

# 1

## Laporan Praktikum CSS Layout

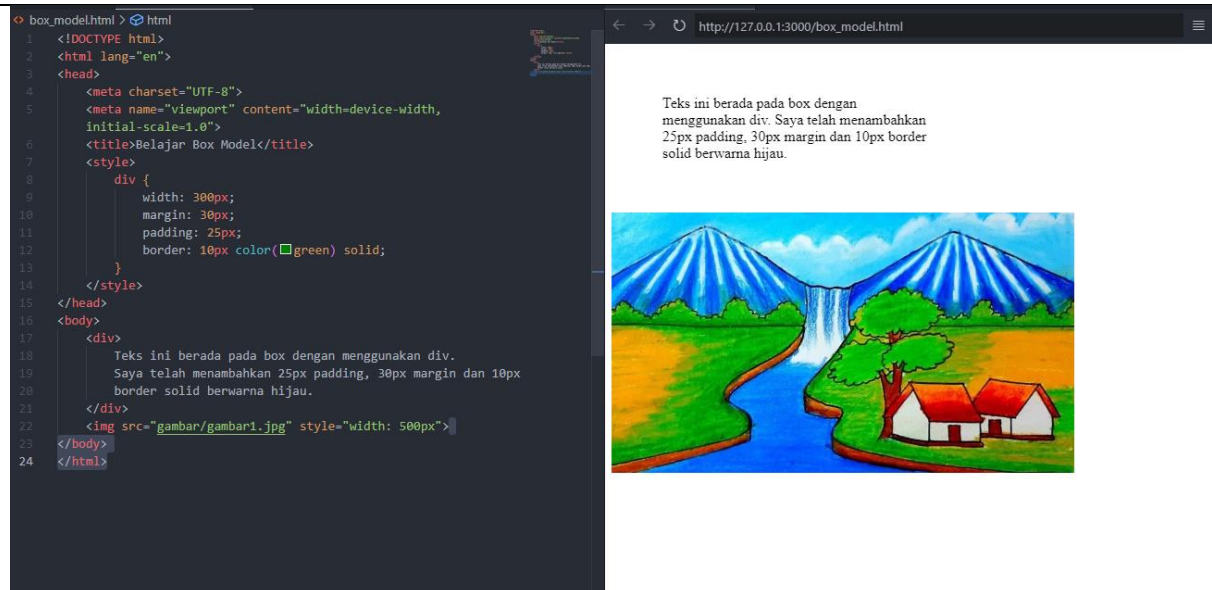
Nama	Muhammad Akbar Fadilah
NPM	244311051
Kelas	2B
Github	<a href="https://github.com/akbrvdv/tugas_praktikum">https://github.com/akbrvdv/tugas_praktikum</a>

### KEMAMPUAN AKHIR YANG DIRENCANAKAN

*Mampu memahami Cascading Style Sheet (CSS) dan merancang aplikasi web berbasis HTML dan CSS.*

## HASIL PEKERJAAN

### No 1 – box\_model



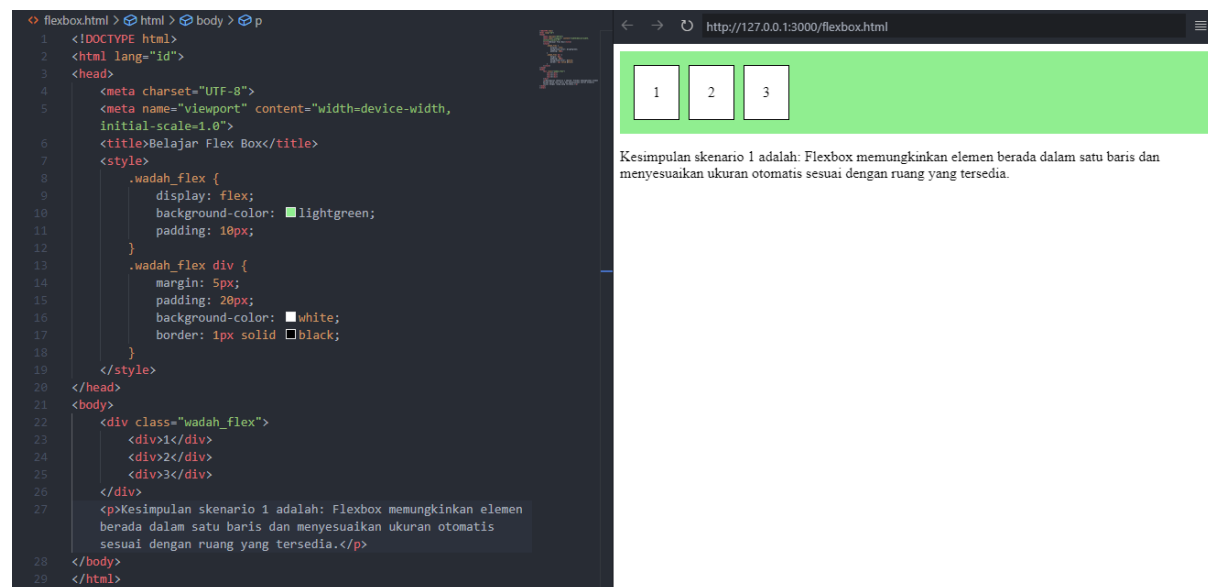
#### Deskripsi/Keterangan :

Kode ini menerapkan Box Model dalam CSS. Elemen `<div>` memiliki lebar `300px`, padding `25px`, margin `30px`, dan border `10px solid hijau` (namun terdapat kesalahan pada border: `10px color(green) solid;`, seharusnya border: `10px solid green;`). Selain itu, terdapat gambar dengan lebar `500px` yang diambil dari folder `gambar/`.

