

Full Stack Web Development

Intro to front-end development, HTML & CSS fundamental

Job Connector Program

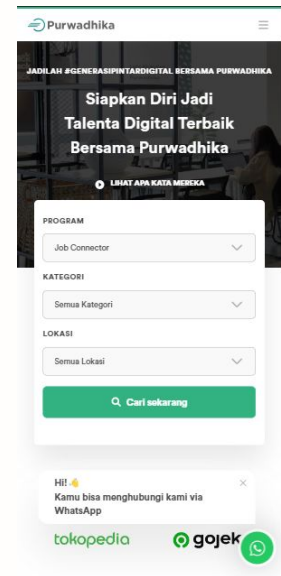
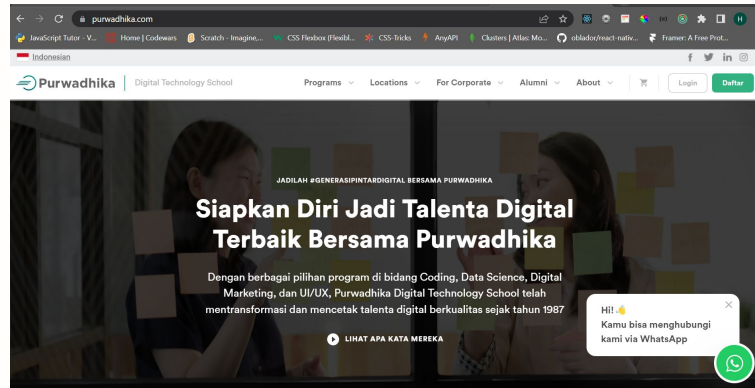
Outline

- Intro to Front-end development
- HTML fundamental
- Basic CSS

What a Front End Developer Does

A **front end developer** has one general responsibility:

To ensure that website visitors can easily interact with the page. They do this through the combination of design, technology and programming to code a website's appearance, as well as taking care of debugging. Whenever you visit a website, anything that you see, click on or otherwise use is the work of a front end developer.



Intro to Front-end Development

Front end development is the development of code that creates the visual front-end elements of a software, application or website. Front end languages include **HTML**, **CSS**, and **Javascript**



HTML defines the content of web pages

CSS specifies the layout of web pages

JS programs the behaviour of web pages

HTML, CSS, and JavaScript are the basic languages you need to know to create a website.

To become a Front-End Developer, start with the subjects below, in the following order:

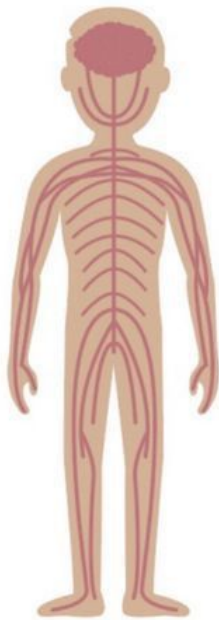
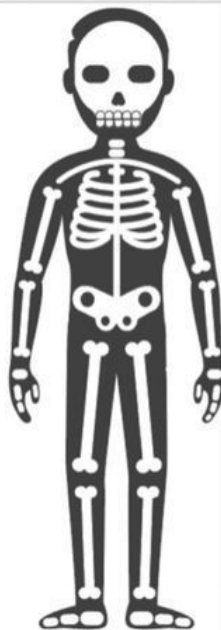
1. **Create the structure with HTML.** The first thing you have to learn is HTML, which is the standard markup language for creating web pages.
2. **Style with CSS.** The next step is to learn CSS, to set the layout of your web page with beautiful colors, fonts, and much more.
3. **Make it interactive with JavaScript.** After studying HTML and CSS, you should learn JavaScript to create dynamic and interactive web pages for your users.

