

# **MODUL PRAKTIKUM**

## **PEMROGRAMAN WEB**



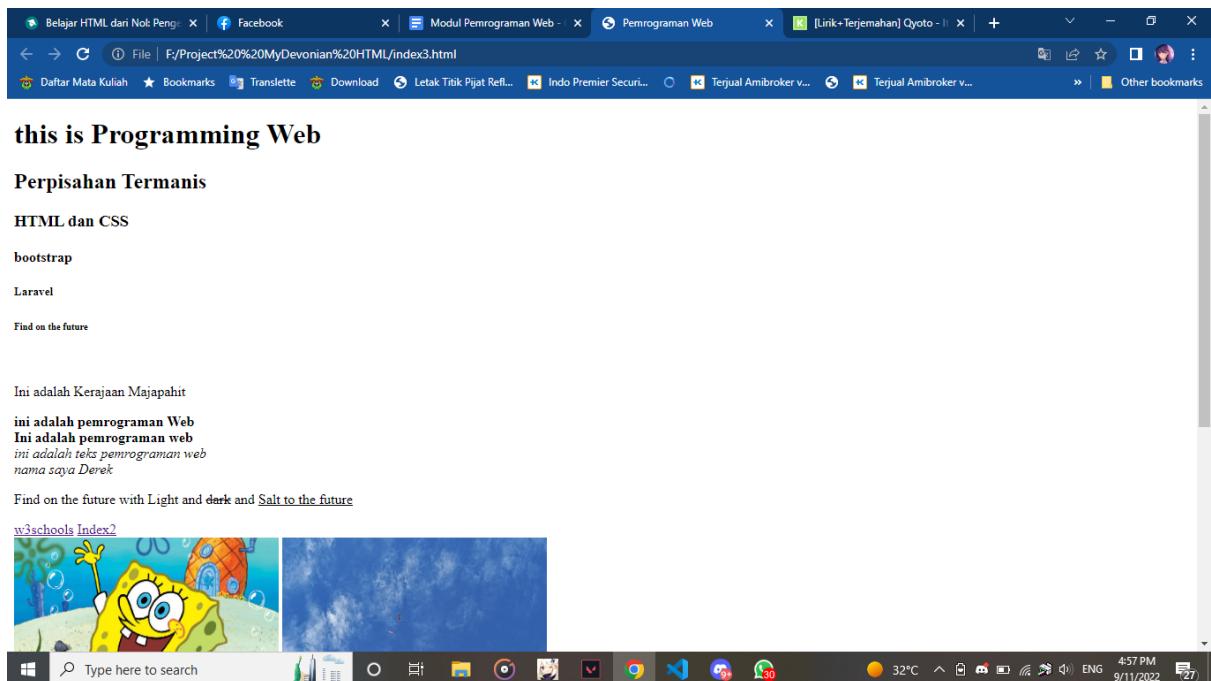
**LABORATORIUM REKAYASA PERANGKAT LUNAK**  
**DEPARTEMEN MATEMATIKA**  
**FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM**  
**UNIVERSITAS HASANUDDIN**  
**MAKASSAR**  
**2022**

# BAB I

## HTML

HTML adalah salah satu web Syntax Hyper Text Markup Language dasar tag dari web browser . Format file html itu yaitu .htm, .html , dan .xhtml (untuk XHTML) jika tidak menggunakan ekstensi, maka tidak bisa baca di web browser. jadi harus nama file format benar. Contoh:  
Index2.html, Index2.htm

contoh Browser web tab :



### Struktur Dasar HTML :

Pada Struktur Dasar HTML ini bisa kita lihat dari kode :

```
<!DOCTYPE html>

<html>

<head>

<title>Judul Halaman Web</title>
```

```
</head>

<body>

    <h1>Modul Praktikum </h1>

    <p>Konten halaman web.</p>

</body>

</html>
```

Struktur dasar html Terbagi atas empat:

1. `<!DOCTYPE html>`
2. `<head>`
3. `<title>`
4. `<body>`

Pengenalan Atribut atribut Elemen Paragraf:

`<p>` = Paragraf  
`<br>` = paragraf Baris baru  
`<hr>` = paragraf garis lurus  
`<div>` = paragraf diluar artikel atau layout

Contoh code :

```
<!DOCTYPE html>

<html>

    <head>

        <title>Judul Halaman Web</title>

    </head>

    <body>

        <h1>Modul Praktikum </h1>
```

```
<p>Konten halaman web.</p>

<br>

<hr>

<div></div>

</body>

</html>
```

### **Pengenalan Tag :**

Sebuah Tag dengan ditandai dengan awalan dan akhiran dari elemen HTML. Tag ini dibuat dalam kurung siku “<...>” seperti ini, jadi berisi di dalam tag itu terdapat nama tag dan kadang juga ditambahkan dengan atribut. Pada dasarnya sebuah tag pembuka dan ada juga tag penutup yang ditulis dengan (/) garis miring “</...>”

### **Membuat Format Text:**

#### **Paragraf :**

Paragraf adalah sebuah Paragraf dimana teks ini dimulai dari baris baru, dan browser secara otomatis menambahkan margin sebelum dan sesudah paragraph.

```
<p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Repellat  
magnam doloribus accusantium facilis possimus ducimus velit a omnis?  
Facere suscipit odio, repudiandae numquam earum eaque. Aspernatur  
veniam fugiat ea voluptatum.</p>
```

### **Bold dan Strong :**

Bold adalah tulisan teks tebal dari tag “**“** untuk mendefinisikan teks tebal tulisan di browser web tanpa kepentingan ekstra.

Contoh :

```
<b> Ini adalah pemrograman web </b>
```

Output:

**ini adalah pemrograman Web**

Strong adalah tulisan teks penting dari tag “**“** untuk mendefinisikan teks yang kuat

Contoh :

```
<strong>ini adalah pemrograman web</strong>
```

Output:

**Ini adalah pemrograman web**

### **Italic dan em**

Italic adalah tulisan teks miring dari tag “*“* untuk mendefinisikan suasana hati alternatif. Teks ini ditampilkan dengan huruf miring.

Contoh :

```
<i>ini adalah teks pemrograman web</i>
```

Output :

*ini adalah teks pemrograman web*

Em adalah tulisan teks yang ditekankan. Konten ini dia outputnya huruf miring.

Contoh :

```
<em> nama saya Derek </em>
```

Output:

*nama saya Derek*

### Del and Ins

Delete adalah teks yang telah dihapus dari dokumen. Konten dalam hasil output web browser ini teks mencoret garis lurus.

Ins adalah teks yang dimasukkan ke dalam dokumen. konten ini hasil output berupa teks garis bawah.

Contoh:

```
<p>Find on the future with Light and <del>dark</del> and <ins>Salt to  
the future</ins></p>
```

Output :

Find on the future with Light and ~~dark~~ and Salt to the future

### Hyperlink :

Seperti yang kita diketahui apa itu Hyperlink. Hyperlink adalah sebuah tag link dimana tag browser link itu meloncat ke WWW sehingga bermunculan pada link tersebut

bahkan ada juga ke dokumen lainnya . Hyperlink ini tag tautan seperti “<a>” . Rumus dari Hyperlink adalah :

```
<a href="url">link text</a>
```

Contohnya bisa kita lihat seperti link dibawah ini:

[https://www.w3schools.com/html/html\\_links.asp](https://www.w3schools.com/html/html_links.asp)

index2.html

contoh codenya :

```
<a  
href="https://www.w3schools.com/html/html_paragraphs.asp">w3schools</a>  
<a href="index2.html">Index2</a>
```

output :

[w3schools Index2](#)

Form :

Form adalah sebuah Membuat HTML dengan Mengisi input dalam HTML .

Form ini terbagi atribut :

1. action => menentukan aksi yang dikirim atau diterima
2. method => Mengirim data dalam aksi pada server.

Apa itu Field ?

Field adalah sebuah ruas yang ditempatkan pada data.

jadi Field terbagi atas 2 :

1. Type : Type dari field yang ditempatkan.
2. name : nama dari field yang menjadi kunci dan variabel akan dipakai program.

Contoh elemen HTML <form> .

```
<form action = "login.php" method = "post">  
<fieldset>  
    <legend>
```

```

        Login
    </legend>
    <p>
        <label>Username</label>
        <input type="text" name="Username"
placeholder="username..."/>
    </p>
    <p>
        <label>Password</label>
        <input type="password" name="password"
placeholder="password..." />
    </p>
    <p>
        <label><input type="checkbox" name="remember"
value="remember" /> Remember me</label>
    </p>
    <p>
        <input type="submit" name="submit" value="Login" />
    </p>
    </fieldset>
</form>

```

Output :

The image shows a simple HTML login form. At the top, there is a legend with the text "Login". Below it is a "p" element containing a "label" for "Username" followed by a text input field with the placeholder "username...". The next "p" element contains a "label" for "Password" followed by a password input field with the placeholder "password...". Below these fields is another "p" element containing a "label" for a checkbox. The checkbox is preceded by the text "Remember me". At the bottom of the form is a "p" element containing a "input" element with the type "submit" and the value "Login".

**Membuat Gambar, Audio, Video , Atau Dokumen lain:**

### Gambar :

Tag HTML <img> digunakan untuk menaruhkan atau menempatkan gambar sehingga ditampilkan gambar pada web browser.

Gambar secara teknis tidak dimasukkan ke dalam halaman web; gambar ditautkan ke halaman web. Tag <img> menciptakan ruang penyimpanan untuk gambar yang direferensikan.

Tag <img> kosong, hanya berisi atribut, dan tidak memiliki tag penutup.

Tag <img> memiliki dua atribut wajib:

src = Menentukan jalur ke gambar

alt = Menentukan teks alternatif untuk gambar yang ditentukan kategori

width and height = untuk menentukan ukuran pada gambar lebar dan tinggi tersebut.

Contoh :

```

```

```


<img src = "20220817_110150.jpg" alt = "langit" width = "300" height="200">
```

Output :



### Video :

video adalah sebuah tag dimana video ini dapat difokuskan dan dimainkan secara langsung atau tidak langsung dalam video maupun film, Video ini berisi tag “<video>”

terdapat tag :

controls => Dimana menambah atribut fokus, biar play, pause bahkan ada volume

width and height = Dimana mengatur ukuran video lebar dan tinggi

HTML Format Dari bisa terbaca type video MP4, OGG, Webm :

- Mozilla => Ok
- Edge => Ok
- Safari => Ogg tidak bisa
- Chrome => Ok

HTML Type Video :

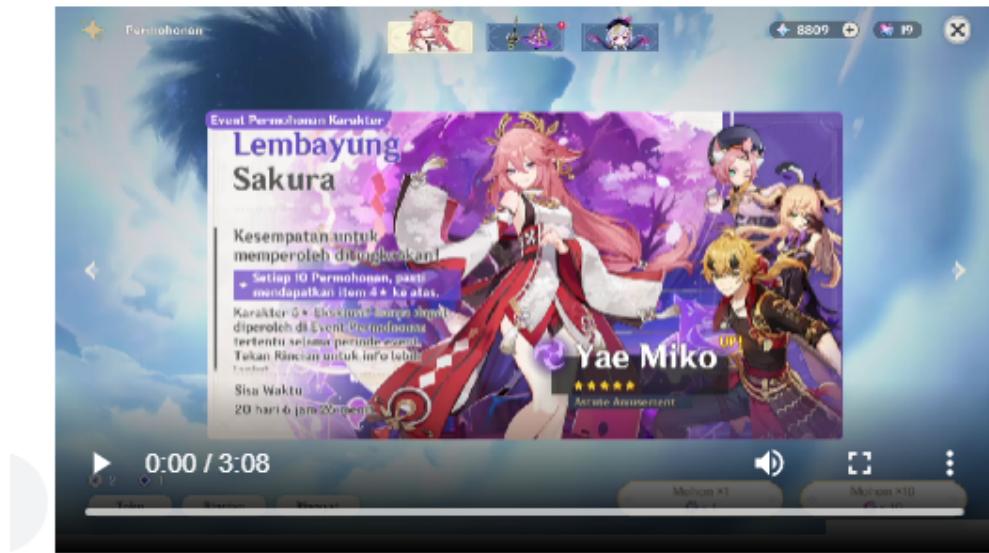
MP4 => video/mp4

Ogg => video/ogg

WebM => video/webm

```
<video width = "500" height="300" controls>
  <source src="Song Of devon/2022-02-16 11-32-56.mp4" type="video/mp4">
</video>
```

output :



Membuat tabel :

Table adalah salah satu mendefinisikan tabel dari web browser untuk mengatur data table ke dalam baris dan kolom.

Tabel tag <tr> singkatan dari Tabel baris.

Tabel tag <th> singkatan dari tabel header.

Tabel tag <td> singkatan dari tabel data.

Contoh :

```
<table border="1">
<tr>
    <th>Nama Mahasiswa</th>
    <th>NIM</th>
    <th>Fakultas</th>
</tr>
<tr>
    <td>Eurico Devon Bura Pakilaran</td>
    <td>H071191048</td>
    <td>MIPA</td>
</tr>
<tr>
```

```

<td>Muh Ikhsan</td>
<td>H071191048</td>
<td>MIPA</td>
</tr>
<tr>
    <td>Muhammad Fajri Rashid</td>
    <td>H071191051</td>
    <td>MIPA</td>
</tr>
</table>

```

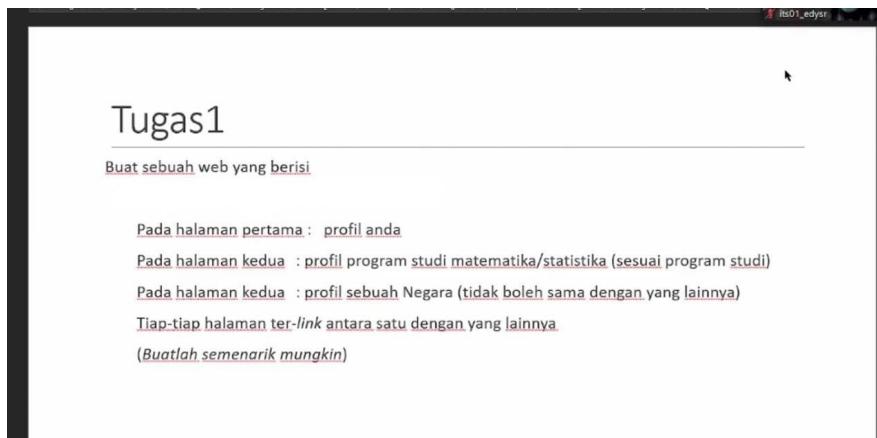
Output :

Nama Mahasiswa	NIM	Fakultas
Eurico Devon Bura Pakilaran	H071191048	MIPA
Muh Ikhsan	H071191048	MIPA
Muhammad Fajri Rashid	H071191051	MIPA

## ***TUGAS PRAKTIKUM***

**KELAS A**

**KELAS B**



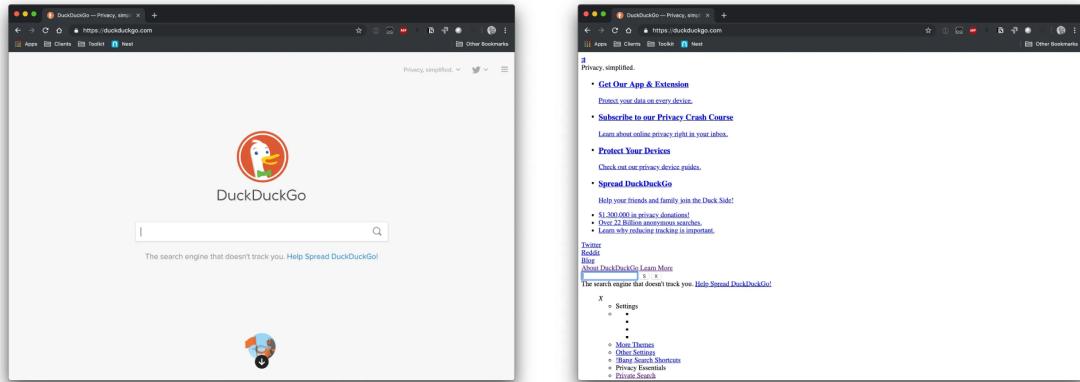
**KELAS C**

**KELAS D**

## BAB II

### CSS

Sebelumnya kita telah mempelajari *Hyper Text Mark-up Language* atau *HTML* yang merupakan pondasi dasar dalam pembuatan sebuah website. Sekarang kita akan mempelajari tentang CSS atau *Cascading Style Sheet* dimana ini akan menjadi unsur utama dalam penentuan tampilan website. CSS unik karena tidak membuat elemen baru, seperti *HTML* atau *JavaScript*. Sebaliknya, ini adalah bahasa yang digunakan untuk menata elemen *HTML*. CSS bertanggung jawab atas gaya teks, ukuran, posisi, warna, dan lainnya di situs web. Itu juga yang mengontrol bagaimana gaya situs web bergerak antara versi desktop dan seluler. Tanpa CSS, situs web akan terlihat sangat membosankan.

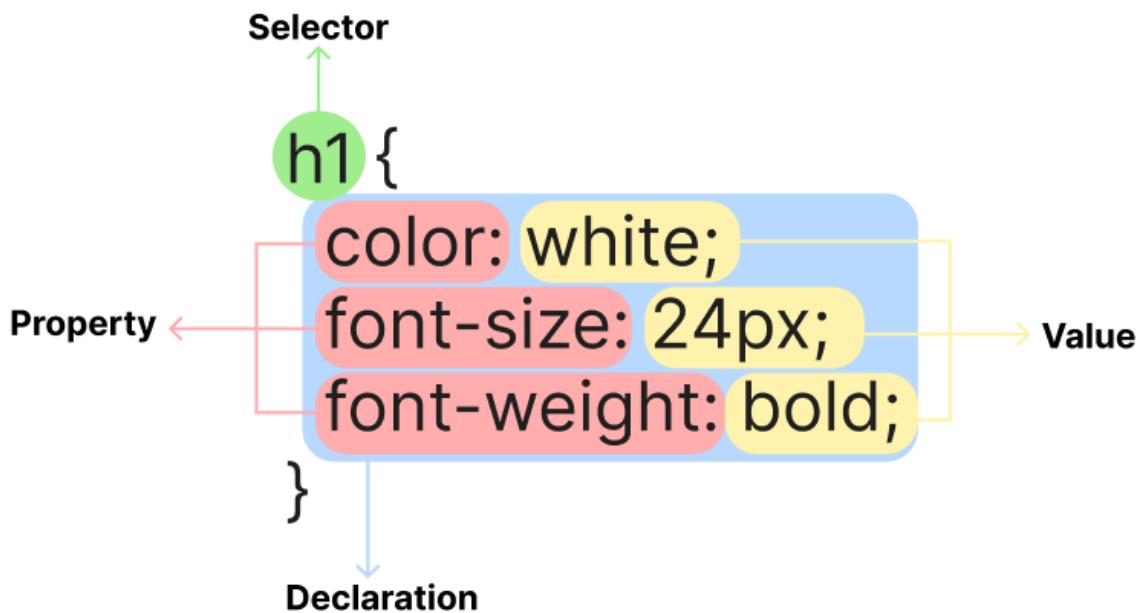


Perbandingan website yang menggunakan css dan tidak

Seperti yang bisa dilihat di gambar CSS akan sangat membantu kita dalam pengaturan tampilan website.

### Struktur CSS

Struktur dasar dari CSS terbagi atas dua aturan, yaitu *Selector* dan *Declaration*. dimana *Selector* Berfungsi untuk memilih *elemen* *HTML* yang akan diatur *styling*-nya. sedangkan *Declaration* berguna sebagai aturan styling yang akan diterapkan pada *elemen* *HTML* yang telah dipilih tadi.



*Struktur syntax umum dari CSS*

Kegunaan dari tiap-tiap keterangan diatas adalah :

- **Selector** : Berfungsi untuk memilih atau membatasi elemen HTML yang akan kita *styling*
- **Declaration** : Menerapkan kumpulan aturan *styling* pada elemen HTML yang telah dipilih
- **Property** : merupakan atribut-atribut *styling* yang dimiliki dan bisa dimodifikasi oleh sebuah elemen HTML
- **Value** : Nilai dari atribut-atribut yang dimiliki oleh setiap elemen HTML

## *Cara Kerja CSS*

1. *Browser* memuat HTML (misalnya menerimanya dari jaringan).
2. Ini mengubah HTML menjadi DOM (*Document Object Model*). DOM mewakili dokumen dalam memori komputer. DOM dijelaskan sedikit lebih detail di bagian selanjutnya.
3. Browser kemudian mengambil sebagian besar sumber daya yang ditautkan oleh dokumen HTML, seperti gambar yang disematkan, video, dan bahkan CSS yang ditautkan! JavaScript ditangani sedikit lebih lambat dalam prosesnya

4. **Browser mem-parsing CSS yang diambil, dan mengurutkan aturan yang berbeda berdasarkan jenis pemilihnya ke dalam *section* yang berbeda**, misal *element*, *Class*, *ID*, dan sebagainya. Berdasarkan penyeleksi yang ditemukan, ia menentukan aturan mana yang harus diterapkan ke *Node* mana di DOM, dan menempelkan styling padanya sesuai kebutuhan (langkah perantara ini disebut pohon render).
5. Pohon render diletakkan dalam struktur yang seharusnya muncul setelah aturan diterapkan padanya.
6. Tampilan visual halaman ditampilkan di layar (tahap ini disebut lukisan).

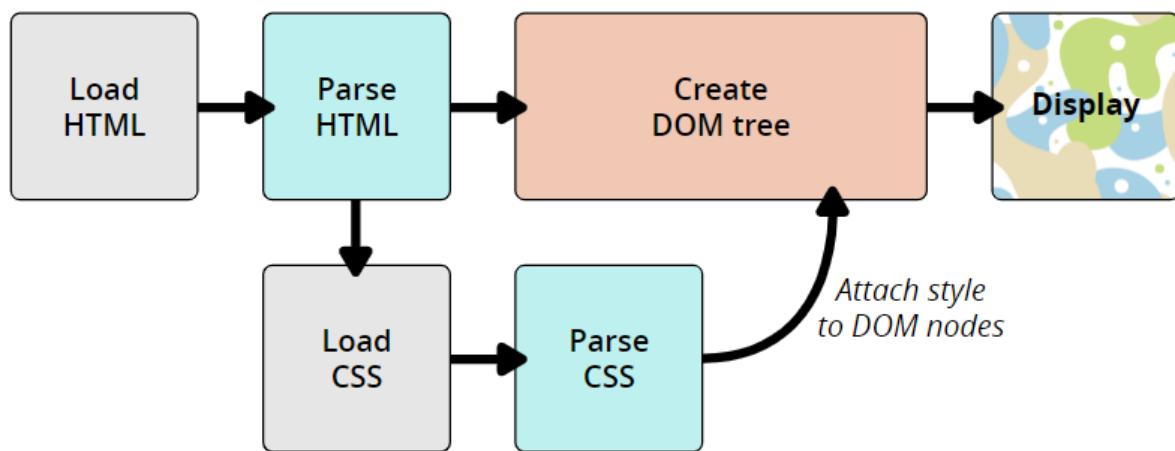


Diagram cara kerja sebuah website

## Meng-embed file CSS ke HTML

Seperti yang kita tahu hasil *styling* dari CSS hanya bisa dilihat ketika ada elemen HTML sehingga CSS harus di *embed* ke *file* HTML. Untuk meng-embed CSS ke HTML bisa dilakukan dengan beberapa cara yaitu *embed inline*, *internal* dan *external*.

### Inline Embed

Inline embedding pada HTML seperti namanya dia menerapkan langsung aturan *styling* pada elemen HTML sehingga jenis *embed* ini kita tidak memerlukan *selector*. inline embed biasa digunakan untuk memodifikasi elemen-elemen yang unik (sedikit) dalam sebuah HTML . Contoh :

```
<h1 style="border: black; padding: 5px">ini adalah header 1</h1>
```

kode diatas hanya akan merubah style pada satu elemen h1 saja dan tidak akan mempengaruhi elemen h1 lainnya.

## Internal Embed

internal embedding ini juga kita masih menggunakan di file HTML nya saja dan syntax nya di letakan di luar element HTML yaitu di dalam *section head* dalam sebuah HTML. Oleh karena itu disini kita akan tetap menggunakan aturan umum penulisan CSS yang sudah kita pelajari sebelumnya. inline style ini biasa digunakan jika satu halaman HTML memiliki gaya yang unik. Contoh :

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <!-- -----STYLE SHEET----- -->
    <style>
      body {
        background-color: linen;
      }

      h1 {
        color: maroon;
        margin-left: 40px;
      }
    </style>
    <!-- -----STYLE SHEET----- -->
    <title>Document</title>
  </head>
  <body>
    <h1 style="border: black; padding: 5px">ini adalah header 1</h1>

    <h3>Ini adalah Header 3</h3>

    <div class="class">ini adalah kelas "class"</div>

    <a href="" id="id">ini adalah id "id"</a>
  </body>
</html>
```

pada kode diatas maka semua elemen yang ada pada selector akan menerapkan aturan styling pada style sheet. perlu diingat aturan-aturan tersebut hanya berlaku pada file HTML ini.

## External embed

berbeda dengan kedua jenis embed sebelumnya kali ini file CSS-nya akan terpisah dari file HTML-nya. Keuntungan dari metode ini, kita akan lebih fleksibel dalam peng-stylean elemen-elemen html nanti karena kita bisa ubah tampilan seluruh situs web dengan mengubah hanya satu file. namun kita harus sertakan referensi ke file style sheet eksternal di dalam elemen <link>, di dalam bagian head. Contoh

File HTML : *index.html*

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <link rel="stylesheet" href="style.css" /> <!--REFERENSI KE FILE CSS-->

    <title>Document</title>
  </head>
  <body>

    <h3>Ini adalah Header 3</h3>

    <div class="class">ini adalah kelas "class"</div>

    <a href="" id="id">ini adalah id "id"</a>
  </body>
</html>
```

File CSS : *style.css*

```
h3 {
  color: red;
  font-weight: bold;
}

.class {
  width: 200px;
  height: 200px;
  background-color: brown;
  color: white;
}
```

```
#id {  
    font-size: 32px;  
    background-color: aqua;  
}
```

seperti pada umumnya ini akan mengubah style semua elemen yang terpilih. dan ini akan berguna jika kita memiliki banyak file HTML.

## ***Selector pada CSS***

*Selector* CSS adalah bagian pertama dari Aturan CSS. Ini adalah pola elemen dan istilah lain yang memberi tahu browser elemen HTML mana yang harus dipilih agar nilai properti CSS di dalam aturan diterapkan padanya. Elemen yang dipilih oleh *Selector* disebut sebagai subjek. pada bab ini kita akan mempelajari mengenai 4 *selector* yang umumnya dipakai pada CSS. jenis-jenis selector tersebut yaitu :

1. *Selector* yang menargetkan ***Element HTML***
2. *Selector* yang menargetkan ***Class*** pada HTML
3. *Selector* yang menargetkan ***ID*** pada HTML
4. *Selector* yang menargetkan semua elemen HTML (Universal)

```
/* selector element html */  
h3 { }  
  
/* selector class */  
.class { }  
  
/* selector id */  
#id { }  
  
/* Selector Universal */  
* { }
```

kita juga bisa membuat aturan *styling* yang sama pada beberapa elemen yang berbeda. Dengan cara *grouping selector*.

```
p, span, a, li {  
    color: chartreuse;  
    font-size: 24px;  
}
```

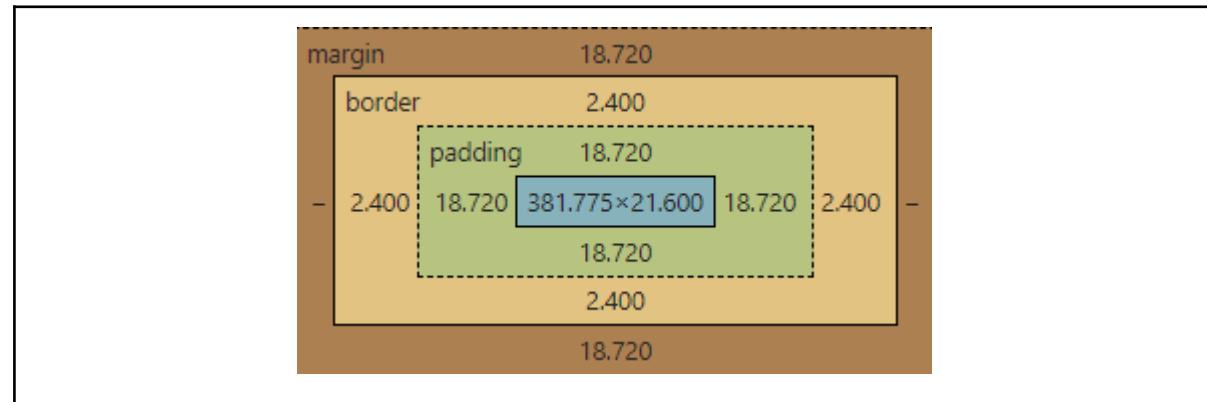
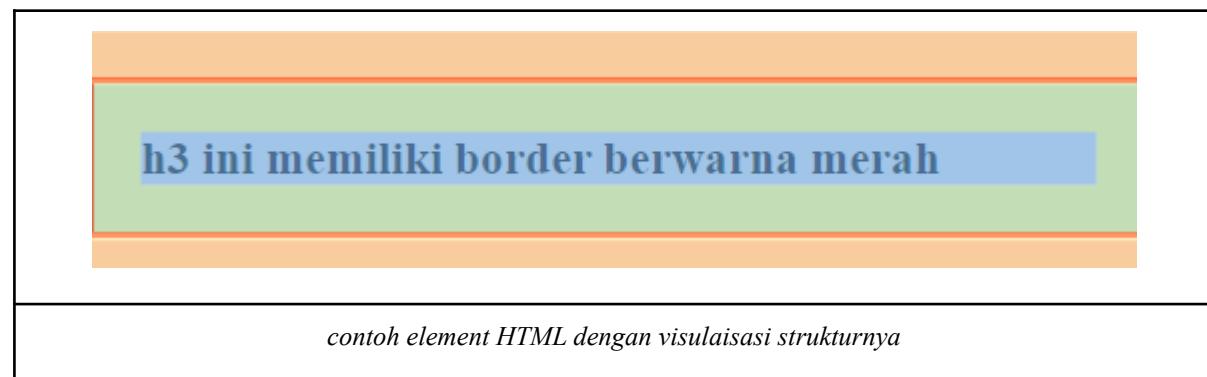
code diatas akan merubah warna dan ukuran dari teks pada elemen : p, span, a, dan li.

## Styling pada CSS

Styling pada CSS sangat bervariasi mulai dari pengaturan tampilan visual sebuah element sampai ke pengaturan posisi dan layout semua elemen. disini kita akan memberikan unsur-unsur yang umumnya muncul dalam penmbuatan CSS pada website.

### Struktur Elemen:

elemen pada HTML direpresentasikan dalam sebuah kotak (*Box*). Kotak ini memiliki struktur *default* yang terdiri dari *margin*, *border*, *padding*, dan elemen itu sendiri.



*diagram struktur element HTML diatas*

struktur ini bisa kita modifikasi melalui CSS dengan properti-properti sebagai berikut:

### ***Margin :***

Margin menentukan jarak box sebuah element dengan box element yang lain. properti margin bisa kita posisikan di empat bagian dalam sebuah element

- *margin-top*
- *margin-bottom*
- *margin-left*
- *margin-right*

atau jika ingin menggunakan di semua sisi kita bisa langsung menggunakan properti `margin`. value dari properti ini menggunakan angka dari beberapa satuan, contoh:

```
.margin{  
    margin: 10px;          /* satuan piksel */  
    margin-top: 10em;      /* satuan rem */  
    margin-bottom: 10%;    /* satuan persen dari ukuran layar */  
    margin-left: 0in;      /* satuan inci */  
    margin-right: 0mm;     /* satuan milimeter */  
}
```

### ***Padding :***

jika margin merupakan jarak antar box element maka padding merupakan jarak antara sebuah element dengan batas box (*border*) dari elemen itu sendiri. seperti margin padding juga bisa di atur dari empat sisi atau semuanya sekaligus

- *padding-top*
- *padding-bottom*
- *padding-left*
- *padding-right*

atau jika ingin menggunakan di semua sisi kita bisa langsung menggunakan properti `padding`. value dari properti ini menggunakan angka dari beberapa satuan, contoh:

```
.padding{  
    padding: 10px;          /* satuan piksel */
```

```
padding-top: 10em; /* satuan rem */
padding-bottom: 10%; /* satuan persen dari ukuran layar */
padding-left: 0in; /* satuan inci */
padding-right: 0mm; /* satuan milimeter */
}
```

### ***border :***

border merupakan batasan terluar dari sebuah element. Batasan ini tentunya bisa di modifikasi dan propertinya adalah border. variasi dari border ada beberapa yaitu

code :

```
<!DOCTYPE html>
<html>
<head>
<style>
p.dotted {border-style: dotted; }
p.dashed {border-style: dashed; }
p.solid {border-style: solid; }
p.double {border-style: double; }
p.groove {border-style: groove; }
p.ridge {border-style: ridge; }
p.inset {border-style: inset; }
p.outset {border-style: outset; }
p.none {border-style: none; }
p.hidden {border-style: hidden; }
p.mix {border-style: dotted dashed solid double; }
</style>
</head>
<body>

<h2>The border-style Property</h2>
<p>This property specifies what kind of border to display:</p>

<p class="dotted">A dotted border.</p>
<p class="dashed">A dashed border.</p>
<p class="solid">A solid border.</p>
<p class="double">A double border.</p>
<p class="groove">A groove border.</p>
```

```

<p class="ridge">A ridge border.</p>
<p class="inset">An inset border.</p>
<p class="outset">An outset border.</p>
<p class="none">No border.</p>
<p class="hidden">A hidden border.</p>
<p class="mix">A mixed border.</p>

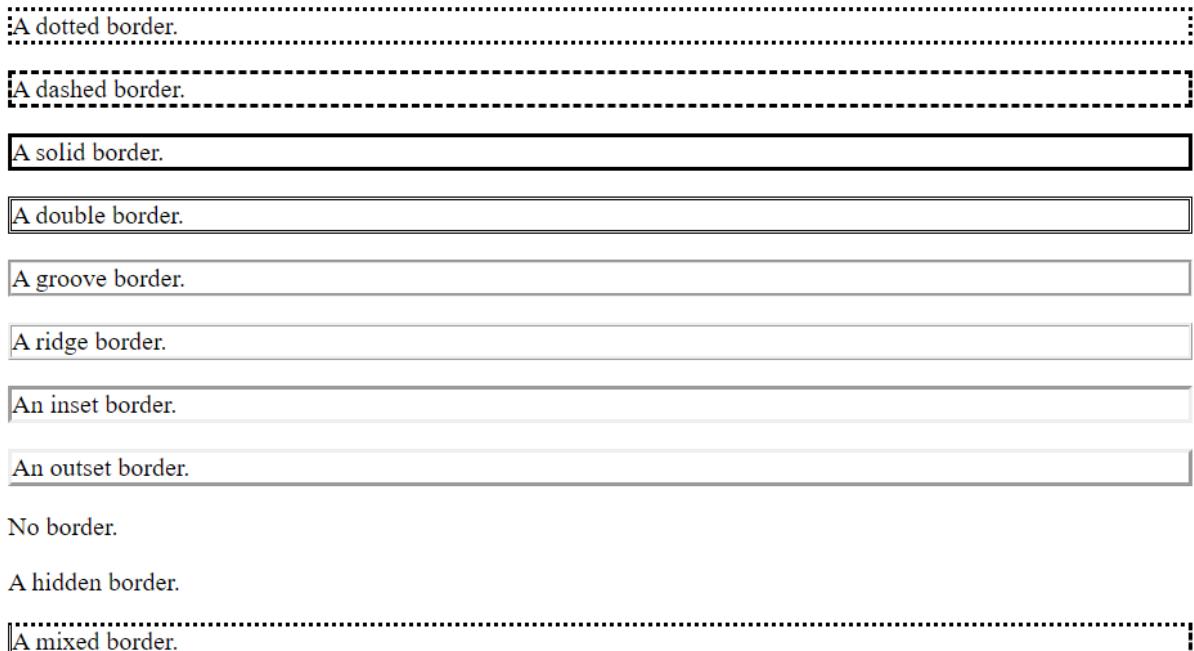
</body>
</html>

```

output:

## The border-style Property

This property specifies what kind of border to display:



### Warna :

Dalam CSS terdapat banyak elemen yang kita bisa modifikasi warnanya mulai dari teks, border, background color dan lain-lain. untuk mengatur warnanya teredapat empat cara yaitu Nama warna, Nilai RGB, Nilai HEX, dan Nilai HSL.

```

.test {
  color: blue;           /* Nama warna */
  color: #000000;        /* Nilai HEX */
  color: hsl(0, 100%, 50%); /* Nilai HSL */

```

```
color: rgb(60, 179, 113); /* Nilai RGB */  
}
```

***color :***

Properti *color* memungkinkan Anda untuk menentukan warna teks di dalamnya sebuah elemen. Anda dapat menentukan apa saja warna dalam CSS dengan salah satu dari empat cara diatas.

code :

```
<h1 style="color: #0000ff">h1 ini berwarna margin biru</h1>
```

output:

**h1 ini berwarna biru**

***background-color :***

CSS memperlakukan setiap elemen HTML seolah-olah itu muncul dalam sebuah kotak, dan properti *background-color* mengatur warna latar belakang untuk kotak itu.

code :

```
<h1 style="background-color: #1919e9">h1 ini berlatar biru</h1>  
<h2 style="background-color: rgb(245, 19, 19);">h2 ini berlatar  
merah</h2>
```

output :

**h1 ini berlatar biru**

**h2 ini berlatar merah**

***border-color :***

setiap element pada HTML memiliki sebuah border namun nilai defaultnya itu 0 sehingga tidak terlihat. Border ini kita bisa modifikasi warnanya menggunakan property *border-color*.

code :

```
<h3 style="border: solid; border-color: red">  
    h3 ini memiliki border berwarna merah  
</h3>
```

output :

**h3 ini memiliki border berwarna merah**

Teks dan font :

CSS memiliki banyak properti untuk memformat atau memodifikasi teks. karena teks tidak akan bisa lepas dari HTML maka unsur teks merupakan bagian inti dalam sebuah website, CSS memiliki banyak properti yang bisa memodifikasi teks-teks dalam HTML, mulai dari jenis font, ukuran font, ketebalan font, dan masih banyak lagi. pada bagian ini kita akan membahas bagian-bagian yang sering digunakan dalam pengayaan teks.