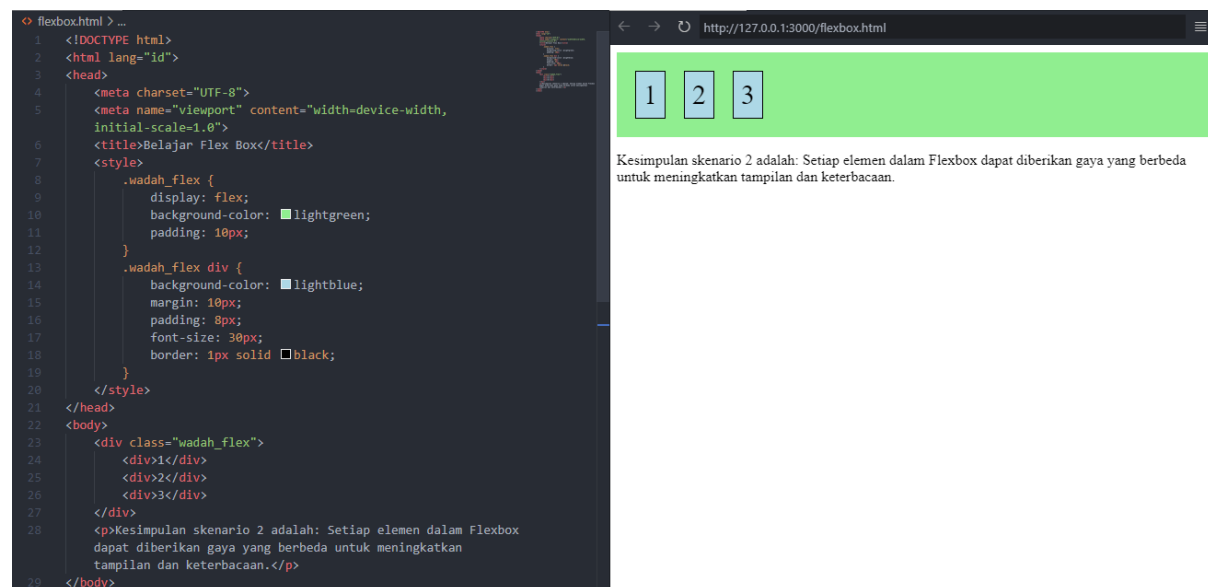
## No 2 – Flexbox

### Hasil Tangkapan Layar :

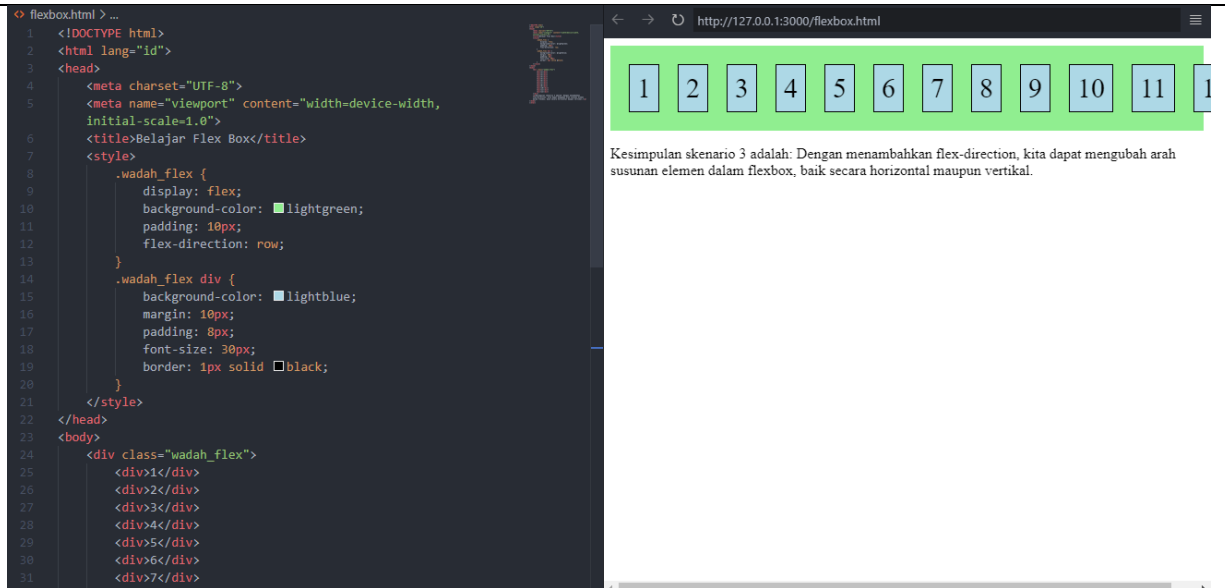
#### Skenario 1



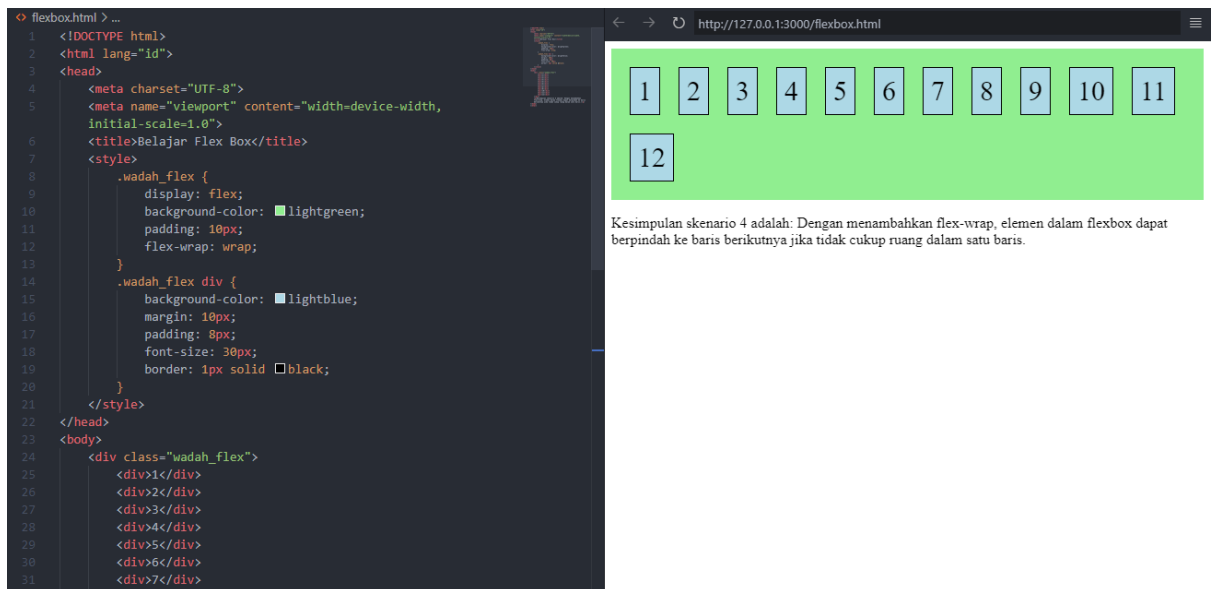
#### Skenario 2



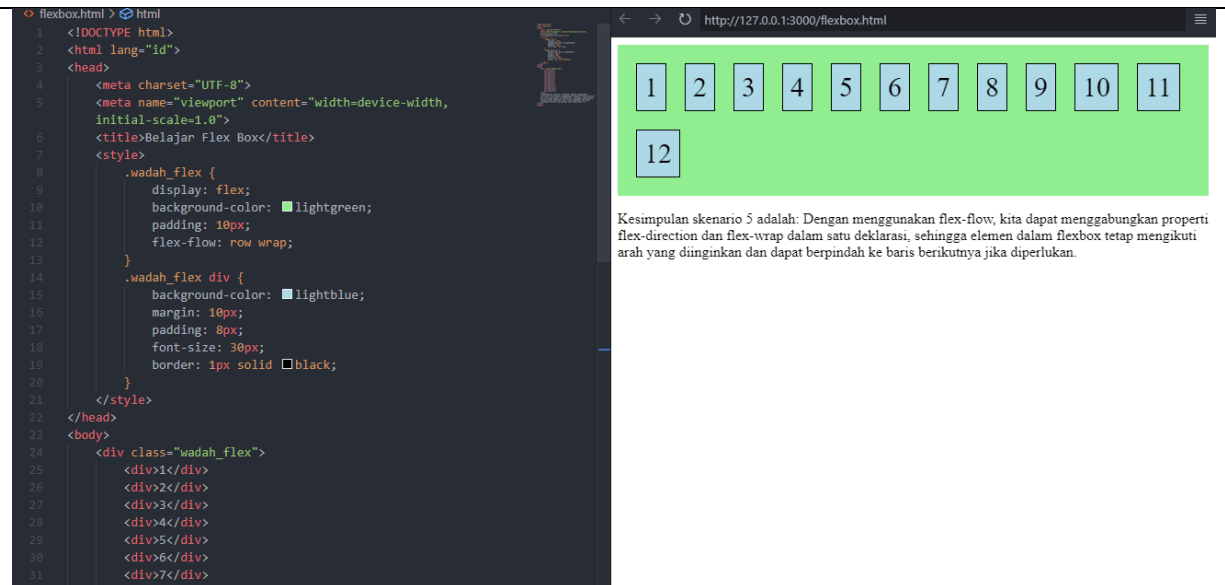
#### Skenario 3



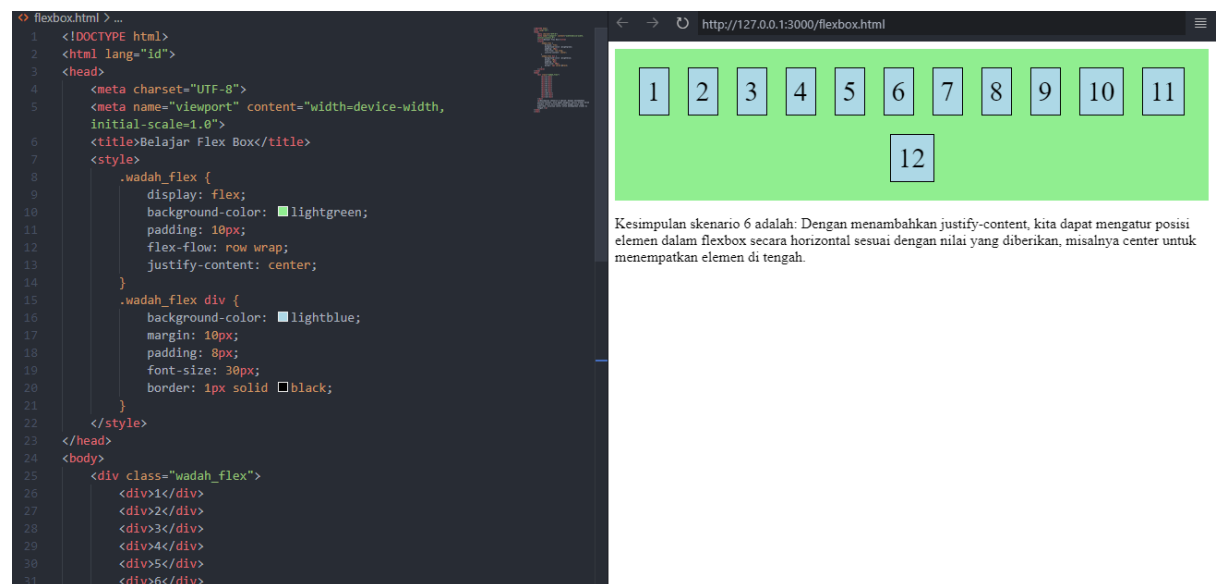
#### Skenario 4



#### Skenario 5



### Skenario 6

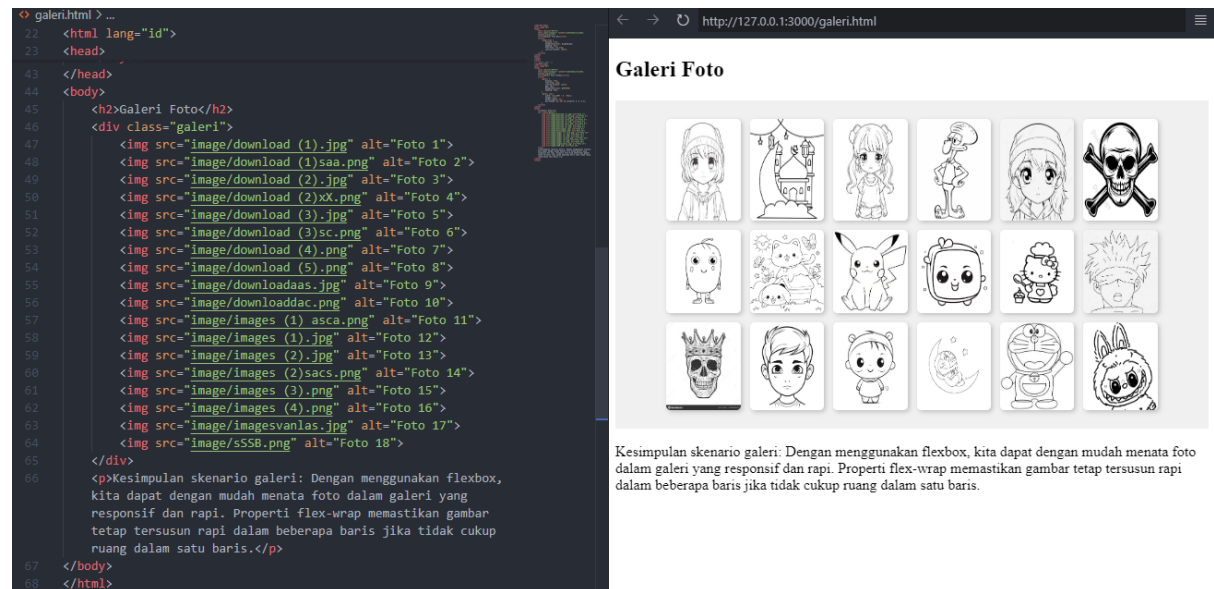


