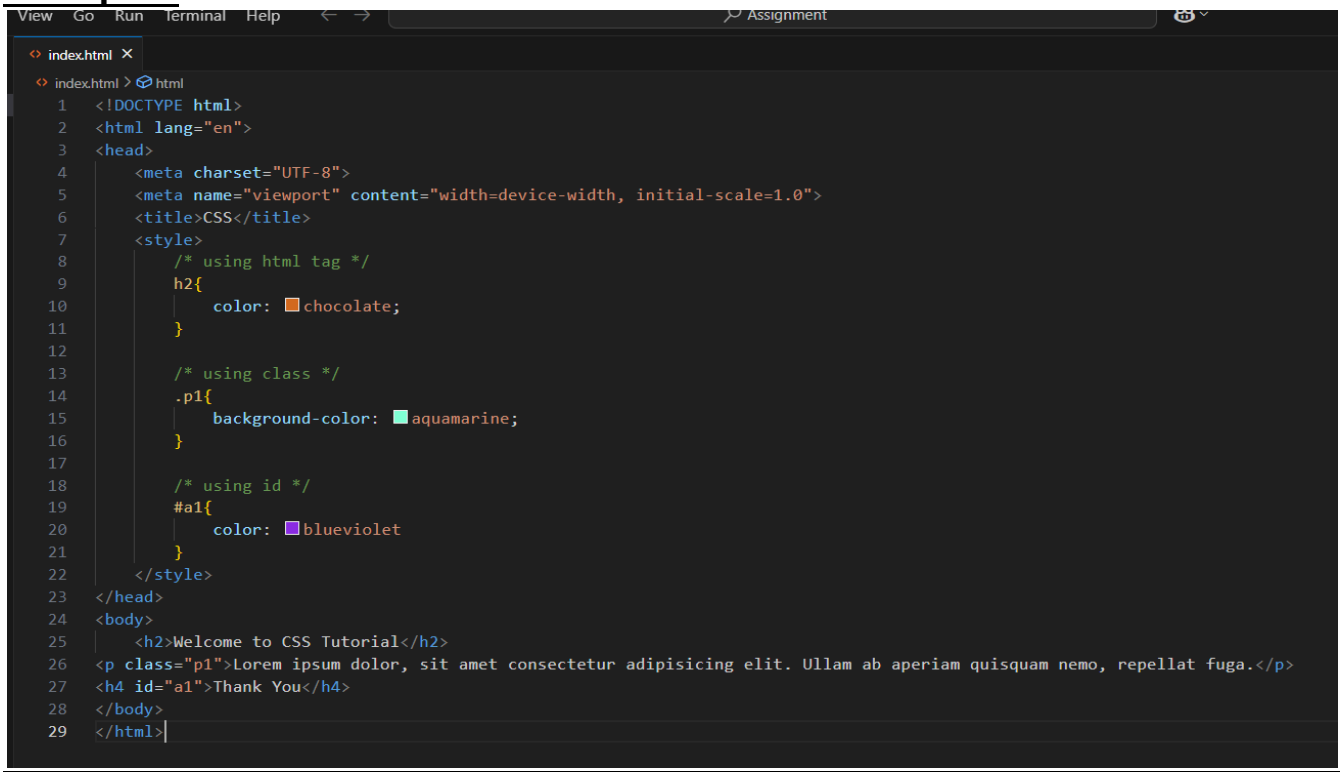
A CSS selector is a pattern used to choose and style specific HTML elements on a web page.