**jenis font:**

Saat memilih jenis huruf, penting untuk dipahami bahwa browser biasanya hanya akan menampilkannya jika diinstal di komputer pengguna tersebut. Akibatnya, situs sering menggunakan sekumpulan kecil tipografi yang dipasang di sebagian besar komputer. Untuk memilih jenis huruf kita bisa menggunakan properti `font-family` pada elemen-elemen yang memiliki teks.

code :

```
<!DOCTYPE html>  
<html>  
<head>  
<style>  
.p1 {  
    font-family: "Times New Roman", Times, serif;  
}  
  
.p2 {  
    font-family: Arial, Helvetica, sans-serif;  
}  
  
.p3 {
```

```

        font-family: "Lucida Console", "Courier New", monospace;
    }
</style>
</head>
<body>

<h1>CSS font-family</h1>
<p class="p1">This is a paragraph, shown in the Times New Roman
font.</p>
<p class="p2">This is a paragraph, shown in the Arial font.</p>
<p class="p3">This is a paragraph, shown in the Lucida Console
font.</p>

</body>
</html>

```

output :

## CSS font-family

This is a paragraph, shown in the Times New Roman font.

This is a paragraph, shown in the Arial font.

**This is a paragraph, shown in the Lucida Console font.**

Nilai properti ini adalah nama jenis huruf yang ingin Anda gunakan. jika nama font terdiri dari lebih dari satu kata, itu harus dimasukkan dalam tanda kutip dua (“\_\_”).

### Ukuran font :

Properti `font-size` memungkinkan Anda untuk menentukan ukuran font. Ada beberapa cara untuk menentukan ukuran font mulai dari satuan *pixel*, *percentage*, *ems* dan *view port width*.

code :

```

<h2 style="font-size: 50px">Contoh elemen h2 pixel</h2>
<h2 style="font-size: 100%">Contoh elemen h2 persen</h2>
<h2 style="font-size: 2em">Contoh elemen h2 ems</h2>
<h2 style="font-size: 5vw">Contoh elemen h2 view port width</h2>

```

output :

# Contoh elemen h2 pixel

Contoh elemen h2 persen

## Contoh elemen h2 ems

## Contoh elemen h2 view port width

*font style :*

Properti `font-weight` memungkinkan Anda membuat teks tebal. Ada dua nilai yang biasanya diambil oleh properti ini yaitu `normal` dan `bold`, namun ada beberapa kasus dimana kita bisa menentukan `font-weight` dengan sebuah angka.

code :

```
<p style="font-weight: bold;">ini adalah paragraph bold</p>
<p style="font-weight: normal;">ini adalah paragraph normal</p>
```

output :

ini adalah paragraph bold

ini adalah paragraph normal

*font style :*

Jika Anda ingin membuat teks miring, Anda dapat menggunakan properti `font-style`. Ada tiga nilai yang dapat diambil oleh properti ini yaitu `normal`, `italic`, dan `oblique`.

*text alignment :*

Properti `text-align` digunakan untuk mengatur perataan horizontal teks. Sebuah teks dapat diratakan kiri atau kanan, dan di tengah. contoh :

code:

```
<h1 style="text-align: center">Heading 1 (center)</h1>
<h2 style="text-align: left">Heading 2 (left)</h2>
```

```
<h3 style="text-align: right">Heading 3 (right)</h3>
```

output :

# Heading 1 (center)

## Heading 2 (left)

### Heading 3 (right)

*Layout :*

CSS memperlakukan setiap elemen HTML seolah-olah berada dalam *box* sendiri. *box* ini akan menjadi block-level elemen atau inline elemen. block-level dimulai pada baris baru dan bertindak sebagai blok penyusun utama tata letak apa pun, sementara inline berada di antara teks di sekitarnya.

<p><b>Lorem Ipsum</b></p> <p>Lore ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit.</p> <ul style="list-style-type: none"><li>• Lore ipsum dolor sit</li><li>• Consectetur adipisicing</li><li>• Elit, sed do eiusmod</li></ul>	<p><i>block-level element</i></p>	<p><b>lorem ipsum</b> dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut <b>labore et dolore</b> magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.</p>  <p>Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum. Lore ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.</p> <p><i>inline element</i></p>
---	-----------------------------------	---

Dalam penentuan posisi dalam CSS kita memiliki tiga skema yang umumnya digunakan untuk mengatur layout sebuah halaman. Skema tersebut antara lain : *normal flow*, *relative positioning*, and *absolute positioning*.

**Normal Flow :**

*Normal Flow* setiap elemen tingkat blok muncul di baris baru, menyebabkan setiap item muncul lebih rendah di halaman daripada yang sebelumnya. Bahkan jika kita menentukan lebar kotak untuk membuat dua element bisa muat dalam satu baris, mereka tidak akan muncul bersebelahan.

Karena ini adalah cara default browser memperlakukan elemen HTML, kita tidak memerlukan properti CSS untuk menunjukkan bahwa elemen harus muncul dalam *normal flow*. namun syntax yang biasa digunakan untuk menentukan skema ini adalah `position: static`.

code :

```
<!DOCTYPE html>
<html>
  <head>
    <style>
      div.static {
        position: static;
        border: 3px solid #73ad21;
      }
    </style>
  </head>
  <body>
    <h2>position: static;</h2>

    <p>
      An element with position: static; is not positioned in any
      special way; it
      is always positioned according to the normal flow of the page.
    </p>

    <div class="static">This div element has position: static;</div>
  </body>
</html>
```

output:

## **position: static;**

An element with `position: static;` is not positioned in any special way; it is always positioned according to the normal flow of the page:

This div element has `position: static;`

### **Relative Positioning:**

Skema ni memindahkan elemen dari posisi yang harusnya berada dalam posisi normal flow, contoh : menggesernya ke atas, kanan, bawah, atau kiri di mana ia akan ditempatkan. Ini tidak mempengaruhi posisi elemen di sekitarnya; mereka tetap di posisi mereka seperti yang seharusnya berada dalam normal flow. kita bisa membuat ini dalam CSS menggunakan syntax position: relative.

```
<!DOCTYPE html>
<html>
  <head>
    <style>
      div.relative {
        position: relative;
        left: 30px;
        border: 3px solid #73ad21;
      }
    </style>
  </head>
  <body>
    <h2>position: relative;</h2>

    <p>
      An element with position: relative; is positioned relative to its
      normal
      position:
    </p>

    <div class="relative">This div element has position:
    relative;</div>
  </body>
</html>
```

output :

### **position: relative;**

An element with position: relative; is positioned relative to its normal position:

---

This div element has position: relative;

---

### **Absolute Positioning :**

Elemen yang diposisikan secara mutlak tidak lagi ada dalam aliran dokumen normal. Sebaliknya, ia duduk di lapisannya sendiri yang terpisah dari yang lainnya.

code :

```
<!DOCTYPE html>
<html>
  <head>
    <style>
      div.relative {
        position: relative;
        width: 400px;
        height: 200px;
        border: 3px solid #73ad21;
      }

      div.absolute {
        position: absolute;
        top: 10px;
        right: 0;
        width: 200px;
        height: 100px;
        border: 3px solid #73ad21;
      }
    </style>
  </head>
  <body>
    <h2>position: absolute;</h2>

    <p>
      An element with position: absolute; is positioned relative to the
      nearest
      positioned ancestor (instead of positioned relative to the
      viewport, like
      fixed):
    </p>

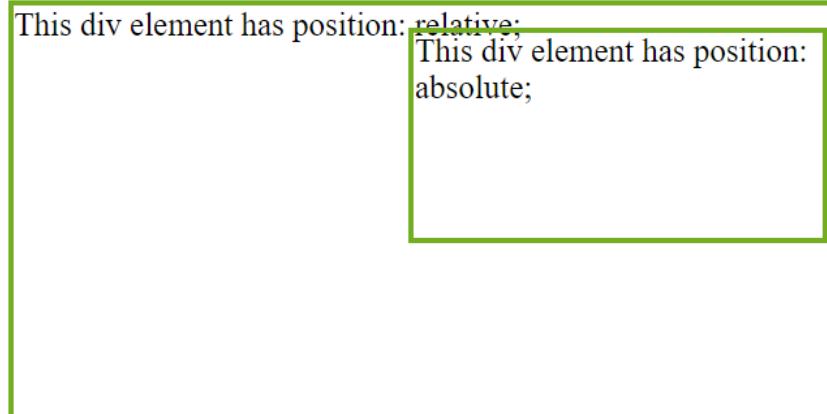
    <div class="relative">
```

```
This div element has position: relative;  
<div class="absolute">This div element has position:  
absolute;</div>  
</body>  
</html>
```

output :

## **position: absolute;**

An element with position: absolute; is positioned relative to the nearest positioned ancestor (instead of positioned relative to the viewport, like fixed):



## **BAB III**

### **Bootstrap**

Setelah kita mempelajari HTML di atas, sekarang kita akan memasuki materi berikutnya yaitu Bootstrap yang sebagaimana adalah framework HTML, CSS dan javascript yang mempermudah untuk mendesain website responsive dengan cepat dan mudah. Bootstrap menyertakan template desain berbasis HTML dan CSS untuk tipografi, formulir, tombol, tabel, navigasi, modal, carousel gambar dan banyak lainnya, serta plugin JavaScript opsional. Bootstrap juga memberi Anda kemampuan untuk membuat desain responsif dengan mudah.

Bootstrap ini bisa kita implementasikan ke website kita dengan beberapa cara yaitu :

1. Compiled CSS and JS

2. Source files
3. CDN via jsDelivr
4. Package managers

namun pada kesempatan kali ini kita akan memberi contoh untuk penggunaan bootstrap melalui CDN via JS Delivr saja. Untuk membuat HTML page menjadi bootstrap template kita hanya perlu menambahkan kode baris yang menghubungkan kita dengan bootstrap server. Kode tersebut merupakan link ke CSS dan ke file javascript nya

code :

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <title>Bootstrap demo</title>

    <!-- Link ke file CSS Bootstrap -->
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-iYQeCzEYFbKjA/T2uDLTpkwGzCiq6soy8tYaI1GyVh/UjpbCx/TYkiZhlZB6+fzT" crossorigin="anonymous" />
  </head>
  <body>
    <h1>Hello, world!</h1>

    <!-- Script ke file Javascript Bootstrap -->
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/js/bootstrap.bundle.min.js" />
```

```
integrity="sha384-u10knCvxWvY5kfmNBILK2hRnQC3Pr17a+RTT6rIHI7NnikvbZlHgT  
POOmMi466C8"  
    crossorigin="anonymous"  
></script>  
</body>  
</html>
```

output :

# Hello, world!

jika font di halaman ini telah berubah maka ini menandakan kita berhasil mengimplementasikan bootstrap ke page kita.

## ***Layoutting dengan Bootstrap:***

pengaturan layout dalam website memang memiliki macam-macam variasi dan setiap framework styling juga memiliki aturan yang berbeda-beda. kali ini kita akan membahas konsep layout dan elemen-elemen layout dalam bootstrap

### **Containter**

*Container* adalah elemen tata letak paling dasar di *Bootstrap* dan diperlukan saat menggunakan sistem *grid*. *Container* pada dasarnya digunakan untuk membungkus konten dengan beberapa *padding*. Mereka juga digunakan untuk menyelaraskan konten secara horizontal di tengah halaman jika tata letak lebar *fixed*.

Bootstrap menyediakan tiga jenis *Container* yang berbeda:

- `.container` yang memiliki lebar maksimum di setiap breakpoint responsif
- `.container-{breakpoint}` yaitu `width: 100%` hingga breakpoint yang ditentukan

- .container-fluid yaitu width: 100% di semua breakpoint

code :

```
<div class="container">
    <!-- Content here -->
</div>
```

efek perubahan dari container mungkin agak susah dilihat karena memang untuk mengatur ukuran box dari suatu element saja.

### Grid System :

Sistem grid bootstrap menyediakan cara yang mudah dan kuat untuk membuat tata letak responsif dari semua bentuk dan ukuran. konsep ini dibangun dengan flexbox dengan pendekatan mobile-first. Juga, ini sepenuhnya responsif dan menggunakan sistem dua belas kolom (12 kolom tersedia per baris) dan enam tingkatan responsif default.

Kita dapat menggunakan kelas grid Bootstrap yang telah ditentukan untuk membuat *layout* dengan cepat untuk berbagai jenis perangkat seperti ponsel, tablet, laptop, desktop, dan sebagainya. Misalnya, Anda dapat menggunakan kelas .col-\* untuk membuat kolom kisi untuk perangkat ekstra kecil seperti ponsel dalam mode *portrait*, dan kelas .col-sm-\* untuk ponsel dalam mode *landscape*. Cara ini juga berlaku untuk media lainnya.

	<b>xs</b> <576px	<b>sm</b> ≥576px	<b>md</b> ≥768px	<b>lg</b> ≥992px	<b>xl</b> ≥1200px	<b>xxl</b> ≥1400px
<b>Container max-width</b>	None (auto)	540px	720px	960px	1140px	1320px
<b>Class prefix</b>	.col-	.col-sm-	.col-md-	.col-lg-	.col-xl-	.col-xxl-
<b># of columns</b>	12					
<b>Gutter width</b>	1.5rem (.75rem on left and right)					
<b>Custom gutters</b>	<a href="#">Yes</a>					
<b>Nestable</b>	<a href="#">Yes</a>					
<b>Column ordering</b>	<a href="#">Yes</a>					

*tabel keterangan ukuran untuk column pada bootstrap*

code :

CSS

```
/* Some custom styles to beautify this example */
.demo-content {
    padding: 15px;
    font-size: 18px;
    background: #dbdfe5;
    margin-bottom: 15px;
}
.demo-content.bg-alt {
    background: #abb1b8;
}
```

HTML

```
<h2 class="text-center mt-3">Bootstrap Responsive Layout</h2>
<div class="text-center my-3">
    Open the output in a new blank tab (Click the arrow next to "Show
    Output"
    button) and resize the browser window to understand how the
    Bootstrap
    responsive grid system works.
</div>
<div class="container">
    <!--Row with two equal columns-->
    <div class="row">
        <div class="col-md-6">
            <div class="demo-content".col-md-6</div>
        </div>
        <div class="col-md-6">
            <div class="demo-content bg-alt".col-md-6</div>
        </div>
    </div>

    <!--Row with two columns divided in 1:2 ratio-->
    <div class="row">
        <div class="col-md-4">
            <div class="demo-content".col-md-4</div>
```

```

        </div>
        <div class="col-md-8">
            <div class="demo-content bg-alt">.col-md-8</div>
        </div>
    </div>

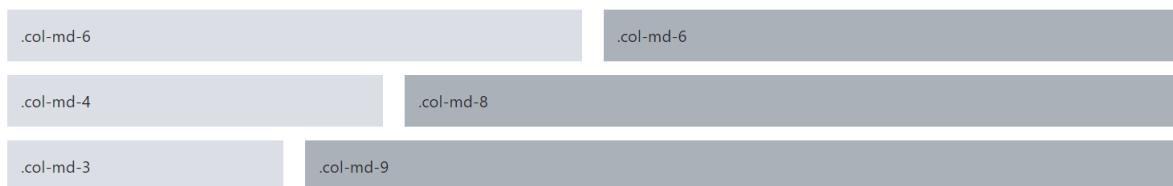
    <!--Row with two columns divided in 1:3 ratio-->
    <div class="row">
        <div class="col-md-3">
            <div class="demo-content">.col-md-3</div>
        </div>
        <div class="col-md-9">
            <div class="demo-content bg-alt">.col-md-9</div>
        </div>
    </div>

```

**output :**

### Bootstrap Responsive Layout

Open the output in a new blank tab (Click the arrow next to "Show Output" button) and resize the browser window to understand how the Bootstrap responsive grid system works.



ini merupakan salah satu contoh penggunaan grid system, lebih lengkapnya bisa dilihat di dokumentasi bootstrap itu sendiri.

### ***Component dan content dalam Bootstrap :***

Setelah kita mempelajari tentang layouting dalam bootstrap sekarang kita akan menulusuri isi-isi dari bootstrap itu sendiri. Isi dalam bootstrap ini bervariasi ada yang mirip dengan element HMTL umumnya namun mengalami sedikit modifikasi, dan ada juga yang memang berbeda dari yang pernah kita pelajari sebelumnya.

## Table :

Tabel HTML digunakan untuk menyajikan data secara grid seperti baris dan kolom.

Menggunakan Bootstrap Anda dapat sangat meningkatkan tampilan tabel dengan cara yang cepat dan mudah. Karena meluasnya penggunaan elemen <table> di widget pihak ketiga seperti kalender dan pemilih tanggal, maka tabel Bootstrap mengalami pengadaptasi dan improvisasi sehingga menjadi lebih optimal. Tambahkan kelas dasar .table ke semua tag <table>, lalu diperluas dengan kelas pengubah opsional atau gaya khusus bootstrap.

code :

```
<table class="table">

    <thead>
        <tr>
            <th scope="col">#</th>
            <th scope="col">First</th>
            <th scope="col">Last</th>
            <th scope="col">Handle</th>
        </tr>
    </thead>
    <tbody>
        <tr>
            <th scope="row">1</th>
            <td>Mark</td>
            <td>Otto</td>
            <td>@mdo</td>
        </tr>
        <tr>
            <th scope="row">2</th>
            <td>Jacob</td>
            <td>Thornton</td>
            <td>@fat</td>
        </tr>
        <tr>
            <th scope="row">3</th>
            <td colspan="2">Larry the Bird</td>
            <td>@twitter</td>
        </tr>
    </tbody>
</table>
```

output :

#	First	Last	Handle
1	Mark	Otto	@mdo
2	Jacob	Thornton	@fat
3	Larry the Bird		@twitter

ini merupakan tabel default dari bootstrap namun bootstrap juga menyediakan beberapa class untuk table-table ini.

Class	Heading	Heading
Default	Cell	Cell
Primary	Cell	Cell
Secondary	Cell	Cell
Success	Cell	Cell
Danger	Cell	Cell
Warning	Cell	Cell
Info	Cell	Cell
Light	Cell	Cell
Dark	Cell	Cell
<i>beberapa varian tabel-table dalam bootstrap</i>		

variasi ini bisa kita implementasikan dengan menambahkan class `table-{class}` pada tag `<table>` (ganti `{class}` dengan variasi tabel di atas) contoh :

code :

```
<table class="table table-success">
  ...
</table>
```

output :

#	First	Last	Handle
1	Mark	Otto	@mdo
2	Jacob	Thornton	@fat
3	Larry the Bird		@twitter

dan masih banyak lagi atribut untuk table yang bisa dilihat di dokumentasi resmi bootstrap.

Image :

Dalam bootstrap gambar mengalami sedikit modifikasi. Dokumentasi dan contoh untuk mengikutsertakan gambar ke dalam perilaku responsif (sehingga gambar tidak pernah menjadi lebih lebar dari induknya) dan menambahkan gaya ringan ke gambar tersebut semuanya melalui kelas.

untuk membuat gambar responsif pada bootstrap kita menggunakan class `.img-fluid` ini membuat `max-width: 100%;` dan `height: auto;` ke gambar sehingga mengikuti skala dengan lebar elemen induknya.

code :

```

```

output :



Responsive image

bootstrap juga memiliki beberapa kelas untuk memodifikasi gambar seperti :

Kelas .rounded menambahkan sudut membulat ke gambar

code :

```
>
```

output :



Kelas .rounded-circle membentuk gambar menjadi lingkaran

code :

```
>
```

output :



Kelas .img-thumbnail membentuk gambar menjadi thumbnail (*bordered*):

code :

```
>
```

output :



Nav :

Bootstrap menyediakan cara mudah dan cepat untuk membuat navigasi dasar yang sangat fleksibel dan elegan. Semua komponen nav Bootstrap berbagi markup dan gaya dasar yang sama melalui kelas .nav dasar. Komponen .nav dasar dibuat dengan flexbox dan memberikan fondasi yang kuat untuk membangun semua jenis komponen navigasi. Ini mencakup beberapa penggantian gaya (untuk bekerja dengan daftar), beberapa bantalan tautan untuk area hit yang lebih besar, dan gaya dasar yang dinonaktifkan. Berikut merupakan cara basic menggunakan *nav*.

code :

```
<nav class="nav">
    <a href="#" class="nav-item nav-link active">Home</a>
    <a href="#" class="nav-item nav-link">Profile</a>
    <a href="#" class="nav-item nav-link">Messages</a>
    <a href="#" class="nav-item nav-link disabled"
tabindex="-1">Reports</a>
</nav>
```

output :

Home    Profile    Messages    Reports

nav ini juga memiliki banyak built-in class yang disediakan oleh bootstrap. Class ini bervariasi mulai dari mengatur alignment, membuat nav nya vertical, dan lain-lain. Kita juga bisa memodifikasi nav ini menjadi *tabs* dengan menambahkan kelas .nav-tabs. contoh sebagai berikut :

code :

```
<nav class="nav nav-tabs">
    <a href="#" class="nav-item nav-link active">Home</a>
    <a href="#" class="nav-item nav-link">Profile</a>
    <a href="#" class="nav-item nav-link">Messages</a>
    <a href="#" class="nav-item nav-link disabled"
tabindex="-1">Reports</a>
</nav>
```

output :



selain dari *tabs* kita juga bisa merubah nav menjadi mode *pills* dengan menambahkan kelas .nav-pills. contoh :

code :

```
<nav class="nav nav-pills">
    <a href="#" class="nav-item nav-link active">
        <i class="bi-house-door"></i> Home
    </a>
    <a href="#" class="nav-item nav-link">
        <i class="bi-person"></i> Profile
    </a>
    <a href="#" class="nav-item nav-link">
        <i class="bi-envelope"></i> Messages
    </a>
    <a href="#" class="nav-item nav-link disabled" tabindex="-1">
        <i class="bi-bar-chart"></i> Reports
    </a>
</nav>
```

output:

Home   Profile   Messages   Reports

## Navbar :

selain menyediakan basic nav bootstrap juga memberikan kita komponen navbar yang siap pakai. Anda dapat menggunakan komponen navbar Bootstrap untuk membuat header navigasi responsif untuk situs web atau aplikasi Anda. Navbar responsif ini akan diciutkan pada perangkat yang memiliki area pandang kecil seperti ponsel, tetapi meluas saat pengguna mengklik tombol saklar

code :

```
<div class="m-4">
    <nav class="navbar navbar-expand-lg navbar-light bg-light">
        <div class="container-fluid">
            <a href="#" class="navbar-brand">Brand</a>
            <button
                type="button"
                class="navbar-toggler"
                data-bs-toggle="collapse"
                data-bs-target="#navbarCollapse"
            >
                <span class="navbar-toggler-icon"></span>
            </button>
            <div class="collapse navbar-collapse" id="navbarCollapse">
                <div class="navbar-nav">
                    <a href="#" class="nav-item nav-link active">Home</a>
                    <a href="#" class="nav-item nav-link">Profile</a>
                    <a href="#" class="nav-item nav-link">Messages</a>
                    <a href="#" class="nav-item nav-link disabled"
                        tabindex="-1"
                    >Reports</a>
                >
            </div>
            <div class="navbar-nav ms-auto">
                <a href="#" class="nav-item nav-link">Login</a>
            </div>
        </div>
```

```
    </div>
</nav>
</div>
```

output :



Brand Home Profile Messages Reports Login

## BAB IV

### Javascript

#### *Pengertian JavaScript*

JavaScript adalah bahasa script yang bisa dijalankan di browser, dan biasa disebut dengan client side programming. Client di sini adalah browser, seperti: Google Chrome, Internet Explorer, Firefox, safari dan sebagainya. Bahasa JavaScript berbentuk kumpulan skrip yang berjalan pada suatu dokumen HTML.

Sepanjang sejarah internet bahasa ini adalah bahasa skrip pertama untuk web. Bahasa ini adalah bahasa pemrograman untuk memberikan kemampuan tambahan terhadap bahasa HTML dengan mengijinkan pengeksekusian perintah-perintah di sisi user, yang artinya di sisi browser bukan di sisi server web. Dalam menjalankan JavaScript tidak dibutuhkan compiler tetapi melalui browser karena browser memiliki engine yang bisa menginterpretasikan semua kode JavaScript.

#### *Sejarah JavaScript*

JavaScript diperkenalkan pertama kali oleh Brendan Eich yang bekerja di Netscape pada tahun 1995. Pada awalnya bahasa ini dinamakan “Live Script” yang berfungsi sebagai bahasa sederhana untuk browser Netscape Navigator 2. Pada masa itu bahasa ini banyak dikritik karena kurang aman,

pengembangannya yang terkesan buru-buru dan tidak ada pesan kesalahan yang ditampilkan setiap kali membuat kesalahan pada saat menyusun suatu program. Kemudian sejalan dengan sedang giatnya kerjasama antara Netscape dan Sun (pengembang bahasa pemrograman “Java”) pada masa itu, maka Netscape memberikan nama “JavaScript” kepada bahasa tersebut pada tanggal 4 Desember 1995. Pada saat yang bersamaan Microsoft sendiri mencoba untuk mengadaptasikan teknologi ini yang mereka sebut sebagai “Jscript” di browser Internet Explorer 3.

Oleh karena ada banyak perusahaan yang mengembangkan bahasa ini kemudian distandardkan dengan nama ECMAScript oleh Netscape melalui Organisasi Internasional ECMA. Standar ini dipublikasikan pertama kali pada bulan Juni 1997 dengan nama dokumen spesifikasi ECMA-262. Melalui publikasi tersebut, semua implementasi JavaScript pada tiap browser akan memiliki standar penerapan (pengkodean) yang sama. Saat ini standar ini telah mencapai rilis Edisi ke-5.1, yang dipublikasikan pada bulan Juni 2011. Setiap browser saat ini memiliki implementasi sendiri-sendiri untuk ECMAScript ini, diantaranya Internet Explorer dengan Jscript, Opera dengan ECMAScript, dan Mozilla Firefox, Google Chrome termasuk juga Safari dengan nama JavaScript.

Selain di browser, sekarang JavaScript juga sudah diterapkan pada banyak aplikasi lainnya seperti Windows 8 Apps (.Net Framework), Adobe Flash ActionScript, KDE Desktop Environment, Node.js, Qt QML, jQuery Mobile, Firefox OS, Ubuntu Touch, dan masih banyak lagi kemungkinan implementasi lainnya. JavaScript bergantung kepada browser (navigator) yang memanggil dan menampilkan halaman web yang tidak hanya berisi HTML tapi juga dapat berisi skrip-skrip JavaScript. JavaScript juga tidak memerlukan penterjemah khusus untuk menjalankannya di sisi user/klien. Berikut ini satu tabel yang berisi beberapa perbandingan mendasar antara Java dan JavaScript.

## ***Penulisan Kode JavaScript***

Pada dasarnya, kita bebas ingin meletakkan kode JavaScript di bagian mana saja, selama berada di dalam tag `<script>`. Umumnya, tag `<script>` diletakkan pada awal kode HTML di akhir tag `<head>`, atau di akhir tag `<body>`. Posisi peletakan ini akan mempengaruhi bagaimana urutan kode JavaScript berjalan.

Halaman HTML di proses oleh web browser dari atas ke bawah secara berurutan. Aturan ini juga berlaku untuk kode HTML, CSS, dan JavaScript, sehingga kode JavaScript yang ada di dalam tag `<head>`, akan diproses lebih dahulu dari pada isi dokumen HTML yang terdapat di dalam tag `<body>`.

## 1. Internal JavaScript

Cara pertama untuk melakukan input kode JavaScript ke dalam halaman HTML adalah dengan menggunakan tag `<script>` secara internal. Internal disini berarti bahwa kode JavaScript ditulis pada halaman yang sama dengan HTML, atau di dalam satu file HTML.

Cara ini merupakan cara yang paling sering digunakan, jika kode JavaScript tidak begitu panjang, dan hanya digunakan di 1 halaman saja. Kode JavaScript yang akan di input diletakkan diantara tag pembuka `<script>` dan tag penutup `</script>`.

Contoh Internal JavaScript:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Belajar JavaScript</title>
    <script>
      alert("Hello, JavaScript!");
    </script>
  </head>
  <body></body>
</html>
```

## 2. External JavaScript

Cara atau metode kedua untuk melakukan input kode JavaScript ke dalam halaman HTML adalah dengan memindahkan kode JavaScript ke dalam sebuah file terpisah, lalu memanggilnya dari HTML. Cara ini sangat disarankan karena akan memberikan banyak keuntungan dan fleksibilitas dalam membuat program JavaScript. Sebuah file JavaScript disimpan dalam ekstensi .js, seperti script.js atau index.js. Dari halaman HTML, kita memanggilnya menggunakan tag `<script>` dengan atribut src. Atribut src berisi alamat dari file JavaScript tersebut.

Contoh External JavaScript:

```

<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Belajar JavaScript</title>
    <script src="script.js"></script>
  </head>
  <body>
    <h1>Contoh External JavaScript</h1>
  </body>
</html>

```

Gambar di atas adalah file HTML, untuk memanggil file JavaScript kita dapat menggunakan attribute src pada tag script dan mengisinya sesuai nama file JavaScript yang ingin kita gunakan pada contoh ini adalah “script.js”.

```

script.js  ×
script.js
1 alert("Hello, Ini External JavaScript");

```

Dan pada file script.js kita dapat menuliskan kode JavaScript tanpa tag <script>.

## *Cara Menjalankan JavaScript*

### 1. Menggunakan Console Browser

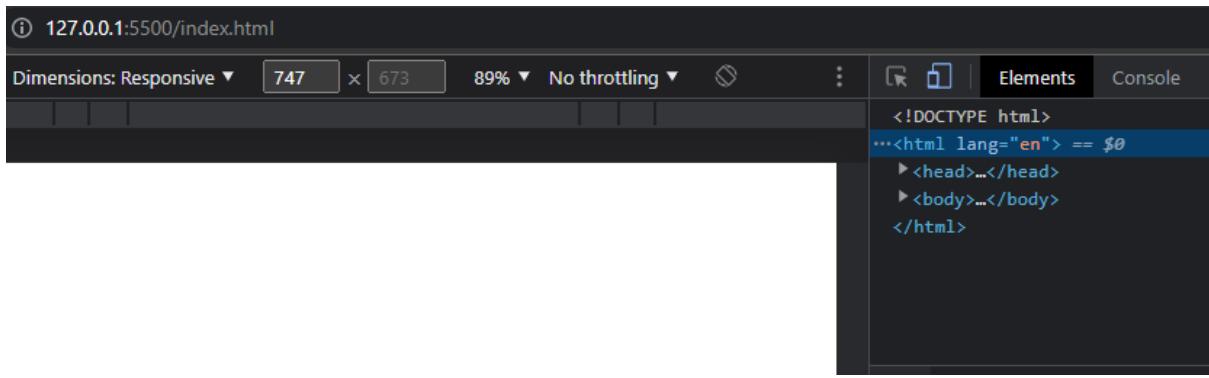
Tuliskan Code berikut pada text editor a masing-masing, untuk mengecek berhasil di jalankan atau tidak .

```

index.html > ...
1  <!DOCTYPE html>
2  <html lang="en">
3    <head>
4      <meta charset="UTF-8" />
5      <meta http-equiv="X-UA-Compatible" content="IE=edge" />
6      <meta name="viewport" content="width=device-width, initial-scale=1.0" />
7      <title>Menjalankan JavaScript</title>
8      <script>
9        console.log("Hello saya lagi belajar JavaScript");
10     </script>
11   </head>
12   <body></body>
13 </html>

```

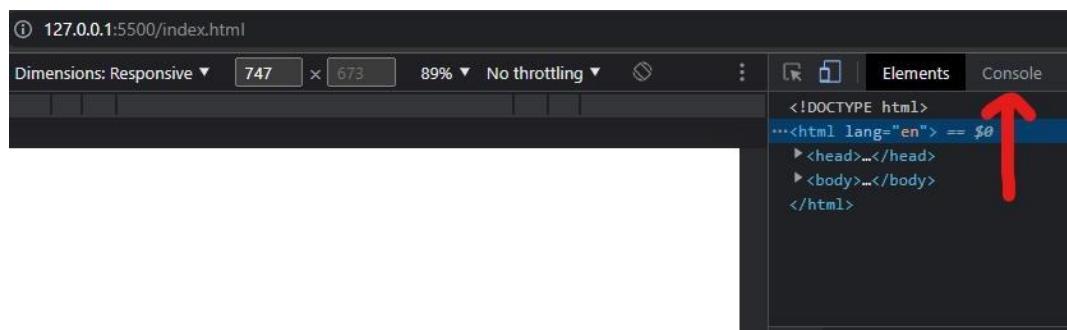
Cara yang pertama ini cukup mudah, kamu tinggal membuka file index.html di browser contohnya Google Chrome, kemudian klik kanan pilih Inspeksi atau tekan (Ctrl+Shift+i).



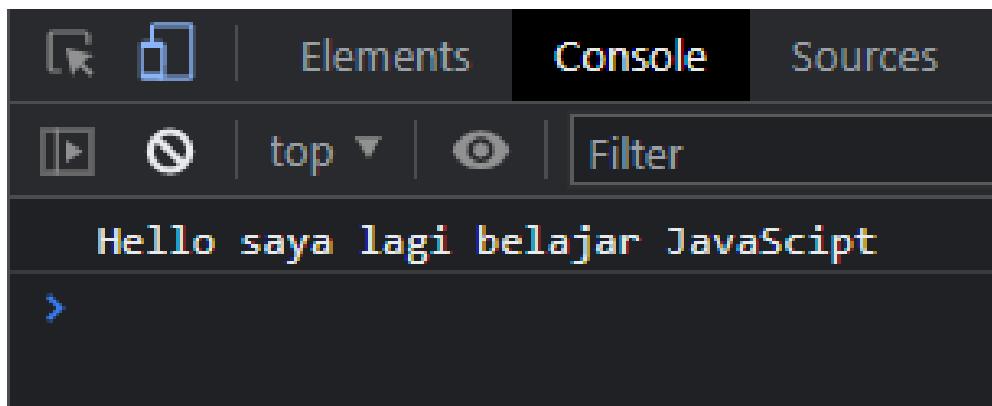
The screenshot shows the Chrome DevTools interface with the 'Elements' tab selected. The left panel displays the DOM tree for the file 'index.html'. The root node is the HTML element, which contains a head and a body. The body contains several script tags, one of which is highlighted with a blue selection bar. The right panel shows the raw HTML code of the page.

```
<!DOCTYPE html>
...<html lang="en"> == $0
  > <head>...</head>
  > <body>...</body>
</html>
```

Kemudian kamu akan mendapatkan tampilan baru di sisi kanan layar seperti gambar di atas. Lalu Pilih tab Console.



Pada tab console kita dapat melihat output dari code javascript pada file HTML di atas.