### Deskripsi/Keterangan :

*Flexbox memberikan fleksibilitas dalam mengatur tata letak elemen dalam suatu container. Dimulai dengan mendeklarasikan display: flex, kita dapat mengontrol arah elemen menggunakan flex-direction, memungkinkan elemen untuk tetap dalam satu baris atau berpindah ke baris berikutnya dengan flex-wrap. Penggunaan flex-flow menyederhanakan deklarasi arah dan pembungkusan elemen. Terakhir, justify-content memungkinkan pengaturan posisi elemen secara horizontal, sehingga elemen dapat disejajarkan ke kiri, tengah, kanan, atau tersebar secara merata sesuai kebutuhan desain.*

### No 3 – Galeri

#### Hasil Tangkapan Layar :



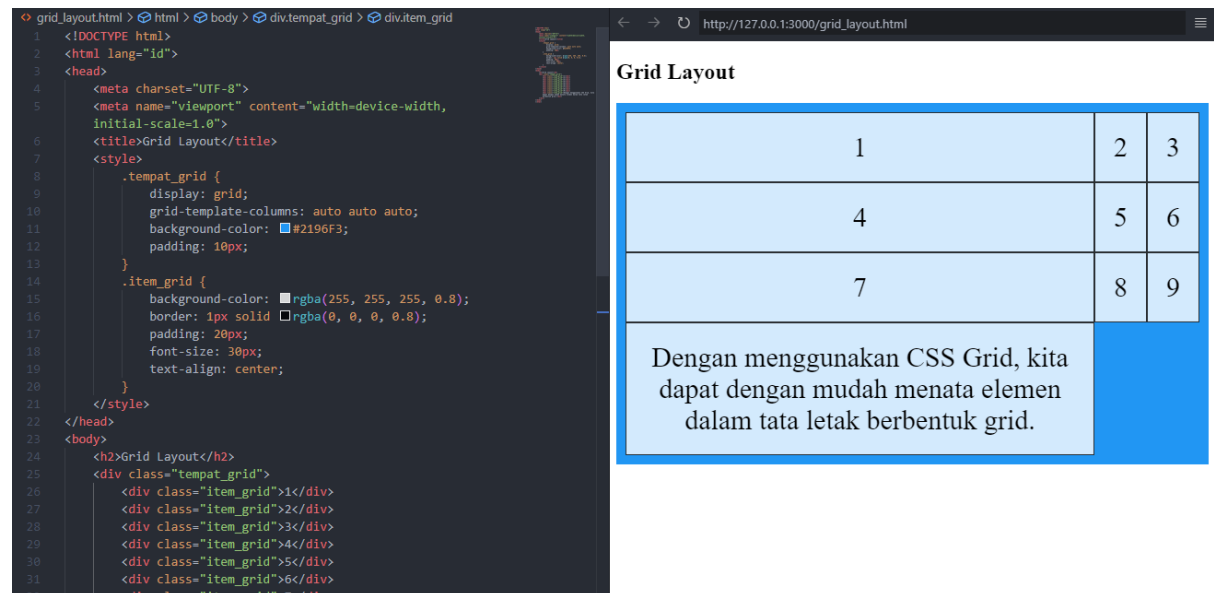
#### Deskripsi/Keterangan :

