

# **Laporan Praktikum**

## **Praktikum Pemrograman Web**

### **Pertemuan 6**

### **Flexbox**



Nama Mahasiswa: Hanan Fijananto (24/538946/SV/24555)

Kelas A1

Dosen Pengampu: Dinar Nugroho Pratomo, S.Kom., M.IM., M.Cs.

March 25, 2025

# Contents

<b>A Tujuan Praktikum</b>	<b>2</b>
<b>B Dasar Teori</b>	<b>2</b>
B.1 Pengertian Flexbox . . . . .	2
B.2 Properti Flexbox . . . . .	2
B.2.1 flex-direction . . . . .	2
B.2.2 justify-content . . . . .	3
B.2.3 align-items . . . . .	4
B.2.4 flex-wrap . . . . .	5
B.2.5 align-self . . . . .	6
B.2.6 order . . . . .	6
<b>C Hasil dan Pembahasan</b>	<b>7</b>
C.1 Latihan Praktikum 1 . . . . .	7
C.2 Latihan Praktikum 2 . . . . .	9
C.3 Tugas Praktikum . . . . .	13
<b>D Kesimpulan</b>	<b>20</b>
<b>E Daftar Pustaka</b>	<b>21</b>

## A Tujuan Praktikum

1. Mahasiswa mampu memahami dasar-dasar komponen flexbox pada css
2. Mahasiswa mampu memahami dan menerapkan komponen flexbox
3. Mahasiswa mampu mengimplementasikan flexbox dalam membuat web

## B Dasar Teori

### B.1 Pengertian Flexbox

Flexible Box atau flexbox css adalah sebuah sistem layout di css yang digunakan untuk mengatur, menyelaraskan, dan mendistribusikan ruang di antara item dalam suatu container. Flexbox memberikan fleksibilitas kepada developer untuk mengelola layout tanpa harus bergantung pada kompleksitas solusi yang sering membuat proses lebih rumit *Belajar Flexbox CSS dengan Contoh Sederhana 2025 — RevoU — revou.co* (n.d.).

Flex box memiliki flex container, dimana di dalamnya terdapat beberapa item flex. Hubungan antara container dan item tersebut seperti parent dan child. Pada bagian container, dapat menggunakan display: flex untuk mengubah layout pada container tersebut menjadi layout flexbox. Item pada container tersebut dapat juga ditambahkan flex item Priwanto (2024).

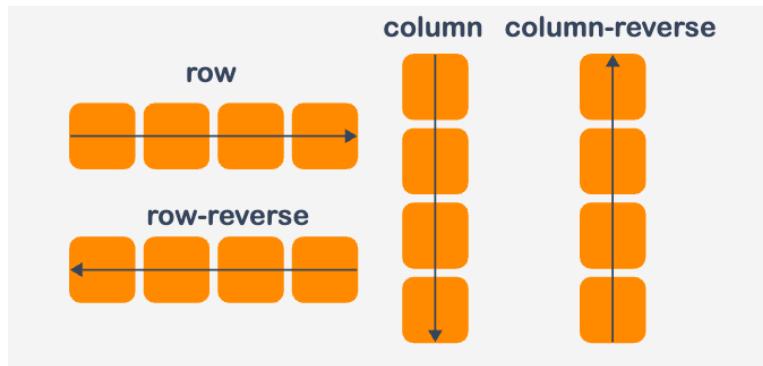
### B.2 Properti Flexbox

Dalam flexbox, terdapat properti flex item yang sering digunakan. Diantaranya yaitu.

#### B.2.1 flex-direction

Properti flex-direction dapat menentukan arah yang akan menampilkan elemen pada container flexbox. nilai yang terdapat di properti flex-direction yaitu

1. row: akan mengubah item dalam container menjadi satu baris (default).
2. column: mengubah item dalam container menjadi satu kolom.
3. row-reverse: mengubah item dalam container menjadi baris secara terbalik.
4. column-reverse: mengubah item dalam container menjadi kolom secara terbalik.