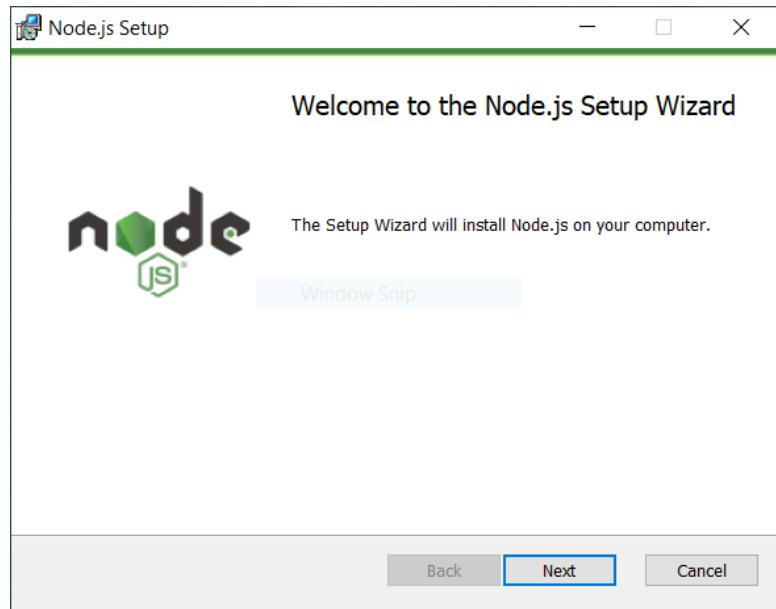


The screenshot shows the Chrome DevTools interface with the 'Console' tab selected. The main area displays the output of a JavaScript command: 'Hello saya lagi belajar JavaScript'. Below the output, there is a blue '>' symbol indicating the next input line.

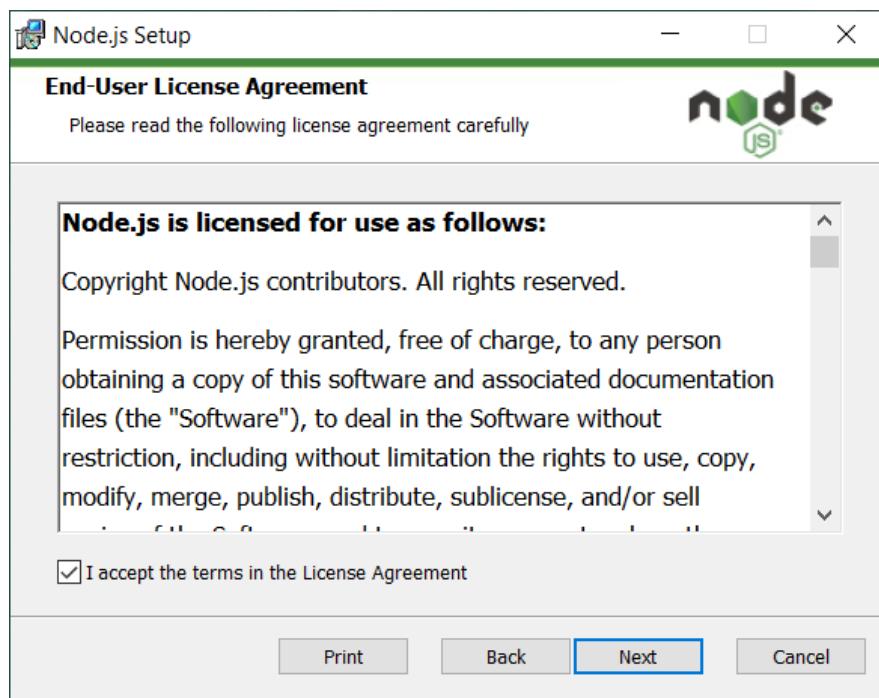
## 2. Menggunakan Node.js

Node.js adalah platform untuk menjalankan code JavaScript secara runtime. Node.js memungkinkan aplikasi dijalankan di luar browser. Itu semua disebabkan karena proses rendering dilakukan di sisi server.

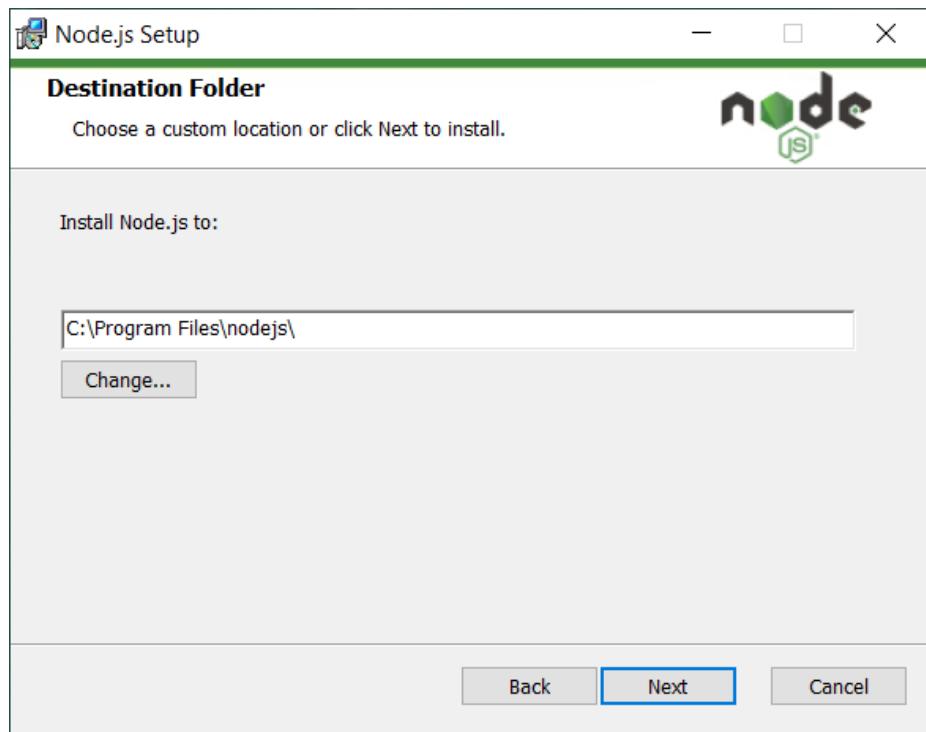
- Cara Instalasi Node.js
1. Download Node.JS Installer pada website resminya di <https://nodejs.org/download/>.
  2. Pilih Sesuai OS yang kalian gunakan dan jalankan file yang telah kalian download.



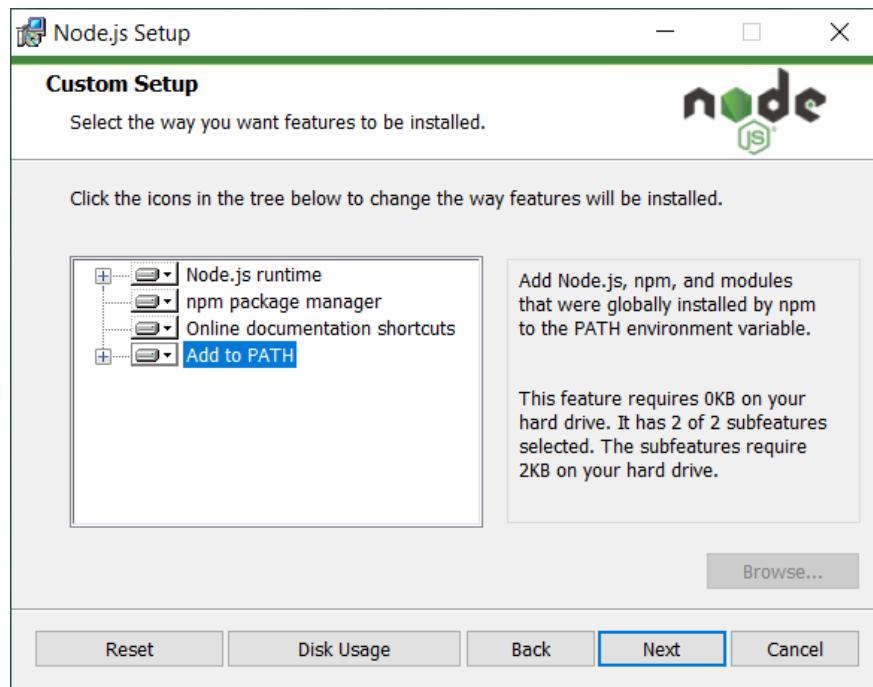
3. Klik *checkbox* yang menandakan kita setuju dengan persyaratan license penggunaan.



4. Tentukan direktori tujuan instalasi jika anda ingin memasang Node.js tanpa menggunakan direktori default.



5. Pada tahap Pengaturan Kostum kita bisa memilih fitur-fitur apa saja yang ingin dan tidak ingin di instal. Pastikan anda memilih Add to PATH agar interpreter Node.js dan npm dapat di eksekusi melalui CMD dari direktori manapun.



6. Proses instalasi akan berjalan tunggu hingga selesai.
7. Jika tidak mengalami error, proses instalasi Node.js berhasil di tandai dengan output Node.js has been successfully installed.

- Untuk Mengecek apakah node.js sudah benar-benar terinstall , buka command prompt(cmd) atau node.js dan ketik node -v (untuk melihat versi node js yang terinstall)

```
Command Prompt
Microsoft Windows [Version 10.0.19044.1889]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Thoriq>node -v
v16.15.1
```

- Dan coba untuk mengecek versi dari npm juga dengan cara npm -v. NPM adalah Node Package Manager, yaitu pengelola package JavaScript bawaan Node.js

```
C:\Users\Thoriq>npm -v
npm WARN config global `--global` --global
8.12.1
```

- Jika muncul versi node.js dan npm seperti di atas berarti node.js sudah berhasil di install di perangkat kalian masing-masing.
- Lalu Coba buka kembali text editor kalian, pada text editor buka terminal dan ketik node lalu nama file yang ingin kalian jalankan. Contohnya seperti gambar di bawah:

```
script.js
1 console.log("Hello saya sedang belajar JavaScript");

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER COMMENTS

Thoriq@DESKTOP-CU1PS3R MINGW64 /e/THORIQ/WEB 2022/JavaScript Modul
$ node script.js
Hello saya sedang belajar JavaScript
```

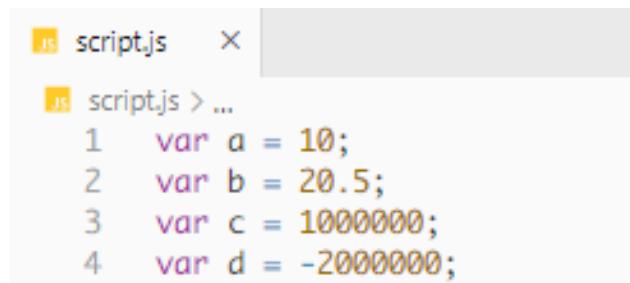
## Tipe Data

Tipe data dalam JavaScript diantaranya adalah tipe data angka, tipe data text (string), dan tipe data Boolean. Selain itu dalam JavaScript ada tipe data objek. Contoh tipe data objek adalah tipe data tanggal (date), array, dan fungsi.

## 1. Tipe Data Angka.

JavaScript tidak membedakan tipe data angka (number) antara angka bulat dengan angka desimal, atau tidak membedakan antara bilangan integer dengan float. Seluruh tipe data angka di dalam JavaScript disimpan dalam bentuk desimal (float).

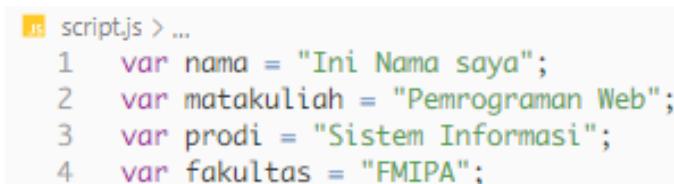
Angka di dalam JavaScript bisa dengan tepat merepresentasikan nilai antara  $-2^{53}$  sampai dengan  $2^{53}$ , atau dari  $-9007199254740992$  sampai  $9007199254740992$ . Karena di dalam JavaScript sebuah variabel tidak perlu di deklarasikan akan bertipe apa, maka jika sebuah variabel diberikan nilai angka, maka variabel tersebut telah menjadi variabel dengan tipe angka. Berikut adalah contohnya:



```
script.js > ...
1 var a = 10;
2 var b = 20.5;
3 var c = 1000000;
4 var d = -2000000;
```

## 2. Tipe Data String.

Tipe data String di dalam JavaScript adalah tipe data yang terdiri dari kumpulan karakter yang berurutan. Atau di dalam penggunaan sehari-hari string adalah tipe data yang menampung nilai text atau kalimat. Untuk membuat sebuah tipe data string, kita hanya tinggal menambahkan tanda kutip (bahasa Inggris: 'quotes') pada awal dan akhir dari text. JavaScript mendukung penggunaan tanda kutip satu (' ') maupun tanda kutip ganda ("").



```
script.js > ...
1 var nama = "Ini Nama saya";
2 var matakuliah = "Pemrograman Web";
3 var prodi = "Sistem Informasi";
4 var fakultas = "FMIPA";
```

## 3. Tipe Data Boolean

Tipe Data Boolean Tipe data Boolean adalah tipe data yang hanya mempunyai dua nilai, yakni benar (True) atau salah (False). Tipe data Boolean sering digunakan untuk membuat alur logika program. Struktur logika seperti if, else, while, dan do while, membutuhkan nilai Boolean sebagai "pengontrol" alur program.

Tipe data Boolean juga merupakan hasil yang didapat dari operasi perbandingan. Misalkan apakah variabel a sama dengan b, atau apakah a lebih besar dari b. Untuk membuat tipe data Boolean di dalam JavaScript, kita cukup memberikan nilai true, atau false ke dalam sebuah variabel. Berikut adalah contoh pembuatan tipe data Boolean di dalam JavaScript:

```
script.js > ...
1 var benar = true;
2 var salah = false;
3 var lapar = true;
4 var haus = false;
```

#### 4. Tipe Data Null dan Undefined

Sepintas null dan undefined terlihat sama, tapi sesungguhnya tidak sama sekali. Null artinya memiliki nilai kosong atau tidak ada dan harus diganti dengan nilai lain yang valid contoh string atau number. Contohnya:

```
script.js > ...
1 var a = null;
2
3 // Hasilnya null
4 console.log(a);
```

Sedangkan Undefined adalah variabel yang sudah di deklarasi tapi belum di beri nilai. Contohnya:

```
script.js > ...
1 var b;
2
3 //Hasilnya undefined
4 console.log(b);
```

### Komentar

JavaScript mendukung 2 jenis cara penulisan komentar, yakni menggunakan karakter // untuk komentar dalam 1 baris, dan karakter pembuka. komentar /\* dan penutup \*/ untuk komentar yang mencakup beberapa baris. Gambar 8.1 merupakan contoh penulisan komentar dalam JavaScript.

```
script.js
1 // ini komentar satu baris
2
3 /* ini komentar beberapa
4      baris pada
5      javascript */
```

## *Variabel Pada JavaScript.*

Variabel adalah penanda identitas yang digunakan untuk menampung suatu nilai. Penulisan Variabel pada JavaScript menggunakan keyword var,let, const, atau tanpa apapun. Pembuatan variabel tanpa menggunakan keyword var,let, atau const memang lebih cepat, akan tetapi tidak disarankan. Contohnya:

```
js script.js > ...
1  var a = 2021;
2  let b = "Sistem Informasi";
3  const c = true;
4  d = "ABC"; // Tidak Disarankan      s
```

VAR, LET, CONST memiliki fungsi yang sama yaitu untuk membuat variabel. Tapi, pada JavaScript menganut 2 sistem scope berbeda, yaitu block scope dan functional scope. Block scope adalah cakupan variabel yang dipisahkan oleh setiap block, block yang dimaksud adalah kurung kurawal ({}). Jadi, jika setiap variabel yang hadir di dalam suatu block, misalnya perulangan, maka jika kita coba mengakses variabel tersebut dari luar perulangan, maka akan mengakibatkan error. Inilah yang dianut oleh LET dan CONST. Contohnya:

```
js script.js > ...
1  for (let i = 1; i < 6; i++) {
2    console.log(i);
3  }
4
5  console.log(i); // Ini akan Error
```

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    JUPYTER    COMMENT

```
$ node script.js
1
2
3
4
5
E:\THORIQ\WEB 2022\JavaScript Modul\script.js:5
console.log(i); // Ini akan Error
^

ReferenceError: i is not defined
```

Bisa di lihat pada gambar di atas jika LET i di panggil di luar perulangan akar menghasilkan error begitu pula dengan menggunakan keyword CONST. Lalu Coba Menggunakan VAR

```
script.js > ...
1  for (var i = 1; i <= 5; i += 2) {
2    console.log(i);
3  }
4  console.log("Ini i d luar for", i);

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER COMMENTS

Thoriq@DESKTOP-CU1PS3R MINGW64 /e/THORIQ/WEB 2022/JavaScript Modul
$ node script.js
1
3
5
Ini i d luar for 7
```

Bisa d lihat jika menggunakan VAR variabel i tetap dapat di panggil di luar perulangan. Lalu coba menggunakan function.

```
script.js > ...
1  function welcome() {
2    var greet = "Hello";
3    return greet;
4  }
5  console.log(greet);

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER COMMENTS

Thoriq@DESKTOP-CU1PS3R MINGW64 /e/THORIQ/WEB 2022/JavaScript Modul
$ node script.js
E:\THORIQ\WEB 2022\JavaScript Modul\script.js:5
  console.log(greet);
          ^
ReferenceError: greet is not defined
```

Bisa di lihat pada gambar di atas console akan menampilkan pesan error 'say is not defined'. Kenapa ini bisa terjadi? Karena VAR menganut sistem functional scope, dimana VAR tidak bisa diakses jika kita mencoba mengakses nya dari luar function, tetapi masih bisa diakses dari luar block lain seperti perulangan atau percabangan (if-else). Sementara LET dan CONST tidak bisa diakses baik di luar function, percabangan, maupun perulangan. Lalu perbedaan lainnya terdapat pada sisi ketetapan data. Apakah ketiganya dapat diubah-ubah data/nilainya, atau tidak dapat diubah. Lihat contoh berikut:

```

script.js > ...
1  let fakultas = "MIPA";
2  fakultas = "FARMASI";
3  console.log("Menggunakan Let", fakultas);
4
5  var Prodi = "Sistem Informasi";
6  Prodi = "Matematika";
7  console.log("Menggunakan var", Prodi);
8
9  const Universitas = "UNHAS";
10 Universitas = "UNM";
11 console.log("Menggunakan const", Universitas);
12

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER COMMENTS

```

Thoriq@DESKTOP-CU1PS3R MINGW64 /e/THORIQ/WEB 2022/JavaScript Modul
$ node script.js
Menggunakan Let FARMASI
Menggunakan var Matematika
E:\THORIQ\WEB 2022\JavaScript Modul\script.js:10
Universitas = "UNM";
^

TypeError: Assignment to constant variable.

```

Berdasarkan gambar di atas dapat dilihat variabel yang menggunakan variabel LET dan VAR dapat diubah nilainya. Sedangkan variabel menggunakan const tidak dapat diubah nilainya. Jadi, gunakan const ketika datanya cenderung tetap.

## *Operator Pada JavaScript*

### 1. Operator Aritmatika

Operator	Tunggal/Biner	Keterangan
+	Biner	Penjumlahan
-	Biner	Pengurangan
*	Biner	Perkalian
/	Biner	Pembagian
%	Biner	Modulus
-	Tunggal	Negasi
++	Tunggal	Penambahan dengan satu
--	Tunggal	Pengurangan dengan satu

Contoh:

```

1 var num1 = 10;
2 var num2 = 20;
3
4 console.log(num1 + num2); // 30
5 console.log(num1 - num2); // -10
6 console.log(num1 + num2 * num1); // 210
7 console.log(num2 / num2 - num1); // -9
8 console.log(num2 % num1); // 0
9 console.log(++num2); // 21

```

Sama seperti pelajaran matematika yang kita pelajari, di dalam JavaScript operator perkalian dan pembagian lebih kuat dari pada operator penambahan dan pengurangan. Jika operasi matematika yang akan diproses memiliki lebih dari 1 operator yang sama kuat dalam sebuah perhitungan, JavaScript akan memprioritaskan urutan perhitungan dari kiri ke kanan.

## 2. Operator Assignment

Operator assignment digunakan untuk memberi nilai pada suatu variabel dengan tanda sama dengan (=), penulisan operator assignment memudahkan kita mempersingkat script.

Operator	keterangan	Contoh	Ekuivalen
=	Sama dengan	X=Y	
+=	Ditambah dengan	X+=Y	X=X+Y
-=	Dikurangi dengan	X-=Y	X=X-Y
*=	Dikali dengan	X*=Y	X=X*Y
/=	Dibagi dengan	X/=Y	X=X/Y
%=	Modulus dengan	X%=Y	X=X%Y
&=	Bit AND dengan	X&Y	X=X&Y
=	Bit OR	X Y	X=X Y

Contoh:

```

script.js > ...
1  var num1 = 10;
2  var num2 = 5;
3
4  console.log((num1 += num2)); // num1 = num1 + num2
5  console.log((num1 -= num2)); // num1 = num1 - num2
6  console.log((num2 *= num1)); // num2 = num2 * num1
7  console.log((num2 /= num1)); // num2 = num2 / num1

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    JUPYTER    COMMENTS

Thoriq@DESKTOP-CU1PS3R MINGW64 /e/THORIQ/WEB 2022/JavaScript Modul
$ node script.js
15
10
50
5

```

### 3. Operator Perbandingan.

Digunakan untuk membandingkan dua buah operan. Operan yang dikenal operator ini dapat bertipe string, numerik, maupun ekspresi lain.

Operator	Keterangan
==	Sama dengan
!=	Tidak sama dengan
>	Lebih besar
<	Lebih kecil
>=	Lebih besar atau sama dengan
<=	Lebih kecil atau sama dengan

Contoh:

```

script.js > ...
1  var num1 = 5;
2  var num2 = 10;
3  var num3 = "5";
4
5  console.log(num1 == num2); // false
6  console.log(num1 != num2); // true
7  console.log(num1 == num3); // true
8  console.log(num1 === num3); // false
9  console.log(num2 > num1); // true
10 console.log(num2 < num1); //false

```

Operasi pembandingan akan mengeluarkan output berupa Boolean (true atau false). Pada baris ke 7 gambar di atas dapat dilihat jika kita membandingkan num1 dan num3 dengan == akan

menghasilkan nilai true, sedangkan pada baris 8 menghasilkan nilai false, ini disebabkan karena operasi pembandingan === akan mengecek kedua tipe data yang dibandingkan sedangkan operasi pembandingan == tidak mengecek tipe data.

#### 4. Operasi Logika

Operasi Logika sering digunakan pada Percabangan atau perulangan.

Operator	Keterangan
&&	Operator logika AND
	Operator Logika OR
!	Operator logika NOT

Untuk operator AND, jika expression sebelah kiri setelah dievaluasi menghasilkan nilai false, maka operator AND akan mengembalikan expression pertama. Kebalikannya, jika expression sebelah kiri setelah dievaluasi menghasilkan nilai true, maka operator AND akan mengembalikan expression kedua. Jika kedua expression/variabel yang bernilai Boolean maka pada operator AND false selalu menang

Untuk operator OR, jika expression sebelah kiri setelah dievaluasi menghasilkan nilai true, maka operator OR akan mengembalikan expression pertama. Kebalikannya, jika expression sebelah kiri setelah dievaluasi menghasilkan nilai false, maka operator OR akan mengembalikan expression kedua. Jika kedua expression/variabel yang bernilai Boolean true dan false, maka pada operator OR true selalu menang

Untuk operator NOT, Jika expression yang diberikan setelah dievaluasi menghasilkan nilai true, maka operator NOT akan mengembalikan false. Kebalikannya, jika expression yang diberikan setelah dievaluasi menghasilkan nilai false, maka operator NOT akan mengembalikan true.

Contoh:

```
js script.js
1 console.log(true && true); // true
2 console.log(true && false); // false
3 console.log(false && true); // false
4
5 console.log(true || false); //true
6 console.log(false || true); // true
7
8 console.log(!true); //false
9 console.log(!false); //true
10
```

## Array

Array adalah tipe data yang digunakan untuk menyimpan berbagai nilai dalam satu nama, dibedakan melalui nilai indexnya. Array sangat populer di berbagai bahasa programming karena penggunaannya dapat menyederhanakan penggunaan variable pada coding

### 1. Membuat Object Array

Membuat / Mendeklarasikan Object Array Sebuah object array dideklarasikan di JavaScript dengan beberapa cara berikut :

- Dengan penggunaan keyword new Array(), dengan contoh sebagai berikut :

```
js script.js > ...
1 var namaArray= new Array();
```

- Dengan penggunaan keyword new Array(jumlah index), contohnya sebagai berikut :

```
js script.js > ...
1 var namaArray = new Array(3);
```

- Dengan penggunaan keyword new Array(item1, item2, ...), dimana antara satu item dengan item lainnya tidak harus memiliki tipe data yang sama. Contoh deklarasinya adalah sebagai berikut :

```
js script.js > ...
1 var namaArray = new Array("Makassar", "Unhas", 2022);
```

- Dengan penggunaan literal [item1, item2, item3, ...], dimana antara satu item dengan item lainnya tidak harus memiliki tipe data yang sama. Contohnya sebagai berikut :

```
js script.js > ...
1 var namaArray = [20, "Nama", "NIM", true];
```

## 2. Mengakses Array

Untuk memberikan dan membaca nilai pada objek array cukup gampang, yaitu dengan menggunakan nilai index - sebagai penunjuk posisi pada objek array. Index selalu dimulai dari angka 0, kemudian diikuti angka 1, 2 dan seterusnya.

The screenshot shows a code editor with a terminal window below it. The code in script.js is:

```
script.js > ...
1 var negara = ["Indonesia", "Malaysia", "Singapura"];
2 console.log(negara[0]);
3
4 // Mengakses Semua Array
5 console.log(negara);
```

The terminal output shows the execution of the script:

```
Thoriq@DESKTOP-CU1PS3R MINGW64 /e/THORIQ/WEB 2022/JavaScript Modul
$ node script.js
Indonesia
[ 'Indonesia', 'Malaysia', 'Singapura' ]
```

## 3. Menambah Nilai Array

The screenshot shows a code editor with a terminal window below it. The code in script.js is:

```
script.js > ...
1 const cars = [];
2 cars[0] = "Saab";
3 cars[1] = "Volvo";
4 cars[2] = "BMW";
5
6 // Menggunakan Method push
7 // Method push akan menambahkan nilai
8 // di index terakhir
9 cars.push("Avanza");
```

The terminal output shows the execution of the script:

```
Thoriq@DESKTOP-CU1PS3R MINGW64 /e/THORIQ/WEB 2022/JavaScript Modul
$ node script.js
[ 'Saab', 'Volvo', 'BMW', 'Avanza' ]
```

#### 4. Mengubah Nilai Array

The screenshot shows a terminal window with the following content:

```
script.js > ...
1 var negara = ["Indonesia", "Malaysia", "Singapura"];
2 console.log("Sebelum di ubah: ", negara);
3 negara[2] = "Korea";
4 negara[1] = "Jepang";
5
6 console.log("Sesudah di ubah: ", negara);

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER COMMENTS
```

```
Thoriq@DESKTOP-CU1PS3R MINGW64 /e/THORIQ/WEB 2022/JavaScript Modul
$ node script.js
Sebelum di ubah: [ 'Indonesia', 'Malaysia', 'Singapura' ]
Sesudah di ubah: [ 'Indonesia', 'Jepang', 'Korea' ]
```

#### 5. Method dan Property pada Object Array

Berikut adalah beberapa method yang dapat digunakan pada variable object array :

5. concat() : Menggabungkan 2 atau lebih array.
6. indexOf() : Mencari suatu nilai pada array dan mengembalikan posisi indexnya.
7. join() : Menggabungkan semua elemen pada array menjadi suatu teks.
8. lastIndexOf() : Mencari suatu nilai pada array, dimulai dari posisi akhir dan kemudian mengembalikan posisi indexnya.
9. pop() : Mengembalikan nilai elemen terakhir, dan kemudian menghilangkannya dari array.
10. push() : Menambahkan nilai baru pada array, dan mengembalikan ukuran array.
11. reverse() : Membalikkan posisi-posisi nilai pada array.
12. shift() : Menghilangkan nilai pertama dari array, dan kemudian mengembalikan nilai tersebut.
13. slice() : Mengembalikan sebagian dari array

Dan berikut adalah beberapa property yang dapat digunakan pada variable object array

14. length : Mengembalikan ukuran array, yakni jumlah nilai / elemen yang terdapat pada array tersebut.
15. constructor : Mengembalikan fungsi yang membuat prototype Array.
16. prototype : Menambahkan property dan method pada objek Array.

#### Percabangan

1. If dan Else Statement

```
// Syntax If Else Statement
if(kondisi){
    // blok kode
}else{
    //blok kode
}
```

Pernyataan ini digunakan untuk menguji sebuah kondisi dan kemudian mengeksekusi pernyataan tertentu bila kondisi tersebut terpenuhi, dan mengeksekusi pernyataan lain bila kondisi tersebut tidak terpenuhi. Contoh:

The screenshot shows a code editor interface with a script.js file open. The code defines a variable umur and uses an if-else statement to log different messages to the console based on whether umur is greater than 18. The terminal below shows the execution of the script and its output.

```
script.js > ...
1 var umur = 17;
2
3 if (umur > 18) {
4     console.log("Boleh Masuk");
5 } else {
6     console.log("Dilarang Masuk");
7 }
8
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER COMMENTS

```
Thoriq@DESKTOP-CU1PS3R MINGW64 /e/THORIQ/WEB 2022/JavaScript Modul
$ node script.js
Dilarang Masuk
```

Karena umur tidak lebih dari 18 , maka isi dari statement if tidak di eksekusi, dan masuk ke statement else yaitu console.log("Di larang masuk").

## 2. Else If

Else if digunakan untuk kasus yang melibatkan lebih banyak kondisi, maka kita dapat meletakkan statement else if setelah statement if. Contoh:

The screenshot shows a code editor interface with a script.js file open. The code defines a variable suhu and uses an if-else-if ladder to log different weather descriptions to the console based on the temperature. The terminal below shows the execution of the script and its output.

```
script.js > ...
1 var suhu = 18;
2
3 if (suhu > 25) {
4     console.log("Hari ini panas");
5 } else if (suhu <= 25 && suhu >= 15) {
6     console.log("Hari ini Hangat");
7 } else {
8     console.log("Hari ini dingin");
9 }
10
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER COMMENTS

```
Thoriq@DESKTOP-CU1PS3R MINGW64 /e/THORIQ/WEB 2022/JavaScript Modul
$ node script.js
Hari ini Hangat
```

Karena suhu = 18 maka isi dari statement if tidak di eksekusi karena syarat statement if adalah suhu > 25 , maka akan di lagi pada statement berikutnya yaitu else if dengan kondisi suhu <= 25 dan suhu >= 15, karena suhu di antara 15 dan 25 maka statement else if ini akan di eksekusi dan menghasilkan output Hari ini hangat. Dan percabangan otomatis berhenti dan tidak mengecek statement berikutnya lagi .

### 3. Switch

```
switch(variabel){  
    case <value>:  
        // blok kode  
        break;  
    case <value>:  
        // blok kode  
        break;  
    default:  
        // blok kode  
}
```

Selain menggunakan if..else, percabangan juga dapat ditangani dengan perintah switch. Dengan kata lain pernyataan switch digunakan untuk menyederhanakan pernyataan if..else yang terlalu banyak. Contoh:

```
script.js > ...  
1  const hari = "Jum'at";  
2  
3  switch (hari) {  
4      case "Minggu":  
5          console.log("Hari ini tidak kuliah");  
6          break;  
7      case "Sabtu":  
8          console.log("Hari ini kuliah MK Aljabar ");  
9          break;  
10     case "Jum'at":  
11         console.log("Hari ini kuliah MK Pemrograman Web");  
12         break;  
13     default:  
14         console.log("Hari tidak valid");  
15 }  
16
```

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    JUPYTER    COMMENTS

```
Thoriq@DESKTOP-CU1PS3R MINGW64 /e/THORIQ/WEB 2022/JavaScript Modul  
$ node script.js  
Hari ini kuliah MK Pemrograman Web
```

Berbeda dengan percabangan if else, pada if else pengecekan akan secara otomatis di berhentikan jika ada kondisi yang memenuhi, sedangkan pada percabagan switch kita membutuhkan keyword break untuk menghentikan pengecekan. Keyword default sama halnya dengan statement else bila tidak ada case yang memenuhi ,maka isi dari keyword default akan di eksekusi. Contoh:

```
script.js > ...
1 const hari = "Senin";
2
3 switch (hari) {
4     case "Minggu":
5         console.log("Hari ini tidak kuliah");
6         break;
7     case "Sabtu":
8         console.log("Hari ini kuliah MK Aljabar ");
9         break;
10    case "Jum'at":
11        console.log("Hari ini kuliah MK Pemrograman Web");
12        break;
13    default:
14        console.log("Hari tidak valid");
15 }
16
```

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    JUPYTER    COMMENTS

```
Thoriq@DESKTOP-CU1PS3R MINGW64 /e/THORIQ/WEB 2022/JavaScript Modul
$ node script.js
Hari tidak valid
```

Bisa di lihat pada gambar karena tidak ada pengecekan/ case hari senin maka isi dari keyword default lah yang di eksekusi.

## Perulangan

Looping atau yang lebih dikenal dengan perulangan adalah suatu metode dalam pemrograman untuk mengeksekusi suatu perintah yang sama terus-menerus hingga kondisi tertentu terpenuhi.

### 1. Perulangan For

```
// Syntax For Loop JavaScript
for([inisialisasi]; [kondisi]; [eksekusi_iterasi]) {
    // blok kode
}
```

- Inisialisasi adalah saat pertama kali mendeklarasi sebuah nilai awal, dimana nilai awal akan berubah selama belum memenuhi syarat kondisi.
- Kondisi berfungsi untuk mengecek perubahan yang terjadi setiap kali terjadi eksekusi iterasi perulangan dengan menggunakan operator perbandingan.

- Eksekusi Iterasi proses akhir setiap kali terjadi eksekusi iterasi, biasanya digunakan untuk proses penambahan (increment) atau pengurangan (decrement).

Perulangan for dibuat untuk menghasilkan output secara berulang dimana jumlah output tersebut sudah diketahui. Contoh:



```
script.js > ...
1  for (let i = 0; i <= 5; i++) {
2    console.log("Ini nomor:", i);
3  }
4

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER COMMENTS

Thoriq@DESKTOP-CU1PS3R MINGW64 /e/THORIQ/WEB 2022/JavaScript Modul
$ node script.js
Ini nomor: 0
Ini nomor: 1
Ini nomor: 2
Ini nomor: 3
Ini nomor: 4
Ini nomor: 5
```

- $i = 0$  adalah variabel yang akan menjadi permulaan perulangan for dijalankan. “ $i=0$ ” berarti perulangan yang dibuat dimulai dari 0.
- $i < 5$  adalah batas akhir program dijalankan. Nilai tertinggi  $i \leq 5$  adalah 5. Sehingga program akan terhenti saat perulangan telah berlangsung dari 0 hingga 4.
- $i++$  adalah nilai penambahan/pengurangan dari variabel  $i$ , bila isi dari perulangan sudah di eksekusi maka  $i$  akan ditambah 1, ini diperlukan agar tidak terjadi infinity loop. Infinity loop adalah perulangan tanpa akhir, hal ini terjadi karena tidak ada penambahan/pengurangan pada permulaan , contoh bila perulangan di mulai dari nol dan berakhir jika  $I \leq 5$  , bila tidak ada penambahan/pengurangan maka  $I$  tidak akan mencapai 6 untuk menghentikan loop, akhirnya infinity loop akan terjadi.

Contoh Perulangan Lainnya, Untuk mengambil Seluruh isi array:

```
js script.js > ...
1 var makanan = ["Coto", "Ayam", "Ikan"];
2
3 for (let i = 0; i < makanan.length; i++) {
4   console.log(makanan[i]);
5 }
6
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER COMMENTS

```
Thoriq@DESKTOP-CU1PS3R MINGW64 /e/THORIQ/WEB 2022/JavaScript Modul
$ node script.js
Coto
Ayam
Ikan
```

## 2. Perulangan While

```
js script.js
1 // Syntax While Loop JavaScript
2 while(kondisi){
3   //blok kode
4 }
```

Perintah while digunakan untuk perulangan yang tidak diketahui berapa kali proses perulangannya. Perintah while terus mengulangi loop selama kondisi memiliki nilai true. Contoh:

```
js script.js
1 i = 0;
2 while (i < 3) {
3   console.log("Hari", i);
4   i++;
5 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER COMMENTS