**1. Element Selector:-**The element selector targets all HTML elements of a specific type.

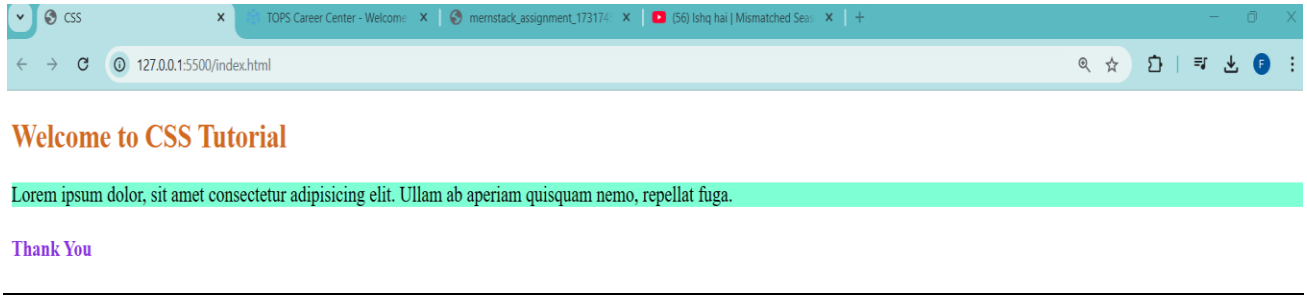
**2. Class Selector:-**The class selector is used to style elements with a specific class attribute. it start with a dot (.) in CSS.

**3. ID Selector:-** The ID selector targets a single element with a specific id attribute. it start with (#) in CSS.

## Example:-




## OUTPUT:-




---

## Ans(2):-

## CSS specificity:-

CSS specificity is a set of rules used by browsers to determine which CSS rule to apply when multiple rules target the same element.

→ More specific selectors taking precedence over less specific ones.

## Resolving Conflicts:-

→CSS rule has a specificity score based on its selectors. The rule with the highest specificity takes precedence.

→ Some properties, like color and font-family, are **inherited** by child elements by default, while others, like margin and border, are not.

## Ans(3):-

| Inline | Internal | External |
|---|---|---|
| Placed directly in the HTML element. | Placed in the head section of the HTML. | Liked to the html page. |
| No selector used. | Use the <style>tag | Use the <link>tag in the <head>tag. |
| Only apply to the HTML element. | Only apply to the current HTML page. | All the style needed for all page. |
| Overrides both internal and external styles | Overrides external CSS | separation of content in HTML. |

## Advantages and Disadvantages :-

## 1.Inline CSS:

**Quick and Simple:** Easy to implement for small changes or specific elements.

**No External Files Required:** Styles are embedded directly, reducing dependency on additional files.

Disadvantages:-

**Redundant Code:** Leads to code duplication, as styles are repeated for each element.

**Not Scalable:** Unsuitable for larger projects due to cluttered HTML and lack of separation between content and presentation.

**Low Reusability:** Styles cannot be reused for multiple elements.

## 2.Internal CSS:

**Easy to Apply to a Single Page:** Useful for styling a single page without affecting others.

**No External File Required:** All styles are included in the HTML file, reducing dependency on external resources.

**Higher Specificity than External Styles:** Useful when overriding external CSS.

Disadvantages:-

**Limited Reusability:** Styles are confined to one document, requiring duplication for other pages.

## 2.External CSS:

**Reusable Styles:** A single style sheet can be used across multiple pages, improving consistency.

**Scalability:** Well-suited for large projects and teams due to centralized styling.

**Faster Maintenance:** Changes in one CSS file automatically reflect across all linked pages.

Disadvantages:-

**Initial Load Time:** Requires an additional HTTP request to fetch the CSS file, which may impact page load time.

**Complex Debugging:** When multiple style sheets are used, debugging specificity conflicts can become challenging.

## Lab Assignment

a1.html | index.html | styles.css

index.html > html > body > form#contact-form > button#btn

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Styled Contact Form</title>
<!-- Used to external CSS -->
  <link rel="stylesheet" href="styles.css">
</head>
<body>
  <form id="contact-form">
    <h2>Contact Us</h2>

    <label for="name">Name:</label>
    <input type="text" id="name" class="form-field" placeholder="Your name" required><br><br>

    <label for="email">Email:</label>
    <input type="email" id="email" class="form-field" placeholder="Your email" required><br><br>

    <label for="message">Message:</label>
    <textarea id="message" class="form" placeholder="Your message" required></textarea><br><br>