HTML ini menampilkan tiga paragraf. style2.css memberikan style dasar pada semua paragraf (hijau, tebal). Paragraf dengan class "group" (paragraf kedua) diubah stylenya (merah, verdana, biru). Paragraf dengan ID "one" (paragraf ketiga) stylenya ditimpa lagi (hijau, latar putih). CSS menggunakan class dan ID untuk memberikan style lebih spesifik pada elemen.

## No 4 – Grid\_layout

### Hasil Tangkapan Layar :

#### Skenario 1



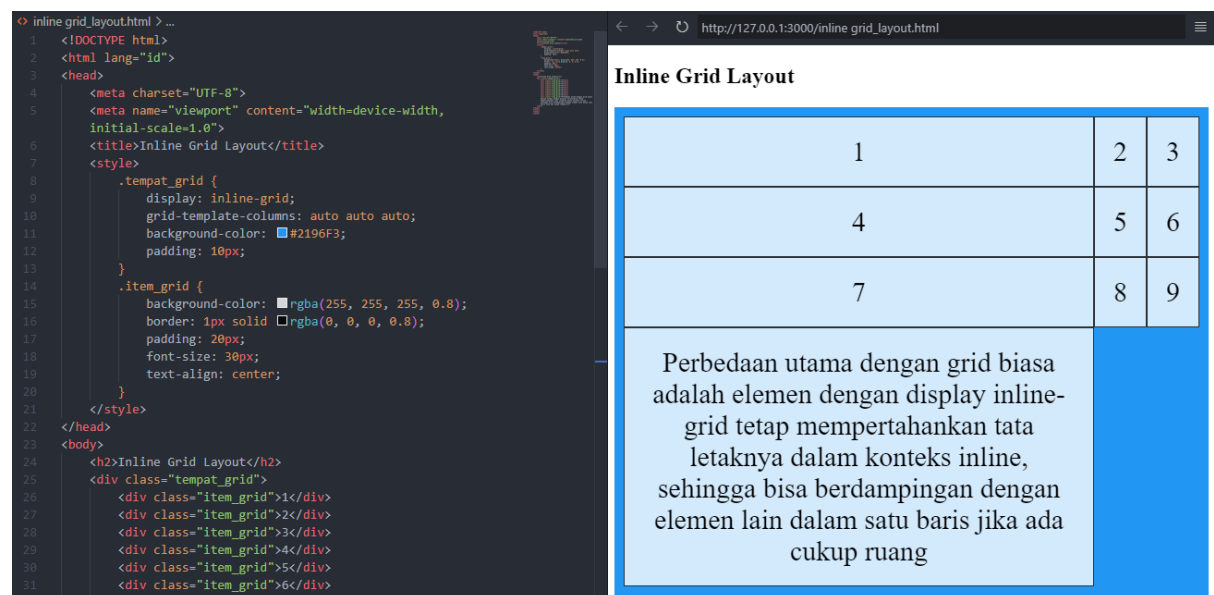
```
1 <!DOCTYPE html>
2 <html lang="id">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width,
6     initial-scale=1.0">
7   <title>Grid Layout</title>
8   <style>
9     .tempat_grid {
10       display: grid;
11       grid-template-columns: auto auto auto;
12       background-color: #2196F3;
13       padding: 10px;
14     }
15     .item_grid {
16       background-color: rgba(255, 255, 255, 0.8);
17       border: 1px solid rgba(0, 0, 0, 0.8);
18       padding: 20px;
19       font-size: 30px;
20       text-align: center;
21     }
22   </style>
23 </head>
24 <body>
25   <h2>Grid Layout</h2>
26   <div class="tempat_grid">
27     <div class="item_grid">1</div>
28     <div class="item_grid">2</div>
29     <div class="item_grid">3</div>
30     <div class="item_grid">4</div>
31     <div class="item_grid">5</div>
32     <div class="item_grid">6</div>
```

Grid Layout

1	2	3
4	5	6
7	8	9

Dengan menggunakan CSS Grid, kita dapat dengan mudah menata elemen dalam tata letak berbentuk grid.