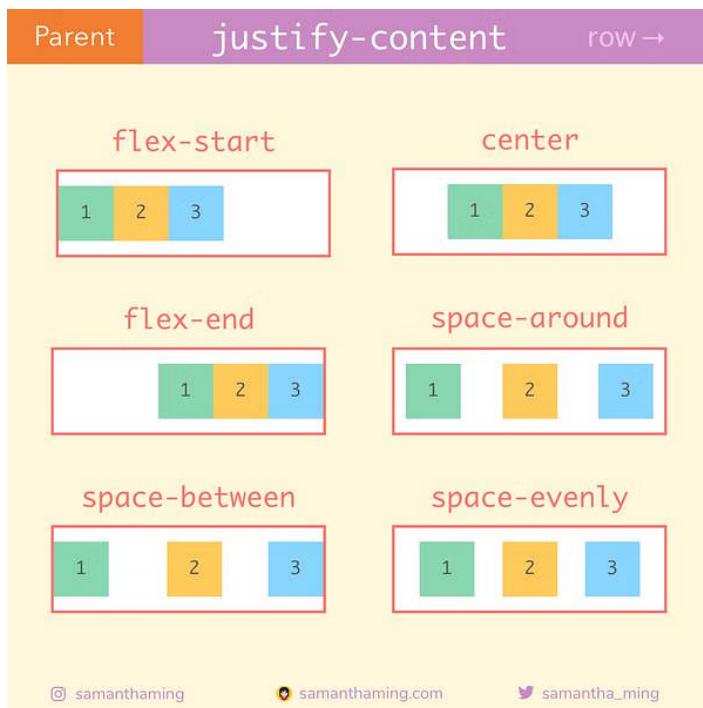


Gambar 1: Visualisasi flex-direction

### B.2.2 justify-content

Properti justify-content digunakan untuk menentukan item agar sejajar di antara item secara horisontal. Nilai yang terdapat pada properti justify-content diantaranya.

1. flex-start: item dikemas sehingga terletak di awal container
2. flex-end: item dikemas sehingga terletak di akhir container
3. center: item dipusatkan sepanjang garis horisontal
4. space-between: item didistribusikan secara merata di baris, item pertama berada di awal, item terakhir berada di akhir.
5. space-around: item didistribusikan secara merata dalam garis dengan ruang yang sama di sekitarnya.
6. space-evenly: jarak antar item dan ruang tepi sama besar.

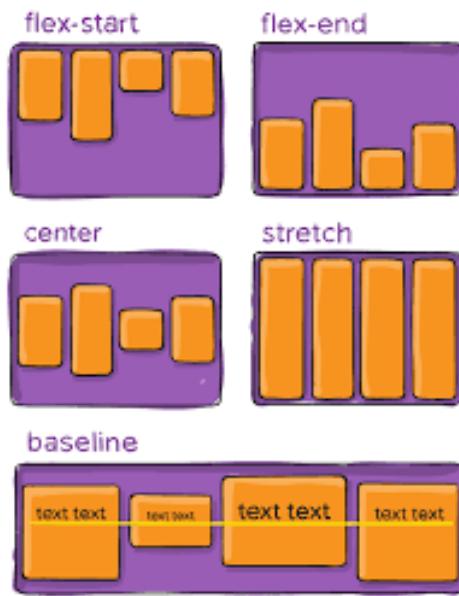


Gambar 2: Visualisasi justify-content

### B.2.3 align-items

Align-items digunakan untuk mengatur item agar berfokus pada cross-axis (garis vertikal). Properti ini juga dapat digunakan untuk mendistribusikan sisa ruang kosong ketika semua item fleksibel pada baris tidak fleksibel. Beberapa nilai yang terdapat pada properti align-items diantaranya.

1. `flex-start`: item dikemas sehingga berada di awal garis vertikal
2. `flex-end`: item dikemas sehingga berada di akhir garis vertikal
3. `center`: item dikemas sehingga terpusat pada container
4. `stretch`: mengisi ruang yang kosong pada container.
5. `baseline`: semua item akan sejajar seperti garis dasar yang sejajar

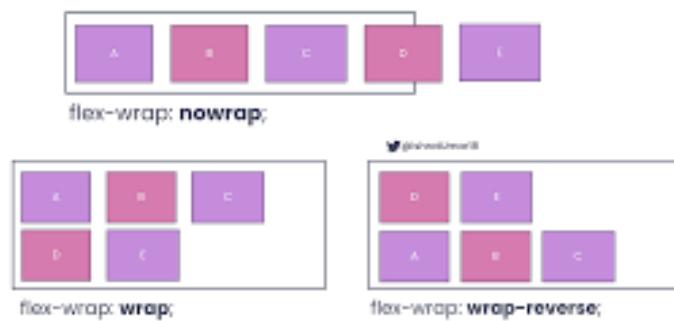


Gambar 3: Visualisasi align-items

#### B.2.4 flex-wrap

Secara default, semua item yang berada di satu container akan mengisi satu baris meskipun ukurannya dapat mengecil. Dengan adanya flex-wrap, item tersebut dapat bergeser ke bawah, sehingga ukurannya tidak berubah (mengecil). Nilai dalam properti flex-wrap diantaranya yaitu.