Intro to Front-end Development

HTML

JS

CSS



Basic HTML Document

Make your first web page

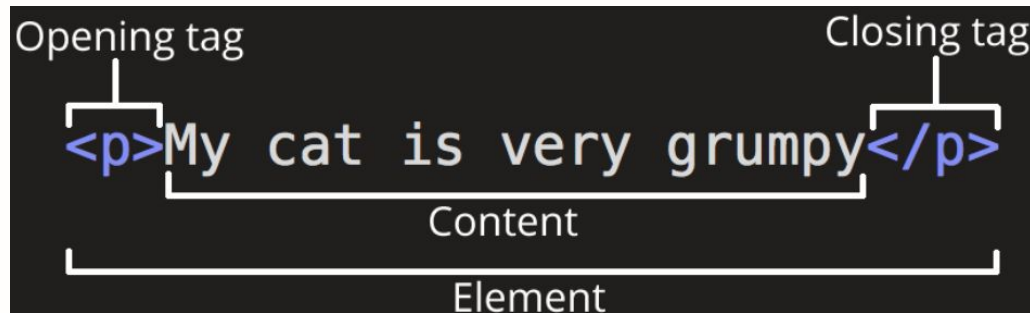


```
<!DOCTYPE html>
<html>
  <head>
    <title>Purwadhika</title>
  </head>
  <body>
    <p>My cat is very grumpy</p>
  </body>
</html>
```

Tag	Description
<!DOCTYPE...>	This tag defines the document type and HTML version.
<html>	This tag encloses the complete HTML document and mainly comprises of document header which is represented by <head>...</head> and document body which is represented by <body>...</body> tags.
<head>	This tag represents the document's header which can keep other HTML tags like <title>, <link> etc.
<title>	The <title> tag is used inside the <head> tag to mention the document title.
<body>	This tag represents the document's body which keeps other HTML tags like <h1>, <div>, <p> etc.

There is **main parts** of the line of code follows:

1. **The opening tag**, define the name of element (in this case, p), wrapped with opening and closing angle brackets.
2. **The closing tag**, similar as the opening tag, this tag includes a forward slash before the element name. This tag places in the end of elements
3. **The content**, content places between the opening and closing tags. In this case, content written as a text.
4. **The element**, the opening tag, the closing tag, and the content together comprise the element.



Heading Tags



```
<h1> This is heading 1 </h1>  
<h2> This is heading 1 </h2>  
<h3> This is heading 1 </h3>  
<h4> This is heading 1 </h4>  
<h5> This is heading 1 </h5>  
<h6> This is heading 1 </h6>
```

This is heading 1

This is heading 2

This is heading 3

This is heading 4

This is heading 5

This is heading 6

Not only document starts with a heading but HTML also.

There are six levels of headings, which use the elements `<h1>`, `<h2>`, `<h3>`, `<h4>`, `<h5>`, and `<h6>`.

While heading tags called, browser adds one line before and one line after that.

Paragraph, Break, & Comment Tag



```
<p>Ini paragraf 1.</p>
```

```
<p>Ini paragraf 2.</p>
```


```
<hr/>
```

```
Ini baris 1 <br/>
```

```
Ini baris 2
```

```
<!-- Ini komentar -->
```

- `<p>` represent for paragraph tag. Text, number or even symbol could be put inside this tags
- `<hr>` used to add a horizontal line
- `
` this tags used to break the line.
- `<!-- -->` used as comment. Content could be put inside the tag and won't show in HTML page



```
<hr/>
<b> Ini bold </b><br/>
<strong>Ini strong</strong><br/>
<i>Ini italic</i><br/>
<em>Ini emphasized</em><br/>
<mark>Ini marked</mark><br/>
<small>Ini small</small><br/>
<del>Ini deketed</del><br/>
<ins>Ini inserted</ins><br/>
<sub>Ini subscript</sub><br/>
<sup>Ini superscript</sup><br/>
<hr/>
```

Here is several tags that could be used in order to modify content inside the tag.

Unordered List Tag



```
<h1>Daftar Belanja:</h1>
<ul>
  <li>Beras</li>
  <li>Minyak Goreng</li>
  <li>Gula</li>
  <li>Santan</li>
</ul>
```

Used to create bullets. Started with `` tag to define the unordered list. To define the list, put `` inside `` element.

Ordered List



```
<h1>Ranking kelas Puwadhika:</h1>  
<ol>  
  <li>Andi</li>  
  <li>Budi</li>  
  <li>Caca</li>  
  <li>Dedi</li>  
</ol>
```

Used to create numbering. Ordered list will show a list using a numbers. Similar to unordered list, but in order to use this, put `` at the beginning of element.