    <button type="submit" id="btn">Submit</button>
  </form>
</body>
</html>

```

View Go Run Terminal Help Assignment

a1.html index.html styles.css

styles.css > ...

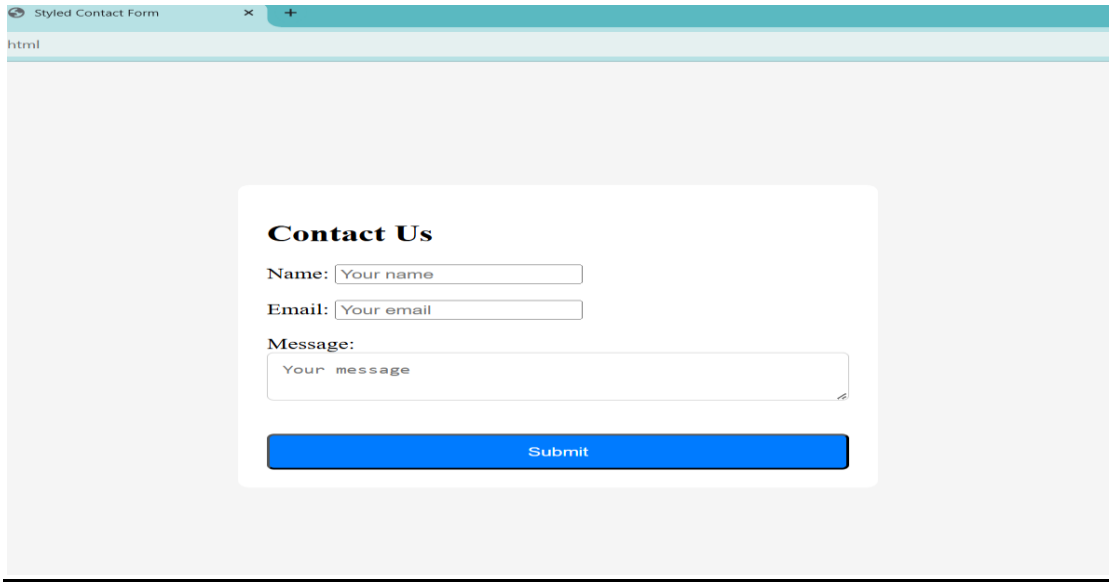
```css
body {
    background-color: #f5f5f5;

    display: flex;
    justify-content: center;
    align-items: center;
    height: 100vh;
  }
  /* Using id */
  #contact-form {
    background-color: #ffffff;
    padding: 20px;
    margin: 0;
    border-radius: 8px;
    width: 400px;
  }
/* Using class */
  .form{
    width: 100%;
    padding: 10px;
    margin-bottom: 15px;
    border: 1px solid #ccc;
    border-radius: 5px;
    box-sizing: border-box;
  }
  #btn {
    width: 100%;
    background-color: #007bff;
    color: #ffffff;
    padding: 10px;
    border-radius: 5px;
    transition: background-color 0.3s ease;
  }