#### Skenario 2



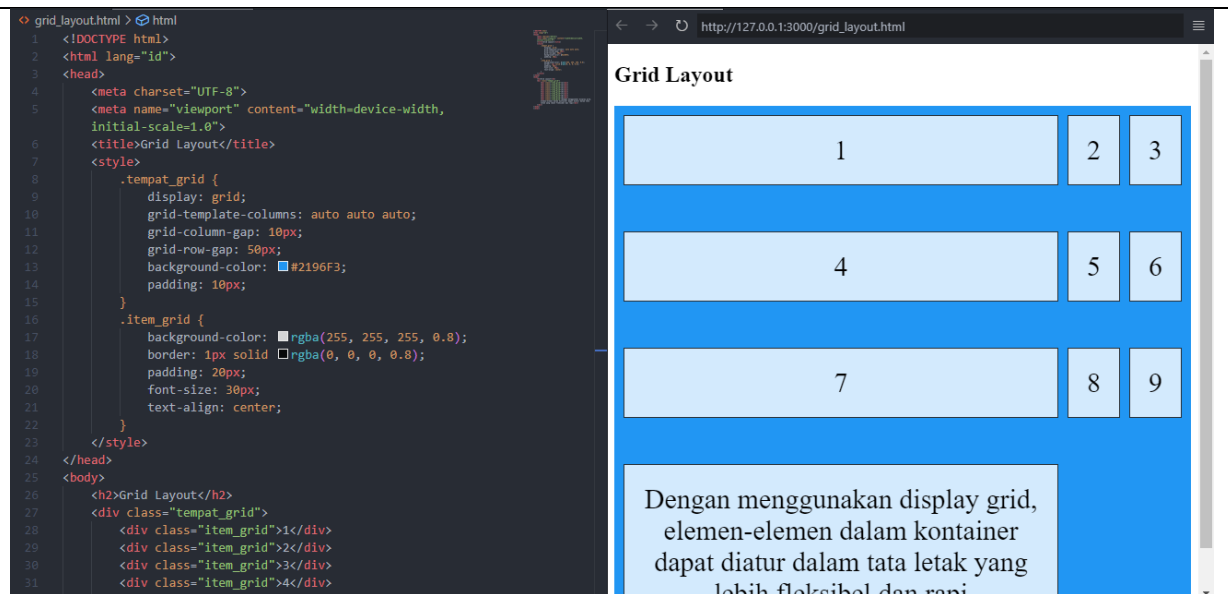
```
1 <!DOCTYPE html>
2 <html lang="id">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width,
6     initial-scale=1.0">
7   <title>Inline Grid Layout</title>
8   <style>
9     .tempat_grid {
10       display: inline-grid;
11       grid-template-columns: auto auto auto;
12       background-color: #2196F3;
13       padding: 10px;
14     }
15     .item_grid {
16       background-color: rgba(255, 255, 255, 0.8);
17       border: 1px solid rgba(0, 0, 0, 0.8);
18       padding: 20px;
19       font-size: 30px;
20       text-align: center;
21     }
22   </style>
23 </head>
24 <body>
25   <h2>Inline Grid Layout</h2>
26   <div class="tempat_grid">
27     <div class="item_grid">1</div>
28     <div class="item_grid">2</div>
29     <div class="item_grid">3</div>
30     <div class="item_grid">4</div>
31     <div class="item_grid">5</div>
32     <div class="item_grid">6</div>
```

Inline Grid Layout

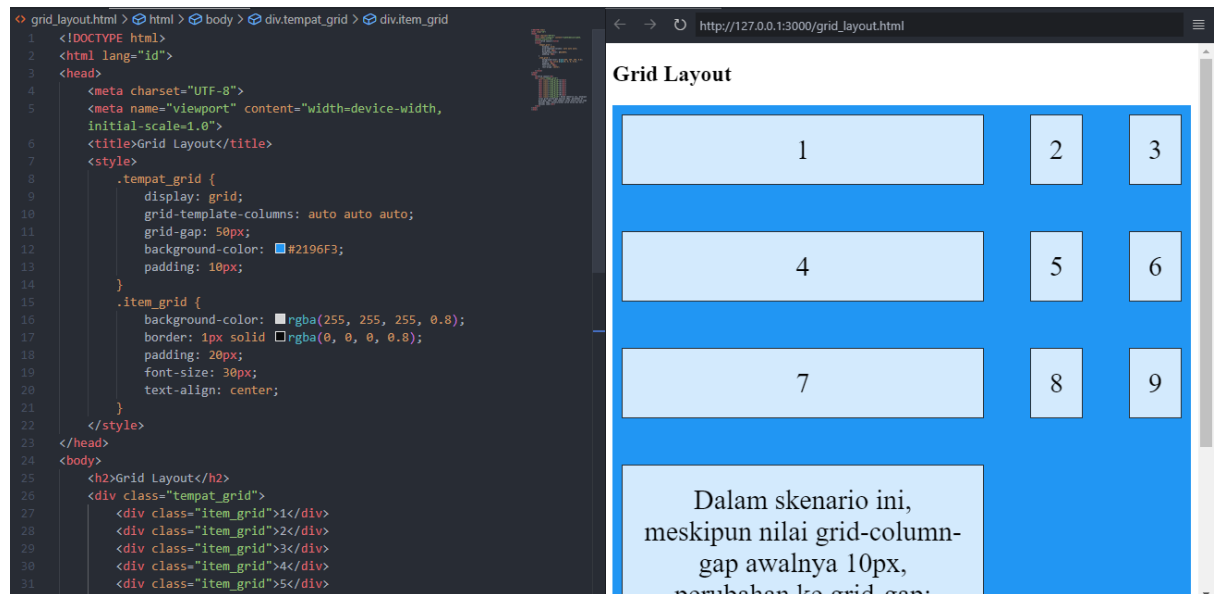
1	2	3
4	5	6
7	8	9

Perbedaan utama dengan grid biasa adalah elemen dengan display inline-grid tetap mempertahankan tata letaknya dalam konteks inline, sehingga bisa berdampingan dengan elemen lain dalam satu baris jika ada cukup ruang