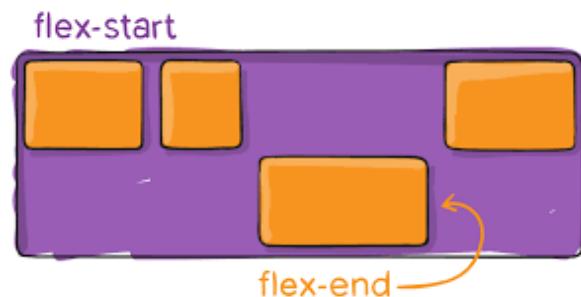
1. wrap: item akan membungkus beberapa baris dari atas ke bawah.
2. wrap-reverse: item akan membungkus beberapa baris dari bawah ke atas.



Gambar 4: Visualisasi flex-wrap

### B.2.5 align-self

properti align-self digunakan untuk mengatur satu elemen di dalam container. nilai yang terdapat dalam properti align-self yaitu flex-start, flex-end, center, stretch.



Gambar 5: Visualisasi align-self

### B.2.6 order

Properti order digunakan untuk mengontrol urutan elemen saat muncul dalam container flex. nilai 0 berarti elemen tetap berada di posisinya (default), nilai -1 akan meletakkan elemen berada di paling kiri baris, sedangkan nilai 1 akan meletakkan elemen di ujung kanan baris.



Gambar 6: Visualisasi order

## C Hasil dan Pembahasan

### C.1 Latihan Praktikum 1

Membuat halaman html dengan mengimplementasikan flexbox secara sederhana.

```
1 <body>
2   <div class="container">
3     <div class="item">1</div>
4     <div class="item">2</div>
5     <div class="item">3</div>
6     <div class="item">4</div>
7     <div class="item">5</div>
8     <div class="item">6</div>
9   </div>
10 </body>
```

Kode html tersebut merupakan struktur awal dari container yang di dalamnya terdapat 6 item. Item dalam container dapat disesuaikan ukuran dan posisinya dengan menggunakan flexbox css.

```
1 .container {
2   padding: 0;
3   margin: 0;
4   flex-flow: row wrap;
5   display: flex;
6   justify-content: space-around;
7   border: solid 2px rgb(109, 0, 217);
8   list-style: none;
9 }
10 .item {
11   line-height: 60px;
12   background-color: plum;
13   text-align: center;
14   width: 100px;
15   height: 60px;
16   padding: 5px;
17   font-weight: bold;
18 }
```

Berdasarkan kode css tersebut, container menggunakan display flex agar item di dalamnya dapat diatur dengan fleksibel. Menggunakan flex-flow: row-wrap agar item di dalam container disusun sesuai barisnya, serta menggunakan justify-content: space-around agar jarak antar item sama besarnya. Hasil dari kode tersebut dapat dilihat pada gambar 7



Gambar 7: Output dari latihan praktikum 1

## C.2 Latihan Praktikum 2

Membuat halaman web semantik dari header hingga footer yang responsive.

```
1 <body>
2   <div class="wrapper">
3     <header class="header">Header</header>
4     <article class="main">
5       <p>
6         Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed
7         do eiusmod tempor incididunt ut labore et dolore magna aliqua.
8         Ut enim ad minim veniam, quis nostrud exercitation ullam
9         laboris nisi ut aliquip ex ea commodo
10        consequat. Duis aute irure dolor in reprehenderit in voluptate
11        velit esse cillum dolore eu fugiat nulla pariatur. Excepteur
12        sint occaecat cupidatat non proident, sunt in culpa qui
13        officia deserunt mollit anim id est laborum
14      </p>
15    </article>
16    <aside class="aside aside-1">Aside 1</aside>
17    <aside class="aside aside-2">Aside 2</aside>
18    <footer class="footer">Footer</footer>
19  </div>
20 </body>
```