  #btn:hover {
    background-color: #0056b3;
  }

```

## OUTPUT:-



## CSS Box Model:-

## Ans (1):-

The CSS Box Model is a fundamental concept in web design that describes how every element on a webpage is structured and sized.

1. **Content:-**The innermost part of the box where text, images, or other content is displayed.

→**Size :-** The size of the content depends on the width and height properties you set for the element.

2. **Padding:-**The space between the content and the element's border.

→ **Size** :**-** because it pushes the border outward. For instance, if your content is 100px wide and you add padding: 10px, the total width becomes 120px.

**3. Borders: -** The edge surrounding the padding and content. Think of it as a frame around the element.

→**Size:-** If your content is 100px wide, with padding: 10px and border: 5px, the total width is now 130px.

**4. Margin:-**The space outside the border that separates the element from other elements.

→Size:- If you set margin: 20px, the element will have 20px of space between its border and nearby elements.

## Ans (2):-

## Difference between border-boxand content-boxbox-sizing:-

The box-sizing property in CSS determines how the total size of an element is calculated. It controls whether padding and borders are included in the element's specified width and height.

→Content Box are defult.

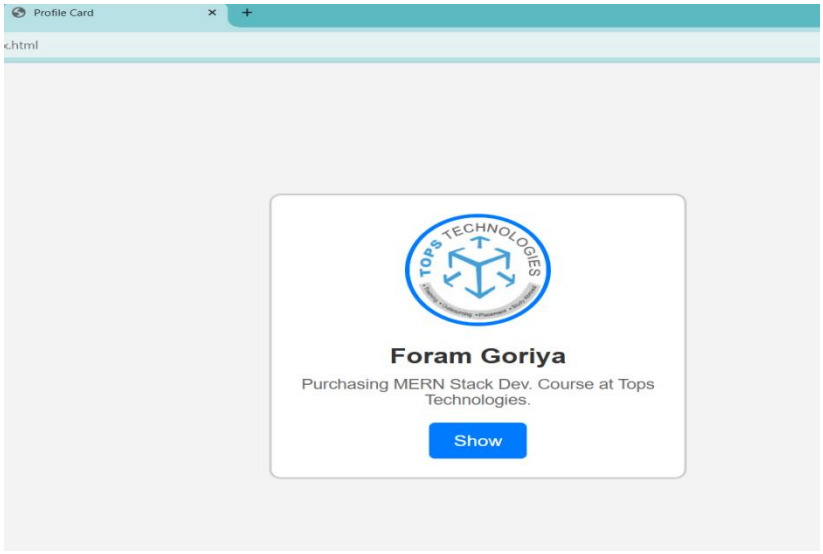
## Lab Assignment

index.html

```
index.html > html > head > style > .btn
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Profile Card</title>
  <style>
    * {
      margin: 0;
      padding: 0;
      box-sizing: border-box;
    }
    body {
      display: flex;
      justify-content: center;
      align-items: center;
      height: 100vh;
      background: #f3f3f3;
      font-family: Arial, sans-serif;
    }
    .profile-card {
      width: 300px;
      padding: 20px;
      border: 2px solid #ccc;
      border-radius: 10px;
      background: #fff;
      text-align: center;
      box-sizing: content-box;
    }
    .profile-pic {
      width: 120px;
      height: 120px;
      border-radius: 50%;
      border: 4px solid #007bff;
      margin-bottom: 15px;
    }
```

```
    margin-bottom: 15px;
  }
  .user-name {
    font-size: 1.4rem;
    color: #333;
    margin-bottom: 10px;
  }
  .user-bio {
    font-size: 0.9rem;
    color: #666;
    margin-bottom: 15px;
  }
  .btn {
    padding: 10px 20px;
    font-size: 1rem;
    color: #fff;
    background: #007bff;
    border: none;
    border-radius: 5px;
    cursor: pointer;
  }
  .btn:hover {
    background: #0056b3;
  }
 </style>
/head>
body>
 <div class="profile-card">
   <img class="profile-pic" src="Screenshot 2025-01-19 153215.png" alt="Profile Picture">
   <h2 class="user-name">Foram Goriya</h2>
   <p class="user-bio">Purchasing MERN Stack Dev. Course at Tops Technologies.  </p>
   <button class="btn">Show</button>
 </div>
/body>
/html>
```

## OUTPUT:-



## CSS Flex box:-

## Ans (1):-

Flex-box is a layout method for arranging items in rows or columns.

1. **Flex-container:**- The flex container is the parent element that holds the flex items. The display: flex; or display: inline-flex; property is applied to the container to enable Flex-box. A flex container organizes its child elements (flex items) along either a row or a column, based on the defined direction.

2. **Flex-item:**- Flex items are the direct children of the flex container. These items are arranged according to the Flex-box model, which allows them to grow, shrink, or remain at their initial size depending on available space and flex properties.

## How it is useful:-

→**Alignment and Distribution:** Flex-box makes it easier to align items vertically and horizontally within a container, both in terms of space distribution and alignment along the main and cross axes.

→**Responsiveness:** Flex-box helps design responsive layouts by making items adapt to different screen sizes and available space.

→**Flexibility:** Flex-box provides a flexible way to arrange elements within a container without needing to explicitly define widths or heights for each item.

## Ans(2):-

## Justify-content,align-items,flex-directionproperties:-

## Justify-content:-

The justify-content property controls the alignment of flex items along the main axis.It distributes space between the items and aligns them relative to the container.

Values:-

→flex-start: Items are aligned to the start of the container.

→flex-end: Items are aligned to the end of the container.

→center: Items are centered within the container.

→space-between: Items are spaced out evenly with the first item at the start and the last item at the end.

→space-around: half the space between the items is at the ends of the container.

→space-evenly: equal space between them and at the edges of the container.

**Example:-**