HTML Table

```
<table>
  <tr>
    <th>Nama</th>
    <th>TTL</th>
  </tr>
  <tr>
    <td>Galih</td>
    <td>Jakarta, 25 Jan 1990</td>
  </tr>
</table>
```

HTML table contain of:

1. <table>
2. <tr>
3. <th> or <td>

Column Span



```
<table>
  <tr>
    <th>Nama</th>
    <th>TTL</th>
  </tr>
  <tr>
    <td colspan="2">
      <center>Galih</center>
    </td>
  </tr>
</table>
```

Row Span

```
<table>
  <tr>
    <th rowspan="2">Jakarta</th>
    <td>A</td>
    <td>B</td>
  </tr>
  <tr>
    <td>C</td>
    <td>D</td>
  </tr>
</table>
```


Absolute Links

```
<p>
  <a href="https://www.google.com">Google</a>
</p>
<p>
  <a href="https://www.yahoo.com" target="_blank">
    Yahoo
  </a>
</p>
```

Relative Links

```
<p>
  <a href="satu.html">Halaman satu</a>
</p>
<p>
  <a href="/index.html" target="_blank">
    Halaman Index
  </a>
</p>
```

Images Tag



```
  
  

```

 tag used to insert image into the page. Source must have extension file



```
<figure>
  
  <figcaption>Ini singa</figcaption>
</figure>
```

Used to add a caption into an image assets

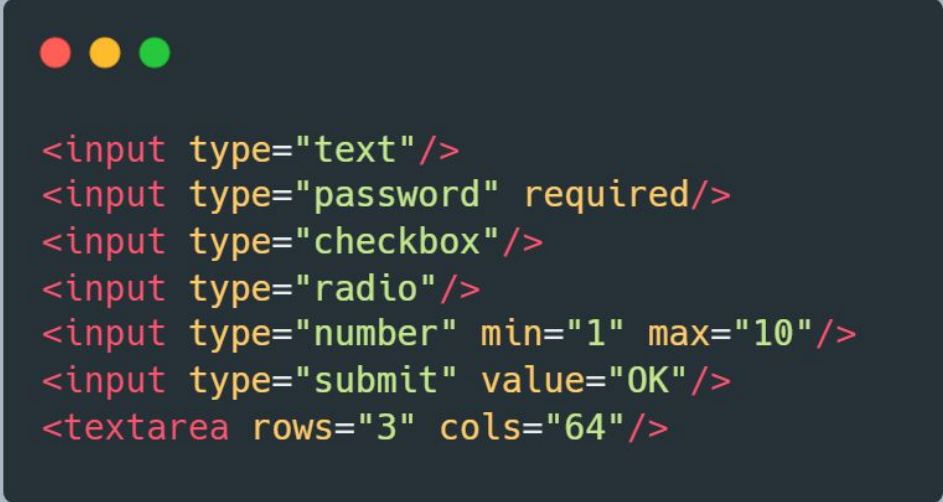
Forms HTML

```

<!DOCTYPE html>
<html>
  <head>
    <title>Forms</title>
  </head>
  <body>
    <form action="" method="post">
      <!-- insert form element -->
    </form>
  </body>
</html>

```

Form could contain several inputs and buttons



```
<input type="text"/>
<input type="password" required/>
<input type="checkbox"/>
<input type="radio"/>
<input type="number" min="1" max="10"/>
<input type="submit" value="OK"/>
<textarea rows="3" cols="64"/>
```

`<input>` tags have several attributes that could be used depends on functionality.

Button Tag



```
<button type="button">Ok</button>  
<button type="button" onclick="doSomething( );">  
  Apply  
</button>
```

<button> tag used to interact between user
and web page

Select Forms



```
<p>Apa warna favoritmu?</p>
<select>
  <option>Merah</option>
  <option>Biru</option>
  <option>Kuning</option>
</select>
<p>Apa makanan favoritmu?</p>
<select>
  <option>Telur gulung</option>
  <option selected>Cilok</option>
  <option>Bakso</option>
</select>
```

Label, Fieldset, & Legend Tag

```
<p>
  <label for="user">Username</label>
  <input type="text" name="user" />
</p>

<p>
  <label for="pass">Password</label>
  <input type="password" name="pass" />
</p>
```

```
<fieldset>
  <legend>Data Diri</legend>
  <p>
    <label for="nama">Nama:</label>
    <input type="text" name="nama" />
  </p>

  <p>
    <label for="usia">Usia:</label>
    <input type="number" name="usia" />
  </p>
</fieldset>
```

Usually, <label> tag is used along with input. <legend> tag also could be used like label tag.