```
Thoriq@DESKTOP-CU1PS3R MINGW64 /e/THORIQ/WEB 2022/JavaScript Modul
$ node script.js
Hari 0
Hari 1
Hari 2
```

Contoh penggabungan perulangan while dengan percabangan if else:

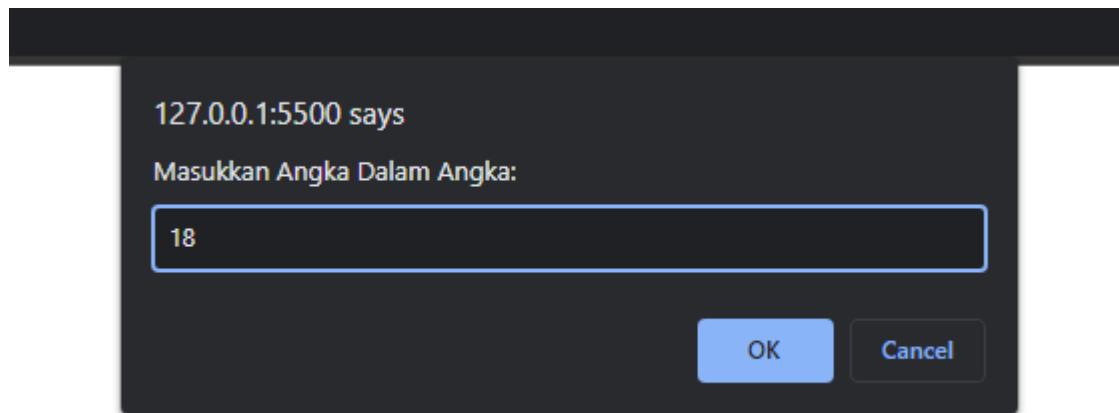
Pada kode di atas d inisialisasi variabel kuliah adalah true dan variabel i =1, lakukan perulangan while jika kuliah=true , bila i==7 maka ubah kuliah menjadi false, jika tidak eksekusi statement else dan tambahkan nilai i dengan 1. Jadi ketika I == 7 perulangan while akan di hentikan.

## **TUGAS PRAKTIKUM**

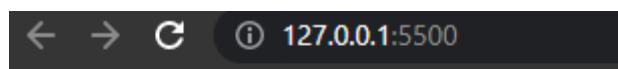
1. Seseorang ingin keluar kantor pada hari itu, buatlah program yang menampilkan informasi apakah hari itu panas , hangat , dingin atau sangat dingin berdasarkan input user. Aturannya:

- Bila Suhu lebih dari 25 maka hari itu panas
- Bila Suhu di antara 15 dan 25 maka hari itu hangat
- Bila Suhu diantara 1 dan 15 maka hari itu dingin
- Bila Suhu kurang dari 1 maka hari itu sangat dingin, dan program akan mengambil input kembali, yakin ingin keluar ? Bila ya maka sistem akan menampilkan “Hari ini sangat dingin, Hati-Hati di jalan”, jika tidak maka sistem sistem akan menampilkan “Jangan Lupa Untuk Tetap Kerja di rumah”.

**Contoh Input:**

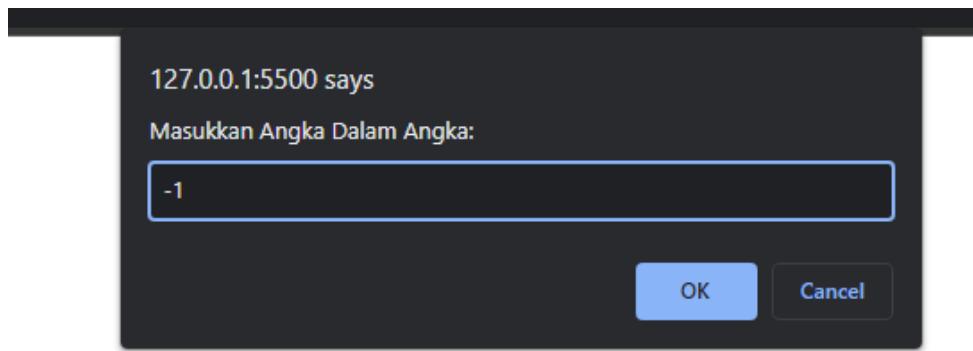


**Contoh Hasil :**

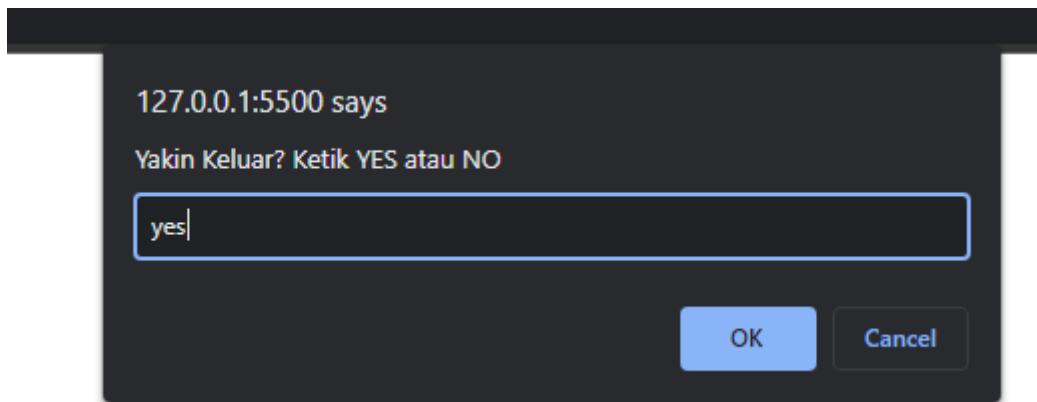


Hari ini hangat karna cuaca: 18 Derajat

**Contoh Input < 1:**



Maka Website akan meminta input lagi seperti berikut:



Maka Hasilnya:

← → C ⓘ 127.0.0.1:5500

Hari ini sangat dingin, Hati-Hati di jalan

2. Buatlah Program yang menampilkan seluruh angka mulai dari 30 sampai 1, dan jumlah tambahnya dari 30 sampai 1 ( $30 + 29 + \dots + 1$ )

Isilah Kode berikut:

```
var sum = 0;
const n = 30;

for (// Lengkapi Kode Ini) {
    //Lengkapi Kode ini
}
document.write("<br /> Jumlah:", sum);
```

Hasilnya:

← → C ① 127.0.0.1:5500

30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1  
Jumlah:465

3. Isilah Kode berikut untuk menghasilkan output yang benar

```
var bahasa = ["JavaScript", "Python", "C++"];

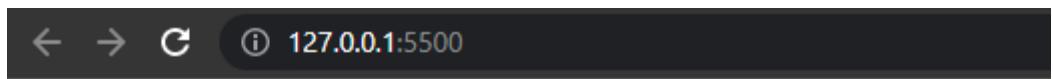
// Tambahkan Bahasa PHP di Akhir Array
document.write("<br>"); // Membuat New Line/ Enter
// Isi Jawaban di sini
document.write("Sesudah PHP ditambah: ", bahasa);

// Ubah nilai C++ menjadi C#
document.write("<br>");
// Isi Jawaban di sini
document.write("Sesudah C++ Di ubah: ", bahasa);

// Balikkan posisi array
document.write("<br>");
// Isi Jawaban di sini
document.write("Sesudah Array Dibalik: ", bahasa);

// Tampilkan Seluruh Array Menggunakan Perulangan
document.write("<br>");
// Isi Jawaban di sini
```

Hasilnya:



Sesudah PHP ditambah: JavaScript,Python,C++,PHP

Sesudah C++ Di ubah: JavaScript,Python,C#,PHP

Sesudah Array Dibalik: PHP,C#,Python,JavaScript

Index 0: PHP

Index 1: C#

Index 2: Python

Index 3: JavaScript

## ***Function***

Fungsi adalah sub-program yang bisa digunakan kembali baik di dalam program itu sendiri, maupun di program yang lain. Fungsi di dalam Javascript adalah sebuah objek. Karena memiliki properti dan juga method.

### **A. Cara Membuat Fungsi di JavaScript**

- Cara Biasa

```
script.js > ...
1  function namaFunction() {
2    // blok kode
3 }
```

- Menggunakan Ekspresi

```
js script.js > ...
1 var namaFungsi = function () {
2     //blok kode
3 };
```

e

Menggunakan variabel, lalu diisi dengan fungsi. Fungsi ini sebenarnya adalah fungsi anonim (anonymous function) atau fungsi tanpa nama.

- Menggunakan Tanda Panah (Arrow Function)

```
js script.js > ...
1 var namaFungsi = () => {
2     //block kode
3 };
4
5 
```

Cara ini sering digunakan di kode Javascript masa kini, karena lebih sederhana. Akan tetapi sulit dipahami bagi pemula. Fungsi ini mulai muncul pada standar ES6. Sebenarnya hampir sama dengan yang menggunakan ekspresi. Bedanya, kita menggunakan tanda panah ( $=>$ ) sebagai ganti function. Pembuatan fungsi dengan cara ini disebut arrow function

Contoh fungsi pada javascript:

```
js script.js > ...
1 // Membuat Fungsi
2 function sayHello() {
3     console.log("Hello, JavaScript!");
4 }
5
6 // Memanggil Fungsi
7 sayHello();
8 
```

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    JUPYTER    COMMENTS

```
Thoriq@DESKTOP-CU1PS3R MINGW64 /e/THORIQ/WEB 2022/JavaScript Modul
$ node script.js
Hello, JavaScript!
```

Contoh Fungsi dengan parameter:

```
js script.js > ...
1 // Membuat Fungsi
2 function greeting(nama) {
3   console.log("Hello", nama);
4 }
5
6 // Memanggil Fungsi
7 greeting("Budi");
8
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER COMMENTS

```
Thoriq@DESKTOP-CU1PS3R MINGW64 /e/THORIQ/WEB 2022/JavaScript Modul
$ node script.js
Hello, JavaScript!
```

## B. Return Function

Agar hasil pengolahan nilai di dalam fungsi dapat digunakan untuk proses berikutnya, maka fungsi harus mengembalikan nilai. Pengembalian nilai pada fungsi menggunakan kata kunci return kemudian diikuti dengan nilai atau variabel yang akan dikembalikan. Contoh:

```
js script.js > ...
1 function kali(a, b) {
2   hasilKali = a * b;
3   return hasilKali;
4 }
5
6 // memanggil fungsi
7 var num1 = 20;
8 var num2 = 5;
9 var hasilPerkalian = kali(num1, num2);
10 console.log(hasilPerkalian);
11
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER COMMENTS

```
Thoriq@DESKTOP-CU1PS3R MINGW64 /e/THORIQ/WEB 2022/JavaScript Modul
$ node script.js
100
```

Tanpa keyword return , outputnya akan undefined .

## **DOM (Document Object Model)**

DOM (Document Object Model) adalah model data standar. DOM adalah sebuah platform dan interface yang memperbolehkan pengaksesan dan perubahan pada konten, struktur, dan style pada sebuah dokumen. Dengan HTML DOM struktur, dan style pada dokumen html dapat diakses dan dirubah dengan menggunakan sintaks javascript. Jadi, fungsi DOM JavaScript adalah untuk memanipulasi halaman website menjadi lebih dinamis. Untuk mengambil elemen HTML, DOM JavaScript menggunakan beberapa metode:

### **1. Mengambil Elemen dengan ID**

Cara ini dilakukan melalui method getElementById(). Berikut contoh:

```
<body>
  <p id="nama">Contoh Mengambil Element dengan ID</p>
</body>
<script src="script.js"></script>

js script.js > ...
1  var nama = document.getElementById("nama");
```

Dari situ, JavaScript DOM dapat mengambil elemen dengan ID `nama` lalu menyimpannya ke dalam sebuah variabel `nama`. Variabel dan nilai id tidak perlu sama dan pastikan sudah mengkoneksikan file HTML dan file JavaScript menggunakan keyword `<script src ="namaFile.js"></script>`.

### **2. Mengambil Elemen dengan Class.**

Anda bisa mendapatkan elemen dengan method getElementsByClassName(). Ini dia contoh:

```
<body>
  <p class="MIPA">Contoh Mengambil Element dengan Class</p>
</body>
<script src="script.js"></script>

js script.js > ...
1  var fakultas = document.getElementsByClassName("MIPA");
```

Di sini kita mendapatkan semua item dengan kelas `MIPA` dan menyimpannya ke dalam variabel `fakultas`.

### **3. Mendapatkan elemen dengan nama tag**

Kita juga bisa mendapatkan elemen kita dengan nama tag menggunakan method getElementsByTagName().

```
<body>
  <h1>contoh mengambil element dengan tag</h1>
</body>
<script src="script.js"></script>
```

```
js script.js > ...
1 var judul = document.getElementsByTagName("h1");
```

Di sini kita mendapatkan semua elemen h1 dari dokumen HTML kita dan menyimpannya ke dalam variabel.

#### 4. Queryselector

Method querySelector() mengembalikan elemen pertama yang cocok dengan CSS selector yang ditentukan. Itu berarti bahwa Anda bisa mendapatkan elemen dengan id, kelas, tag dan semua CSS selector yang valid. Berikut adalah daftar beberapa opsi yang paling populer.

- Contoh Get by id:

```
js script.js > ...
1 var header = document.querySelector("#header");
```

- Contoh Get by class:

```
js script.js > ...
1 var items = document.querySelector(".list-items");
```

- Contoh Get by tag:

```
js script.js > ...
1 var headings = document.querySelector("h1");
```

Kita juga bisa mendapatkan elemen yang lebih spesifik menggunakan CSS Selector, Contoh:

```
<body>
  <h1 class="judul" id="judul">Contoh Spesifik Query</h1>
</body>
<script src="script.js"></script>
```

```
js script.js > ...
1 var spesifikQuery = document.querySelector("h1.judul#judul");
```

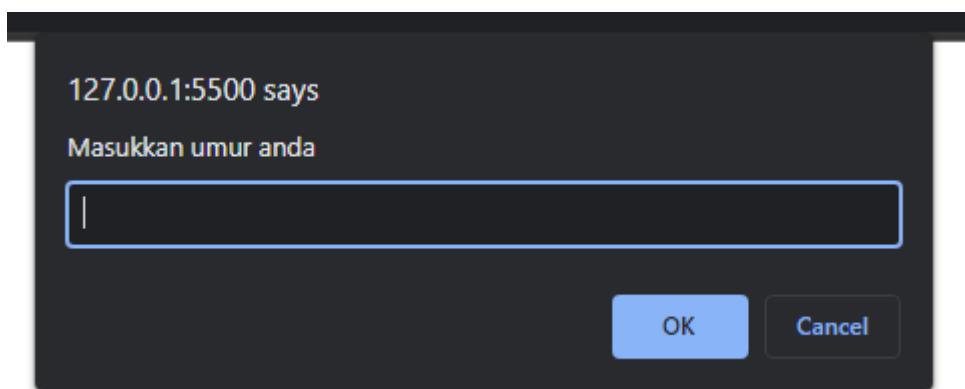
Dalam contoh ini kita mencari tag h1, kelas judul , dan id judul secara bersamaan dan menyimpannya pada variabel spesifikQuery.

Contoh function dan DOM javaScript:

```
index.html > ...
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8" />
5      <meta http-equiv="X-UA-Compatible" content="IE=edge" />
6      <meta name="viewport" content="width=device-width, initial-scale=1.0" />
7      <title>Belajar JavaScript</title>
8  </head>
9  <body>
10     <p id="nama">Disini akan muncul nama</p>
11  </body>
12  <script src="script.js"></script>
13 </html>

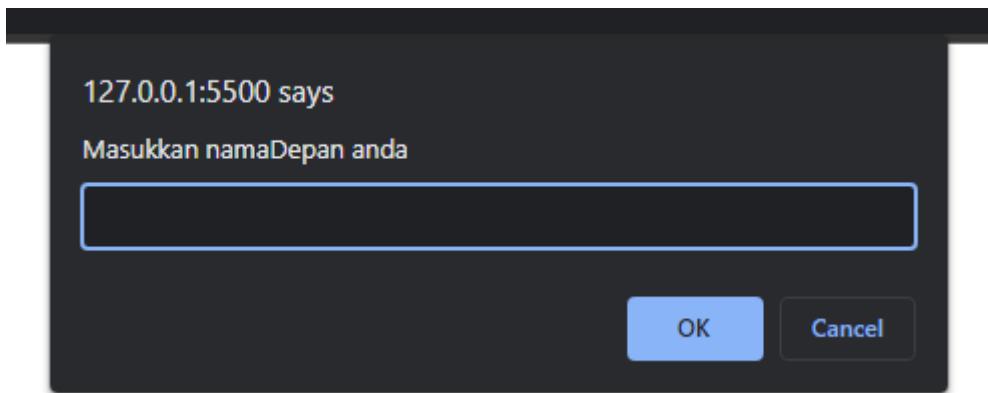
script.js > ...
1  var nama = document.getElementById("nama");
2
3  function namaLengkap(namaDepan, namaBelakang) {
4      namaDepan = prompt("Masukkan namaDepan anda");
5      namaBelakang = prompt("Masukkan namaBelakang anda");
6      nama.innerHTML = namaDepan + " " + namaBelakang + " anda bisa masuk";
7  }
8  function umurPenonton(umur) {
9      umur = prompt("Masukkan umur anda");
10     if (umur > 18) {
11         namaLengkap();
12     } else {
13         alert("Anda tidak bisa masuk!");
14     }
15 }
16 umurPenonton();
```

Ketika di lihat pada halaman web, website akan meminta input umur karena pada file script.js kita hanya memanggil fungsi umurPenonton(), prompt pada baris 4 ,5 dan 9 adalah fungsi untuk mengambil input dari user pada halaman website.



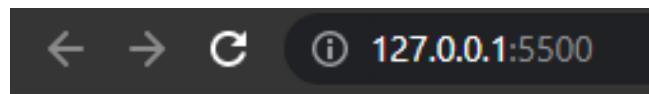
Ketika umur yang dimasukkan lebih dari 18 (lihat baris 10 ) maka fungsi namaLengkap() akan di eksusi, jika tidak maka website akan menampilkan pesan “anda tidak bisa masuk!”.

Pada fungsi namaLengkap() terdapat 2 parameter yaitu namaDepan dan namaBelakang, dan fungsi ini akan meminta input lagi dari user berupa namaDepan dan nama Belakang.



Setelah memassukan nilai namaDepan dan namaBelakang, website akan menampilkan hasil output berupa namaDepan dan nama Belakang (lihat baris 6), untuk menampilkan output ke halaman website pertama kita harus mengambil terlebih dahulu tag pada file HTML menggunakan DOM JavaScript, pada file HTML baris 10 dapat di lihat terdapat tag p dengan id="nama" lalu gunakan method getElementById (Lihat baris 1) dan masukkan id="nama", di sini kita mengambil tag p melalui idnya yaitu "nama" dan menyimpannya ke dalam variabel nama.

Lalu pada baris 6 terdapat fungsi innerHTML, innerHTML fungsinya adalah untuk menentukan dan mengembalikan semua teks, termasuk tag html, yang dikandung oleh sebuah elemen. Adapun yang dinamakan innerText yang mengembalikan semua teks yang dikandung oleh elemen dan semua elemen turunannya. innerHTML inilah salah satu cara untuk dapat membuat output tampil di halaman website.



**Budi Andi anda bisa masuk**

Maka hasilnya akan seperti gambar di atas , budi adalah contoh input user untuk nama depan dan andi adalah input user untuk nama belakang.

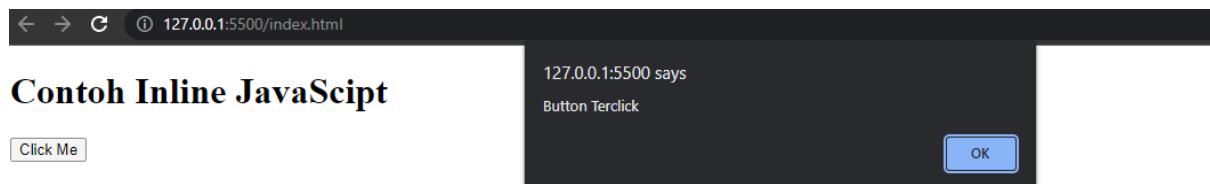
## ***Event Handler***

Event handler adalah pemanggilan kode JavaScript ketika “event” terjadi dalam tag HTML. Event handler di dalam JavaScript ditulis dengan penambahan kata “on”, seperti onclick, onload, onError, dan lain-lain Contoh Menggunakan Event OnClick:

Pada file HTML kita coba buat event onclick dengan isi alert("Button Terclick) pada tag button seperti gambar di bawah.

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Belajar JavaScript</title>
  </head>
  <body>
    <h1>Contoh Inline JavaScript</h1>
    <button onclick="alert('Button Terclick')">Click Me</button>
  </body>
</html>
```

Jadi, Ketika button di click akan browser akan memunculkan alert "Button Terclick".



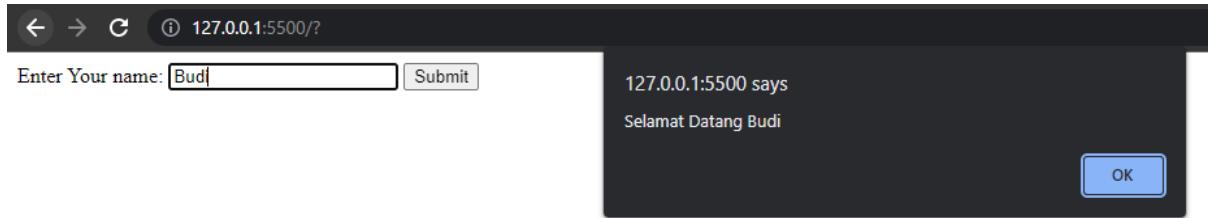
Contoh Menggunakan event onSubmit dan function:

```
<body>
  <form onsubmit="welcome()">
    Enter Your name: <input id="nama" type="text" />
    <input type="submit" value="Submit" />
  </form>
</body>
<script src="script.js"></script>

script.js > ...
1  function welcome(nama) {
2    nama = document.getElementById("nama").value;
3    alert("Selamat Datang " + nama);
4 }
```

Event onSubmit ini dijalankan setelah terjadi penekanan tombol submit yang dimiliki sebuah form. Seperti contoh gambar di atas bisa di lihat pada tag form kita memanggil event onSubmit dengan isi fungsi welcome(). Pada file script.js terdapat fungsi welcome dengan parameter nama, dengan isi nama = document.getElementById("nama").value, ini artinya kita megambil value-nilai dari tag input pada file HMTL dengan id nama.

Lalu kita buat alert untuk menampilkan selamat datang nama , seperti penjelasan sebelumnya nama adalah nilai dari input yang di masukkan user pada website. Berikut contoh hasilnya

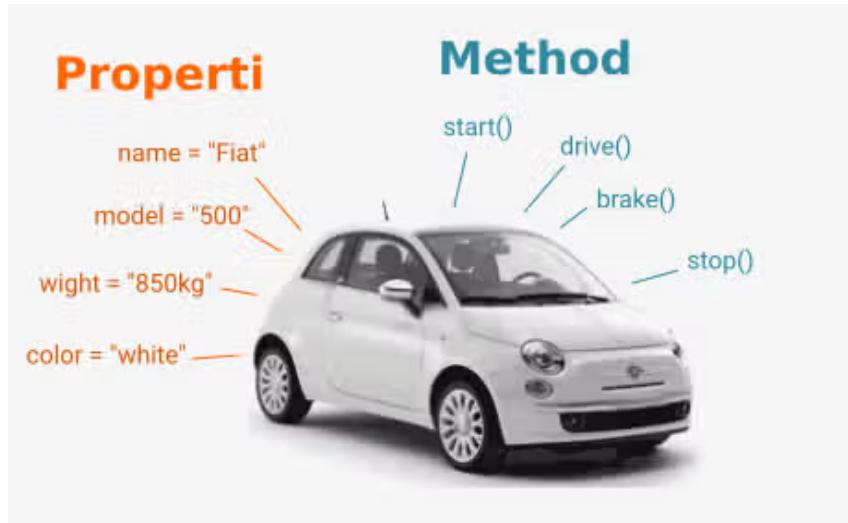


Berikut Beberapa event yang sering di gunakan:

Event	Keterangan
onClick	Event ini dibangkitkan ketika terjadi aksi klik (dilakukan oleh pengguna) terhadap element, dimana element yang dimaksud adalah semua element dari form yang dapat diklik seperti element button (tombol). Aksi klik yang dimaksud adalah adanya penekanan tombol klik kir mouse terhadap suatu element
onChange	Event ini dibangkitkan ketika sebuah element (dalam hal ini adalah element select, text dan textarea) telah diubah nilainya sebelum element tersebut kehilangan focusnya
onmousedown	Event ini dibangkitkan, ketika tombol mouse ditekan ke bawah
onMouseover	Event ini dibangkitkan ketika panah mouse berada diatas element , yaitu element hyperlink dan element area .
onSubmit	Event ini dibangkitkan setelah terjadi penekanan tombol submit yang dimiliki sebuah form
onLoad	Event ini dibangkitkan ketika suatu objek selesai ditampilkan pada halaman web. Objek yang dimaksud pada keterangan ini adalah objek Window, Frame dan Image
onMouseOut	Event ini dibangkitkan ketika panah mouse keluar dari daerah lingkupan suatu element, yaitu element hyperlink dan element area.

## Objek

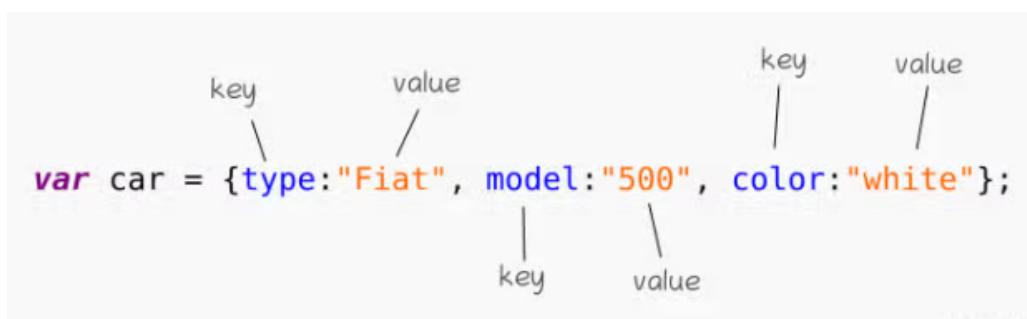
Objek sebenarnya adalah sebuah variabel yang dapat menyimpan banyak nilai (properti) dan fungsi (*method*). Contoh dalam kehidupan sehari-hari adalah Mobil.



Salah satu cara kita untuk mendeskripsikan mobil di atas sebagai berikut:

```
script.js > ...
1 var namaMobil = "Fiat";
2 var warnaMobil = "Putih";
3 var modelMobil = 500;
```

Pada gambar di atas kita membutuhkan 3 variabel untuk menyimpan nama, warna, dan model mobil. Dengan object pada javascript kita bisa menyimpan beberapa properti dalam 1 variabel. Contohnya:



Objek pada javascript, dapat dibuat dengan tanda kurung kurawal dengan isi berupa key/properti dan value yang merupakan isi dari property. Agar lebih rapi, kode di atas bisa ditulis seperti berikut:

```
script.js > ...
1  var car = {
2    type: "Fiat",
3    model: 500,
4    color: "white",
5  };
```

## 1. Perbedaan Properti dan Method

Properti adalah ciri khas dari objek (variabel). Sedangkan *method* adalah perilaku dari objek (fungsi). Method dapat dibuat dengan cara mengisi nilai (value) dengan sebuah fungsi. Contoh Method pada objek:

```
script.js > ...
1  var car = {
2    type: "Fiat",
3    model: 500,
4    color: "white",
5  };
6
7  var car = {
8    // Properti
9    type: "Fiat",
10   model: "500",
11   color: "white",
12
13  // method
14  start: function () {
15    console.log("ini method menyalakan mobil");
16  },
17  stop: function () {
18    console.log("ini method mematikan mobil");
19  },
20};
```

## 2. Cara Mengakses Property dan Method Objek

Untuk mengakses property dan method pada objek, kita menggunakan tanda titik atau dot “.” , lalu diikuti dengan nama properti atau method. Contoh:

```

js script.js > ...
1 var car = {
2   type: "Fiat",
3   model: "500",
4   color: "white",
5
6   start: function () {
7     console.log("ini method menyalakan mobil");
8   },
9   stop: function () {
10    console.log("ini method mematikan mobil");
11 },
12 };
13
14 //Mengakses Properti
15 console.log(car.model);
16 console.log(car.color);
17
18 //Mengakses method
19 car.start();
20 car.stop();

```

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    JUPYTER    COMMENTS

```

Thoriq@DESKTOP-CU1PS3R MINGW64 /e/THORIQ/WEB 2022/JavaScript Modul
$ node script.js
500
white
ini method menyalakan mobil
ini method mematikan mobil

```

Untuk mengakses properti, kita cukup gunakan nama objek.NamaProperti. Sedangkan untuk method, kita gunakan objek.NamaMethod (). Untuk mengakses method kita harus menggunakan tanda kurung.

### 3. Menggunakan Keyword this

Kata kunci this digunakan untuk mengakses property dan method dari dalam method (objek). Contohnya:

```

js script.js > ...
1 var person = {
2   firstName: "Budi",
3   lastName: "Dian",
4   showName: function () {
5     console.log(`Nama: ${this.firstName} ${this.lastName}`);
6   },
7 }
8
9 person.showName();

```

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    JUPYTER    COMMENTS

```

Thoriq@DESKTOP-CU1PS3R MINGW64 /e/THORIQ/WEB 2022/JavaScript Modul
$ node script.js
Nama: Budi Dian

```

Keyword this.firstname dan this.lastname pada contoh diatas hanya akan mengacu pada objek person.

#### 4. Membuat Class Objek

Pada pemrograman berorientasikan objek, kita biasanya membuat objek dengan membuat instance dari class. Akan tetapi pada contoh-contoh di atas, kita membuat objeknya secara langsung. Lalu bagaimana kalau kita ingin membuat objek yang lain dengan properti yang sama. Contoh tidak efektif:



```
script.js > ...
1 var person = {
2   firstName: "Muh",
3   lastName: "Anton",
4   showName: function () {
5     console.log(`Nama Pertama: ${this.firstName} ${this.lastName}`);
6   },
7 };
8
9 var person2 = {
10   firstName: "Thoriq",
11   lastName: "AS",
12   showName: function () {
13     console.log(`Nama Kedua: ${this.firstName} ${this.lastName}`);
14   },
15 };
16 person.showName();
17 person2.showName();
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER COMMENTS

```
Thoriq@DESKTOP-CU1PS3R MINGW64 /e/THORIQ/WEB 2022/JavaScript Modul
$ node script.js
Nama Pertama: Muh Anton
Nama Kedua: Thoriq AS
```

Ini tentu tidak efektif, jika kita ingin membuat banyak objek. Karena kita harus menulis ulang kode yang sama. Solusinya kita bisa gunakan class. Pada Javascript versi ES5, class belum ada. Fitur ini baru ditambahkan pada Javascript versi ES6. Contohnya:



```
script.js > ...
1 class Mahasiswa {
2   constructor(namaLengkap, tahun, fakultas) {
3     this.namaLengkap = namaLengkap;
4     this.tahun = tahun;
5     this.fakultas = fakultas;
6
7     this.fullName = function () {
8       return `Nama Lengkap: ${this.namaLengkap} \nUmur: ${this.tahun}`;
9     };
10    this.fakultasMhs = function () {
11      return `Fakultas: ${this.fakultas}`;
12    };
13  }
14 }
15 var Person1 = new Mahasiswa("Thoriq", 2020, "MIPA");
16 var Person2 = new Mahasiswa("Ali", 2021, "FARMASI");
17 console.log(Person1.fullName());
18 console.log(Person1.fakultasMhs());
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER COMMENTS

```
Thoriq@DESKTOP-CU1PS3R MINGW64 /e/THORIQ/WEB 2022/JavaScript Modul
$ node script.js
Nama Lengkap: Thoriq
Umur: 2020
Fakultas: MIPA
```

Pada gambar di atas Constructor adalah special method yang dieksekusi sebelum method yang lain. Kita membuat objek baru dengan keyword new, lalu diberikan nilai parameter namaLengkap, tahun, dan fakultas. Jadi, berapapun objek yang ingin kita buat cukup tambah keyword new.

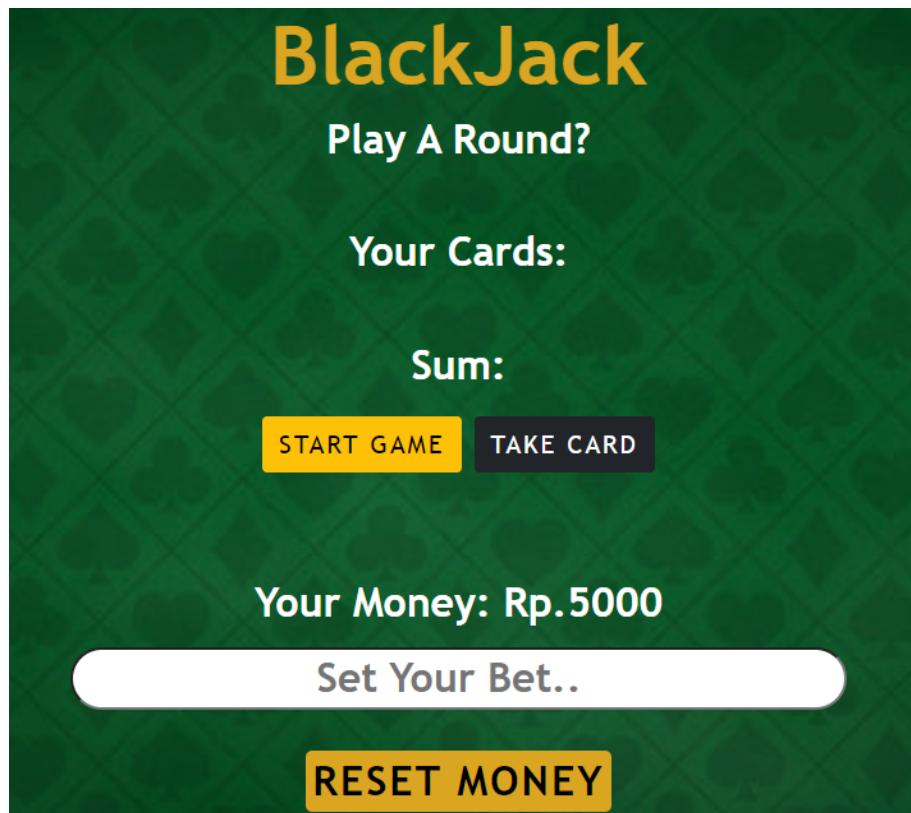
## ***TUGAS PRAKTIKUM***

Buatlah sebuah game yang berjudul BlackJack Game (Game Kartu) dengan aturan berikut:

1. Saat Awal game pemain diberi uang sebesar 5000.
2. Untuk memulai game pemain harus bertaruh (bet) terlebih dahulu dan taruhan harus  $\leq$  uang yang dimiliki pemain saat itu.
3. Pada game ini satu kartu minimal bernilai 1 dan maksimal 11.
4. Ketika menekan tombol mulai, uang pemain otomatis terpotong dari taruhan yang telah ia tentukan, dan pemain akan diberi 2 kartu acak.
5. Pemain dapat menarik kartu random, dengan syarat jika jumlah nilai kartunya tidak sama dengan 21 atau belum melebihi 21.
6. Bila Jumlah Kartu = 21, maka pemain menang dan uang pemain akan bertambah 5x lipat dari taruhan yang ia pasang.
7. Bila Jumlah nilai kartu yang dimiliki pemain lebih dari 21, maka pemain kalah dan uang yang ditaruhnya tidak akan kembali.
8. jika uang yang dimiliki pemain habis , maka permainan berakhir (game over) dan harus mereset uang kembali

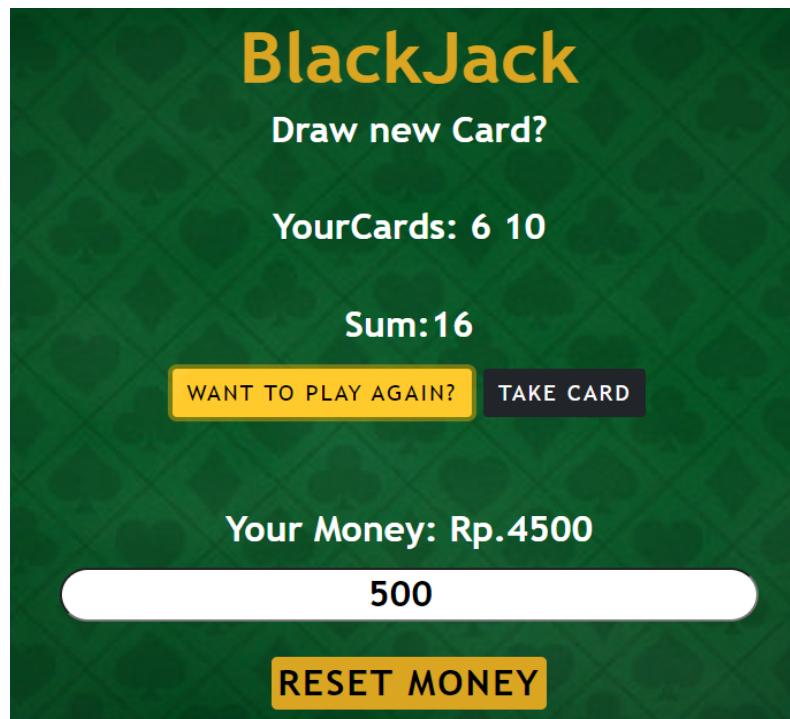
**Note: Tampilan/ Desain Dari Website tidak harus sama dengan contoh di bawah!.**

**Contoh Sebelum Memulai Game:**



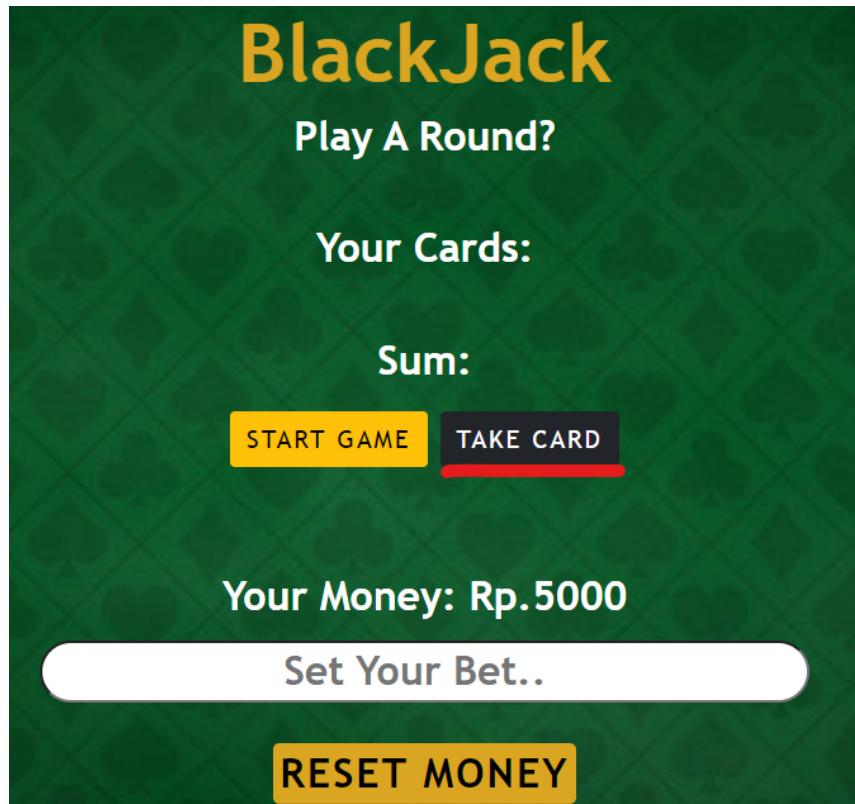
1. Untuk memulai game harus bertaruh(bet) uang terlebih dahulu, Ketika Menekan Tombol Start maka pemain akan langsung diberi 2 kartu random.