```
.container {
  display: flex;
  justify-content: center;
}
```

**align-items:-**The align-items propertyallows you to align items within the flex container based on their height.

**Values:**

→stretch: Items stretch to fill the container.

→flex-start: Items are top if the main axis is horizontal.

→flex-end: Items are aligned bottom if the main axis is horizontal.

→center: Items are aligned in the center of the cross axis.

→baseline: Items are aligned the text baseline of the items.

**Example:**

```
.container {
  display: flex;
  align-items: center;
}
```

**3.flex-direction:-**The flex-direction property defines the direction of the main axiswhich the flex items are placed within the container.

**Values:**

→row (default): Items arehorizontally from left to right.

→row-reverse: Items are horizontally but right to left.

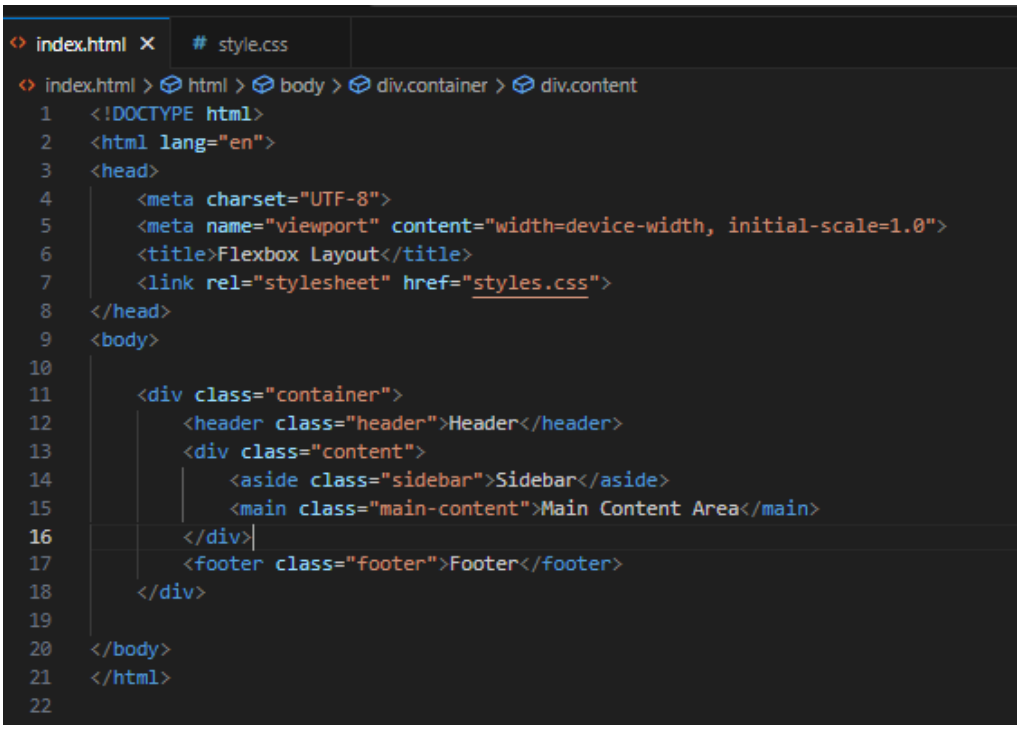
→column: Items are vertically from top to bottom.

→column-reverse: Items are vertically bottom to top.

**Example:**