Division Tag



```
<!DOCTYPE html>
<html>
  <head>
    <title>Division</title>
  </head>
  <body>
    <div>Division 1</div>
    <div>Division 2</div>
    <div>Division 3</div>
  </body>
</html>
```

The div tag represents a generic container, because it defaults to a block. As a block, it starts on its own new line, similar to how <p> tags work.

Find out more on,

<https://developer.mozilla.org/en-US/docs/Learn/HTML/Cheatsheet>

What is Cascading Style Sheets?

CSS (Cascading Style Sheets) describes how HTML elements are to be displayed on screen, paper, or in other media.

CSS saves a lot of work. It can control the layout of multiple web pages all at once. External stylesheets are better stored in CSS files separately.



How to Write CSS?

There are several ways to write down css:

- Inline Styles
- Internal Styles
- External Styles



```
<!DOCTYPE html>
<html>
  <head>
    <title>CSS Styles</title>
  </head>
  <body>
    <p style="color: green">Hola 😊</p>
  </body>
</html>
```



```
<!DOCTYPE html>
<html>
  <head>
    <title>CSS Styles</title>
    <style>
      h2 { color: green; }
    </style>
  </head>
  <body>
    <p>Hola 😊</p>
  </body>
</html>
```

External Styles

index.html

```

<!DOCTYPE html>
<html>
  <head>
    <title>CSS Styles</title>
    <link rel="stylesheet" type="text/css"
href="style.css" />
  </head>
  <body>
    <p>Ini Paragraf.</p>
  </body>
</html>

```

style.css

```

p { color: blue }

```



```
<!DOCTYPE html>
<html>
  <head>
    <style>
      h2 { color: orange; }
      p { color: red; }
      .mobil { color: blue; }
      #avanza { color: greenyellow; }
    </style>
  </head>
  <body>
    <h2>Halo</h2>
    <p>Hai</p>
    <p class="mobil" id="avanza">Ini Avanza</p>
    <p class="mobil" id="alya">Ini Alya</p>
  </body>
</html>
```

Selector used to tell which element would given style through CSS.

`h2`, `p`, `.mobil` & `#avanza` are called *Selectors*.

`{color: orange;}` are *Property* & *Value*.

To select an element to style, simply:

- call its **tag** e.g `h2 {color: orange;}`
- call its **class** e.g `.mobil {color: blue;}`
- call its **id** e.g `#avanza {color: green;}`

```

<!DOCTYPE html>
<html>
  <head>
    <style>
      a[href] {
        color: red;
      }
    </style>
  </head>
  <body>
    <p><a href="#">Jaya jaya jaya !</a></p>
  </body>
</html>

```

Color

```
<!DOCTYPE html>
<html>
  <head>
    <style>
      body {
        background-color: yellow;
      }

      h1 {
        color: rgb(0, 0, 255);
      }

      h2 {
        color: #00ff00;
      }

      p {
        color: hsl(360, 100%, 75%);
      }
    </style>
  </head>
  <body>
    <h1>Selamat datang 🇮🇩</h1>
    <h2>Purwadhika 🌟</h2>
    <p>Digital Technology School 🖥️</p>
  </body>
</html>
```

Color used to change the color of the text

Red, Green, Blue Color Values

■ {color: **rgb(0, 0, 255);**}

Hexadecimal Value

■ {color: **#00FF00;**}

Hue, Saturation, Lightness Value

■ {color: **hsl(360, 100%, 75%);**}

Alpha Transparency

■ {color: **rgba(0, 0, 255, 0.782);**}

■ {color: **hsla(360, 100%, 75%, 0.5);**}

There are several ways to choose the palette color

Background Color

```

<head>
  <style>
    body {
      background-color: blue;
      /* option 1 */
      background: linear-gradient(blue,
yellow);
      /* option 2 */
      background: linear-gradient(90deg,
blue, yellow);
      /* option 3 */
      background: linear-gradient(blue,
yellow);
    }
  </style>
</head>
<body> 🚀 </body>

```

Background color would give a color into the whole content on the tag

Background Image

```
<!DOCTYPE html>
<html>
  <head>
    <style>
      body {
        background-color: lightgray;
        background: url("lin.jpg");
        background-position: left top;
        background-size: 1280px 720px;
      }
    </style>
  </head>
  <body></body>
</html>
```