**Contoh Ketika Menekan Tombol Start:**

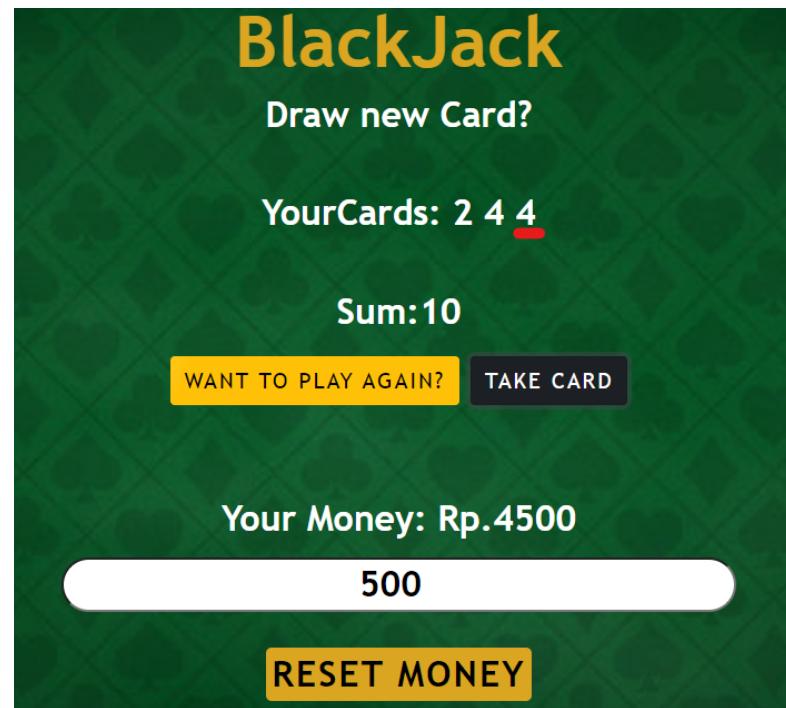


2. Pemain dapat menarik kartu random dengan, syarat jumlah nilai kartunya tidak sama dengan 21 atau belum melebihi 21.

**Contoh jumlah nilai kartu < 21:**



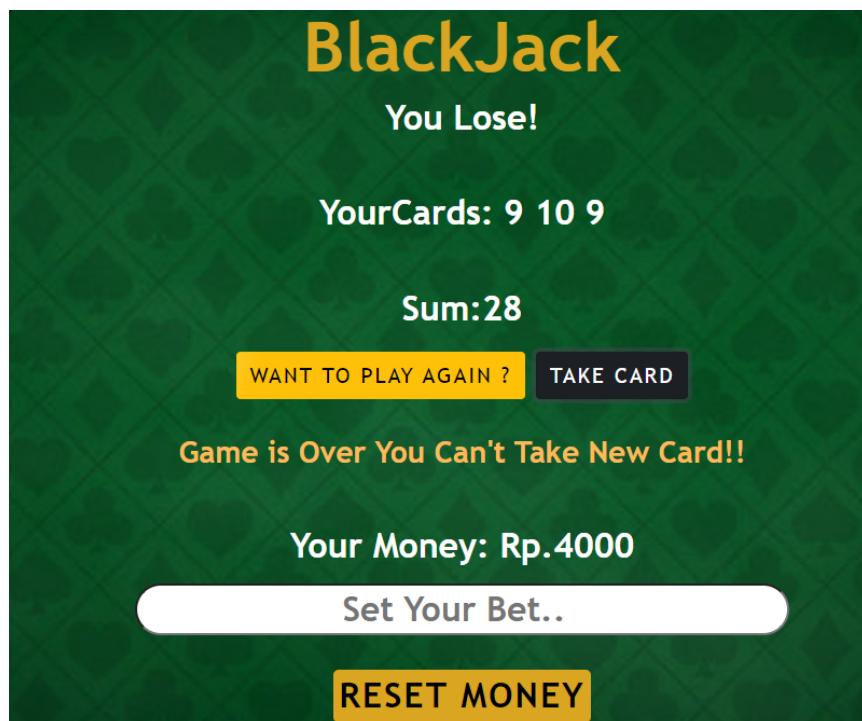
Ketika Menekan tombol take card, maka hasilnya:



Maka akan masuk kartu baru yaitu 4 pada contoh di atas.

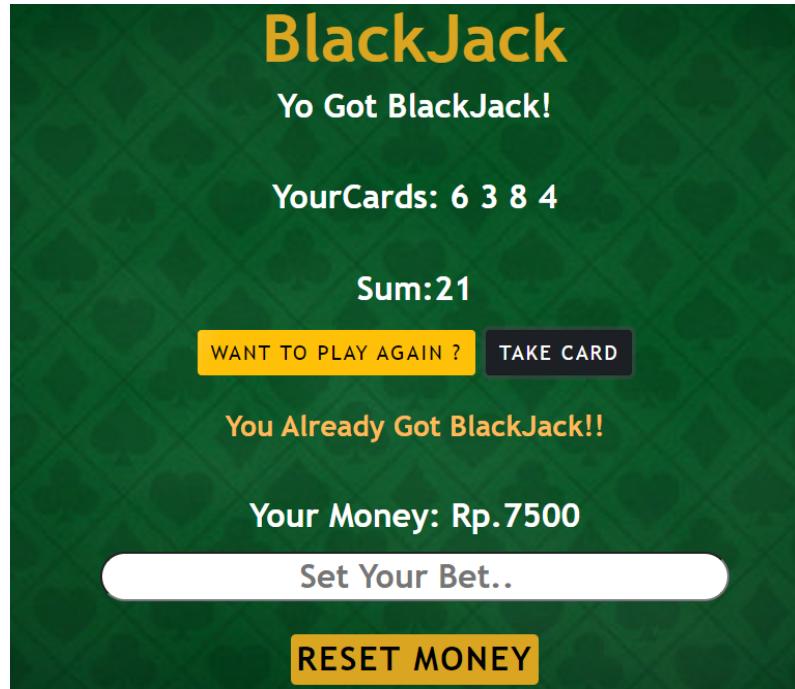
Jika jumlah kedua kartu belum mencapai 21 atau lebih dari 21 , Maka pemain dapat menarik kartu lagi. Tapi jika jumlah = 21 dan jumlah kartu > 21 , maka game berakhir dan harus dimulai dari awal.

**Contoh Jumlah Kartu > 21:**



Bila jumlah nilai kartu pemain > 21, dan mencoba untuk menarik kartu lagi maka akan muncul text berupa “Game is Over You Can’t take new card” seperti gambar di atas.

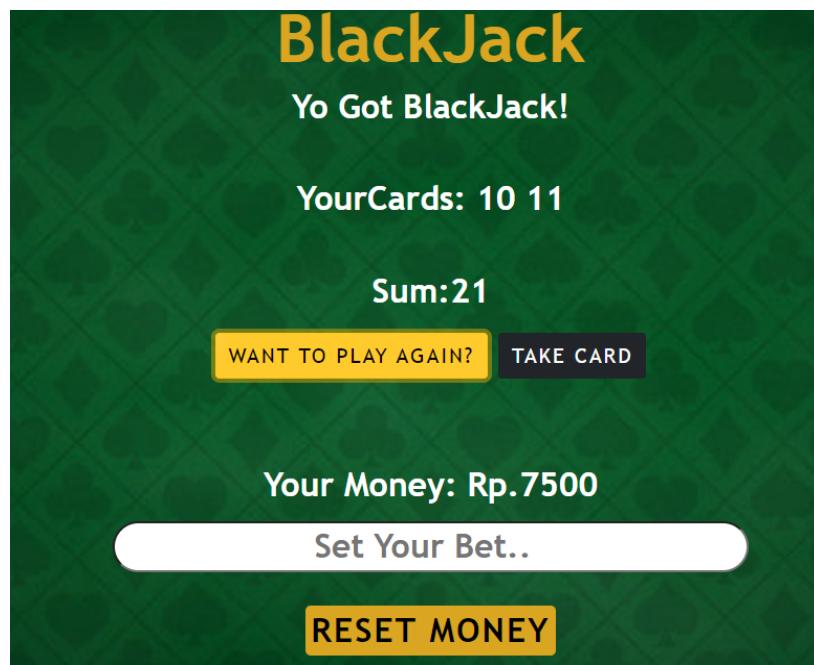
**Contoh Jumlah Kartu = 21:**



Bila jumlah nilai kartu pemain = 21, dan mencoba untuk menarik kartu lagi maka akan muncul text berupa “You Already Got BlackJack” seperti gambar di atas.

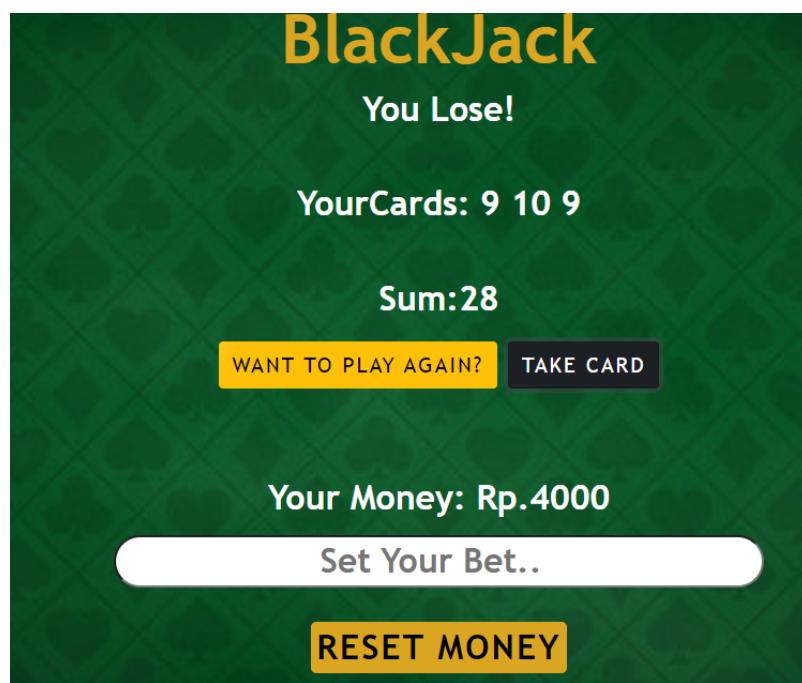
3. Game dimenangkan jika jumlah semua kartu adalah 21 (BlackJack) dan Jika Lebih dari 21 Pemain Kalah.

**Contoh Pemain Menang:**



Pada gambar di atas pemain bertaruh 500, karena jumlah nilai kartu pemain adalah 21 (BlackJack) maka uang pemain bertambah  $500 * 5 = 2500$ .

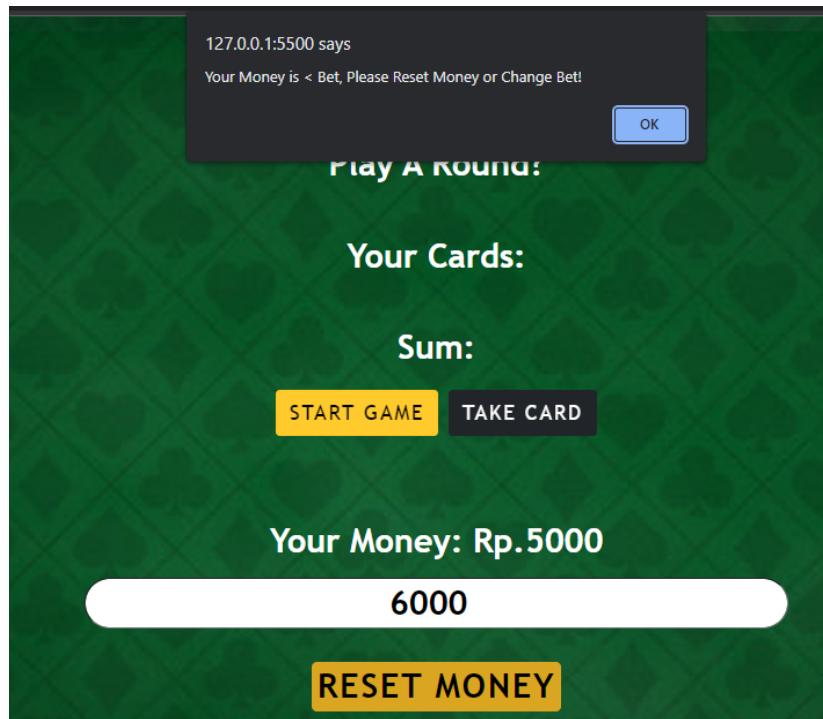
**Contoh Pemain Kalah:**



Pada gambar di atas pemain bertaruh 1000, karena jumlah nilai kartu pemain lebih dari 21 maka uang pemain berkurang 1000.

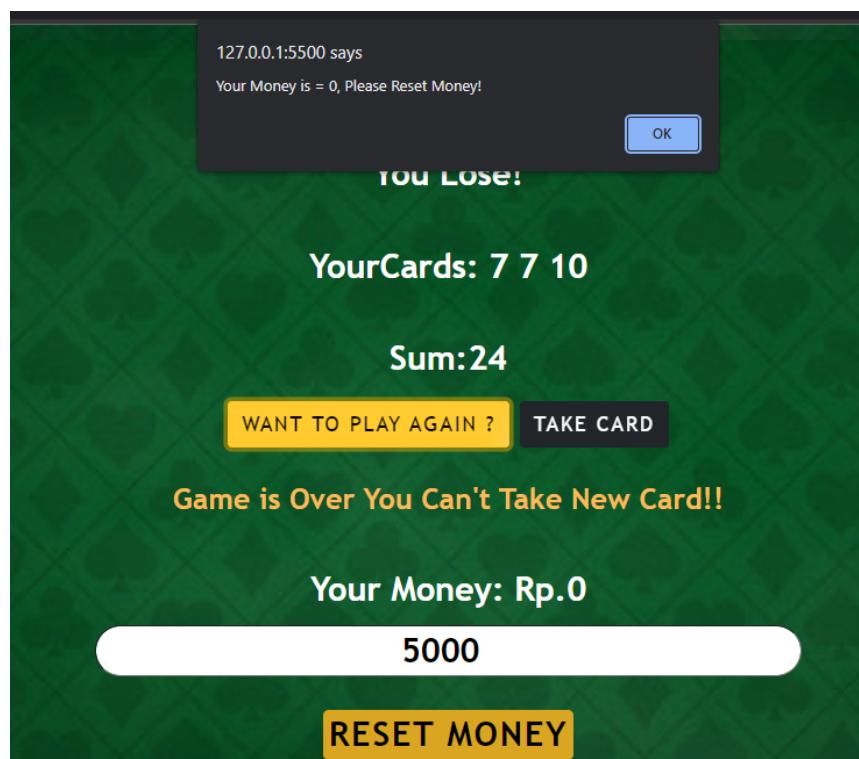
4. Jika uang pemain kurang dari taruhannya makan akan muncul alert

**Contohnya:**



5. Jika Uang Pemain Habis , Maka akan muncul alert Your money = 0 Please Reset Your Money

**Contohnya:**



## **BAB V**

### **Serverside**

#### ***Web Statis dan Web Dinamis***

Web merupakan salah satu aplikasi yang berisi dokumen multimedia (teks, gambar, suara, animasi, video ) di dalamnya yang menggunakan protokol HTTP (Hypertext Transfer Protocol) dan untuk mengaksesnya menggunakan browser. Situs web adalah kumpulan dari halaman web yang sudah dipublikasikan ke jaringan internet dan memiliki domain/URL (Uniform Resource Locator) yang dapat diakses oleh seluruh pengguna internet dengan cara mengetikkan alamatnya. Jika di tinjau dari aspek content atau isinya, web dapat dibagi menjadi 2 jenis, yaitu :

##### **1. Web Statis**

Web statis merupakan Website yang penggunaanya tidak dapat merubah content atau isi dari web tersebut secara langsung melalui browser. Web statis menggunakan Client Side Scripting untuk teknologinya seperti HTML dan Cascading Style Sheet (CSS) sehingga membuat dokumen web statis tidak memungkinkan dilakukan perubahan isi/data. Perubahan isi/data pada web statis ini hanya dapat dilakukan dengan cara megubah 2 langsung isinya pada file web mentah tersebut. Contoh web statis : Web Profil Perusahaan.

##### **2. Web Dinamis**

Web dinamis merupakan jenis web yang isi atau content-nya dapat berubah-ubah setiap saat. Melalui web dinamis ini user dapat langsung merubah data melalui halaman control panel atau administrasi yang biasanya telah disediakan untuk user Administrator.

Beberapa komponen yang diperlukan untuk membuat web dinamis yaitu Client Side Scripting (HTML, Java Script, Cascading Style Sheet) dan Server Side Scripting (seperti PHP, program basis data seperti MySQL). Contoh web Dinamis: Situs Web berita, Situs Web E-Commerce.

## ***Web Server***

Web Server Web Server adalah sebuah komputer yang terdiri dari perangkat keras dan perangkat lunak dan bekerja sebagai penyedia layanan yang dapat diakses oleh banyak pengguna sehingga membutuhkan kapasitas dan kapabilitas yang lebih besar daripada PC biasa. Web server bertanggung jawab untuk menerima permintaan HTTP dari komputer klien dan merupakan program aplikasi yang berfungsi sebagai tempat menyimpan semua dokumen[1]dokumen web kedalam direktori utama web server (document root).

Web server terdiri dari dua jenis yaitu Online Mode dan Offline Mode. Pada Online Mode, selain komputer, anda harus menyiapkan domain dan hosting serta koneksi internet yang memadai untuk mengelolanya. Sedangkan pada Offline mode, cukup komputer dan beberapa software untuk membuat web server lokal.

Beberapa Web Server yang banyak digunakan di internet antara lain:

1. Apache Web Server
2. Internet information Service, IIS
3. Xitami Web Server
4. Sun Java System Web Server

Selain itu Ada beberapa jenis software untuk membangun web server lokal atau localhost yang support dengan sistem operasi Windows, diantaranya adalah WampServer, Appserv, **XAMPP**, PHP Triad atau Vertrigo.

## ***Server Side Scripting***

Server side scripting merupakan salah satu jenis Bahasa pemrograman web yang proses pengolahannya dilakukan di sisi server. Server Side Scripting inilah yang memungkinkan untuk menghasilkan halaman web yang dinamis. Berikut adalah beberapa bahasa pemrograman yang biasa digunakan dalam Server-side scripting adalah :

1. **PHP**
2. ASP
3. Python

4. Javascript(NodeJS)

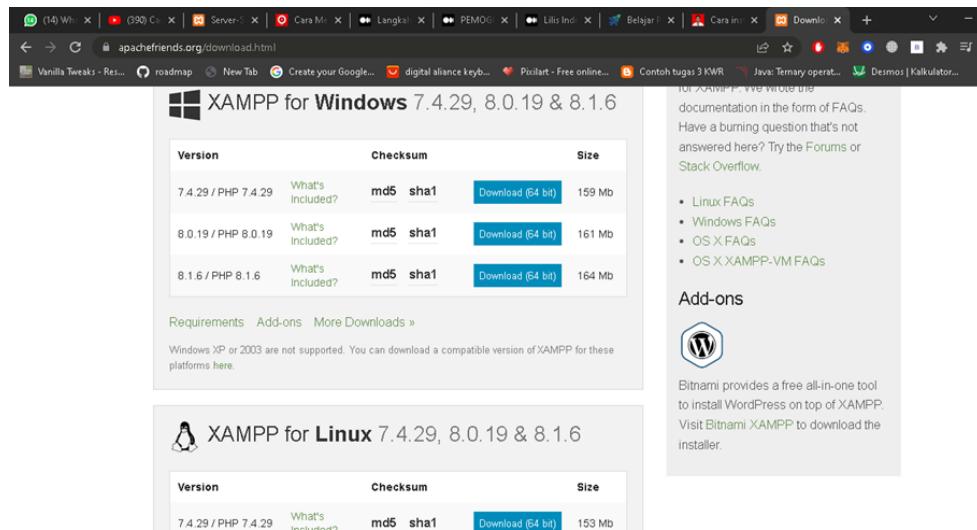
5. GO

Yang akan kita bahas di Server Side Scripting hanya PHP

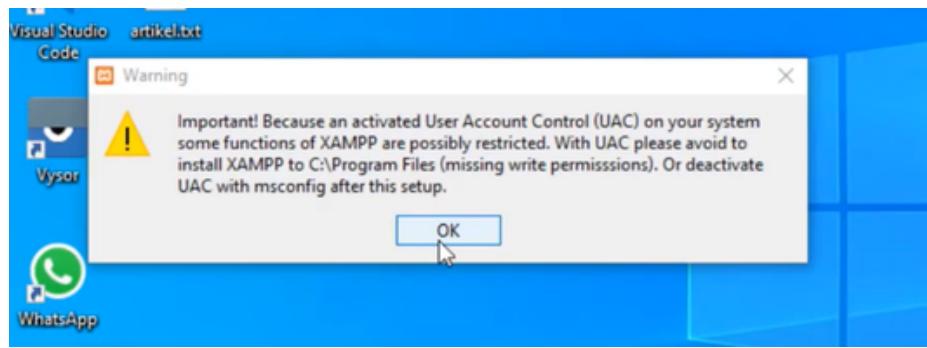
## ***Instalasi Apache, PHP dan MySQL dengan XAMPP***

XAMPP merupakan perangkat lunak bebas yang mendukung banyak system operasi. XAMPP terdiri dari web server Apache, database MySQL, dan modul PHP / penerjemah bahasa yang ditulis dalam bahasa pemrograman PHP dan PERL. XAMPP memiliki fungsi sebagai server yang berdiri sendiri (localhost). Melalui XAMPP, developer dapat menguji aplikasi web yang dibuat di komputer tanpa harus terkoneksi internet, dengan kata lain membuka web secara offline.

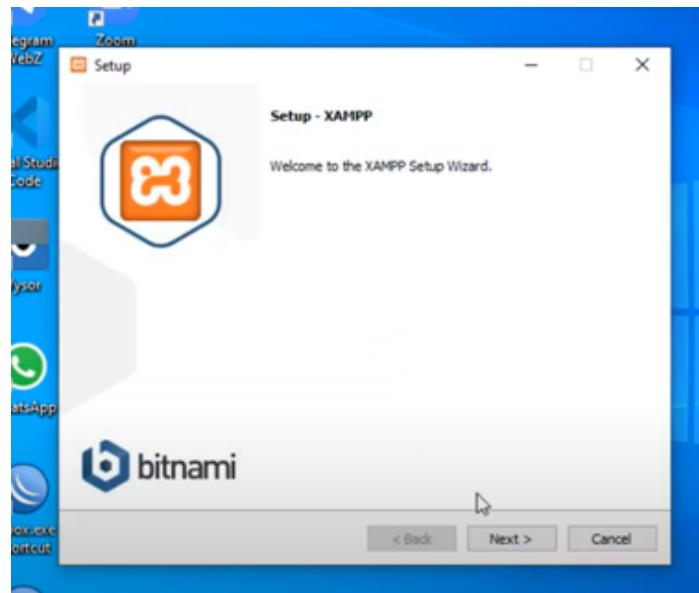
1. Download aplikasi XAMPP versi stabil terbaru di: <https://www.apachefriends.org/download.html>. Pilih versi yang sesuai dengan system operasi yang anda gunakan.



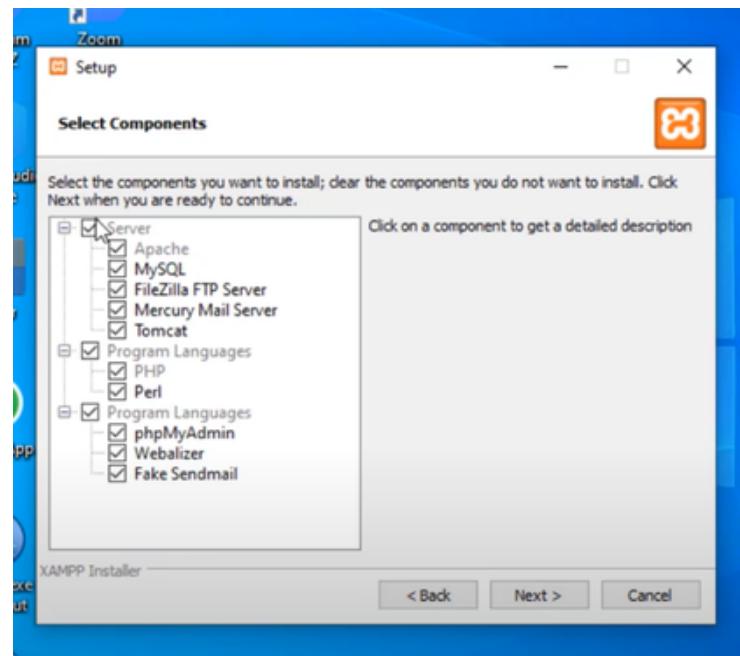
2. Buka folder lokasi dimana file setup XAMPP telah terunduh.
3. Klik kanan pada file srtup XAMPP, dan pilih Run As Administrator.
4. Jika ada peringatan User Account Control, anda klik OK saja



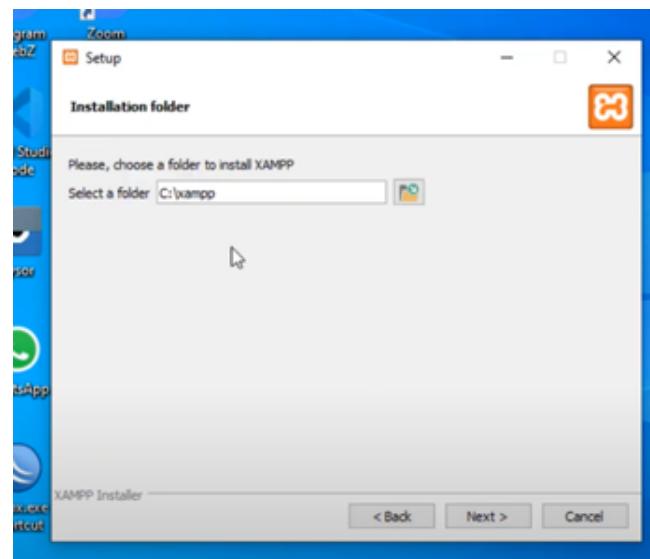
5. Jendela instalasi XAMPP akan terbuka, klik Next >



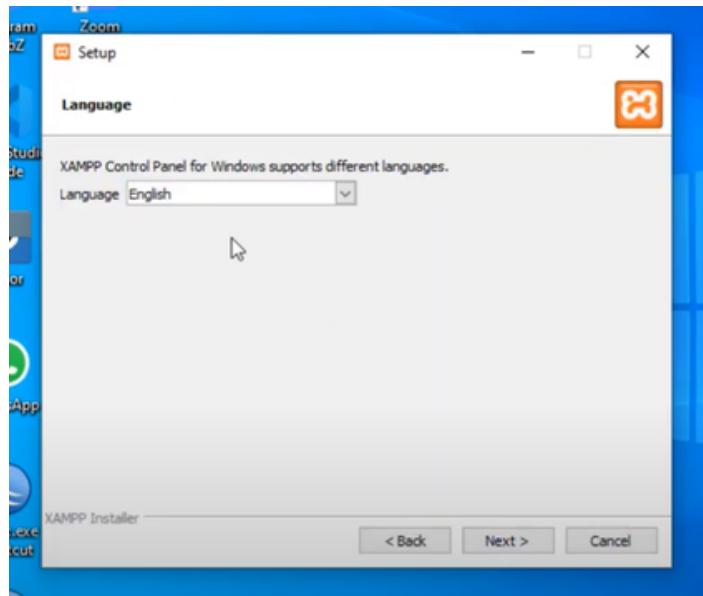
6. Ceklist semua lalu klik Next >



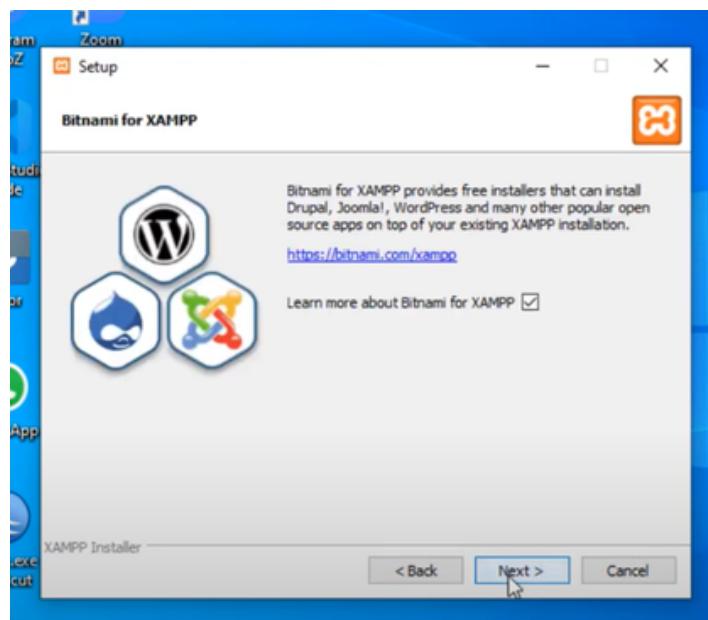
7. Pemilihan lokasi folder instalasi, secara default ada di C:\xampp. Tapi jika ruang hardisk menipis, silahkan ditempatkan di Drive Hardisk yang lebih lega, seperti D atau E (jika ada beberapa partisi hardisk). Pastikan juga nama folder xampp tidak ada di drive C:, klik Next >



8. Pilihan Bahasa silahkan disesuaikan saja, klik Next >



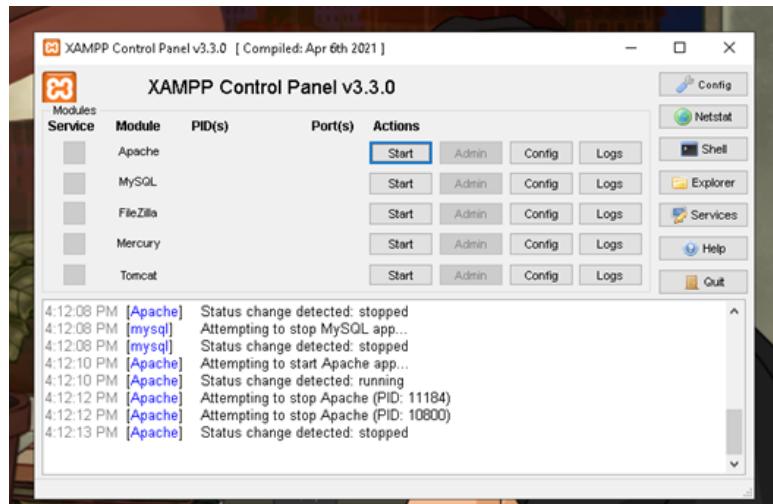
9. Instalasi akan mulai berjalan, klik Next >



10. Tunggu hingga finish

Sampai tahap ini, berarti kita sudah menginstal XAMPP. Itu berarti kita sudah selesai menginstall PHP, APACHE dan MYSQL. Langkah selanjutnya adalah menjalankan servicenya.

11. Jalankan XAMPP Control Panel yang ada di desktop. Nyalakan Apache dan Mysql dengan mengklik tombol Start.



12. Buka web browser anda, lalu ketikkan `http://localhost`. Jika tampilannya seperti di bawah ini, maka apache sudah terinstall dengan benar.

The screenshot shows a web browser window with the URL `localhost/dashboard/` in the address bar. The page title is 'XAMPP Apache + MariaDB + PHP + Perl'. Below the title, it says 'Welcome to XAMPP for Windows 8.1.6'. It includes a message about successful installation and links for 'Applications', 'FAQs', 'HOW-TO Guides', 'PHPInfo', and 'phpMyAdmin'. A note at the bottom states: 'XAMPP has been around for more than 10 years – there is a huge community behind it. You can get involved by joining our [Forums](#), adding ...'.

## **Struktur Penulisan Server Side Scripting (PHP)**

Format penulisan script PHP dibuka oleh tag **<?php dan ditutup dengan ?>**. Perlu digaris bawahi bahwa tugas PHP adalah mengolah data sesuai dengan sintaks yang dituliskan (keinginan pengguna), sedangkan untuk mengatur format tampilan di layar merupakan tugas dari HTML. Oleh sebab itu, perlu diketahui cara menyisipkan skrip PHP ke dalam HTML. Ada 2 cara aturan penulisan skrip PHP yaitu:

## 1. Embedded Script

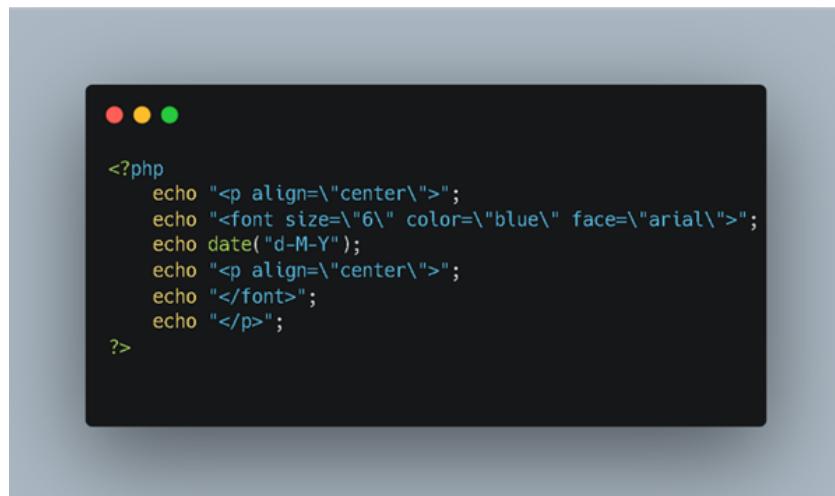
Dengan cara meletakkan tag php diantara tag-tag HTML. Contoh :



```
<html>
  <head>
    <title> Embedded Script </title>
  </head>
  <body>
    <?php
      echo "Pemograman Web";
      echo date ("d-M-Y");
    ?>
  </body>
</html>
```

## 2. Non Embedded Script

Dengan cara meletakkan skrip HTML di dalam skrip PHP. Contoh :



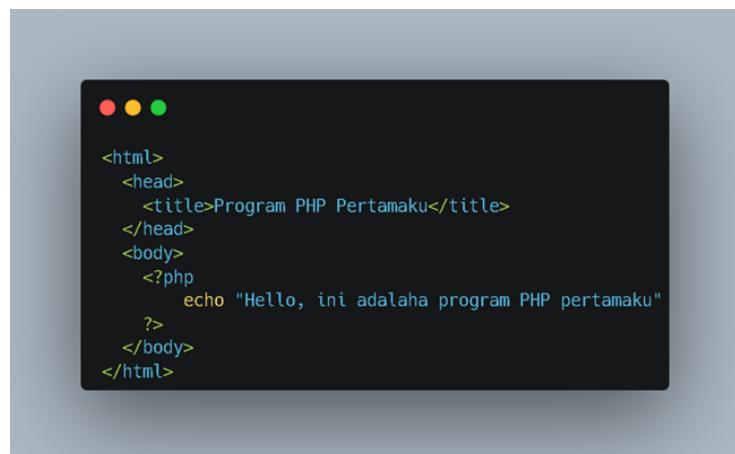
```
<?php
  echo "<p align=\\"center\\>";
  echo "<font size=\\"6\\" color=\\"blue\\" face=\\"arial\\>";
  echo date("d-M-Y");
  echo "<p align=\\"center\\>";
  echo "</font>";
  echo "</p>";
?>
```

## *Menampilkan Program PHP di localhost*

1. Buatlah folder LatihanPHP di folder htdocs (folder htdocs berada di file instalasi XAMPP kalian)

This PC > DATA2 (D:) > XAMPP > htdocs			
Name	Date modified	Type	Size
New folder	9/12/2020 20:08	File folder	

2. Ketikkan perintah ini pada code editor anda.



```
<html>
<head>
    <title>Program PHP Pertamaku</title>
</head>
<body>
    <?php
        echo "Hello, ini adalah program PHP pertamaku"
    ?>
</body>
</html>
```

3. Simpan dengan nama hello.php (Ingat, simpan semua file web anda di dalam XAMPP/htdocs/(nama folder yang sudah anda buat tadi)).
4. Untuk melihat hasilnya di browser. Buka Browser Ketikkan <http://localhost/LatihanPHP/hello.php>



Hello, ini adalah Program PHP Pertamaku

## **BAB VI**

### **PHP**

#### **DASAR-DASAR PHP**

##### ***PENGERTIAN DAN SEJARAH PHP***

PHP (Hypertext Preprocessor) adalah bahasa skrip yang dapat ditanamkan atau disisipkan ke dalam HTML. PHP banyak dipakai untuk memrogram situs web dinamis. PHP dapat digunakan untuk membangun sebuah CMS.