```
.container {
  display: flex;
flex-direction: column;
```

}


index.html
style.css
index.html > html > body > div.container > div.content
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4 <meta charset="UTF-8">
5 <meta name="viewport" content="width=device-width, initial-scale=1.0">
6 <title>Flexbox Layout</title>
7 <link rel="stylesheet" href="styles.css">
8 </head>
9 <body>
10
11 
12 <header class="header">Header</header>
13 
14 <aside class="sidebar">Sidebar</aside>
15 <main class="main-content">Main Content Area</main>
16 
17 <footer class="footer">Footer</footer>
18 
19
20 </body>
21 </html>
22

# Lab Assignment

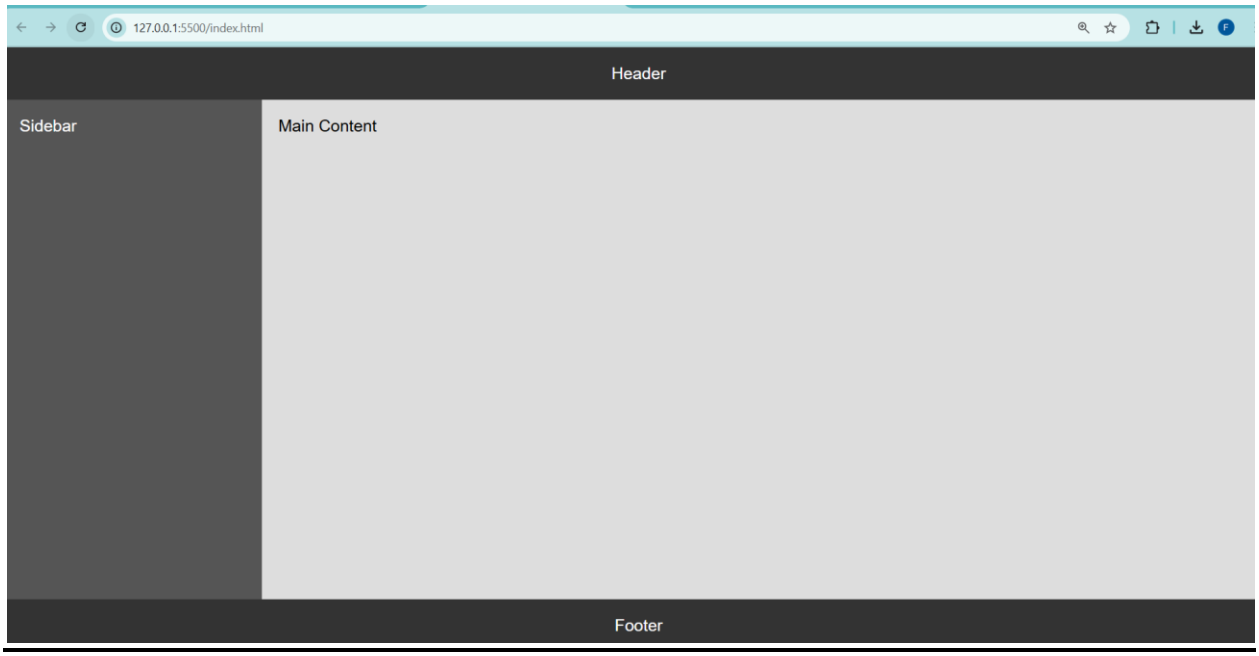
f1.html | index.html

index.html > html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Flexbox Layout</title>
    <style>
        * {
            margin: 0;
            padding: 0;
            box-sizing: border-box;
        }
        body {
            font-family: Arial, sans-serif;
            display: flex;
            flex-direction: column;
            height: 100vh;
        }
        header, footer {
            background: #333;
            color: white;
            text-align: center;
            padding: 1rem;
        }
        .container {
            display: flex;
            flex: 1;
            flex-wrap: wrap;
        }
        .sidebar {
            background: #555;
            color: white;
            padding: 1rem;
            width: 250px;
            flex-shrink: 0;
        }
```

```
        }
        .main-content {
            background: #ddd;
            flex: 1;
            padding: 1rem;
        }
        @media (max-width: 768px) {
            .container {
                flex-direction: column;
            }
            .sidebar {
                width: 100%;
                text-align: center;
            }
        }
    </style>
</head>
<body>
    <header>Header</header>
    <div class="container">
        <aside class="sidebar">Sidebar</aside>
        <main class="main-content">Main Content</main>
    </div>
    <footer>Footer</footer>
</body>
</html>
```

## OUTPUT:-



# CSS Grid

# Ans (1):-

## CSS Grid and how it differs from Flex box:-

CSS Grid and Flex box are both layout systems in CSS that help design responsive and dynamic web layouts.