Not only color, but also image could be set as a background for content inside the tag



```
<head>
  <style>
    p {
      font-family: "Impact", Arial;
      font-style: italic;
      text-transform: uppercase;
      text-decoration: line-through;
      text-shadow: -4px 4px 4px red;
      line-height: 50%;
      letter-spacing: 2px;
      word-spacing: 4px;
      text-align: left;
      text-indent: 2rem;
    }
  </style>
</head>

<body>
  <p>Halo kamu ❤️</p>
</body>
```

font-family (*web safe fonts*):

Arial, Helvetica, Times New Roman, Times, Courier New, Courier, Verdana, Georgia, Palatino, etc.

font-style:

normal, italic, oblique

text-transform:

capitalize, uppercase, lowercase, none

text-decoration:

underline, overline, line-through, wavy, none

text-align:

left, center, right

Width & Height

```
<head>
  <style>
    .konten {
      background-color: pink;
      width: 900px;
      height: 100px;
    }
  </style>
</head>

<body>
  <div class="konten">Halo Semuanya!</div>
</body>
```

Width and **height** used to define the size of class named as **konten**

Absolute Lengths

px: the unit for pixels

pt: the unit for points

cm: the unit for centimeters

mm: the unit for millimeters

in: the unit for inches

pc: the unit for picas

Relative Lengths

%: the unit for percentages

em: relative to current font size

rem: relative to current font size on the element

vw: relative to the width of viewport divided by 100

vh: relative to the height of viewport/100

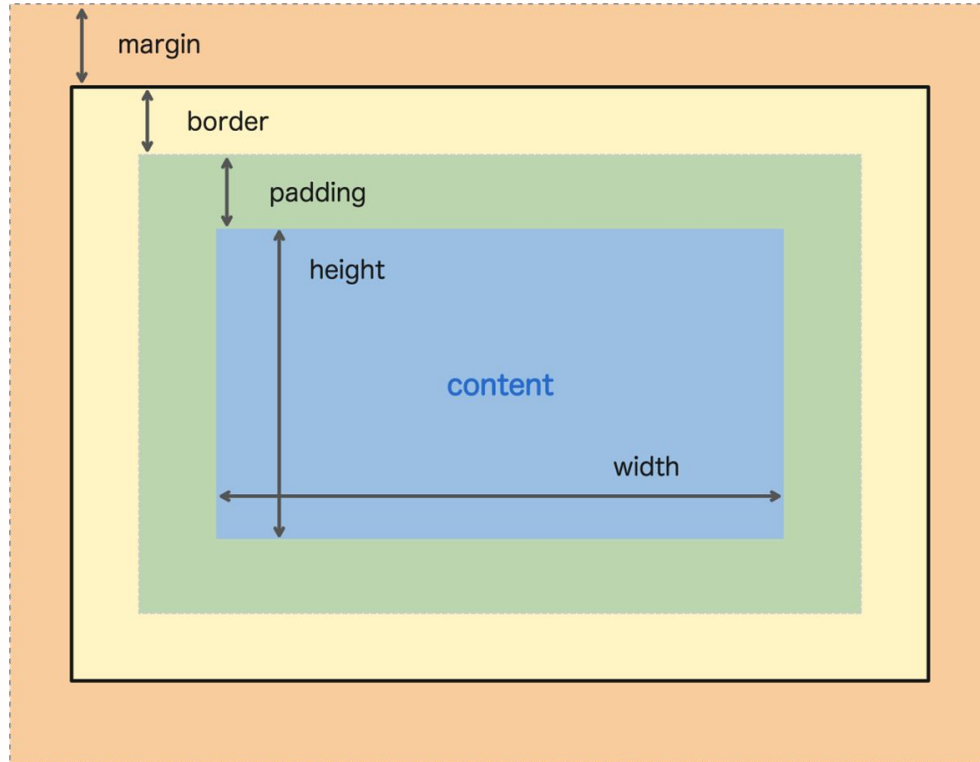
vmin: relative to the smaller viewport's dimension/100

vmax: relative to the larger viewport's dimension/100

ch: relative to 0

ex: relative to the x-height of font

Margin, Border, & Padding



Margin

```
<head>
<style>
  div {
    background-color: lightblue;
    width: 900px;
    height: 500px;
    margin-top: 200px;
    margin-right: 200px;
    margin-bottom: 200px;
    margin-left: 200px;
  }
</style>
</head>

<body>
  <div>
    <h1>Contoh Margin</h1>
  </div>
</body>
```