Pada awalnya PHP merupakan kependekan dari Personal Home Page (Situs personal). PHP pertama kali dibuat oleh Rasmus Lerdorf pada tahun 1995. Pada waktu itu PHP masih bernama Form Interpreted (FI), yang wujudnya berupa sekumpulan skrip yang digunakan untuk mengolah data formulir dari web.

Selanjutnya Rasmus merilis kode sumber tersebut untuk umum dan menamakannya PHP/FI. Dengan perilisan kode sumber ini menjadi sumber terbuka, maka banyak pemrogram yang tertarik untuk ikut mengembangkan PHP.

Pada November 1997, dirilis PHP/FI 2.0. Pada rilis ini, interpreter PHP sudah diimplementasikan dalam program C. Dalam rilis ini disertakan juga modul-modul ekstensi yang meningkatkan kemampuan PHP/FI secara signifikan.

Pada tahun 1997, sebuah perusahaan bernama Zend menulis ulang interpreter PHP menjadi lebih bersih, lebih baik, dan lebih cepat. Kemudian pada Juni 1998, perusahaan tersebut merilis interpreter

baru untuk PHP dan meresmikan rilis tersebut sebagai PHP 3.0 dan singkatan PHP diubah menjadi akronim berulang PHP: Hypertext Preprocessing.

Pada pertengahan tahun 1999, Zend merilis interpreter PHP baru dan rilis tersebut dikenal dengan PHP 4.0. PHP 4.0 adalah versi PHP yang paling banyak dipakai pada awal abad ke-21. Versi ini banyak dipakai disebabkan kemampuannya untuk membangun aplikasi web kompleks tetapi tetap memiliki kecepatan dan stabilitas yang tinggi.

Pada Juni 2004, Zend merilis PHP 5.0. Dalam versi ini, inti dari interpreter PHP mengalami perubahan besar. Versi ini juga memasukkan model pemrograman berorientasi objek ke dalam PHP untuk menjawab perkembangan bahasa pemrograman ke arah paradigma berorientasi objek.

Versi terbaru dari bahasa pemograman PHP adalah versi 8.0.23 yang resmi dirilis pada tanggal 01 September 2022.

## ***SINTAKS PHP***

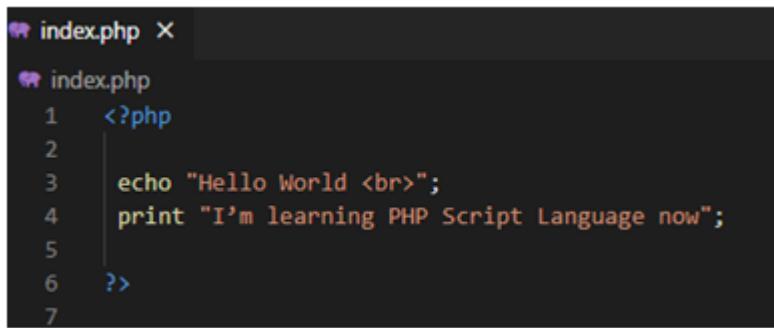
Skrip PHP dieksekusi di server dan hasil eksekusi yang berupa kode HTML dikirim ke komputer klien.

### **Tag PHP**

Skrip PHP selalu di awali dengan tanda ‘<?php’ dan ditutup dengan ‘?>’. Skrip PHP dapat diletakkan dimana saja dalam suatu dokumen HTML ataupun menuliskannya secara terpisah di file PHP dengan **ekstensi.php**. Beberapa server yang telah diatur konfigurasi directive ‘shorthand-support’, dapat mengawali skrip dengan tanda ‘<?’ dan diakhiri dengan ‘?>’. Tetapi demi kompatibilitas maksimum, disarankan menggunakan bentuk standar ‘<?php’.

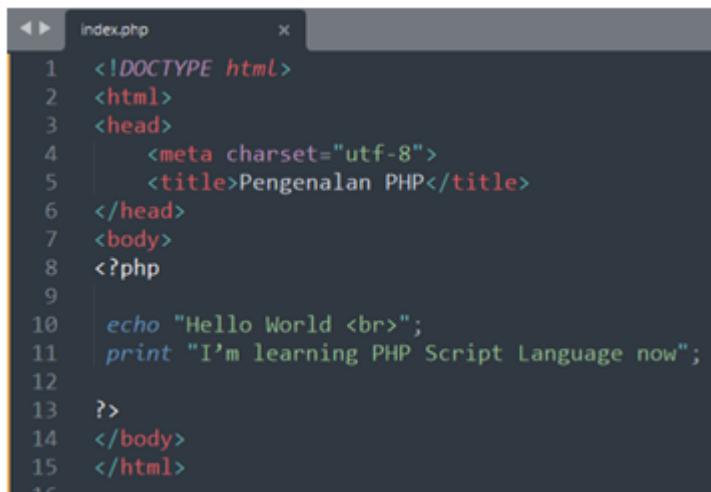
Berikut beberapa contoh penulisan skrip PHP.

- Penulisan skrip PHP pada umumnya.



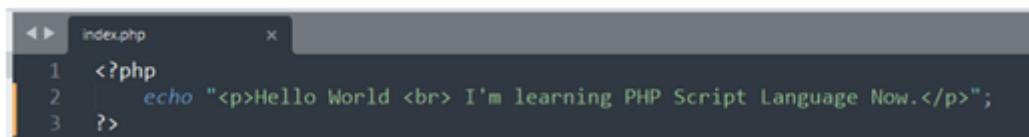
```
index.php
1 <?php
2
3 echo "Hello World <br>";
4 print "I'm learning PHP Script Language now";
5
6 ?>
7
```

- Penulisan skrip PHP di dalam HTML



```
index.php
1 <!DOCTYPE html>
2 <html>
3 <head>
4     <meta charset="utf-8">
5     <title>Pengenalan PHP</title>
6 </head>
7 <body>
8 <?php
9
10 echo "Hello World <br>";
11 print "I'm learning PHP Script Language now";
12
13 ?>
14 </body>
15 </html>
```

- Penulisan skrip HTML di dalam PHP



```
index.php
1 <?php
2     echo "<p>Hello World <br> I'm learning PHP Script Language Now.</p>";
3 ?>
```

## Statement PHP

Suatu skrip PHP dapat berisi satu atau lebih intruksi / statement. Setiap intruksi PHP harus diakhiri dengan symbol semikolon ‘ ; ’. Tugas simbol ini untuk membedakan atau memisahkan satu intruksi PHP dengan instruksi PHP lainnya.

The screenshot shows a code editor window titled "statement.php" containing the following PHP code:

```
1 <?php
2
3     $a = 1;
4     $b = 2;
5     $c = $a + $b;
6     $d = $a * $b;
7     echo "a + b = ";
8     echo $c;
9     echo "<br>";
10    echo "a * b = ";
11    echo $d;
12 ?>
```

To the right, a browser window titled "localhost/modul-php/statement" displays the output of the code:

a + b = 3  
a \* b = 2

## Komentar PHP

Ada kalanya kita sebagai pemrogram, perlu menandai atau memberi komentar ataupun deskripsi pada suatu blok kode di dalam program. Komentar pada program merupakan tulisan pada program yang tidak dieksekusi pada PHP, ada 3 macam cara penulisannya:

### 1. /\* komentar \*/

Tulisan apapun yang berada di antara '/\*' dan '\*/' akan dianggap sebagai komentar. Cara seperti ini sangat berguna dan efisien untuk pemberian komentar yang memakan banyak baris.

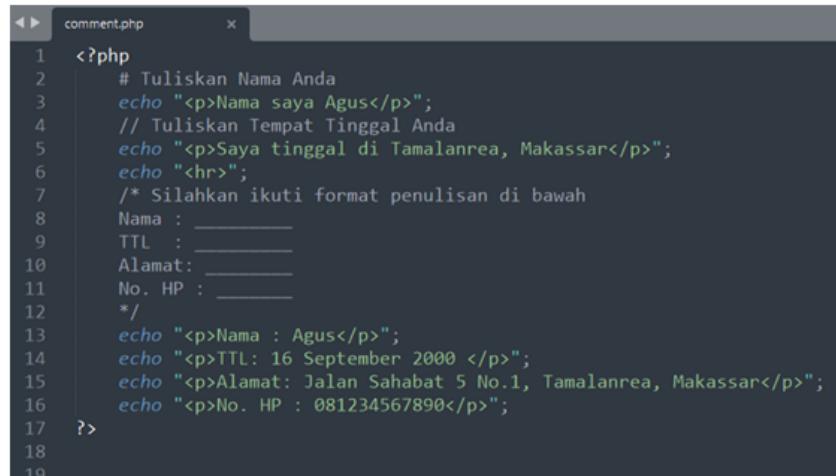
### 2. // komentar

Tulisan di baris yang sama setelah '//' akan dianggap sebagai komentar. Cara ini berguna untuk pemberian komentar singkat yang tak lebih dari 1 baris saja.

### 3. # komentar

Sama seperti '//' , tulisan di baris yang sama setelah '#' akan dianggap sebagai komentar. Cara ini berguna untuk pemberian komentar singkat yang tak lebih dari 1 baris saja

## Contoh penggunaan komentar pada sintaks PHP



```
<?php
# Tuliskan Nama Anda
echo "<p>Nama saya Agus</p>";
// Tuliskan Tempat Tinggal Anda
echo "<p>Saya tinggal di Tamalanrea, Makassar</p>";
echo "<br>";
/* Silahkan ikuti format penulisan di bawah
8   Nama : _____
9   TTL : _____
10  Alamat: _____
11  No. HP : _____
12 */
echo "<p>Nama : Agus</p>";
echo "<p>TTL: 16 September 2000 </p>";
echo "<p>Alamat: Jalan Sahabat 5 No.1, Tamalanrea, Makassar</p>";
echo "<p>No. HP : 081234567890</p>";
?>
18
19
```

## VARIABEL DAN TIPE DATA PHP

Suatu variable digunakan untuk menyimpan suatu nilai, dapat berupa teks, angka, atau array. Variabel dalam PHP menggunakan simbol ‘\$’ di awal namanya. Sintaks perintah membuat variabel:

`$nama_var = nilai`

### Aturan Pemberian Nama Variabel

1. Harus dimulai dengan huruf atau garis bawah (underscore) ‘\_’ setelah karakter ‘\$’.
2. Hanya dapat menggunakan karakter alphanumeric dan underscore (A-Z / a-z / 0 - 9, dan \_ ).
3. Sebaiknya tidak menggunakan spasi, jika nama variable terdiri lebih dari satu kata, pisahkan dengan underscore (\$nama\_depan, \$nilai\_tugas) atau kapitalisasi (\$namaDepan, \$nilaiTugas)

Variabel pada PHP bersifat “case sensitif”, yang berarti Anda harus memperhatikan penulisan huruf besar dan huruf kecil. Variabel **\$jumlah** berbeda dengan **\$Jumlah**.

Contoh penulisan yang benar:

`$variabel1`

`$variabel_2`

Contoh penulisan yang salah:

`$02variabel`

## \$variabel/03

Tipe data variabel tidak perlu didekalarasikan, PHP akan otomatis mengkonversi atau menentukan tipe data variabel berdasarkan nilai yang disimpannya.

Ada beberapa jenis dari tipe data yang dapat dioperasikan PHP, antara lain:

1. Integer, yang nilainya berupa bilangan bulat, seperti 1, 0, -1, 0123 (octal) dan 0xfb (hexadesimal).
2. Floating Point, yang nilainya berupa bilangan pecahan seperti, 0.123 dan 23e2.
3. String, yang nilainya berupa teks atau kombinasi karakter. Pada tipe data string, terdapat escape character, yaitu karakter khusus yang digunakan mewakili karakter ASCII dengan fungsi khusus. Untuk mendefinisikan karakter yang akan ditampilkan jika karakter tersebut merupakan escape character yaitu dengan mengawali karakter tersebut dengan tanda \ (backslash).

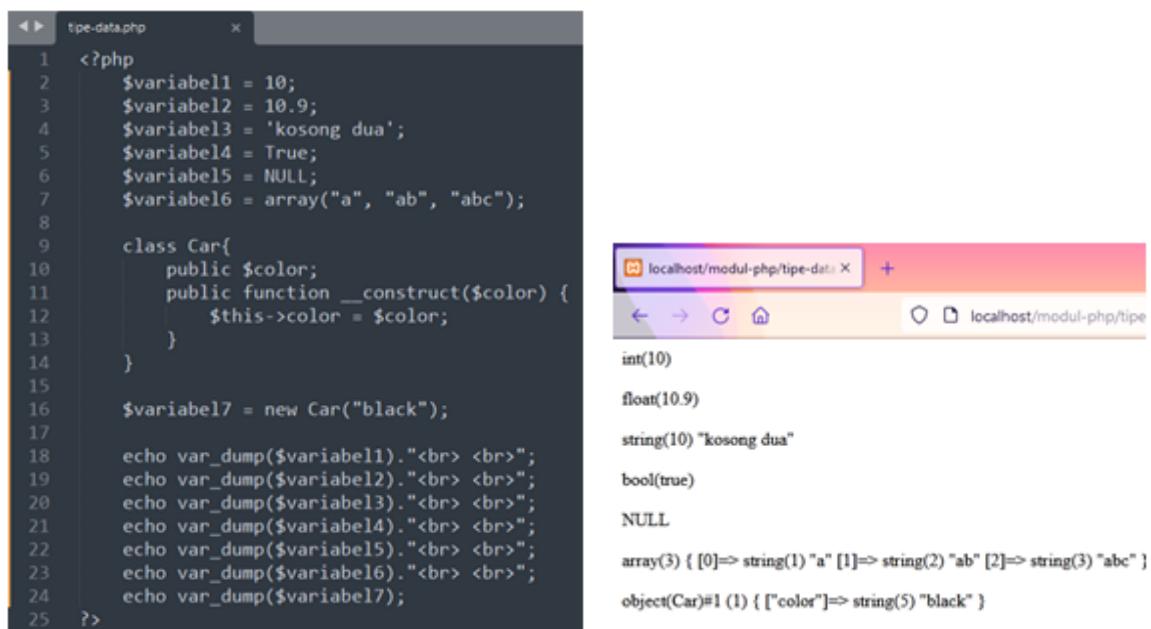
### Daftar escape character

\n	Baris baru (LF or 0x0A (10) in ASCII)
\r	Carriage return (CR or 0x0D (13) in ASCII)
\t	Horizontal tab (HT or 0x09 (9) in ASCII)
\\"	Backslash
\\$	Dollar sign
\\"	Double-quote
\[0 – 7] {1 – 3}	Karakter dalam notasi octal yaitu urutan karakter yang cocok dengan ekspresi regular.
\x[0-9A-Fa-f] {1, 2}	Karakter dalam notasi hexadecimal yaitu urutan karakter yang cocok dengan ekspresi regular.

4. Boolean, tipe data yang nilainya menyatakan kebenaran berupa True atau False.

5. Null, tipe data khusus yang hanya berisi 1 nilai yaitu NULL. Jika Anda membuat variabel tanpa mengisi nilai, maka secara otomatis variabel itu berisi nilai NULL.
6. Array, kumpulan nilai dari satu atau lebih tipe data.
7. Object, tipe data ini merupakan pengembangan PHP berorientasi objek. Tipe data objek merupakan tipe data yang didalamnya mempunyai data dan method. Data tersebut didefinisikan sebagai class terlebih dahulu.

Contoh-contoh tipe data.



```

1 <?php
2     $variabel1 = 10;
3     $variabel2 = 10.9;
4     $variabel3 = 'kosong dua';
5     $variabel4 = True;
6     $variabel5 = NULL;
7     $variabel6 = array("a", "ab", "abc");
8
9     class Car{
10         public $color;
11         public function __construct($color) {
12             $this->color = $color;
13         }
14     }
15
16     $variabel7 = new Car("black");
17
18     echo var_dump($variabel1)."<br> <br>";
19     echo var_dump($variabel2)."<br> <br>";
20     echo var_dump($variabel3)."<br> <br>";
21     echo var_dump($variabel4)."<br> <br>";
22     echo var_dump($variabel5)."<br> <br>";
23     echo var_dump($variabel6)."<br> <br>";
24     echo var_dump($variabel7);
25 ?>

```

localhost/modul-php/tipe-data X +

int(10)

float(10.9)

string(10) "kosong dua"

bool(true)

NULL

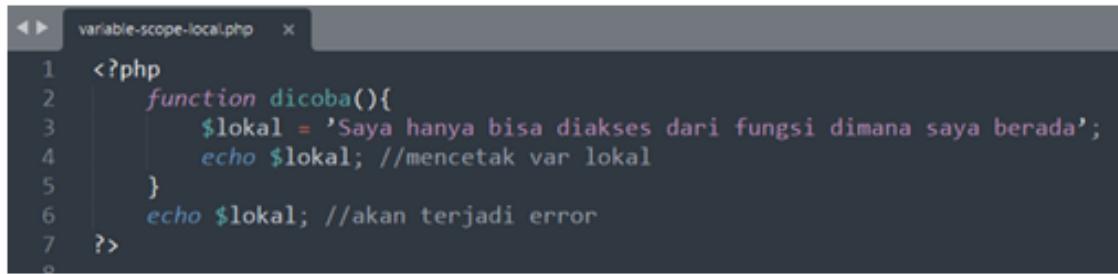
array(3) { [0]=> string(1) "a" [1]=> string(2) "ab" [2]=> string(3) "abc" }

object(Car)#1 (1) { ["color"]=> string(5) "black" }

Scope atau ruang lingkup variabel adalah bagian dari skrip yang dapat mereferensikan variabel tersebut. Ada 4 scope variabel dalam PHP, antara lain:

## 1. Scope Local

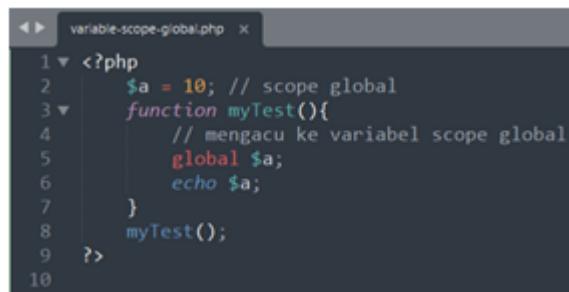
Suatu variabel yang dibuat pada suatu fungsi akan menjadi variabel lokal (memiliki scope local) dan hanya bisa diakses di dalam fungsi. Nama variabel yang sama dapat dibuat dalam fungsi yang berbeda, sebab variabel local hanya dikenali oleh fungsi yang membentuk variabel tersebut. Variabel local akan dihapus setelah fungsi usai dieksekusi. Contoh:



```
variable-scope-local.php
1 <?php
2     function dicoba(){
3         $lokal = 'Saya hanya bisa diakses dari fungsi dimana saya berada';
4         echo $lokal; //mencetak var lokal
5     }
6     echo $lokal; //akan terjadi error
7 ?>
```

## 2. Scope Global

Scope global dimiliki oleh variabel yang dibuat diluar fungsi. Variabel dengan scope global dapat diakses dari bagian manapun dari program selama perintah tersebut ditulis diluar suatu fungsi. Variabel global dapat diakses dari dalam suatu fungsi dengan menggunakan kata kunci ‘global’. Contoh:



```
variable-scope-global.php
1 <?php
2     $a = 10; // scope global
3     function myTest(){
4         // mengacu ke variabel scope global
5         global $a;
6         echo $a;
7     }
8     myTest();
9 ?>
10
```

PHP menyimpan semua variabel global dalam array bernama `$GLOBAL[]`. Indeks dari array adalah nama dari variabel-variabel tersebut. Array ini dapat diakses dari dalam fungsi, tetapi juga dapat digunakan untuk mengupdate variabel global secara langsung. Contoh:



```
variable-scope-global-b.php
1 <?php
2     global $a;
3     $a="abc";
4     // adalah sama dengan menggunakan perintah:
5     // $GLOBALS["a"]="abc";
6
7     function myTest(){
8         echo $GLOBALS["a"];
9     }
10    myTest();
11
12 ?>
```

## 3. Scope Static

Ketika suatu fungsi selesai digunakan, secara normal semua variabelnya akan dihapus. Jika diinginkan variabel-variabel tersebut tidak dihapus ketika fungsi selesai dipakai, gunakan kata kunci ‘static’ saat membuat variabel. Contoh pembuatan variabel statik :

```

variable-scope-static.php  x
<?php
function myTest(){
    static $x = 0;
    echo $x;
    $x++;
}
myTest();
echo "<br>";
myTest();
echo "<br>";
myTest();
echo "<br>";
?>

```

Variabel \$varStatik sekarang menjadi variabel statik.

## ***OPERATOR***

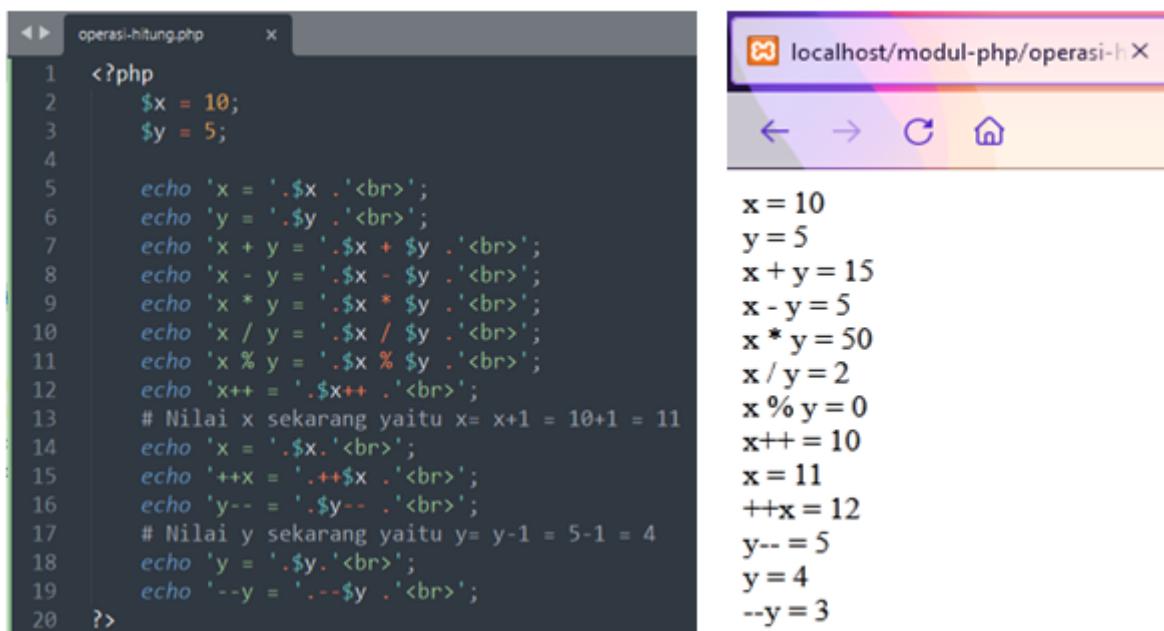
Operator digunakan untuk mengolah nilai. PHP memiliki beberapa kategori operator sebagai berikut:

### **Operator Hitung**

Operator	Penjelasan	Contoh	Hasil
+	Pertambahan	x=2  y=x+2	y=4
-	Pengurangan	x=2  y=5-x	y=3
*	Perkalian	x=4  y=x*5	y=20
/	Pembagian	y=15/5	y=3
%	Sisa hasil bagi	x=10%5 y=10%8  z=5%2	x=0  y=2  z=1

++	Inkremen	x=5 x++	x=6
--	dekremen	x=5 x--	x=4

Contoh operasi hitung :



```

1 <?php
2     $x = 10;
3     $y = 5;
4
5     echo 'x = '.$x . '<br>';
6     echo 'y = '.$y . '<br>';
7     echo 'x + y = '.$x + $y . '<br>';
8     echo 'x - y = '.$x - $y . '<br>';
9     echo 'x * y = '.$x * $y . '<br>';
10    echo 'x / y = '.$x / $y . '<br>';
11    echo 'x % y = '.$x % $y . '<br>';
12    echo 'x++ = '.$x++ . '<br>';
13    # Nilai x sekarang yaitu x= x+1 = 10+1 = 11
14    echo 'x = '.$x . '<br>';
15    echo '++x = '.++$x . '<br>';
16    echo 'y-- = '.$y-- . '<br>';
17    # Nilai y sekarang yaitu y= y-1 = 5-1 = 4
18    echo 'y = '.$y . '<br>';
19    echo '--y = '.--$y . '<br>';
20 ?>

```

localhost/modul-php/operasi-hitung.php

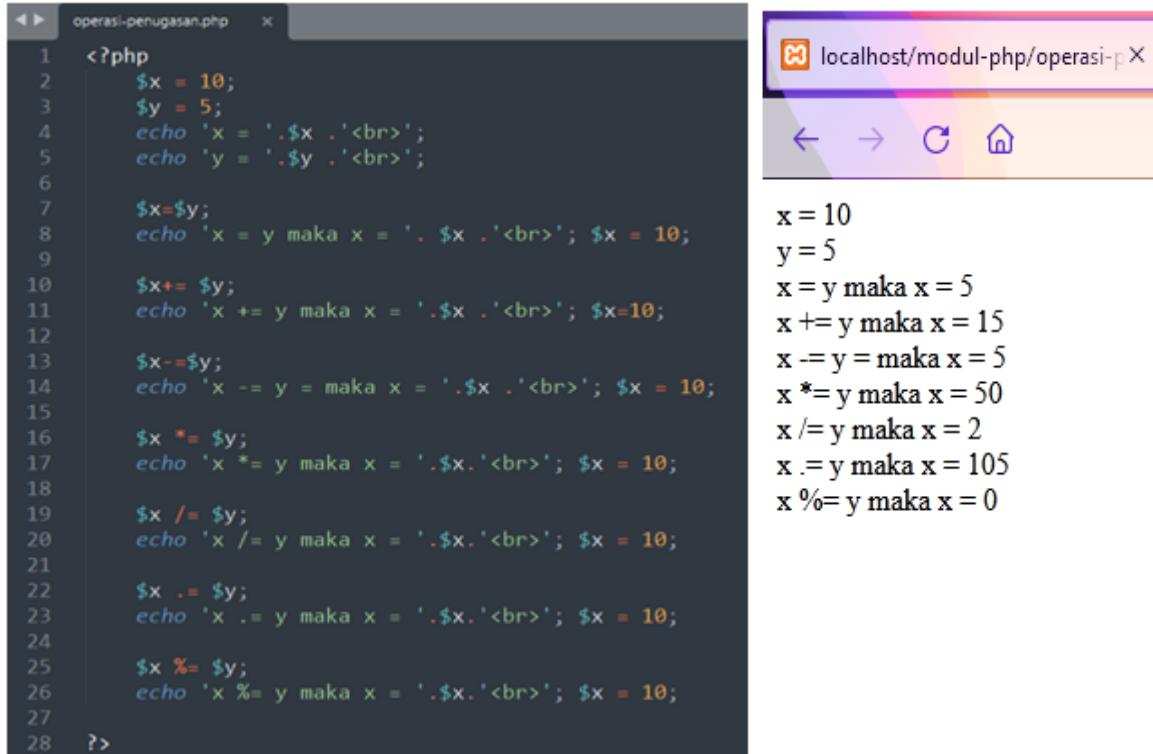
x = 10  
y = 5  
x + y = 15  
x - y = 5  
x \* y = 50  
x / y = 2  
x % y = 0  
x++ = 10  
x = 11  
++x = 12  
y-- = 5  
y = 4  
--y = 3

### Operator Penugasan

Operator	Contoh	Sama dengan
=	x=y	x=y
+=	x+=y	x=x+y
-=	x-=y	x=x-y
*=	x*=y	x=x*y
/=	x/=y	x=x/y

<code>.=</code>	<code>x.=y</code>	<code>x=x.y</code>
<code>%=</code>	<code>x%-=y</code>	<code>x=x%y</code>

Contoh operasi penugasan



The screenshot shows a PHP script named 'operasi-penugasan.php' in a code editor and its output in a browser window.

```

1 <?php
2 $x = 10;
3 $y = 5;
4 echo 'x = ' . $x . '<br>';
5 echo 'y = ' . $y . '<br>';

6 $x=$y;
7 echo 'x = y maka x = ' . $x . '<br>; $x = 10;

8 $x+= $y;
9 echo 'x += y maka x = ' . $x . '<br>; $x=10;

10 $x-= $y;
11 echo 'x -= y = maka x = ' . $x . '<br>; $x = 10;

12 $x *= $y;
13 echo 'x *= y maka x = ' . $x . '<br>; $x = 10;

14 $x /= $y;
15 echo 'x /= y maka x = ' . $x . '<br>; $x = 10;

16 $x .= $y;
17 echo 'x .= y maka x = ' . $x . '<br>; $x = 10;

18 $x %= $y;
19 echo 'x %= y maka x = ' . $x . '<br>; $x = 10;
20 ?>

```

The browser output shows the results of each assignment operation:

```

x=10
y=5
x=y maka x=5
x+=y maka x=15
x-=y = maka x=5
x*=y maka x=50
x/=y maka x=2
x.=y maka x=105
x%-=y maka x=0

```

## Operator Perbandingan

Operator	Penjelasan	Contoh
<code>==</code>	sama dengan	<code>5==8</code> mengembalikan nilai <i>false</i>
<code>===</code>	Identik dengan	<code>5===8</code> mengembalikan nilai <i>false</i>
<code>!=</code>	tidak sama dengan	<code>5!=8</code> mengembalikan nilai <i>true</i>
<code>!==</code>	tidak identik dengan	<code>5 !== 8</code> mengembalikan nilai <i>true</i>
<code>&lt;&gt;</code>	tidak sama dengan	<code>5&lt;&gt;8</code> mengembalikan nilai <i>true</i>

$\Leftrightarrow$	Spaceship	$5 \Leftrightarrow 8$ mengembalikan nilai
$>$	lebih besar dari	$5 > 8$ mengembalikan nilai <i>false</i>
$<$	lebih kecil dari	$5 < 8$ mengembalikan nilai <i>true</i>
$\geq$	lebih besar dari atau sama dengan	$5 \geq 8$ mengembalikan nilai <i>false</i>
$\leq$	lebih kecil dari atau sama dengan	$5 \leq 8$ mengembalikan nilai <i>true</i>

Contoh operasi perbandingan.



```

1 <?php
2     $x = 10; $y = 5;
3     echo 'x = '.$x . '<br>'; echo 'y = '.$y . '<br>';
4
5     echo 'Apakah x sama dengan y ? '; var_dump($x==$y); echo '<br>';
6
7     echo 'Apakah x identik dengan y ? '; var_dump($x===$y); echo '<br>';
8
9     echo 'Apakah x tidak sama dengan y ? '; var_dump($x!=$y); echo '<br>';
10
11    echo 'Apakah x tidak sama dengan y ? '; var_dump($x<>$y); echo '<br>';
12
13    echo 'Apakah x tidak identik dengan y ? '; var_dump($x!===$y); echo '<br>';
14
15    echo ' Apakah x lebih besar, sama dengan atau lebih kecil dari y ? ';
16    echo ($x<=$y); echo '<br>';
17
18    echo 'Apakah x lebih besar dari y ? '; var_dump($x>$y); echo '<br>';
19
20    echo 'Apakah x lebih kecil dari y ? '; var_dump($x<$y); echo '<br>';
21
22    echo 'Apakah x lebih besar dari atau sama dengan y ? ';
23    var_dump($x>=$y); echo '<br>';
24
25    echo 'Apakah x lebih kecil dari atau sama dengan y ? ';
26    var_dump($x<=$y); echo '<br>';
27
28 ?>

```

The screenshot shows a browser window with the URL `localhost/modul-php/operasi-perbandingan.php`. The page displays the output of a PHP script comparing variables `x` and `y`. The output includes various boolean expressions and their results:

```

x = 10
y = 5
Apakah x sama dengan y ? bool(false)
Apakah x identik dengan y ? bool(false)
Apakah x tidak sama dengan y ? bool(true)
Apakah x tidak sama dengan y ? bool(true)
Apakah x tidak identik dengan y ? bool(true)
Apakah x lebih besar, sama dengan atau lebih kecil dari y ? 1
Apakah x lebih besar dari y ? bool(true)
Apakah x lebih kecil dari y ? bool(false)
Apakah x lebih besar dari atau sama dengan y ? bool(true)
Apakah x lebih kecil dari atau sama dengan y ? bool(false)

```

## Operator Logika

Operator	Penjelasan	Contoh
<code>&amp;&amp;</code>	and	<code>x=6; y=3; (x &lt; 10 &amp;&amp; y &gt; 1)</code> mengembalikan <i>true</i>
<code>  </code>	or	<code>x=6; y=3; (x==5    y==5)</code> mengembalikan <i>false</i>
<code>!</code>	not	<code>x=6; y=3; !(x==y)</code> mengembalikan <i>true</i>

Contoh operasi logika

The screenshot shows a code editor with a file named `operasi-logika.php`. The code demonstrates various logical operators (`&&`, `||`, `!`) and their usage with `var_dump` to show the resulting boolean values.

```

1 <?php
2 $x = 10; $y = 5;
3 echo 'x = '.$x .'  
'; echo 'y = '.$y .'  
';
4
5 echo 'x > y and x == y ?'; var_dump($x==$y and $x>$y); echo '  
';
6
7 echo 'x > y && y < x ?'; var_dump($x > $y && $y < $x); echo '  
';
8
9 echo 'x > y or x == y ?'; var_dump($x > $y or $x == $y); echo '  
';
10
11 echo 'x == y || x < y ?'; var_dump($x == $y || $x < $y); echo '  
';
12
13 echo '! (x == y) ?'; var_dump(!(x == y)); echo '  
';
14
15 ?>

```

## STRUKTUR KENDALI

Dalam dunia pemrograman umumnya, terdapat 2 jenis struktur kendali yaitu struktur kendali kondisional dan struktur kendali perulangan. Struktur kendali kondisional (bersyarat) dan struktur kendali perulangan (looping). Struktur kendali kondisional adalah struktur kendali yang digunakan untuk mengatur kapan suatu perintah akan dijalankan. Dengan statement ini kita bisa mengatur kapan suatu perintah akan dijalankan, yaitu ketika telah dipenuhinya suatu syarat tertentu. Sedangkan

struktur kendali perulangan digunakan untuk mengatur perintah yang dijalankan secara berulang-ulang.

Dalam PHP, terdapat dua buah struktur kendali yang termasuk struktur kendali kondisional, yaitu IF dan CASE OF. Sedangkan yang termasuk struktur kendali perulangan adalah: FOR, WHILE, DO WHILE dan FOREACH.