## Key Features of CSS Grid:-

→**Rows & Columns:** Allows precise placement of items in both directions.

→**Explicit and Implicit Grids:** grid-template-rows & grid-template-columns or let Grid automatically create one.

→**Grid Areas:** You can assign multiple elements to specific named areas.

→**Flexible Sizing:** Supports auto-sizing, and min-max().

→**Alignment Controls:** Supports align-items, justify-items, place-items, and gap

## When to Use Grid vs. Flex box:-

→Layout type in grid 2-dimensional(row columns).

→Layout type in grid 2-dimensional (row columns).

→ Complex layouts in grid system.

→ Simple layouts in Flex box.

# Ans (2):-

## Grid Example:-

1. **grid-template-columns:-** This property defines the number and size of the columns in a CSS grid.

**Example:-**

```
.container {
  display: grid;
  grid-template-columns: 100px 200px auto;
}
```

**2. grid-template-rows:-** This property defines the number and size of the rows in a CSS grid.

**Example:-**

```
.container {
  display: grid;
  grid-template-rows: 100px 200px auto;
}
```

3. **grid-gap :-**The grid-gap property defines spacing between rows and columns.

**Example:-**

```
.container {
  display: grid;
  grid-template-columns: repeat(3, 1fr);
  grid-template-rows: repeat(2, 150px);
  gap: 20px;        }
```

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Task</title>
    <style>
        .container {
  display: grid;
  grid-template-columns: repeat(3, 1fr);
  grid-template-rows: 100px 200px auto;
  gap: 15px 10px;
}
.item {
  background-color: lightblue;
  padding: 20px;
  text-align: center;
  border: 1px solid blue;
}
    </style>
</head>
<body>
    <div class="container">
        <div class="item">1</div>
        <div class="item">2</div>
        <div class="item">3</div>
        <div class="item">4</div>
        <div class="item">5</div>
        <div class="item">6</div>
    </div>
</body>
</html>
```

## Output:-

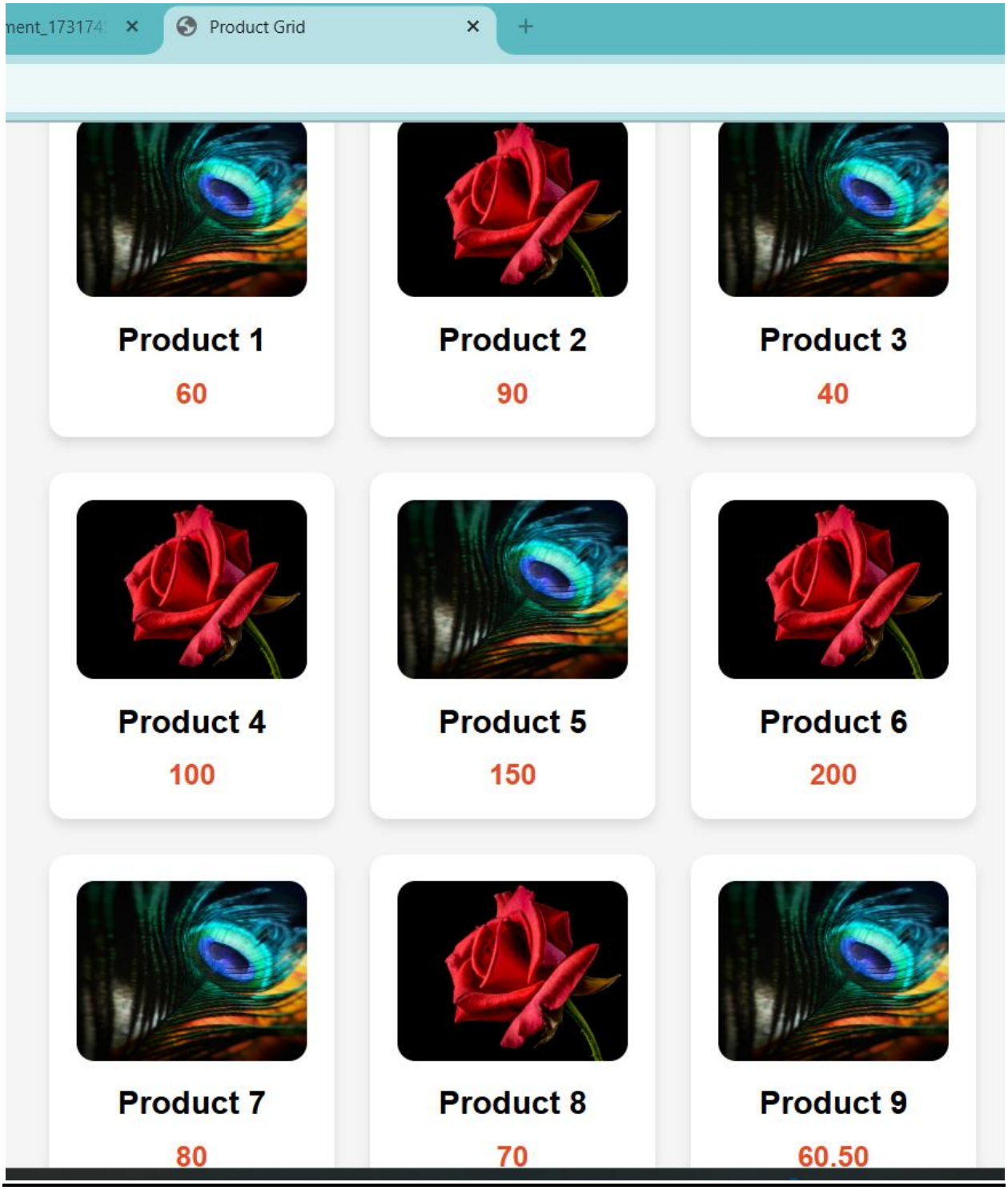
# Lab Assignment

index.html style.css

index.html > html > body > div.grid-container > div.product-card > img