Here is shorthand to write down margin

margin: 200px;

*this can be used if all margin position have same value

margin: 200px 150px;

*the first value will represent top and bottom, the second one represent left and right

margin: 200px 100px 150px 80px;

*if every position has different value, use this shorthand. This represent value from top, right, bottom, and left

```
<head>
  <style>
    div {
      background-color: lightblue;
      width: 900px;
      height: 500px;
    }
    h1 {
      color: white;
      background-color: blue;
      padding: 25px;
      border: 20px ridge yellow;
      border-radius: 10px;
      box-shadow: -0.5rem 0.5rem 1rem gray;
    }
  </style>
</head>

<body>
  <div>
    <h1>Contoh Border</h1>
  </div>
</body>
```

Border style

solid, dotted, dashed, double,
inset, outset, groove, ridge

Padding

```
<head>
  <style>
    div {
      background-color : lightblue;
      width : 900px;
      height : 500px;
    }
    h1 {
      color : white;
      background-color : blue;
      padding-top : 10px;
      padding-bottom : 20px;
      padding-left : 25px;
      padding-right : 30px;
    }
  </style>
</head>

<body>
  <div>
    <h1>
      Ini Padding
    </h1>
  </div>
</body>
```

Here is shorthand to write down padding

padding: 200px;

*this can be used if all padding position have same value

padding: 200px 150px;

*the first value will represent top and bottom, the second one represent left and right

padding: 200px 100px 150px 80px;

*if every position has different value, use this shorthand. This represent value from top, right, bottom, and left

Pseudo-class


A **CSS pseudo-class** is a keyword added to a selector that specifies a special state of the selected element(s). For example, **:hover** can be used to change a button's color when the user's pointer hovers over it.

For example, it can be used to:

Style an element when a user mouses over it

Style visited and unvisited links differently

Style an element when it gets focus



```
selector::pseudo-element {  
  property : value;  
}
```

Pseudo-class Example



```
/* unvisited link */
```

```
a:link {  
  color: #FF0000;  
}
```



```
/* mouse over link */
```

```
a:hover {  
  color: #FF00FF;  
}
```



```
/* visited link */
```

```
a:visited {  
  color: #00FF00;  
}
```



```
/* selected link */
```

```
a:active {  
  color: #0000FF;  
}
```


A **CSS pseudo-element** is a keyword added to a selector that lets you style a specific part of the selected element(s). For example, `::first-line` can be used to change the font of the first line of a paragraph.

A CSS pseudo-element is used to style specified parts of an element.

For example, it can be used to:

Style the first letter, or line, of an element

Insert content before, or after, the content of an element



```
selector::pseudo-element {  
  property : value;  
}
```


Pseudo-elements Example

```
p::first-line {  
  color: #ff0000;  
  font-variant: small-caps;  
}
```

The `::first-line` pseudo-element can only be applied to block-level elements.

The following properties apply to the `::first-line` pseudo-element:

- font properties
- color properties
- background properties
- word-spacing
- letter-spacing
- text-decoration
- vertical-align
- text-transform
- line-height
- clear

Exercise

Create a component with styling like example output:



Beautiful!

Modal with a custom image

Lovely!

Exercise

Create a table with data and styling like example output:

No.	Image	Name	Food	Group
1		SmilingCat	HealthyFish	<input checked="" type="radio"/> Vertebrate <input type="radio"/> Invertebrate
2		HappyDog	FreshMeat	<input checked="" type="radio"/> Vertebrate <input type="radio"/> Invertebrate
3		ConfuseSimpanse	SweetBanana	<input checked="" type="radio"/> Vertebrate <input type="radio"/> Invertebrate
4		FriskyJellyfish	Saltwaterplankton	<input type="radio"/> Vertebrate <input checked="" type="radio"/> Invertebrate
5		SwimingSquid	SmallFish	<input type="radio"/> Vertebrate <input checked="" type="radio"/> Invertebrate

Thank You!