### 1. Struktur kendali kondisional

Ketika Anda menulis kode program, suatu saat Anda perlu menggunakan mekanisme percabangan sehingga berdasarkan kondisi program Anda akan melakukan aksi yang berbeda. Misalnya saja Anda ingin membuat program yang dapat menampilkan salam yang berbeda tergantung jenis hari pada saat itu, maka kode programnya akan seperti ini:



```
if.php
1 <?php
2     $hari=date("D");
3     if ($hari=="Mon"){
4         echo "Selamat berlibur";
5     }else{
6         echo "Selamat bekerja dan berkarya";
7     }
8 ?>
```

Jika program diatas dijalankan pada hari Minggu, maka Anda akan memperoleh ucapan ‘Selamat berlibur’, tapi jika dijalankan selain hari Minggu akan memperoleh ucapan ‘Selamat bekerja dan berkarya’.

Terkadang untuk menyatakan suatu syarat, kita menggunakan operator pembanding atau relasional. Hasil penggunaan operator relasional ini akan diperoleh nilai BENAR atau SALAH. Operator relasional dapat dilihat di sub bab Operator.

Struktur if mempunyai beberapa format, berikut penjelasannya.

#### a. IF

Adapun sintaks atau aturan penulisan IF adalah sebagai berikut:

```
if (syarat){
    statement;
}
```

Dalam sintaks di atas, bagian ‘statement’ akan dijalankan atau dilakukan jika ‘syarat’ terpenuhi atau ‘syarat’ bernilai benar (true). Apabila ‘syarat’ tidak terpenuhi, bagian ‘statement’ tidak akan dijalankan atau dalam hal ini tidak melakukan apa-apa. Dengan demikian, nilai ‘syarat’ haruslah hanya ada dua kemungkinan, yaitu BENAR atau SALAH itu saja

Contohnya :



The image shows a comparison between a code editor and a web browser. On the left, a code editor window titled 'if-1.php' displays the following PHP code:

```
1 <?php
2     $hujan = True;
3     if ($hujan){
4         echo "Tutup pintu dan jendela";
5     }
6 ?>
7
```

On the right, a browser window titled 'localhost/modul-php/if-1.php' shows the output: 'Tutup pintu dan jendela'. This demonstrates that the condition '\$hujan' is true, so the echo statement is executed.

#### b. IF ... ELSE

Adapun sintaks atau aturan penulisan IF ... ELSE adalah sebagai berikut.

```
if (syarat){
    statement1;
} else{
    statement2;
}
```

Dalam sintaks di atas, bagian ‘statement1’ akan dijalankan atau dilakukan jika ‘syarat’ terpenuhi atau ‘syarat’ bernilai benar (true). Apabila ‘syarat’ tidak terpenuhi, maka bagian ‘statement2’ akan dijalankan.

Contoh:



A screenshot of a code editor window titled "if-else.php". The code is written in PHP and demonstrates an if-else statement. It uses the date() function to get the current minute and second, then checks if the minute is divisible by 2. If true, it outputs "Berhenti!", otherwise it outputs "Jangan berhenti!". The code is numbered from 1 to 15.

```
<?php
/* Misalnya suatu lampu lalu lintas akan
berganti warna setiap 1 menit*/
$menit= date("i");
$detik= date("s");
echo $menit.":".$detik;
echo "<br>";
if ( $menit %2 == 1){
    echo "Berhenti!";
} else{
    echo "Jangan berhenti!";
?>
```

### c. IF ... ELSEIF

```
if(syarat1){
    statement1;
} elseif (syarat2){
    statement2;
}
.....
else {
    statement3;
}
```

Dalam sintaks di atas, bagian ‘statement1’ akan dijalankan atau dilakukan jika ‘syarat1’ terpenuhi atau ‘syarat1’ bernilai benar (true). Apabila ‘syarat’ tidak terpenuhi, maka bagian ‘statement2’ akan dijalankan. Apabila ‘syarat2’ tidak terpenuhi maka bagian ‘statement3’ akan dijalankan. Bila masih terdapat beberapa syarat yang diperlukan, bagian elseif dapat ditulis berulangkali.

Contoh:

The screenshot shows a code editor window titled 'if-elseif.php' containing the following PHP code:

```

1 <?php
2     /*jarak tempuh suatu kendaraaan
3      dalam satuan km */
4     $kondisi_mesin = NULL;
5     $jarak_tempuh = 10;
6     if ($jarak_tempuh <= 500){
7         $kondisi_mesin = "Sangat baik";
8     } elseif ($jarak_tempuh < 5000){
9         $kondisi_mesin = "Cukup Baik";
10    } else{
11        $kondisi_mesin = "Kurang Baik";
12    }
13    echo "Kondisi mesin : ".$kondisi_mesin;
14 ?>

```

To the right, a browser window displays the output: "Kondisi mesin : Sangat baik".

#### d. Switch

Jika dalam program ada banyak pilihan, maka lebih baik menggunakan switch. Switch digunakan untuk memilih salah satu dari banyak blok kode yang akan dieksekusi. Adapun sintaks atau aturan penulisan Switch adalah sebagai berikut:

```

switch(variabel) {
    case label_1:
        statement1;
        break;
    case label_2:
        statement_2;
        break;
    .....
    default:
        statement_last;
}

```

Keterangan:

n = variabel penentu.

label = nilai variabel penentu.

Dalam sintaks di atas, bagian ‘statement1’ akan dijalankan atau dilakukan jika ‘label\_1 terpenuhi atau ‘label\_1 bernilai benar (true). Apabila ‘label\_1 tidak terpenuhi, maka bagian ‘statement\_2’ akan dijalankan. Apabila ‘label\_2’ tidak terpenuhi maka bagian default akan dijalankan. Bila masih terdapat beberapa syarat yang diperlukan, bagian case dapat ditulis berulangkali. Contoh:

The image shows a split-screen view. On the left is a code editor window titled "switch-case.php" containing the following PHP code:

```

1 <?php
2     $hariKerja; $agenda; $seragam;
3     $hari = "Senin";
4     switch ($hari) {
5         case 'Senin':
6             case 'Selasa':
7                 $hariKerja = True;
8                 $agenda = "Mengajar";
9                 $seragam= "putih";
10                break;
11            case 'Rabu':
12            case 'Kamis':
13                $hariKerja = True;
14                $agenda = "Mengajar";
15                $seragam= "batik";
16                break;
17            case 'Jumat':
18                $hariKerja = True;
19                $agenda = "Mengajar";
20                $seragam= "olahraga";
21                break;
22        default:
23            $hariKerja = False;
24            $agenda = "Liburan";
25            $seragam= "Bebas";
26            break;
27    }
28    echo "Agenda Guru <br>";
29    echo "Agenda: ".$agenda."<br>";
30    echo "Seragam: ".$seragam."<br>";
31 ?>

```

On the right is a browser window titled "localhost/modul-php/switch-case.php" displaying the output of the script:

Agenda Guru  
Agenda: Mengajar  
Seragam: putih

## 2. Struktur kendali perulangan

Sedangkan struktur kendali perulangan digunakan untuk mengatur perintah yang dijalankan secara berulang-ulang. Dalam PHP, terdapat empat macam struktur kendali perulangan yaitu FOR, WHILE, DO WHILE dan FOREACH.

### a. FOR

Struktur perulangan *for* sering digunakan dalam pemrograman. Sintaksnya:

```
for (inisialisasi; kondisi; inkremen){
```

kode yang akan dieksekusi;

```
}
```

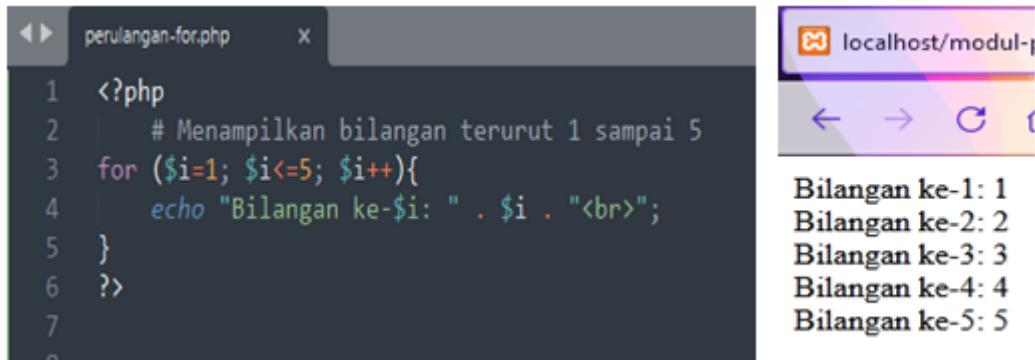
Tiga parameternya yaitu:

Inisialisasi : Nilai awal counter.

Kondisi : Parameter yang akan dievaluasi pada setiap iterasi perulangan. Jika hasil evaluasi true, perulangan akan dilanjutkan. Bila false perulangan akan dihentikan.

Inkremen : Nilai pertambahan konter setiap satu iterasi perulangan diselesaikan. Setiap parameter di atas bersifat opsional.

Contoh:



The screenshot shows a code editor window titled "perulangan-for.php" containing the following PHP code:

```
1 <?php
2     # Menampilkan bilangan terurut 1 sampai 5
3     for ($i=1; $i<=5; $i++){
4         echo "Bilangan ke-$i: " . $i . "<br>";
5     }
6 ?>
```

To the right of the code editor is a browser window titled "localhost/modul-p" showing the output of the code:

Bilangan ke-1: 1  
Bilangan ke-2: 2  
Bilangan ke-3: 3  
Bilangan ke-4: 4  
Bilangan ke-5: 5

b. WHILE

Perulangan while akan menjalankan suatu blok kode (sekelompok kode) selama kondisinya bernilai true. Sintaks perulangan ini:

```
while (kondisi) {
    blok kode yang akan dieksekusi;
}
```

Contohnya:

```
while (kondisi) {
    blok kode yang akan dieksekusi;
}
```



The screenshot shows a code editor window titled "perulangan-while.php" containing the following PHP code:

```
1 <?php
2     $i=1;
3     while($i<=5){
4         echo "Bilangan ke-$i: " . $i . "<br />";
5         $i++;
6     }
7 ?>
```

To the right of the code editor is a browser window titled "localhost/modul-p" showing the output of the code:

Bilangan ke-1: 1  
Bilangan ke-2: 2  
Bilangan ke-3: 3  
Bilangan ke-4: 4  
Bilangan ke-5: 5

c. DO WHILE

Satu yang unik dari perulangan ini adalah, dia pasti akan melakukan iterasi, minimal satu kali, meskipun nilai kondisinya tidak pernah true. Hal ini disebabkan struktur perulangan yang menyebabkan blok kode akan dijalankan lebih dulu, lalu kondisinya dievaluasi belakangan. Sintaks perulangan ini:

```

do {
    kode yang akan dieksekusi;
} while (kondisi);

```

Contoh:

```

<?php
$i=1;
do{
    $i++;
    echo "Bilangan ke-$i: " . $i . "<br />";
}
while ($i<=5);
?>

```

Bilangan ke-2: 2  
Bilangan ke-3: 3  
Bilangan ke-4: 4  
Bilangan ke-5: 5  
Bilangan ke-6: 6

#### d. FOREACH

Perulangan ini disediakan PHP untuk memudahkan kita mengakses elemen-elemen suatu array. Sintaks:

```

foreach ($array as $value) {
    kode yang akan dieksekusi;
}

```

Pada setiap iterasi perulangan, nilai dari elemen array yang sedang ditunjuk oleh pointer akan diletakkan ke variabel `$value` kemudian pointer akan bergerak menunjuk elemen array berikutnya sehingga pada iterasi selanjutnya, elemen array berikutnya yang akan diolah.

Contoh:

```

<?php
$x=array("satu","dua","tiga");
foreach ($x as $value){
    echo $value . "<br />";
}
?>

```

satu  
dua  
tiga

## **FUNCTION (FUNGSI)**

Fungsi merupakan salah satu teknik pemrograman modular. Sebuah aplikasi besar disusun dari modul-modul yang berupa sebuah fungsi atau prosedur. Fungsi berisi sekelompok kode dengan tugas dan tujuan spesifik. Fungsi tidak akan dieksekusi ketika program dijalankan. Fungsi hanya akan dieksekusi jika dilakukan pemanggilan terhadapnya. Pemanggilan dapat dilakukan dari mana saja dalam program. Keuntungan teknik ini, modul-modul yang dibuat dapat digunakan berkali-kali (reuse).

## **MEMBUAT FUNGSI**

Sebuah fungsi dibuat dengan aturan sintaks:

```
function namaFungsi() {  
    kode-kode yang akan dieksekusi;  
}
```

Beberapa petunjuk dalam membuat sebuah fungsi:

- Namai fungsi yang menggambarkan fungsinya
- Nama fungsi dimulai dengan huruf atau garis bawah (*underscore*), tidak boleh angka.

### **Pemanggilan Fungsi**

Ketika fungsi sudah dibuat, dia dapat dijalankan dengan cara dipanggil. Pemanggilan suatu fungsi mengikuti pola: nama fungsi lalu diikuti tanda kurung dan nilai parameter jika ada.

Contoh:

```
tambah(10,20);
```

Memanggil sebuah fungsi bernama *tambah* dengan nilai parameter 10 dan 20. Jika tidak ada nilai parameter, maka pemanggilan fungsi seperti di bawah:

```
cetak();
```

### **Parameter Fungsi**

Untuk menambah daya guna fungsi dapat ditambahkan parameter fungsi yang tidak lain adalah serupa variabel. Parameter ini dituliskan sesudah nama fungsi didalam tanda kurung. Dengan parameter ini, hasil dari fungsi dapat diatur sesuai dengan keinginan.

## Nilai Balik Fungsi

Fungsi dapat diatur agar mengembalikan hasil berupa nilai dengan cara menggunakan kata kunci *return*. Nilai yang dikembalikan pada fungsi diatas adalah jumlah dari nilai variabel \$x dan \$y yang ada didalam variabel \$total. Hasil di layar browser adalah tampilan  $5 + 20 = 25$ .

Contoh:



The image shows a comparison between a code editor and a web browser. On the left, a code editor window titled 'fungsi.php' displays the following PHP code:

```
<?php
    #fungsi yang mengembalikan nilai
    function pangkat($a, $b){
        $c = $a**$b;
        return $c;
    }
    $x = 3; $y = 4;
    echo $x." pangkat ".$y." = ".pangkat(3, 4)."

On the right, a browser window titled 'localhost/modul-php/fu' shows the output of the code execution:



3 pangkat 4 = 81  
Selamat Pagi!


```

## ARRAY

Array adalah jenis tipe data khusus yang menyimpan sejumlah nilai data dalam sebuah variabel. Nilai-nilai data tersebut, yang disebut elemen array, memiliki indeks yang menunjukkan urutannya dalam array. Elemen array pertama akan memiliki indeks 0, elemen kedua berindeks 1, dan seterusnya.

Dengan array, proses pencarian data tertentu akan mudah dilakukan. Misalkan Anda memiliki data nama barang elektronik yang disimpan dalam variabel tunggal seperti ini:

```
$brg1="DVD";
```

```
$brg2="Televisi";
```

```
$brg3="Lemari es";
```

## Jenis Array

Dalam PHP terdapat 3 jenis array, yaitu array numerik (*indexed arrays*), array assosiatif (*associative array*), dan array multidimensi (*multidimensional array*).

- a. Array numerik, array numerik yaitu jenis array yang berindeks numerik. Array ini yang paling banyak digunakan karena terindeks numerik secara otomatis.
- b. Array assosiatif, yaitu jenis array yang memiliki indeks berupa string. Array ini digunakan ketika data-data yang digunakan memiliki kunci yang unik di setiap nilainya. Tampak dalam menentukan elemen-elemen array diatas menggunakan simbol “=>” yang dibuat dengan simbol sama dengan (“=” ) dikombinasikan dengan simbol lebih besar (“>”). Untuk mencetak seluruh nilai elemen array jenis ini dapat menggunakan perulangan *foreach*.

**Array multidimensi, yaitu jenis array yang indeksnya juga array. Artinya, elemen dari array jenis ini berupa suatu array juga.**

### Membuat Array

Dalam membuat array dalam PHP fungsi yang digunakan yaitu `array()`. Contoh:

```
#Membuat array numerik
$person = array("Joni", "Indra", "Susi");

#Membuat array assosiatif
$umur = array ("Joni"=>"17", "Indra"=>"18", "Susi"=>"16");

#Membuat array multi-dimensi

$mhs = array (
    array ("A12.2010.04567", "Anita Larasati", 3.5),
    array ("A12.2010.05678", "Dude Harmono", 3),
    array ("A12.2010.06789", "Ernawati Listyani", 2.75),
);
```

### Mengakses elemen Array

Untuk mengakses elemen array tersebut dengan menggunakan indeksnya. Dikarenakan ketiga array di atas memiliki indeks yang berbeda-beda, maka untuk mengakses elemennya juga menggunakan cara yang berbeda.

Contohnya:

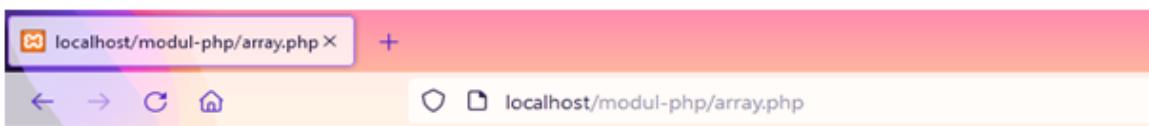
```

#Mengakses elemen array numerik
echo "<h2>Mengakses elemen array numerik</h2>";
echo "<p>Nilai dari indeks 0 adalah ".$person[0]."</p>";
echo "<p>Nilai dari indeks 2 adalah ".$person[2]."</p>";

#Mengakses elemen array asosiatif
echo "<h2>Mengakses elemen array asosiatif</h2>";
echo "<p>Umur Joni adalah ".$umur["Joni"]."</p>";
echo "<p>Umur Indra adalah ".$umur["Indra"]."</p>";

#Mengakses elemen array multi dimensional
echo "<h2>Mengakses elemen array multi dimensional</h2>";
echo "<p>Mahasiswa dengan nim ".$mhs[0][0]." memiliki IPK senilai ".$mhs[0][2]."</p>";
echo "<p>Mahasiswa dengan nim ".$mhs[1][0]." memiliki IPK senilai ".$mhs[1][2]."</p>";

```



## Mengakses elemen array numerik

Nilai dari indeks 0 adalah Joni

Nilai dari indeks 2 adalah Susi

## Mengakses elemen array asosiatif

Umur Joni adalah 17

Umur Indra adalah 18

## Mengakses elemen array multi dimensional

Mahasiswa dengan nim A12.2010.04567 memiliki IPK senilai 3.5

Mahasiswa dengan nim A12.2010.05678 memiliki IPK senilai 3

## Mencari Panjang Suatu Array

Seperti diketahui array terdiri dari sejumlah elemen array. Untuk mengetahui jumlah elemen dalam suatu array atau panjang suatu array dapat menggunakan fungsi count().

Contohnya:

```

#Panjang elemen array numerik
$n_person = count($person); // $n_person = 3
#Panjang elemen array asosiatif
$n_umur = count($umur); // $n_umur = 3
#Panjang elemen array multi dimensional
$n_mhs = count($mhs); // $n_mhs = 3

```

## Mencetak Seluruh Elemen Array

Untuk mencetak seluruh elemen array dapat digunakan suatu perulangan for atau for each sebagai berikut:

```

#Menampilkan seluruh elemen array numerik satu per satu
echo "<h2>Nilai array person beserta dengan indeksnya </h2>";
for ($i = 0; $i <= count($person)-1; $i++) {
    echo "<p>Indeks " . $i . " bernilai " . $person[$i] . "</p>";
}
#Menampilkan seluruh elemen array assosiatif satu per satu
echo "<h2>Nilai array umur beserta dengan indeksnya </h2>";
foreach ($umur as $x=>$nilaiX) {
    echo "<p>Indeks " . $x . " bernilai " . $nilaiX . "</p>";
}
#Menampilkan seluruh elemen array multi dimensional satu per satu
echo "<h2>Nilai array mhs beserta dengan indeksnya </h2>";
foreach($mhs as $x){
    echo "<p>Nama : ".$x[1]."; Nim : ".$x[0]."; IPK : ".$x[2]."</p>";
}

```

## Array Assosiatif

Penulisan indeks untuk jenis array ini bagi sedikit berbeda bila dibandingkan dengan array berindeks. Kode pembuatan array jenis tampak seperti berikut ini:

## Mengurutkan Array

Berikut beberapa fungsi yang dapat digunakan untuk mengurutkan array.

Nama Fungsi	Kegunaan Fungsi
array_push()	Fungsi ini akan menambahkan satu atau lebih elemen pada elemen terakhir array.
array_pop()	Fungsi ini akan menghapus elemen terakhir dari array.
array_search()	Fungsi ini akan mencari suatu element array dan mengembalikan sebuah key.
array_merge()	Fungsi ini akan menggabungkan satu atau lebih array menjadi satu array.
array_slice()	Fungsi ini akan mengembalikan bagian yang dipilih dari sebuah array.

sort()	Fungsi ini akan mengurutkan elemen secara menaik atau <i>ascending</i> . Jika elemen-elemen array berupa string, maka akan diurutkan menurut urutan alfabet (a, b, c, dan seterusnya). Dan jika berupa nilai numerik, maka elemen array akan diurutkan secara numeris (1, 2, 3, dan seterusnya).
rsort()	Fungsi ini akan mengurutkan elemen array secara menurun atau <i>descending</i> .
asort()	Fungsi ini digunakan untuk jenis array assosiatif yang akan mengurutkan array secara menaik berdasarkan nilai data elemen array.
ksort()	Fungsi akan mengurutkan secara menaik array assosiatif berdasarkan nilai indeks elemen array.
arsort()	Fungsi ini merupakan gabungan sifat dari fungsi asort() dan rsort() yang akan menghasilkan pengurutan array assosiatif secara menurun ( <i>descending</i> ).
krsort()	Fungsi ini gabungan sifat dari fungsi ksort() dan rsort() yang akan menghasilkan pengurutan array assosiatif secara menurun ( <i>descending</i> ) berdasarkan nilai indeks elemen array.

## VARIABEL SUPERGLOBALS GET DAN POST

Variabel superglobal yaitu variabel-variabel yang sudah dimiliki oleh PHP yang bisa diakses dimanapun dan kapanpun di program php. Contoh superglobal seperti `$_GET`, `$_POST`, `$_REQUEST`, `$_SESSION`, `$_COOKIE`, `$_SERVER`, `$_ENV` yang semuanya adalah array assosiatif. Namun yang akan dibahas dalam modul ini yaitu hanya `$_GET` dan `$_POST`.

### `$_GET`

`$_GET` adalah variabel superglobal PHP yang digunakan untuk mengumpulkan data setelah submit di dalam form HTML dengan atribut method="get".

Misalkan, kita memiliki file **form.html** yang didalamnya terdapat form dengan atribut `action="getAbsensi.php"` dan `method="get"`. Artinya, data pada form akan dikirimkan ke **getAbsensi.php** dengan method get. Sehingga data-data yang dikirim akan diproses oleh **getAbsensi.php**.

### **form.html**

```
index2.html      x
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title>
        Form
    </title>
</head>
<body>
    <h1>Absensi</h1>
    <form action="getAbsensi.php" method="get">
        <label for="nama">Nama</label><br>
        <input type="text" name="nama" id="nama"><br>
        <label for="nim">NIM</label><br>
        <input type="text" name="nim" id="nim"><br>
        <input type="submit">
    </form>
</body>
</html>
```

← → ⌂ ⌂ localhost/modul-php/getdanpost/index2.html

## Absensi

Nama  
  
NIM

## getAbsensi.php

```
getAbsensi.php      x
<?php
    echo "<h3>".$_GET['nama']."' dengan nim '".$_GET['nim']."' telah mengisi absen.</h3>";
?>
```

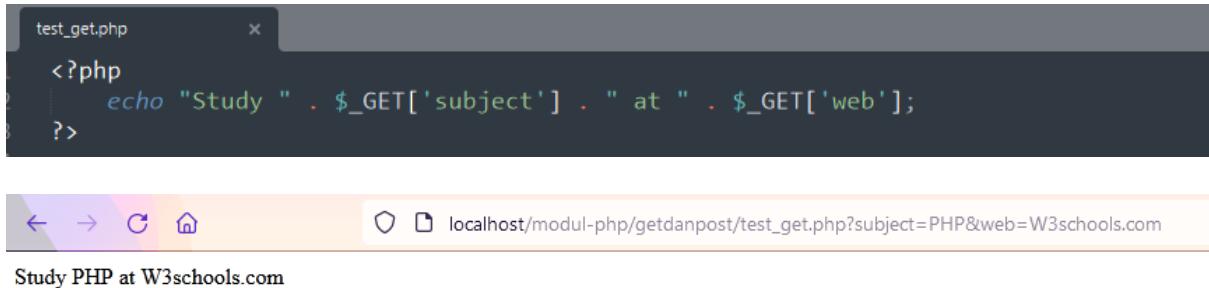
← → ⌂ ⌂ localhost/modul-php/getdanpost/getAbsensi.php?nama=Agus&nim=H071230001

**Agus dengan nim H071230001 telah mengisi absen.**

\$\_GET bisa juga mengumpulkan data yang dikirim dari sebuah URL. Misalkan, kita memiliki file html **index.html** yang didalamnya terdapat hyperlink dengan parameter sebagai berikut.

<a href="test\_get.php?subject=PHP&web=W3schools.com">Test \$GET</a>

Ketika pengguna mengklik link "Test \$GET", parameter "subject" dan "web" dikirimkan ke "test\_get.php", dan nilai dari masing-masing parameter tersebut dapat diakses di test\_get.php menggunakan variabel \$\_GET.



A screenshot of a web browser window. The title bar says "test\_get.php". The code area contains:

```
<?php  
| echo "Study " . $_GET['subject'] . " at " . $_GET['web'];  
?>
```

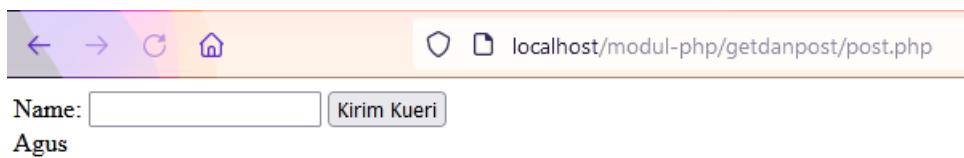
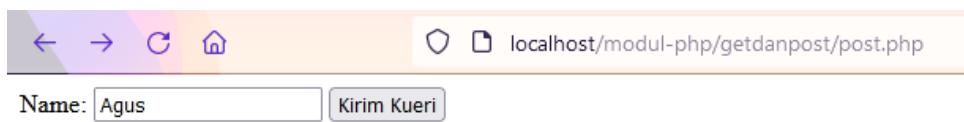
The address bar shows "localhost/modul-php/getdanpost/test\_get.php?subject=PHP&web=W3schools.com". The page content displays "Study PHP at W3schools.com".

## \$\_POST

\$\_POST adalah variabel super global PHP yang digunakan untuk mengumpulkan data form setelah submit pada form HTML dengan method="post". \$\_POST juga banyak digunakan untuk melewatkkan variabel.

Contoh di bawah ini menunjukkan form atau formulir dengan tag input dan tombol Submit. Ketika pengguna mengirimkan data dengan mengklik "Submit", data formulir dikirim ke file yang ditentukan dalam atribut action dari tag <form>. Dalam contoh ini, kami menunjuk ke file itu sendiri untuk memproses data formulir. Kemudian, kita dapat menggunakan variabel super global \$\_POST untuk mengumpulkan nilai dari field input:

```
post.php
x
<!DOCTYPE html>
<html>
<body>
<form method="post" action=<?php echo $_SERVER['PHP_SELF'];?>>
    Name: <input type="text" name="fname">
    <input type="submit">
</form>
<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    // collect value of input field
    $name = $_POST['fname'];
    if (empty($name)) {
        echo "Name is empty";
    } else {
        echo $name;
    }
}
?>
</body>
</html>
```



## OOP PHP

Object Oriented Programming (OOP) atau dalam bahasa Indonesia lebih dikenal sebagai pemrograman berorientasi objek merupakan paradigma pemrograman berdasarkan konsep "objek", yang dapat berisi data dalam bentuk *field* atau atribut; serta kode dalam bentuk fungsi/prosedur (*method*). Semua data dan fungsi di dalam paradigma ini dibungkus dalam *kelas-kelas* atau *objek-objek*. Bandingkan dengan logika pemrograman terstruktur. Setiap objek dapat menerima pesan, memproses data, dan mengirim pesan ke objek lainnya.

Model data berorientasi objek dikatakan dapat memberi fleksibilitas yang lebih, kemudahan mengubah program, dan digunakan luas dalam teknik peranti lunak skala besar. Lebih jauh lagi, pendukung OOP mengklaim bahwa OOP lebih mudah dipelajari bagi pemula dibanding dengan pendekatan sebelumnya, dan pendekatan OOP lebih mudah dikembangkan dan dirawat.

## CLASS

Kelas atau Class merupakan struktur data dari suatu objek. Lebih jelasnya adalah sebuah bentuk dasar yang mendefinisikan **variabel** dan **method** umum pada semua obyek.

### Aturan Pembuatan Class

1. Definisi suatu class dimulai dengan keyword Class, diikuti dengan nama classnya.
2. Aturan penamaan class sama dengan penamaan pada variabel. Umumnya nama class dimulai dengan huruf besar, dan apabila katanya lebih dari dua kata bisa menggunakan teknik camel case (contohnya lemariBuku).
3. Isi suatu class diapit dengan kurung kurawal ( { } ).
4. Umumnya isi class diawali dengan menuliskan properti-propertinya dan diikuti dengan method-methodnya.

### Sintaks class

```
class namaKelas{  
    // properti – properti  
    // method - method  
}
```

Contoh pembuatan class:

```
<?php  
    class Orang{  
        public $nama;  
        function UcapSalam(){  
            echo "Hallo. Nama Saya adalah $this->nama";  
        }  
    }  
?>
```

## OBJECT

Sederhananya, sebuah obyek adalah kumpulan dari variabel dan fungsi yang dibungkus menjadi satu entitas. Entitas tersebut dapat berupa variabel biasa. Sebuah obyek diciptakan melalui sebuah kelas atau dengan istilah *instance of class*. Sebuah class tidak dapat digunakan tanpa diinstansiasikan dulu (kecuali untuk Static Class). Obyek memiliki 2 elemen utama: Obyek sendiri adalah kumpulan variabel dan fungsi yang dihasilkan dari template khusus atau disebut class.

Instansiasi kelas bisa dilakukan dengan perintah new. Ketika suatu instansiasi kelas (pembuatan objek) dilakukan, maka secara otomatis akan memanggil fungsi Constructor milik class tersebut.

Sintaks instansiasi kelas

```
$nama_obyek = new namaKelas();
```

Contoh instansiasi kelas.

```
<?php
    class Orang{
        public $nama;

        function UcapSalam(){
            echo "Hallo. Nama Saya adalah $this->nama";
        }
    }

    $orang_1 = new Orang();
?>
```

## PROPERTY

Properti adalah suatu variabel berisi nilai-nilai yang tersimpan dalam objek tersebut dan secara langsung maupun tidak langsung menentukan karakteristik dari objek tersebut. Misalnya di dalam kelas Orang, propertinya adalah nama, jenisKelamin, alamat, noHP dan lain-lain. Properti memiliki tipe visibility yang ditulis di depan properti itu sendiri. Berikut sintaks penulisan properti:

```
accessModifier $namaProperti;
accessModifier $namaProperti1;
```

Contoh properti dalam OOP PHP:

```
<?php
    class Orang{

        public $nama; $jenisKelamin
        protected $noHP; $noRekening;
        private $pinATM;
    }

?>
```

## METHOD

Method yaitu suatu aksi yang akan dijalankan atau dikerjakan oleh objek tersebut. Method adalah fungsi yang ada di dalam object. Sama halnya dengan function, suatu method juga bisa

mengembalikan nilai dan tidak mengembalikan nilai. Misalnya di dalam class Orang, *method*-nya seperti berjalan, duduk, makan, minum, tidur, dan lain-lain.

Sama seperti properti, method juga memiliki tipe accessModifier yang ditulis di depan keyword ‘function’.

Sintaks

```
accessModifier function namaFunction(){
    //instruksi yang akan dijalankan
}
```

Contoh:

```
class Orang{
    public function UcapSalam(){
        echo "Selamat pagi";
    }
}
```

## Constructor dan Destructor

Constructor adalah suatu function khusus yang akan dieksekusi ketika suatu objek dibuat (instansiasi class). Umumnya constructor dibuat untuk memberikan suatu operasi awal yang harus dilakukan ketika sebuah objek dilahirkan (inisialisasi objek). Sedangkan destructor adalah function khusus yang dilakukan ketika suatu objek akan dihapus. Umumnya destructor dibuat untuk mengembalikan kembali sumber daya komputer (misalnya memori, file). Contoh operasi dalam destructor yaitu menghapus kembali memori yang telah digunakan atau menutup koneksi ke suatu file.

Sintaks:

```
function __construct(){
    // instruksi yang akan dijalankan
}

function _destruct(){
    // instruksi yang akan dijalankan
}
```

Contoh :

```
class Orang{
    private $nama;
    function __construct($nama){
        $this->nama=$nama;
        echo "Contructor: $this->nama dilahirkan<br>";
    }
    function UcapSalam(){
        echo "Hallo. Nama Saya adalah ".$this->nama."<br>";
    }
    function __destruct(){
        echo "Destructor: $this->nama meninggal dunia<br>";
    }
}
```

## ACCESS MODIFIER

Property dan method dapat memiliki pengubah akses yang mengontrol di mana mereka dapat diakses. Dengan adanya access modifier, kita akan leluasa mengatur setiap property dan method agar digunakan secara tepat.

Macam-macam access modifier dalam PHP:

- Public, property dan method dapat diakses dimanapun, walaupun dari luar classnya. Access modifier ini bersifat default.
- Private, property dan method hanya dapat diakses di dalam pendeklarasian classnya saja. Property dan method tidak dapat dikenal di class turunannya.
- Protected, mirip dengan private, tetapi property dan method dapat dikenal di class turunannya.

Contoh:

```
class Orang {
    public $nama;
    public $jenisKelamin;
    protected $noHP;
    protected $noRekening;
    private $pinATM;
}

$orang1 = new Orang();
$orang1->nama = 'Hasanuddin'; // OK
$orang1->noHP = '081111111111'; // ERROR
$orang1->pinATM = '1234'; // ERROR
```

Dari potongan kode di atas, properti yang hanya dapat diakses di luar class Fruit yaitu **name**. Sedangkan pengaksesan properti noHP dan pinATM membuat program error, karena kedua properti itu masing-masing menggunakan properti protected dan private.

## Function Setter dan Getter

Method Setter adalah fungsi yang digunakan untuk memberikan (set) nilai untuk suatu properti. Umumnya Method ini digunakan untuk properti yang mempunyai visibilitas protected atau private. Method setter sering digunakan untuk memvalidasi data yang masuk sebelum data tersebut diisikan ke suatu properti.

Method Getter adalah suatu Method yang digunakan untuk menghasilkan (get) suatu nilai baik dari member public, private, protected atau pun dari hasil perhitungan.

Contoh :

```
class Orang{
    private $nama;
    private $jenisKelamin;
    private $noHP;
    private $noRekening;
    private $pinATM;
    function __construct($namaa, $jenisKelaminn, $noHPP, $noRekeningg, $pinATMm){
        $this->nama = $namaa;
        $this->jenisKelamin = $jenisKelaminn;
        $this->noHP = $noHPP;
        $this->noRekening = $noRekeningg;
        $this->pinATM = $pinATMm;
    }
    public function getNama(){
        echo "Namanya ".$this->nama."<br>";
    }
    public function getJenisKelamin(){
        echo "Jenis kelaminnya ".$this->jenisKelamin."<br>";
    }
    public function setNoHP($hpBaru){
        $this->noHP = $hpBaru;
        echo "Nomor Hp berhasil diperbarui.<br>";
    }
    public function getNoHP(){
        echo "Nomor hpnya adalah ".$this->noHP."<br>";
    }

    public function getNoRekening(){
        echo "Nomor Rekeningnya adalah ".$this->noRekening."<br>";
    }
    public function setPinATM($pinBaru){
        $this->pinATM = $pinBaru;
        echo "Pin ATM berhasil diganti.<br>";
    }
}
$orang1 = new Orang("Andi", "Laki-laki", "081234567890", "001001001", "1234");
$orang1->getNama();
$orang1->getJenisKelamin();
$orang1->getNoHP();
$orang1->getNoRekening();
$orang1->setNoHP("088888888888");
$orang1->setPinATM("4321");
$orang1->getNoHP();
```

Pada kode di atas, method setter yaitu setNoHP() dan setPinATM. Sedangkan method getter-nya yaitu getNama(), getJenisKelamin(), getNoHp, dan getNoRekening(). Semua property dijadikan private, sehingga pengubahan saldo hanya dapat dilakukan di class Bank saja, guna mencegah pengubahan jumlah saldo oleh pihak selain bank. Function setNoHP() dan setPinATM() untuk mengontrol nilai property \$noHP dan \$pinATM. Karena tanpa mengontrol atribut tersebut maka siapapun bisa mengganti nomor HP pelanggan atau pin ATM nya. Sedangkan function getNama(), getJenisKelamin(), getnoHP(), dan getNoRekening() untuk mengembalikan nilai property-property tersebut tanpa mengakses property itu sendiri. Property seperti \$nama, \$jenisKelamin dan \$noRekening tidak memiliki function setter, karena property tersebut hanya berubah ketika di set di

awal program atau pendaftaran rekening. Property \$pinATM tidak memiliki function getter karena property \$pinATM bersifat rahasia.