```
<html lang="en">
<body>
    <div class="grid-container">
        <div class="product-card">
            <div class="product-price">60</div></div>
        <div class="product-card">
            <img src="Images/rose.jpg" alt="Product 2" height="100px" width="100px">
            <div class="product-title">Product 2</div>
            <div class="product-price">90</div></div>
        <div class="product-card">
            <img src="Images/feadhers.jpg" alt="Product 3" height="100px" width="100px">
            <div class="product-title">Product 3</div>
            <div class="product-price">40</div></div>
        <div class="product-card">
            <img src="Images/rose.jpg" alt="Product 4" height="100px" width="100px">
            <div class="product-title">Product 4</div>
            <div class="product-price">100</div></div>
        <div class="product-card">
            <img src="Images/feadhers.jpg" alt="Product 5" height="100px" width="100px">
            <div class="product-title">Product 5</div>
            <div class="product-price">150</div></div>
        <div class="product-card">
            <img src="Images/rose.jpg" alt="Product 6" height="100px" width="100px">
            <div class="product-title">Product 6</div>
            <div class="product-price">200</div></div>
        <div class="product-card">
            <img src="Images/feadhers.jpg" alt="Product 7" height="100px" width="100px">
            <div class="product-title">Product 7</div>
            <div class="product-price">80</div></div>
        <div class="product-card">
            <img src="Images/rose.jpg" alt="Product 8" height="100px" width="100px">
            <div class="product-title">Product 8</div>
            <div class="product-price">70</div></div>
        <div class="product-card">
            <img src="Images/feadhers.jpg" alt="Product 9" height="100px" width="100px">
            <div class="product-title">Product 9</div>
            <div class="product-price">60.50</div></div></div>
</body>
</html>
```

## OUTPUT:-


Product Grid
Product 1
60
Product 2
90
Product 3
40
Product 4
100
Product 5
150
Product 6
200
Product 7
80
Product 8
70
Product 9
60.50