#### Skenario 3



#### Skenario 4



#### Deskripsi/Keterangan :

Penggunaan CSS Grid mempermudah pengaturan elemen dalam tata letak yang rapi dan terstruktur. Grid Layout menyusun elemen dalam kolom dan baris otomatis, sementara Inline Grid tetap dalam aliran inline. Penambahan grid-gap meningkatkan estetika dengan mengatur jarak antar elemen. Secara keseluruhan, CSS Grid fleksibel dan efektif untuk desain responsif.




## No 5 – Grid\_layout2

### Hasil Tangkapan Layar :

#### Skenario 1

```
1 <!DOCTYPE html>
2 <html lang="id">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width,
6     initial-scale=1.0">
7   <title>Grid Layout 2</title>
8   <style>
9     .grid-container {
10       display: grid;
11       grid-template-rows: 100px 200px 100px;
12       grid-template-columns: 200px 300px 100px;
13       gap: 10px;
14       background-color: #f4f4f4;
15       padding: 10px;
16     }
17     .grid-item {
18       background-color: #4CAF50;
19       color: white;
20       padding: 20px;
21       text-align: center;
22     }
23   </style>
24 </head>
25 <body>
26   <h2>Grid Layout 2</h2>
27   <div class="grid-container">
28     <div class="grid-item">1</div>
29     <div class="grid-item">2</div>
30     <div class="grid-item">3</div>
31     <div class="grid-item">4</div>
32     <div class="grid-item">5</div>
33   </div>
34 </body>
35 </html>
```




Grid Layout 2

Kesimpulan skenario 1: Dengan mengubah kombinasi nilai grid-template-rows dan grid-template-columns, tampilan grid berubah secara dinamis.

#### Skenario 2

```
1 <!DOCTYPE html>
2 <html lang="id">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width,
6     initial-scale=1.0">
7   <title>Grid Layout 2</title>
8   <style>
9     .grid-container {
10       display: grid;
11       grid-template-columns: auto auto auto;
12       gap: 10px;
13       background-color: #f4f4f4;
14       padding: 10px;
15     }
16     .grid-item {
17       background-color: #4CAF50;
18       color: white;
19       padding: 20px;
20       text-align: center;
21     }
22   </style>
23 </head>
24 <body>
25   <h2>Grid Layout 2</h2>
26   <div class="grid-container">
27     <div class="grid-item">1</div>
28     <div class="grid-item">2</div>
29     <div class="grid-item">3</div>
30     <div class="grid-item">4</div>
31     <div class="grid-item">5</div>
32     <div class="grid-item">6</div>
33   </div>
34 </body>
35 </html>
```



Grid Layout 2

Kesimpulan skenario 2: Dengan mengubah nilai grid-template-columns menjadi berbagai kombinasi seperti auto auto, 70px 300px auto 40px, dan 10px 500px 20px 10px, tampilan grid berubah secara fleksibel mengikuti ukuran kolom yang telah ditentukan.

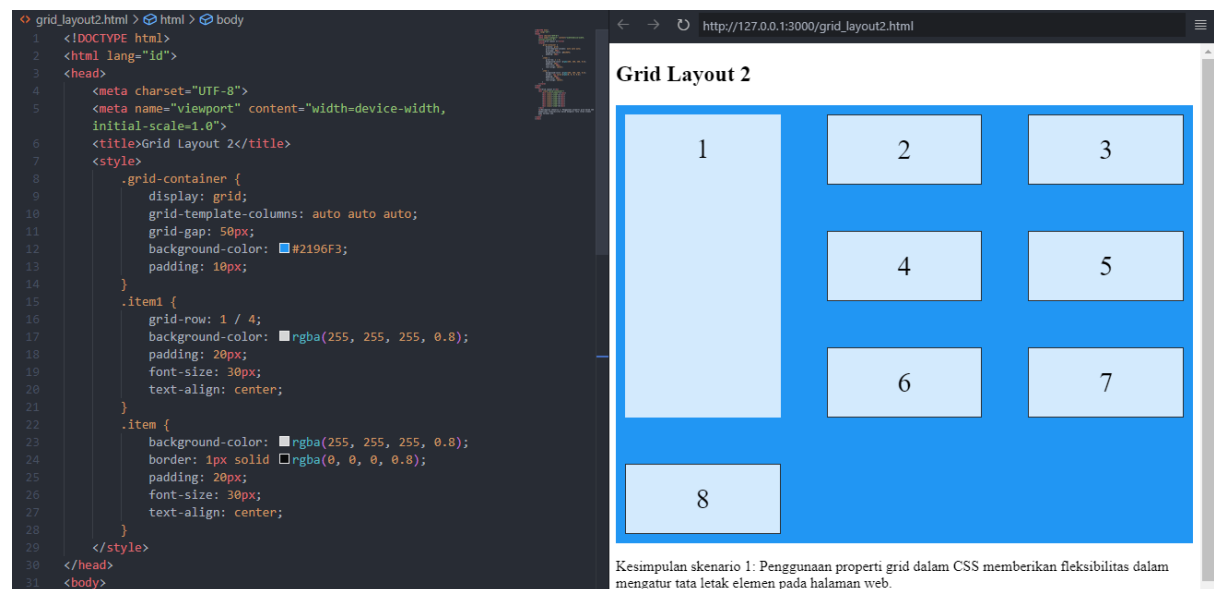
### Deskripsi/Keterangan :