Kode html tersebut digunakan dalam merancang struktur awal dalam pembuatan semantik. Terdapat tag header, article, aside, serta footer yang dibalut dalam div class wrapper.

```
1 .wrapper {
2   display: flex;
3   flex-flow: row wrap;
4   font-weight: bold;
5   text-align: center;
6 }
7 .wrapper > * {
8   padding: 10px;
9   flex: 1 100%;
10}
11 .header {
```

```

12     background: tomato;
13 }
14 .footer {
15     background: lightgreen;
16 }
17 .main {
18     text-align: left;
19     background: deepskyblue;
20 }
21 .aside-1 {
22     background: gold;
23 }
24 .aside-2 {
25     background: hotpink;
26 }

```

Css tersebut berguna untuk menambahkan warna pada setiap tag semantik. class wrapper sebagai container dari semua elemen semantik yang dibuat menjadi flex dan memiliki properti flex-flow: row wrap. Artinya memiliki properti flex-direction: row dan flex-wrap: wrap. Secara default, css tersebut berfungsi untuk layar desktop dan akan menghasilkan output seperti pada gambar 8



Gambar 8: Tampilan desktop

Agar tampilan website dapat teratur saat dikecilkan (saat dibuka di hp), maka perlu disesuaikan dengan menggunakan properti @media pada css.

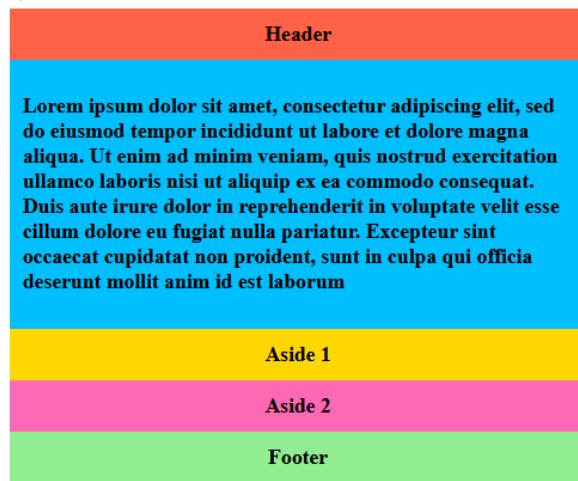
```
1 @media all and (min-width: 600px) {  
2     .aside {  
3         flex: 1 0 0;  
4     }  
5 }  
6 @media all and (min-width: 800px) {  
7     .main {  
8         flex: 30px;  
9     }  
10    .aside-1 {  
11        order: 1;  
12    }  
13    .main {  
14        order: 2;  
15    }  
16    .aside-2 {  
17        order: 3;  
18    }  
19    .footer {  
20        order: 4;  
21    }  
22    body {  
23        padding: 2em;  
24    }  
25 }
```

Pada media query pertama, aturan diterapkan pada layar dengan lebar minimum 600 piksel. Elemen dengan kelas .aside diberikan properti flex: 1 0 0, yang berarti elemen ini akan tumbuh (flex-grow: 1), tidak menyusut (flex-shrink: 0), dan memiliki basis ukuran nol (flex-basis: 0).

Pada media query kedua, yang berlaku untuk layar dengan lebar minimum 800 piksel, beberapa aturan tambahan diterapkan. Elemen dengan kelas .main diberikan properti flex: 30px, yang berarti memiliki basis ukuran 30 piksel tanpa pengaturan pertumbuhan atau penyusutan eksplisit. Selain itu, elemen-elemen dengan kelas .aside-1, .main, .aside-2, dan .footer diberikan properti order, yang menentukan urutan tampilannya dalam flex container.

.aside-1 memiliki urutan pertama (order: 1), diikuti oleh .main (order: 2), .aside-2 (order: 3), dan .footer (order: 4). Dengan demikian, elemen-elemen tersebut akan ditampilkan dalam urutan yang telah ditentukan.

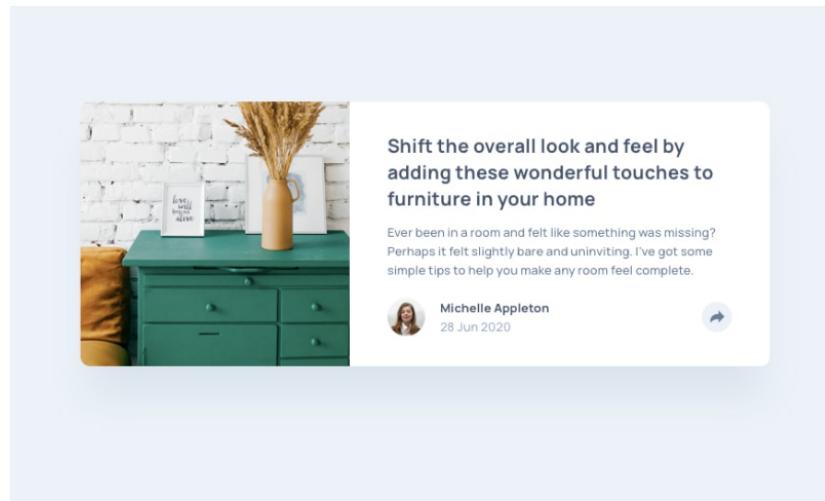
Terakhir, elemen body diberikan properti padding: 2em, yang menambahkan jarak sebesar 2 em di sekeliling kontennya, sehingga memberikan tata letak yang lebih rapi dan nyaman dilihat pada layar dengan ukuran lebih besar.



