Report Day: 3.1

Submitted By: Ayesha Abid

Submitted To: Mr. Ali Hyder Hidayat

Group: Frontend Development @ProSensia

Dated: 07/07/25

Learning Outcomes:

Introduction to CSS Grid:

CSS Grid Layout is a powerful 2-dimensional system in CSS for designing web page layouts. Unlike Flexbox (which works best in one direction), Grid allows us to create complex layouts using **rows and columns** simultaneously.

1. Grid Rows and Columns

In a grid container, the space is divided into rows (horizontal) and columns (vertical).

Grid-template-rows

This property defines the height of each row. We can use units like px, %, fr, or auto.

Grid-template-columns

This property defines the width of each column.

2.The gap property:

The gap property is a shorthand property for row-gap and column-gap:

Syntax:

```
.container {
  display: grid;
  gap: 50px 100px;
}
```

3.Grid Lines

The lines between columns are called *column lines*.

The lines between rows are called *row lines*.

4. Grid Properties

- grid-column-start
- grid-column-end
- grid-row-start
- grid-row-end
- grid-column
- grid-row

4. Grid Template Areas

Grid-template-areas allows us to assign **names to areas** of your layout and position elements accordingly.

5. The grid-area Property

The grid-area property is a shorthand property for the grid-row-start, grid-column-start, grid-row-end and the grid-column-end properties.

The syntax is grid-row-start / grid-column-start / grid-row-end / grid-column-end

```
.item4 {
  grid-area: 1 / 2 / 3 / 2;
}
```

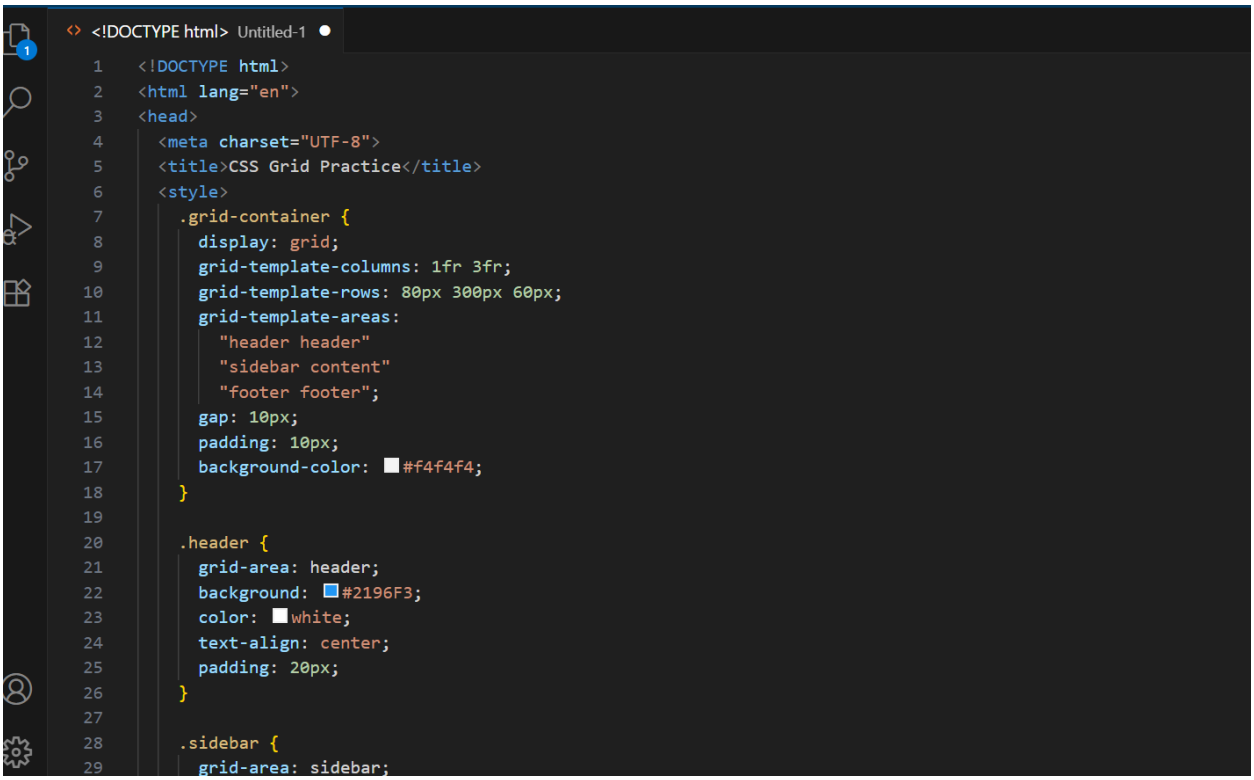
6. Naming Grid Items with grid-area

The grid-area property can also be used to assign names to grid items.

The named grid items can then be referred to by the grid-template-areas property of the grid container.

```
.item1 {
  grid-area: myArea;
}

.grid-container {
  grid-template-areas: 'myArea myArea myArea myArea myArea';
}
```



```
<!DOCTYPE html> Untitled-1
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>CSS Grid Practice</title>
  <style>
    .grid-container {
      display: grid;
      grid-template-columns: 1fr 3fr;
      grid-template-rows: 80px 300px 60px;
      grid-template-areas:
        "header header"
        "sidebar content"
        "footer footer";
      gap: 10px;
      padding: 10px;
      background-color: #f4f4f4;
    }

    .header {
      grid-area: header;
      background: #2196F3;
      color: white;
      text-align: center;
      padding: 20px;
    }

    .sidebar {
      grid-area: sidebar;
```

```
  <style>
    }

    .content {
      grid-area: content;
      background: #4CAF50;
      color: white;
      padding: 20px;
    }

    .footer {
      grid-area: footer;
      background: #333;
      color: white;
      text-align: center;
      padding: 20px;
    }
  </style>
</head>
<body>
  <div class="grid-container">
    <div class="header">Header</div>
    <div class="sidebar">Sidebar</div>
    <div class="content">Main Content</div>
    <div class="footer">Footer</div>
  </div>
</body>
</html>
```

Conclusion:

CSS Grid is a robust layout system ideal for structuring full web pages or application layouts. With grid-template-rows, grid-template-columns, and grid-template-areas, we can create responsive, readable, and maintainable layouts without relying heavily on external frameworks.