## INHERITANCE (PEWARISAN)

Pewarisan/Penurunan merupakan fitur OOP yang sangat berguna. Dengan pewarisan, sebuah class (parent class) dapat diturunkan menjadi class baru (child class) yang mempunyai sebagian atau seluruh karakteristik dari class parentnya.

Karakteristik pewarisan:

- a. Pewarisan dilakukan dengan keyword ‘extends’.
- b. Child class akan mewarisi semua property dan method dengan access modifier public atau protected dari parent class.
- c. Access Modifier berlaku di pewarisan.
- d. Untuk mencegah pewarisan kelas dan mencegah method overriding bisa menggunakan keyword ‘final’.
- e. Dalam pewarisan dikenal istilah Override yaitu mendefinisikan ulang suatu function yang telah tersedia di class child sehingga mempunyai perilaku yang berbeda dari perilaku function class parentnya.
- f. Dalam child class, boleh ditambah property atau method baru.

Sintaks:

```
class parentClass{  
    public $property1;  
    public function method1(){  
        // set intruksi1  
    }  
}  
  
class childClass1{  
    public $property1;  
    public $property2;  
    public function method1(){  
        // set intruksi2  
    }  
}
```

Contoh:

```

class Fruit {
    public $name;
    public $color;
    public function __construct($name, $color) {
        $this->name = $name;
        $this->color = $color;
    }
    protected function intro() {
        echo "The fruit is {$this->name} and the color is {$this->color}." ;
    }
    final function outro(){
        $this->name = NULL;
        $this->color = NULL;
        echo "Class Fruit has been closed";
    }
}
class Strawberry extends Fruit {
    public function message() {
        echo "Am I a fruit or a berry? ";
        // Memanggil protected function dari parent class - OK
        $this -> intro();
    }
}
$strawberry = new Strawberry("Strawberry", "red"); // OK. __construct() bersifat publik
$strawberry->message(); // OK. message() bersifat publik dan dipanggil oleh method intro() (yang bersifat protected) dari parent
classnya
$strawberry->outro();
final class Apple extends Fruit{
}
class FujiApple extends Apple{ // ERROR. Class FujiApple tidak bisa mewarisi final class Apple
}
$apple1 = new FujiApple();

```

Class Fruit merupakan parent class dari class Strawberry, class Apple dan class FujiApple. Yang artinya class Strawberry, class Apple dan class FujiApple mewarisi seluruh property dan method yang memiliki access modifier public dan protected pada class Fruit. Sehingga seluruh property dan function pada class Fruit dapat diakses di child classnya. Pada class Strawberry terjadi penambahan function message() yang hanya bisa dipakai di class Strawberry dan child classnya.

Selain class Strawberry yang menjadi child class dari class Fruit, juga terdapat class Apple. Class Apple tidak memiliki penambahan property dan method lagi, bahkan tidak ada overriding di method. Namun class Apple, menggunakan keyword final. Artinya class Apple tidak dapat diwariskan ke class lainnya. Sehingga error terjadi ketika class Apple diwariskan ke class FujiApple.

Method outro() pada class Fruit juga menggunakan keyword final. Artinya method ini dapat diturunkan (diwariskan) di child class, namun tidak bisa di-override. Hal ini akan mencegah pengubahan setiap intruksi di dalam method outro().

## ***TUGAS PRAKTIKUM***

Lengkapi program web simulasi transaksi Sistem Bank berikut dengan menggunakan prinsip OOP. Dengan rincian:

- Minimal dua class yaitu NasabahBiasa dan NasabahPremium.

- b. Nasabah biasa dapat melakukan pengecekan saldo, transfer maksimal Rp 1 Juta perhari, tarik tunai maksimal Rp 1 Juta per hari, minimal saldo setelah transfer atau tarik tunai adalah Rp 100 ribu. Biaya Transfer Rp 1000 per transaksi. Tidak dapat mengajukan pinjaman.
- c. Nasabah premium dapat pengecekan saldo, transfer maksimal Rp 10 Juta perhari, tarik tunai maksimal Rp 10 Juta per hari, minimal saldo setelah transfer atau tarik tunai adalah Rp 0 . Biaya Transfer Rp 0 per transaksi. Dapat mengajukan pinjaman maksimal 2 kali lipat dari saldoanya.
- d. Gunakan modifier access pada method atau atribut agar transaksi terkontrol dan lebih aman.

Lengkapi program berikut.

```

class NasabahBiasa {
    $saldo;
    $nama;
    $noRekening;
    $pinATM;
    $riwayatTransaksi;
    function __construct($saldo, $nama, $noRekening, $pinATM){
        //Masukkan informasi akun
    }function setSaldo($nominal){
        // Lakukan pengubahan nilai saldo, jika transaksi dilakukan
    }function getSaldo(){
        // cek saldo
    }function setorTunai($nominal){
        // operasi pada setor tunai
    }function tarikTunai($nominal){
        // operasi pada tarik tunai
    }function login($noRekening, $pinATM){
        //Cek ke valid atau tidaknya data yang dimasukkan
    }function mintaPinjaman($nominal){
        echo "Pinjaman Tidak diperbolehkan, silahkan upgrade akun Anda menjadi akun premium";
    }function riwayatTransaksi(){
        // berisi array riwayat transaksi mengenai nama, noRekening, tanggal transaksi,
        jenisTransaksi, jumlah saldo terakhir
    }
}
class AkunBankPremium extends AkunBankBiasa{
    $jumlahHutang;
    function mintaPinjaman($nominal){
        //Lakukan operasi minta pinjaman
    }
}

```

## BAB VII

### Laravel I

#### *Apa itu Framework?*

Frame artinya kerangka, dan work artinya kerja. Secara harfiah, framework berarti kerangka kerja. Jika dihubungkan dengan pemrograman web, maka pengertian framework adalah kerangka kerja yang digunakan programmer dalam membuat website.

Alih-alih menulis seluruh kode program dari nol, framework menyediakan kerangka kerja yang bisa digunakan programmer untuk mempercepat pekerjaannya.

Framework menyediakan sekumpulan kelas, fungsi, variabel, dan komponen-komponen lain yang kerap dibutuhkan programmer dalam mengembangkan sebuah website.

Dengan menggunakan framework, Anda tidak perlu menulis seluruh kode program secara manual. Cukup menggunakan kerangka dasar yang sudah disiapkan di dalam framework.

#### *Cara Kerja Framework*

Pertama-tama, bayangkan web development layaknya proses perakitan mobil. Dalam merakit mobil, Anda harus menyediakan semua komponen yang dibutuhkan agar mobil dapat beroperasi dengan baik.

Secara fungsional, Anda harus menyiapkan komponen-komponen bagian dalam mobil seperti mesin, suspensi, rem, dan lain sebagainya. Ini sama dengan back-end website yang berisi sekumpulan fungsi, perintah, dan cara kerja yang berkaitan dengan fungsionalitas website seperti ilustrasi fitur pencarian yang kita bahas di atas.

Secara tampilan, mobil juga harus terlihat menarik, futuristik, dan sesuai dengan selera pasar. Anda harus merancang model, ukuran, warna, corak, dan hal-hal lain yang berkaitan dengan aspek visual dari sudut pandang pengguna.

Dalam web development, ini adalah bagian front-end yang berfungsi untuk merancang dan mempercantik tampilan website seperti background, font, kombinasi warna, dan sebagainya.

Sekarang, bayangkan jika Anda harus memproduksi seluruh komponen mobil secara manual. Anda harus merancang dan membuat mesin, mendesain chasis, mengatur mekanisme keamanan, dan seluruh aspek rumit dalam sebuah mobil. Tentu membutuhkan banyak waktu, tenaga, pikiran, dan biaya. Sama halnya dengan mengembangkan website dari nol.

Lain ceritanya jika Anda menggunakan framework. Layaknya puzzle, framework terdiri dari kepingan-kepingan komponen yang bisa Anda susun sesuka hati hingga membentuk gambar yang utuh.

Ini seperti komponen siap pakai yang tinggal disusun dan dirangkai untuk membuat mobil. Dalam web development, framework menyediakan fungsi-fungsi berkualitas tinggi yang bisa langsung Anda gunakan.

Dengan begitu, proses pengembangan website bisa dilakukan secara lebih mudah, cepat, hemat, serta dengan hasil yang lebih berkualitas. Seperti itulah cara kerja framework dalam web development.

## ***Framework VS Library***

Banyak orang keliru mengartikan framework sebagai library, padahal keduanya memiliki arti dan fungsi yang berbeda. Konsepnya memang mirip, library dan framework sama-sama menyediakan fitur dan fungsionalitas yang dibutuhkan dalam web programming. Tapi dari segi kompleksitas, framework jauh lebih lengkap dan luas dari library.

Perbedaan pertama dapat dilihat dari jenis objek. Library berisi sekumpulan kode yang terdiri dari kelas, fungsi, dan variabel siap pakai. Framework pun sama, namun fungsinya jauh lebih luas dari sekedar menyediakan sepaket fitur.

Library bisa diinstall secara individual layaknya plugin tambahan di web browser. Pernah menginstall plugin dari Chrome Web Store? Kurang lebih seperti itulah proses instalasi library. Mekanismenya tergantung dari bahasa pemrograman yang Anda gunakan, biasanya melalui Command Line.

Sedangkan framework adalah aplikasi utuh yang membutuhkan aplikasi-aplikasi pendukung lain. Laravel misalnya, membutuhkan banyak ekstensi [PHP](#) beserta aplikasi-aplikasi lain yang harus diinstall ke perangkat. Biasanya, programmer [menginstall XAMPP](#) yang menyediakan paket lengkap sesuai kebutuhan Laravel. Ukuran dan konsumsi sumber daya framework juga lebih besar karena berwujud aplikasi.

Perbedaan lainnya bisa dilihat dari mekanisme kontrol antara keduanya. Saat menggunakan library, kode program Anda memanggil library dengan metode import. Sebaliknya, saat menggunakan framework, framework-lah yang memanggil dan mengimport kode program Anda. Mekanisme ini disebut dengan Inversion of Control (IoC).

*Jadi, jangan keliru menyebut library sebagai framework, atau sebaliknya.*

#### Fungsi Framework dalam Web Development

1. Meningkatkan Efisiensi Programmer
2. Mempermudah Kolaborasi Tim
3. Kualitas Kode Lebih Terjamin
4. Minim Cela Keamanan
5. Hemat Biaya

Yang di sebut dengan Native artinya asli, yaitu pemrograman PHP yang murni yang di susun dengan cara di coding/dibangun oleh para programmer itu sendiri tanpa ada istilah tambahan buat setting/konfigurasi lainnya. Manfaat dari PHP Native sederhana jika kita sudah menguasainya maka akan lebih mudah menggunakannya dengan PHP Framework.

#### ***Kelebihan PHP Native :***

- Pemrograman yang dibangun atas dasar pemikiran seorang programmer itu sendiri. Sehingga menerbitkan suatu program yang akan membuat program akan lebih baik.
- Bisa dibentuk dalam sebuah format OOP (Object Oriented Programming) sehingga dapat membentuk suatu program.
- Kita bisa menerapkannya dalam skala mudah sampai kita tau tergantung tingkat kesulitannya.

### ***Kelemahan PHP Native :***

- Dokumentasi pemrograman yang tidak jelas, karena PHP Native dibangun atas dasar pemahaman seorang programmer “tersebut” maka belum tentu pemikiran seorang programmer lainnya selaras dengan programmer yang merancang programmer tersebut. Maka dari itu dokumentasi merupakan salah satu hal wajib yang diperlukan oleh programmer, agar ia dapat melakukan maintenance (perbaikan) ataupun Upgrade.
- Biasanya tidak cocok untuk proses bisnis ataupun sistem yang sangat luas. Karena framework cukup rendah untuk suatu bisnis yang terlalu luas.
- Tidak ada Coding Style Consistence, apabila sistem kita masih dikelola oleh programmer lainnya maka akan ada perbedaan dalam penulisan source code program.
- Tidak ada Security Concern framework, dan tidak ada security/pengamanan default pada sistem yang sudah dibangun. Alias kita harus membangun sendiri pengamanannya. Supaya terhindar dari kesalahan-kesalahan dalam hal apapun.

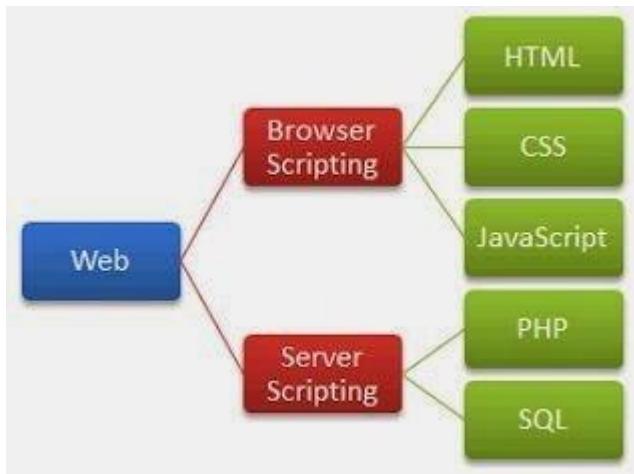
### ***Client-side Web Programming***

Client side scripting merupakan salah satu jenis bahasa pemrograman web yang proses pengolahannya dilakukan di sisi client.

Proses penerjemahan atau pengolahannya dilakukan oleh web browser sebagai client-nya, didalam web browser sudah terdapat library yang mampu menerjemahkan semua perintah dihalaman web yang menggunakan client side scripting.

Contoh dari client side scripting ini antara lain

- HTML,
- XHTML,
- CSS,
- JavaScript,
- XML.
- JQUERY



## ***Server-side Web Programming***

Server side scripting merupakan bahasa pemrograman web yang pengolahannya dilakukan dalam server, maksudnya ialah web server yang sudah telah terintegrasi oleh web engine.

Disini peran web engine ialah memproses semua script yang ada termasuk kategori client side scripting. Web engine biasanya harus diinstal dalam komputer terlebih dahulu dari bagian terpisah web server.

Contoh dari server side scripting ialah

- Active server pages (ASP),
- PHP : Hypertext preprocessor (PHP),

## **Apa itu Laravel?**

Sebelum membahas **Laravel**, mari kita belajar dahulu mengenai konsep atau definisi dari framework PHP dan arsitektur MVC. Karena Laravel merupakan sebuah framework PHP berbasisis MVC.

## **Apa itu MVC?**

MVC adalah suatu arsitektur aplikasi dengan cara memisahkan aplikasi menjadi tiga bagian yang saling terhubung dengan sesuai dengan bagaimana informasi disajikan kepada pengguna dan didapat dari pengguna. MVC terdiri dari 3 bagian, yaitu:

Model : Adalah bagian yang memiliki fungsi-fungsi untuk mengelola database

View : adalah bagian yang bertugas menampilkan data

Controller : adalah bagian yang menjadi penghubung antara model dan view. Controller memiliki perintah-perintah yang berfungsi untuk memproses bagaimana data ditampilkan dari Model ke View, atau bagaimana data dikirimkan dari View ke Model.

## **Apakah ada framework PHP yang berbasis MVC selain Laravel?**

## **Mengapa Laravel?**

Saat ini Laravel adalah salah satu framework PHP paling besar dan paling banyak digunakan di dunia. Hal ini menjadikan Laravel memiliki banyak sekali tutorial dan komunitas yang dapat membantu kita bila mengalami kesulitan dalam membangun aplikasi. Laravel juga memiliki banyak sekali *third-party module* yang dapat kita unduh secara gratis dan mudah.

## **Server Requirements**

### **Apa yang dibutuhkan untuk membangun sebuah aplikasi dengan Laravel?**

Berikut spesifikasi server yang dibutuhkan untuk membangun sebuah aplikasi dengan menggunakan Laravel 8 :

1. PHP minimal versi 7.3

2. Mysql 5.7.39
3. Web Server
4. Composer 2.0
5. Mamp/Mysql (Optional)

## Install Composer

Laravel membutuhkan Compose to manajemen dependencies. Jadi sebelum menggunakan laravel kita harus memastikan Composer telah terinstal di mesin.

## Install Laravel

### Via Laravel Installer

First, download menggunakan Laravel installer..

```
composer global require "laravel/installer=~1.1"
```

Pastikan `~/.composer/vendor/bin` directory sudah ada dalam PATH sehingga `laravel` executable dapat dieksekusi.

Ketika `laravel new` command dieksekusi will membuat Laravel installation di directory yang sudah dispesifikasi. Untuk instance, `laravel new blog` akan membuat direktori dengan nama `blog` mengandung Laravel installation dengan semua dependensi terinstal. Metode instalasi ini lebih cepat dari instal menggunakan Composer:

```
laravel new blog
```

### Via Composer Create-Project

Bisa juga menggunakan Composer `create-project` command in your terminal:

```
composer create-project laravel/laravel:^8.0 example-app
```

## *Konsep dasar MVC*

Seperti penjelasan sebelumnya, MVC adalah konsep yang terpisah menjadi tiga bagian, yakni *model*, *view*, dan *controller*. Tiga komponen ini sebenarnya mempunyai pengertian dan tugasnya masing-masing dalam mendukung pembangunan aplikasi *web* atau *mobile*.

## 1. Model

Komponen *model* berhubungan dengan database dan interaksi. *Model* umumnya merepresentasikan struktur data dari suatu aplikasi dengan bentuk basis data seperti *file teks*, *file XML*, atau *web service*. *Model* menentukan data apa yang harus ada di dalam aplikasi. Jika suatu keadaan aplikasi berubah, *model* akan memberitahu tampilan dan *controller*.

Apabila masih kurang paham, mudahnya begini, contoh suatu aplikasi *web toko online* mempunyai fitur barang favorit atau keranjang belanja. *Model* yang akan menentukan apa saja data yang terdapat pada barang favorit atau keranjang belanja tersebut, misalnya harga, nama barang, nama toko, jumlah dan lainnya.

## 2. View

Komponen kedua dari konsep MVC adalah *view*. *View* menjadi bagian yang berkaitan langsung dengan tampilan pada pengguna atau menangani presentation *logic*. Pada suatu aplikasi *web*, *view* berbentuk *file template* HTML yang diatur oleh *controller*.

*View* akan menentukan bagaimana daftar disajikan pada pengguna dan juga menerima data yang perlu ditampilkan dari *model*. Meski begitu, *view* tidak memiliki akses langsung untuk mencampuri bagian *model*.

## 3. Controller

Konsep terakhir dari bagian MVC adalah *controller*. *Controller* berisikan logika yang dapat memperbarui *model* atau tampilan sebagai respon dari tindakan pengguna aplikasi. Tugas *controller* sebenarnya cukup banyak, yakni menyediakan variabel yang akan tampil pada bagian *view*, menyediakan penanganan *error*, mengerjakan proses logika aplikasi, melakukan validasi, dan memanggil *model* untuk akses ke basis data.

Lebih mudahnya, *controller* menjadi bagian yang bertugas untuk menghapus atau menambahkan suatu barang pada toko *online*. Bila kamu memutuskan untuk menghapus barang dari keranjang belanjaan toko *online*, maka *controller* perlu pembaruan *model* lalu menerima *input*, kemudian memanipulasi *model* sesuai keinginan pengguna, baru ditampilkan pada bagian *view*.

Namun, *controller* juga bisa langsung memperbarui tampilan tanpa membutuhkan *model*. Contohnya, saat pengguna ingin mengubah daftar barang sesuai abjad. Bisa dikatakan bahwa *controller* merupakan ‘otak’ dari semua bagian MVC.

## *Cara kerja MVC*

Setelah mengetahui fungsi dan konsep dasar, selanjutnya ada cara kerja dari MVC yang punya proses cukup unik. MVC adalah konsep desain arsitektur yang berfungsi sebagai pendukung pembangunan

suatu aplikasi *web* atau *mobile*. Selama masa percobaan atau *testing* aplikasi, terdapat cara kerja MVC yang perlu diketahui, di antaranya:

### **1. View akan menampilkan user interface aplikasi**

Pada aplikasi uji coba konsep MVC, *view* pertama kali akan menampilkan *user interface* dan informasi tambahan lain yang terdapat di aplikasi pada pengguna. Tahap ini harus dilakukan semenarik mungkin, karena tampilan awal sebuah aplikasi menjadi penentu pengguna menyukainya atau tidak.

Ketika pengguna melakukan *request* di aplikasi, *view* akan menampung hal tersebut dan melanjutkannya ke bagian *controller*. Apabila sudah dilanjutkan, *controller* menerima *request* aplikasi dari bagian *view*.

### **2. Controller memberikan instruksi pada model**

Sesudah menerima *request* aplikasi dari *view*, *controller* perlu memberikan instruksi pada *model* untuk menyiapkan informasi yang berkaitan dengan permintaan bagian *view*.

Setelah menemukan informasi yang berkaitan dengan permintaan bagian *view*, *model* tidak bisa langsung mengirimkannya begitu saja ke *controller*. *Model* perlu mengelola informasi yang telah terkumpul di *database*. Hal ini dilakukan untuk menyaring kembali mana informasi yang benar-benar penting dan tidak.

### **3. Model menyerahkan hasil pengolahan informasi database**

Saat mengelola informasi di *database*, *model* tak melakukannya sendirian saja, ada bantuan logika pemrograman yang membuat proses pengelolaan lebih cepat. Setelah selesai, *model* harus menyerahkan hasil pengolahan informasi *database* ke *controller*, bukan bagian *view*. Kemudian, *view* menggunakan data yang telah siap diterima *controller* dari *model* untuk ditampilkan pada pengguna.

## ***Kelebihan MVC***

Penggunaan konsep MVC sangat membantu para Developer baik itu Developer Web atau Mobile. Developer dibantu dari segi hal seperti manajemen *source code*, keamanan, validasi data, *query database*, dan lain-lain. MVC adalah konsep yang membuat implementasi aplikasi jadi jauh lebih sederhana, sehingga jumlah baris program pun sedikit. Di bawah ini masih ada kelebihan lain dari penggunaan MVC.

### **1. Mampu mendukung pemrograman asinkron**

MVC adalah konsep desain arsitektur dengan *multi-purpose programming technique*, karena adanya integrasi bersama *framework JavaScript*. Dari kondisi ini, aplikasi yang menggunakan konsep MVC berarti dapat bekerja dengan *file PDF*, *widget desktop*, dan *browser* untuk situs tertentu. Selain itu,

komponen logika tersebut juga mendukung teknik pemrograman asinkron yang mampu membantu Developer mengembangkan aplikasi dengan cepat.

## 2. Dapat mengembangkan web menjadi SEO friendly

Siapa yang tidak ingin *website*-nya mendapatkan banyak kunjungan dari pengguna? Tentu saja semua orang menginginkannya. MVC adalah salah satu *platform* desain arsitektur yang dapat mendukung aplikasi *web* masuk dalam kategori SEO *friendly*. Hal ini memudahkan kamu untuk mengembangkan aplikasi *web* melalui banyaknya *traffic* kunjungan.

## 3. Adanya perubahan kode program tidak berpengaruh

Pada aplikasi *web* mana pun, *user interface* adalah komponen yang paling sering berubah dibandingkan lainnya. Perubahan tersebut bisa terjadi di *font*, warna, *layout* fitur tertentu, dan penambahan atau pengurangan fitur untuk *web* maupun *mobile*. Perubahan pada *user interface* biasanya akan memengaruhi komponen lainnya. Pilihannya hanya ada dua, yaitu merubah semua komponen aplikasi *web* atau tetap membiarkan *user interface* tampil seperti itu.

Dengan bantuan konsep MVC, kamu tidak perlu mengkhawatirkan hal tersebut. Sebab, MVC melalui bagian *view* dan *model* membuat kita bebas menambahkan atau mengubah sesuatu pada tampilan *user interface*.

# BAB VIII

## Laravel II

### **Basic Routing**

#### **Routing pada Laravel**

Routing atau Rute dalam bahasa indonesia pada laravel berguna untuk mengarahkan halaman yang ingin ditampilkan. Routing akan muncul pada URL browser yang dapat diakses oleh pengguna, Routing pada laravel berhubungan dengan controller tetapi juga bisa langsung diarahkan menuju view tanpa melalui controller.

```
● ● ●

<?php

use Illuminate\Support\Facades\Route;
use App\Http\Controllers\ArticleController;

Route::get('/', function () {
    return view('welcome');
});

Route::get('/article', function () {
    return "Welcome to my world !";
});

Route::get('/article/v2', [ArticleController::class, 'article']);
```

## Method Router

Router mengizinkan beberapa method yang merespon HTTP Verb;



```
Route::get($uri, $callback);
Route::post($uri, $callback);
Route::put($uri, $callback);
Route::patch($uri, $callback);
Route::delete($uri, $callback);
Route::options($uri, $callback);
```

## The Route List

The `route:list` Artisan command untuk mengetahui daftar routes yang ada di application

```
php artisan route:list
```

## Route Parameters

### Required Parameters

Terkadang Anda perlu menangkap segmen URI dalam rute Anda. Misalnya, Anda mungkin perlu mengambil ID pengguna dari URL. Anda dapat melakukannya dengan menentukan parameter rute:

```
Route::get('/user/{id}', function ($id) {
    return 'User ' . $id;
});

// Juga bisa menggunakan lebih dari 1 route parameter

Route::get('/posts/{post}/comments/{comment}', function ($postId, $commentId) {
    //
});
```

## Optional Parameters

Terkadang Anda mungkin perlu menentukan parameter rute yang mungkin tidak selalu ada di URI. Anda dapat melakukannya dengan menempatkan ? tandai setelah nama parameter. Pastikan untuk memberikan nilai default pada variabel terkait rute:

```
Route::get('/user/{name?}', function ($name = null) {
    return $name;
});

Route::get('/user/{name?}', function ($name = 'John') {
    return $name;
});
```

## Named Routes

Rute bernama memungkinkan pembuatan URL atau pengalihan yang mudah untuk rute tertentu. Anda dapat menentukan nama untuk rute dengan merantai metode nama ke definisi rute:

```
Route::get('/user/profile', function () {
    //
})->name('profile');

Route::get('/user/profile',[UserProfileController::class, 'show']
)->name('profile');
```

## Route Prefixes

Metode awalan dapat digunakan untuk mengawali setiap rute dalam grup dengan URI yang diberikan. Misalnya, Anda mungkin ingin mengawali semua URI rute dalam grup dengan admin:

```
Route::prefix('admin')->group(function () {
    Route::get('/users', function () {
        // Matches The "/admin/users" URL
    });
});
```

## **Route Name Prefixes**

Metode nama dapat digunakan untuk mengawali setiap nama rute dalam grup dengan string yang diberikan. Misalnya, Anda mungkin ingin mengawali semua nama rute yang dikelompokkan dengan admin. String yang diberikan diawali dengan nama rute persis seperti yang ditentukan, jadi kami pasti akan memberikan trailing . karakter di awalan:



```
Route::name('admin.')->group(function () {
    Route::get('/users', function () {
        // Route assigned name "admin.users"...
    })->name('users');
});
```

## ***Apa itu Laravel Model?***

Model merupakan salah satu dari bagian MVC yang akan berkomunikasi dengan database. Model yang sudah terhubung ke database akan digunakan/dipanggil via Controller sebagaimana konsep MVC itu berjalan.

Migration merupakan salah satu fitur Laravel yang berfungsi seperti version control untuk database. Melalui fitur ini sebuah team pengembangan web development akan dapat bekerja dalam team untuk mengelola dan modifikasi skema basis data aplikasi. Migration biasanya dipasangkan dengan Schema Builder dari Laravel untuk dengan mudah membangun skema basis data aplikasi Anda. Facade Skema Laravel menyediakan dukungan untuk membuat dan memanipulasi tabel di semua sistem basis data yang didukung Laravel.

## ***Membuat dan Memanfaatkan Laravel Model***

Dalam contoh kasus berikut, akan dibuat database yang diberi nama Blog.

## ***Membuat Database di MySQL/MariaDB***

Silahkan membuat sebuah database dengan menggunakan PHPMyAdmin. Beri nama database tersebut dengan nama yang diinginkan. Pada contoh selanjutnya database diberi nama “Blog”. Jangan lupa untuk mendapatkan username dan password yang akan dipergunakan untuk mengakses Database. Pada umumnya default username untuk instalasi database adalah menggunakan username root dan tanpa password. Hindari praktek seperti ini pada versi production dari web yang dikembangkan.

## ***Membuat Model dengan Artisan***

Berikut adalah perintah untuk membuat Model. Bagian akhir perintah diberi kode migration –m dengan tujuan nantinya akan dihasilkan sebuah table dalam database.

```
php artisan make:model Post -m
```

## ***Struktur Migrations***

Class class terdiri dari 2 methods: up dan down. Method up dipergunakan untuk menambahkan tabel, columns, atau index pada database. Sebaliknya down dipergunakan untuk mengembalikan status pada posisi sebelum method up dijalankan.

Contoh sebuah migrations untuk tabel flights:

```
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

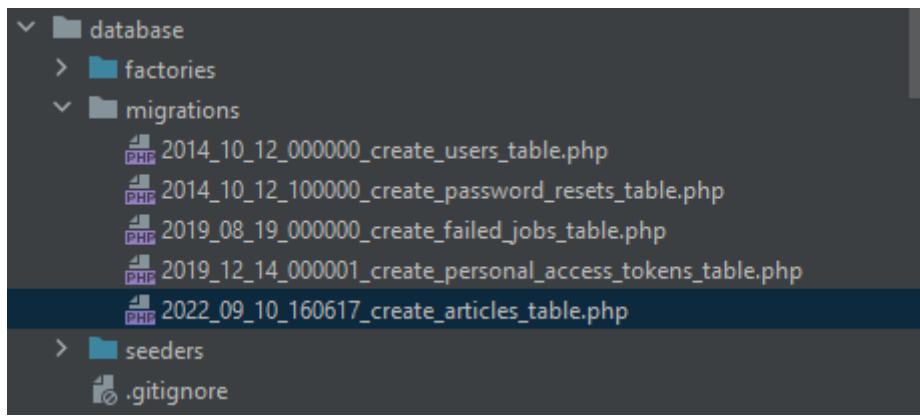
class CreateArticlesTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('articles', function (Blueprint $table) {
            $table->id();
            $table->string('judul')->nullable();
            $table->text('isi_berita')->nullable();
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('articles');
    }
}
```

## ***Migration***

Perintah artisan make:model dengan option –m juga akan menghasilkan file /database/migrations/yyy\_mm\_dd\_063343\_create\_articles\_table.php. Di folder tersebut telah terdapat 2 file lain yang dihasilkan secara default berfungsi untuk fitur

user authentication dan reset password.



File Migration yang dihasilkan

File migration yang dihasilkan dapat dijalankan kemudian setelah definisi table database yang akan dibuat disempurnakan. Database akan dibuat berdasarkan konfigurasi di function up(), sedangkan function down() akan dijalankan untuk menghapus table. Perintah untuk berbagai tipe data dapat diakses pada dokumentasi Laravel di link berikut:

## ***Migrate***

Sebelum menjalankan Migrations, pastikan semua credential dari Database telah benar. Silahkan mengubah nama database, username, dan password pada file .env yang terletak di root folder.

```

.env
9  DB_CONNECTION=mysql
10 DB_HOST=127.0.0.1
11 DB_PORT=3306
12 DB_DATABASE=portfolio
13 DB_USERNAME=root
14 DB_PASSWORD=
15
```

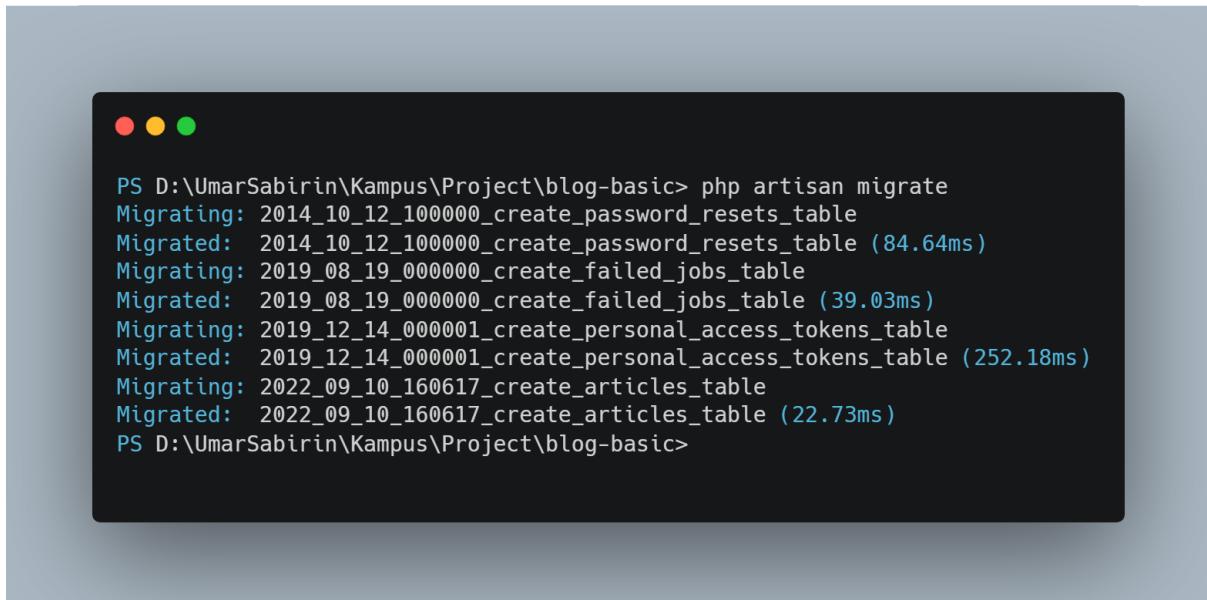
File .env yang berisi database connection credentials

Pastikan juga untuk memastikan database server sedang berjalan database telah di create sebelum menjalankan Migrations. Jalankan migrations dengan perintah:

```
$ php artisan migrate
```

Migrations akan mendapati mana saja file migrations yang belum dijalankan berdasarkan data yang tersimpan di database. Untuk file migrations yang telah dijalankan tidak akan dijalankan kembali.

```
$ php artisan migrate
```



```
PS D:\UmarSabirin\Kampus\Project\blog-basic> php artisan migrate
Migrating: 2014_10_12_100000_create_password_resets_table
Migrated: 2014_10_12_100000_create_password_resets_table (84.64ms)
Migrating: 2019_08_19_000000_create_failed_jobs_table
Migrated: 2019_08_19_000000_create_failed_jobs_table (39.03ms)
Migrating: 2019_12_14_000001_create_personal_access_tokens_table
Migrated: 2019_12_14_000001_create_personal_access_tokens_table (252.18ms)
Migrating: 2022_09_10_160617_create_articles_table
Migrated: 2022_09_10_160617_create_articles_table (22.73ms)
PS D:\UmarSabirin\Kampus\Project\blog-basic>
```

Tampilan apabila migrations berhasil

Hasil dari proses migrations dapat diperiksa melalui aplikasi phpMyAdmin.

## ***Controller***

Laravel Controller merupakan salah satu bagian dimana seluruh fungsional web dibuat. Pada Controller dilakukan pengaturan untuk mengakses Model terkait dengan Database dan juga bagaimana mengirimkan datanya ke View. Berbagai pemrosesan juga dilakukan di dalam Controller

Ada 2 cara membuat controller pada laravel, yang pertama kita bisa membuat controller laravel dengan cara membuat langsung file controller baru nya dalam folder dan cara yang kedua kita bisa menggunakan perintah php artisan dari laravel

Cara membuat controller :



```
php artisan make:controller ArticleController
```



```
<?php

namespace App\Http\Controllers;

use App\Models\Article;
use Illuminate\Http\Request;

class ArticleController extends Controller
{

    public function article() {
        return "test only";
    }

    public function list() {
        $articles = Article::all();
        return view('article.list', compact('articles'));
    }
}
```

## Views

### Basic Usage

Tampilan berisi HTML yang disajikan oleh aplikasi Anda, dan berfungsi sebagai metode praktis untuk memisahkan logika kontroler dan domain dari logika presentasi Anda. Tampilan disimpan di direktori resources/views.

Tampilan sederhananya seperti ini:

```
<!-- View stored in resources/views/greeting.php -->

<html>
  <body>
    <h1>Hello, <?php echo $name; ?></h1>
  </body>
</html>

Route::get('/', function()
{
    return view('greeting', [ 'name' => 'James' ]);
});
```

Seperti yang Anda lihat, argumen pertama yang diteruskan ke view helper sesuai dengan nama file tampilan di direktori resources/views. Argumen kedua yang diteruskan ke helper adalah variabel data yang harus tersedia untuk dilihat.