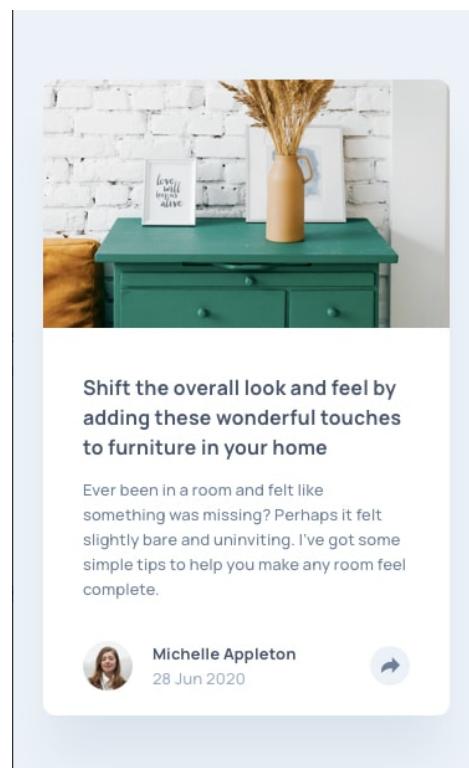
Gambar 9: Tampilan layar dengan ukuran 400px

### C.3 Tugas Praktikum

Mahasiswa diminta untuk menerapkan flexbox dalam mendesain ulang (*slashing*) suatu tampilan web. Berikut merupakan tampilan web yang akan di-*desain* ulang.



Gambar 10: referensi web desktop yang akan di-*slashing*



Gambar 11: referensi web mobile yang akan di-*slashing*

Untuk membuat tampilan kartu seperti soal, dapat membuat container sebagai class parent dari item-item di bawahnya.

```
1 <body>
2   <div class="container">
3     
4     <div class="main">
5       <p class="bold">Shift the overall look and feel by adding these
wonderful touches to furniture in your home</p>
6       <p class="description">Ever been in a room and left like something
was missing? Perhaps it felt slightly bare and uninviting. I've got some
simple tips to help you make any room feel complete</p>
7       <div class="profil">
8         
9         <div class="nama-tgl">
10           <p class="nama">Michelle Appleton</p>
11           <p class="tgl">28 Jun 2020</p>
12         </div>
13         <div class="icon"></div>
14       </div>
15     </div>
16   </div>
17 </body>
```

Elemen pertama di dalam container adalah sebuah gambar dengan tag `<img>` yang memiliki sumber `./images/drawers.jpg` serta atribut `alt="aesthetic"`.

Selanjutnya, terdapat elemen `<div>` dengan kelas `main` yang berisi teks utama dan informasi tentang penulis. Di dalamnya, terdapat sebuah paragraf dengan kelas `bold` yang menampilkan teks utama dalam format yang lebih menonjol, diikuti oleh paragraf dengan kelas `description` yang berisi deskripsi tambahan.

Di bagian bawah, terdapat `<div>` dengan kelas `profil`, yang digunakan untuk menampilkan informasi penulis. Di dalamnya, terdapat sebuah gambar profil dengan sumber `avatar-michelle.jpg` serta teks alternatif "profil". Selain itu, terdapat elemen `<div>` dengan kelas `nama-tgl`, yang berisi dua paragraf: satu untuk menampilkan nama penulis, yaitu "Michelle Appleton", dan satu lagi untuk menampilkan tanggal publikasi, yaitu "28 Jun 2020".

```

1 * {
2   font-family: Arial, Helvetica, sans-serif;
3 }
4 body {
5   min-height: 100vh;
6   display: flex;
7   justify-content: center;
8   align-items: center;
9   background-color: aliceblue;
10 }
11 .container {
12   display: flex;
13   flex-direction: row;
14   position: absolute;
15   box-shadow: rgba(17, 17, 26, 0.1) 0px 8px 24px, rgba(17, 17, 26, 0.1) 0px
16   16px 56px, rgba(17, 17, 26, 0.1) 0px 24px 80px;
17   background-color: rgb(255, 255, 255);
18   height: 230px;
19   border-radius: 12px;
20   max-width: 630px;}
```

Pada bagian pertama, selektor universal (\*) digunakan untuk menerapkan jenis font Arial, Helvetica, atau font sans-serif lainnya ke seluruh elemen dalam halaman.