Dengan mengubah kombinasi nilai `grid-template-rows` dan `grid-template-columns`, tampilan grid menjadi lebih fleksibel. Pada skenario 1, perubahan nilai `grid-template-rows` dan `grid-template-columns` berdampak pada bentuk dan ukuran grid secara keseluruhan. Sedangkan pada skenario 2, perubahan nilai `grid-template-columns` dengan kombinasi yang berbeda seperti `auto auto, 70px 300px auto 40px`, dan `10px 500px 20px 10px` memungkinkan grid untuk menyesuaikan tata letaknya sesuai dengan ukuran kolom yang telah ditentukan.

## No 6 – Grid\_layout2

### Hasil Tangkapan Layar :

#### Skenario 1



#### Skenario 2

grid\_layout2.html > html > body > div.grid-container > div.grid-item

```
1 <!DOCTYPE html>
2 <html lang="id">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width,
6     initial-scale=1.0">
7   <title>Grid Layout2</title>
8   <style>
9     .grid-container {
10       display: grid;
11       grid-template-columns: auto auto auto auto;
12       gap: 10px;
13       background-color: #f4f4f4;
14       padding: 10px;
15     }
16     .grid-item {
17       background-color: #4CAF50;
18       color: white;
19       padding: 20px;
20       text-align: center;
21     }
22     .grid-container.variant1 {
23       grid-template-columns: auto auto;
24     }
25     .grid-container.variant2 {
26       grid-template-columns: 70px 300px auto 40px;
27     }
28     .grid-container.variant3 {
29       grid-template-columns: 10px 500px 20px 10px;
30     }
31   </style>
```

http://127.0.0.1:3000/grid\_layout2.html

## Grid Layout Skenario 2

Default (auto auto auto auto)

1	2	3	4
5	6	7	8

Varian 1 (auto auto)

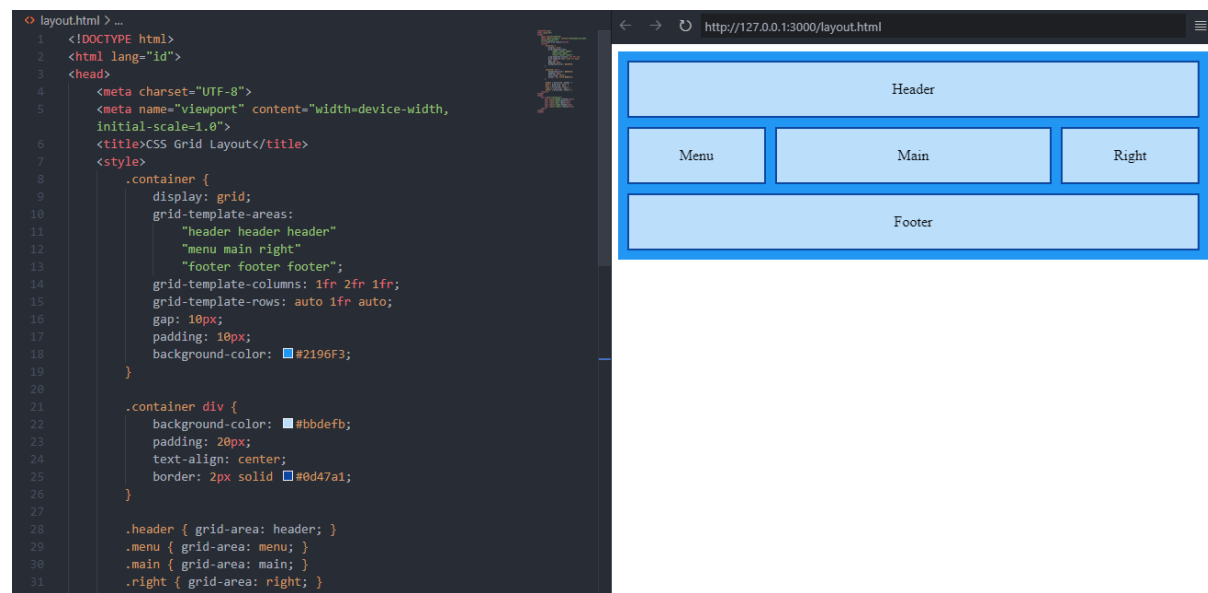
1	2
3	4
5	6
7	8

**Deskripsi/Keterangan :**

*Skenario 1 dan 2 mengajarkan bahwa dengan memanfaatkan grid-row dan grid-template-columns, kita bisa mengontrol bagaimana elemen-elemen dalam grid ditampilkan, baik dari segi tinggi maupun lebar. Grid CSS memungkinkan desain yang lebih responsif dan terstruktur dengan memberikan kendali penuh atas ukuran dan posisi elemen dalam tata letak web.*

## No 7 – Layout

### Hasil Tangkapan Layar :



### Deskripsi/Keterangan :

Kode di atas menggunakan CSS Grid untuk membuat tata letak halaman yang terdiri dari Header, Menu, Main, Right, dan Footer. Dengan `grid-template-areas`, setiap bagian diatur posisinya dalam grid, sedangkan `grid-template-columns` dan `grid-template-rows` mengontrol ukuran kolom dan baris agar responsif. Hasilnya adalah layout yang rapi, fleksibel, dan mudah disesuaikan.