



Welcome 🎉

Front-End Bootcamp

Learn to build modern websites with:

- HTML → Structure
- CSS → Styling
- JavaScript → Interactivity



About Your Instructor



Name: Lahcen Nidhal

Background: Software Developer & infoSec student

Experience: more then 3 years with software dev

Passion: Teaching & helping students become front-end developers



What We'll Cover in This Bootcamp

Part 1: HTML

Part 2: CSS

Part 3: JavaScript

Projects after each module

Final capstone project



The Internet: How It Works

Internet = Global network of computers.

Works with:

- **Client (browser)** → sends request.
- **Server** → responds with data.

Uses **HTTP/HTTPS protocol**.

Example:

- You type `www.google.com` → Browser asks server → Server sends back HTML page → Browser renders it.

client



DNS

IP

HTTP Request

HTTP Response

Server





what is HTML?

HyperText Markup Language

Defines the structure of a webpage.


Not a programming language → a markup language.

Core of every website.

HTML Document Structure & Syntax

Every HTML document starts with:

html

 Copy code

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <title>My First Page</title>
  </head>
  <body>
    <h1>Hello World!</h1>
  </body>
</html>
```





Essential HTML Tags & Elements

Headings: `<h1>` to `<h6>`

Paragraphs: `<p>`

Text formatting: ``, ``, `<mark>`

Lists: ``, ``, ``

Div & Span: generic containers




Document Head & Meta Tags

Contains information **about** the document.

Examples:

html

 Copy code


```
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<meta name="description" content="Bootcamp website">
<title>Front-End Bootcamp</title>
```



Forms & Input Types

Form basics:

html

 Copy code

```
<form action="/submit" method="POST">
  <input type="text" name="username" placeholder="Enter name">
  <input type="email" name="email" required>
  <button type="submit">Submit</button>
</form>
```

Common input types: text, password, email, number, date, file, checkbox, radio, range, color.



Form Validation Attributes

Built-in attributes:

- `required` → field must be filled.
- `min`, `max`, `maxlength`
- `pattern` → regex validation.

Example:

html

```
<input type="email" required>  
<input type="text" pattern="[A-Za-z]{3,}">
```



Semantic HTML5 Elements

Meaningful tags → Improve readability & SEO.


Examples:

- `<header>` → page header
- `<nav>` → navigation links
- `<main>` → main content
- `<article>` → independent content
- `<section>` → grouped content
- `<footer>` → page footer

Media Elements

- Images: ``
- Audio:


html

 Copy code

```
<audio controls>
  <source src="sound.mp3" type="audio/mpeg">
</audio>
```

- Video:

html


 Copy code

```
<video controls>
  <source src="video.mp4" type="video/mp4">
</video>
```

Tables & Data Presentation

Table example:

html

 Copy code

```
<table border="1">
  <tr><th>Name</th><th>Email</th></tr>
  <tr><td>Nidal</td><td>nidal@mail.com</td></tr>
</table>
```


Use `<thead>`, `<tbody>`, `<tfoot>` for structure.

Links & Navigation

Basic link: `Go to Page`

Navigation:

html

 Copy code

```
<nav>
```

```
  <a href="index.html">Home</a>
```

```
  <a href="about.html">About</a>
```

```
</nav>
```



Project 1: Contact Form & Media Gallery

Build a contact form using:

- Text, email, date, and file inputs.
- Validation attributes.


Add a media gallery with images, audio, and video.

Accessibility Best Practices

Provide `alt` text for images.

Use labels for inputs:

html

 Copy code

```
<label for="email">Email:</label>
<input type="email" id="email">
```

Maintain color contrast.

Keyboard navigation support.




ARIA Attributes & Roles

ARIA = Accessible Rich Internet Applications.

Example:

html

 Copy code

```
<nav role="navigation" aria-label="Main navigation">
```

Use when native HTML is not enough.



SEO-Friendly HTML Structure

Use semantic tags (header, main, footer).

Use `<title>` and `<meta description>`.

Proper headings hierarchy (h1 → h6).

Meaningful link text (Read more about HTML instead of Click here).



HTML Debugging & Troubleshooting

Use **browser dev tools** (Inspect → Elements).

Check **console errors**.

Validate with W3C.

Common mistakes:

- Missing closing tags.
- Wrong nesting (e.g., `<p><div>...</p></div>`).



Project 2: Accessible Multi-Page Website

Create a **multi-page site** with:

- Navigation (<nav>).
- Semantic layout (header, main, footer).
- Accessibility features (labels, ARIA).
- SEO best practices.