Selanjutnya, elemen `|body|` diatur agar memiliki tinggi minimum sebesar 100

Bagian utama dari cuplikan artikel dibungkus dalam elemen `|div class="container"|`, yang menggunakan `display: flex` dengan `flex-direction: row`, sehingga elemen-elemen di dalamnya ditampilkan secara horizontal. Properti `position: absolute` digunakan agar elemen ini dapat diatur posisinya secara lebih bebas dalam halaman.

Untuk memberikan efek bayangan, digunakan properti `box-shadow` dengan beberapa nilai transparansi warna hitam untuk menciptakan efek kedalaman. Warna latar belakang kontainer diatur menjadi putih (`background-color: rgb(255, 255, 255)`). Selain itu, tinggi kontainer ditetapkan sebesar 230 piksel (`height: 230px`), dan sudutnya dibuat membulat dengan `border-radius: 12px`. Agar tidak terlalu melebar pada layar yang lebih besar, digunakan properti `max-width: 630px` untuk membatasi lebar maksimum elemen ini.

```
1 .container img {  
2     background-size: cover;  
3     object-position: left;  
4     object-fit: cover;  
5     width: 237px;  
6     border-radius: 10px;  
7     border-top-right-radius: 0;  
8     border-bottom-right-radius: 0;  
9 }
```

Kode CSS tersebut digunakan untuk mengatur tampilan gambar yang berada di dalam elemen dengan kelas .container. Properti background-size: cover; memastikan bahwa gambar akan mengisi seluruh area elemen tanpa mengalami distorsi, dengan menyesuaikan ukuran dan memotong bagian yang tidak sesuai. Properti object-position: left; digunakan untuk menempatkan posisi utama gambar di bagian kiri, sehingga jika terjadi pemotongan, bagian kanan akan lebih banyak terpotong dibandingkan bagian kiri. Selain itu, properti object-fit: cover; memastikan bahwa gambar tetap memenuhi area yang telah ditentukan tanpa mengubah aspek rasio aslinya, sehingga tampilan gambar tetap proporsional dan tidak terdistorsi.

```
1 .bold {  
2     background-color: rgb(255, 255, 255);  
3     margin-top: -5px;  
4     padding-left: 0px;  
5     line-height: 25px;  
6     font-size: 20px;  
7     font-weight: bold;  
8     color: rgb(71, 76, 91);  
9     margin-bottom: 5px;  
10    padding-right: 20px;  
11 }  
12  
13 .main {  
14     display: flex;  
15     flex-direction: column;  
16     border-radius: 10px;
```

```

17 background-color: rgb(255, 255, 255);
18 position: relative;
19 padding: 25px;
20 padding-right: 30;
21 }
22
23 .description {
24   line-height: 17px;
25   color: rgba(54, 59, 119, 0.625);
26   font-size: 12px;
27   font-weight: lighter;
28 }
```

Kelas bold digunakan untuk mengatur tampilan teks yang bersifat menonjol, seperti judul atau bagian utama dari suatu artikel. Warna latar belakang elemen ini diatur menjadi putih dengan background-color: rgb(255, 255, 255);. Properti margin-top: -5px; digunakan untuk sedikit menggeser elemen ke atas, sedangkan margin-bottom: 5px; memberikan ruang di bagian bawah. Properti padding-left: 0px; dan padding-right: 20px; mengatur jarak antara teks dengan batas elemen, dengan jarak lebih besar di sebelah kanan agar teks tidak terlalu rapat ke tepi. Baris teks dibuat lebih renggang dengan line-height: 25px;, sementara ukuran huruf diatur sebesar font-size: 20px; agar lebih mudah dibaca. Untuk memberikan efek penekanan, properti font-weight: bold; digunakan agar teks terlihat lebih tebal. Warna teks diatur dengan color: rgb(71, 76, 91);, yang merupakan warna abu-abu kebiruan untuk menciptakan kontras yang nyaman di mata.

Kelas main digunakan sebagai wadah utama untuk teks dan elemen lainnya di dalam komponen. Properti display: flex; dengan flex-direction: column; membuat elemen-elemen di dalamnya tersusun secara vertikal. Sudut elemen dibuat sedikit membulat menggunakan border-radius: 10px;, dan warna latar belakangnya ditetapkan menjadi putih dengan background-color: rgb(255, 255, 255);. Properti position: relative; memungkinkan elemen ini untuk diposisikan secara lebih fleksibel dalam tata letak. Ruang dalam elemen ini diatur menggunakan padding: 25px; untuk memberikan jarak dari tepi elemen ke kontennya, namun properti padding-right: 30; tidak memiliki satuan yang jelas, sehingga kemungkinan besar tidak akan berfungsi sebagaimana mestinya.

Kelas `description` digunakan untuk mengatur tampilan teks deskriptif yang menjelaskan isi artikel secara ringkas. Jarak antar baris diatur dengan `line-height: 17px;` agar teks tetap terbaca dengan nyaman. Warna teks diberikan efek transparansi dengan `color: rgba(54, 59, 119, 0.625);`, yang menghasilkan warna biru keabu-abuan.

Untuk membuat tampilan dapat dilihat pada mobile, dapat mengatur layout menggunakan properti `@media` seperti pada kode berikut.

```
1 @media (max-width: 630px) {  
2     .container {  
3         height: 75vh;  
4         flex-direction: column;  
5         width: 350px;  
6     }  
7     .container img {  
8         width: 100%;  
9         border-radius: 0px;  
10        border-top-right-radius: 10px;  
11        border-top-left-radius: 10px;  
12    }  
13    .profil img {  
14        border-radius: 50%;  
15        width: 10%;  
16    }  
17}
```

### 1. Media Query (@media (max-width: 630px))

Digunakan untuk menyesuaikan tampilan saat lebar layar kurang dari atau sama dengan 630 piksel (versi responsif untuk perangkat kecil).

### 2. Kelas container

`height: 75vh;` → Mengubah tinggi elemen menjadi 75% dari tinggi viewport.

`flex-direction: column;` → Mengubah susunan elemen dalam kontainer menjadi vertikal.

`width: 350px;` → Menetapkan lebar maksimum kontainer menjadi 350 piksel.

### 3. Kelas container img

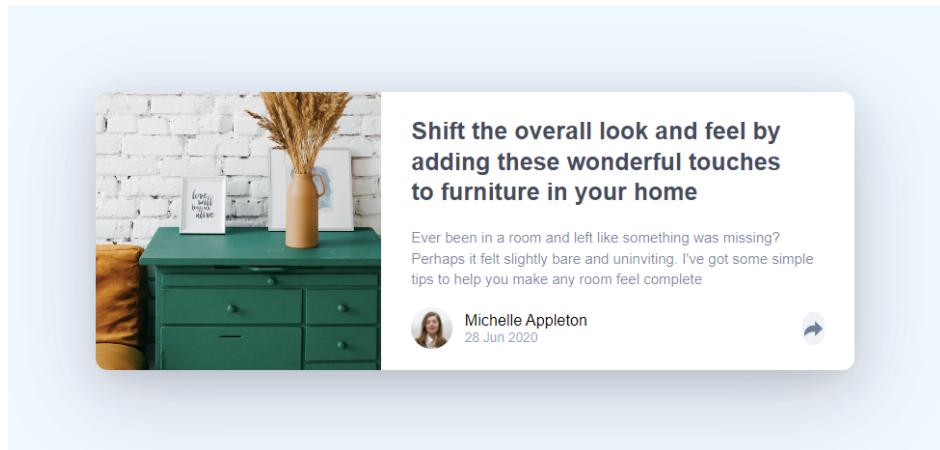
`width: 100%;` → Membuat gambar memenuhi seluruh lebar kontainer.

border-radius: 0px; → Menghilangkan efek sudut membulat pada semua sisi gambar.

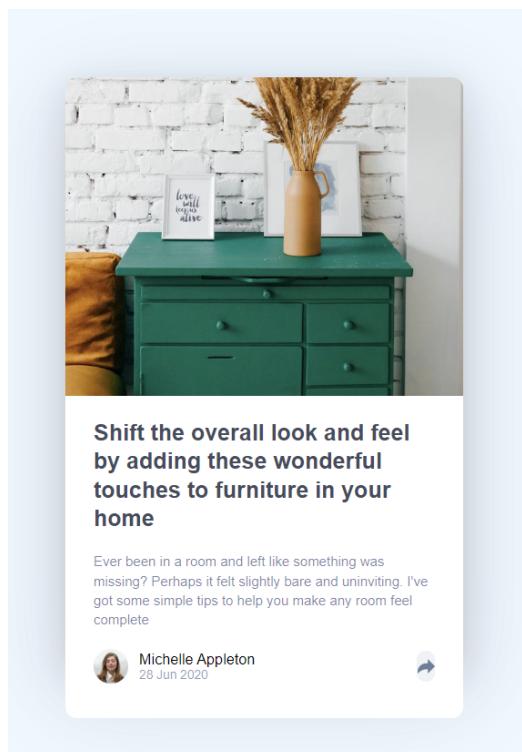
#### 4. Kelas profil img

border-radius: 50%; → Membuat gambar profil menjadi lingkaran.

width: 10%; → Mengatur ukuran gambar profil menjadi 10% dari lebar elemen induknya.



Gambar 12: Output slashing desktop view



Gambar 13: Output slashing mobile view

## D Kesimpulan

Berdasarkan praktikum yang telah dilakukan, dapat diambil kesimpulan bahwa kode html dan css tersebut bertujuan untuk membangun tampilan komponen artikel yang responsif, estetis, dan mudah dibaca. Struktur HTML digunakan untuk menyusun elemen-elemen utama, seperti gambar, teks, serta informasi profil. Sementara itu, CSS digunakan untuk mengatur tata letak, warna, ukuran, dan jarak antar elemen agar tampilan lebih menarik serta nyaman dilihat. Dalam kode CSS ini, digunakan konsep Flexbox, yaitu teknik tata letak yang memungkinkan elemen dalam suatu kontainer tersusun lebih fleksibel dan dinamis. Properti yang digunakan dalam Flexbox meliputi justify-content, yang mengatur perataan elemen secara horizontal; align-items, yang mengontrol penyelarasan elemen secara vertikal; serta flex-direction, yang menentukan apakah elemen disusun dalam baris (row) atau kolom (columns).

Selain itu, terdapat properti order yang menentukan urutan tampilan elemen dalam kontainer flex, di mana nilai yang lebih kecil akan ditampilkan lebih awal. Properti flex-wrap juga diterapkan untuk mengontrol apakah elemen tetap dalam satu baris (nowrap) atau berpindah ke baris berikutnya (wrap) saat ruang tidak mencukupi. Penerapan Flexbox dalam kode ini memungkinkan tata letak yang lebih responsif dan rapi, terutama ketika dikombinasikan dengan aturan @media. Pada tampilan dengan lebar kurang dari 630 piksel, tata letak kontainer diubah menjadi vertikal (flex-direction: column), serta ukuran dan bentuk gambar disesuaikan agar lebih optimal pada perangkat dengan layar kecil. Properti border-radius: 50% pada gambar profil juga memberikan efek lingkaran untuk tampilan yang lebih estetis dan modern.

Secara keseluruhan, kombinasi HTML dan CSS dalam kode ini menunjukkan bagaimana teknik desain web modern dapat diterapkan untuk menciptakan tampilan yang bersih, responsif, dan mudah digunakan. Dengan mengoptimalkan penggunaan Flexbox, tata letak elemen menjadi lebih fleksibel dan adaptif terhadap berbagai ukuran layar. Selain itu, pemanfaatan berbagai properti CSS membantu menciptakan pengalaman pengguna yang lebih baik dengan tampilan yang profesional dan nyaman dilihat di berbagai perangkat.

## E Daftar Pustaka

*Belajar Flexbox CSS dengan Contoh Sederhana 2025 — RevoU — revou.co.* (n.d.).

<https://www.revou.co/panduan-teknis/flexbox-css#:~:text=flexbox%20CSS%20adalah%20sistem%20layout,Begini%20cara%20menggunakannya!&text=Belajar%20di%20RevoU!,dari%20praktisi%20terbaik%20di%20bidangnya>. ([Accessed 20-03-2025])

Priwanto, D. (2024). *Belajar CSS Bagian 12: Fungsi dan Contoh CSS Flexbox Layout* — rumahweb.com. <https://www.rumahweb.com/journal/belajar-css-bagian-12/>. ([Accessed 24-03-2025])