

Lecture II:

JavaScript & React Part 1

August 30, 2024

Training Session Information (cont.)

Class Schedule

~~August 23, Lecture I: Intro to Web, HTML & CSS~~

August 30, Lecture II: JavaScript & React Part 1 <- TODAY!!!

September 11, Lecture III: JavaScript & React Part 2

September 18, Lecture IV: TailwindCSS, ChakraUI, Fonts/Icons

September 25, Lecture V: NextJS

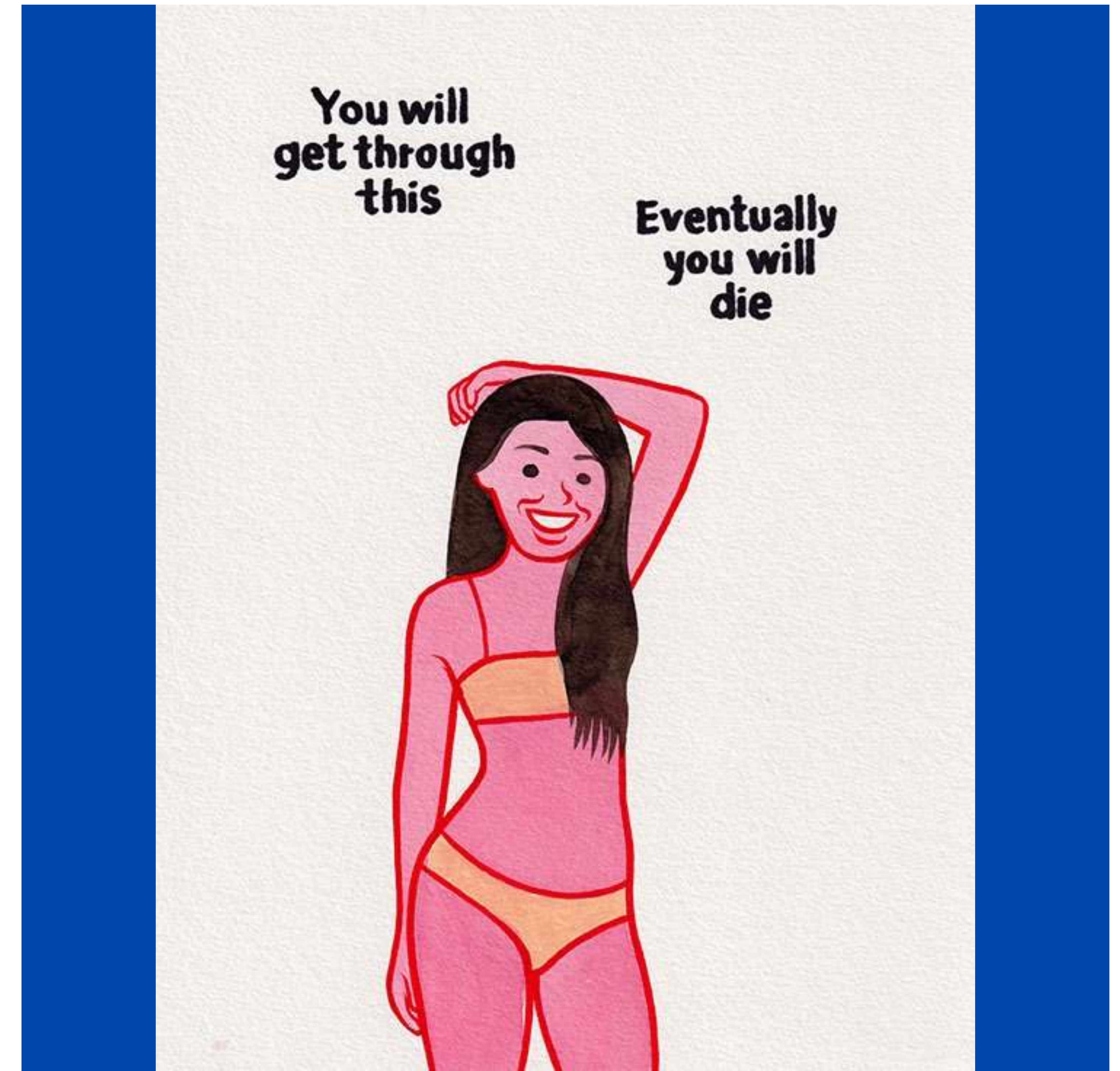
Classes will be held on Microsoft Teams

3 hours a class, 1PM - 4PM

Contents

HTML & CSS Review
JavaScript Basic Syntax
React Intro: Components, props
Hosting on Vercel (optional)

Note: we have rearranged to slow down the pace now

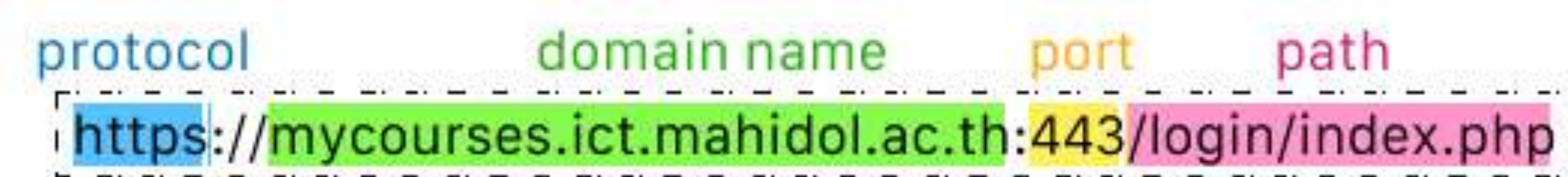


Week 1 Review

Week 1 Review

Week 1 Review

Key takeaways:



protocol domain name port path

`https://mycourses.ict.mahidol.ac.th:443/login/index.php`

- HTML used to render content on the webpage
- CSS used to style and decorate the webpage

Extra Tips:

- Type "!" in VSCode to autocomplete **boilerplate html**
- There are 100+ HTML tags and 200+ CSS styles, google them!

Week 1 Review - commonly used HTML tags & CSS properties

HTML Basic tags

- h1, h2, h3,
- ``
- `<div style="color: red;">...</div>`
- `Press me`
- `</br>`
- `<iframe src="https://ict.mahidol.ac.th" />`

HTML Form

`<form>`

`<input type='text'>Name</input>`

`<input type='text'>Name</input>`

`<input type='text'>Name</input>`

`<input type='text'>Name</input>`

`</form>`

Other inputs types: button, checkbox, select, date, file, email, password, etc.

Week 1 Review

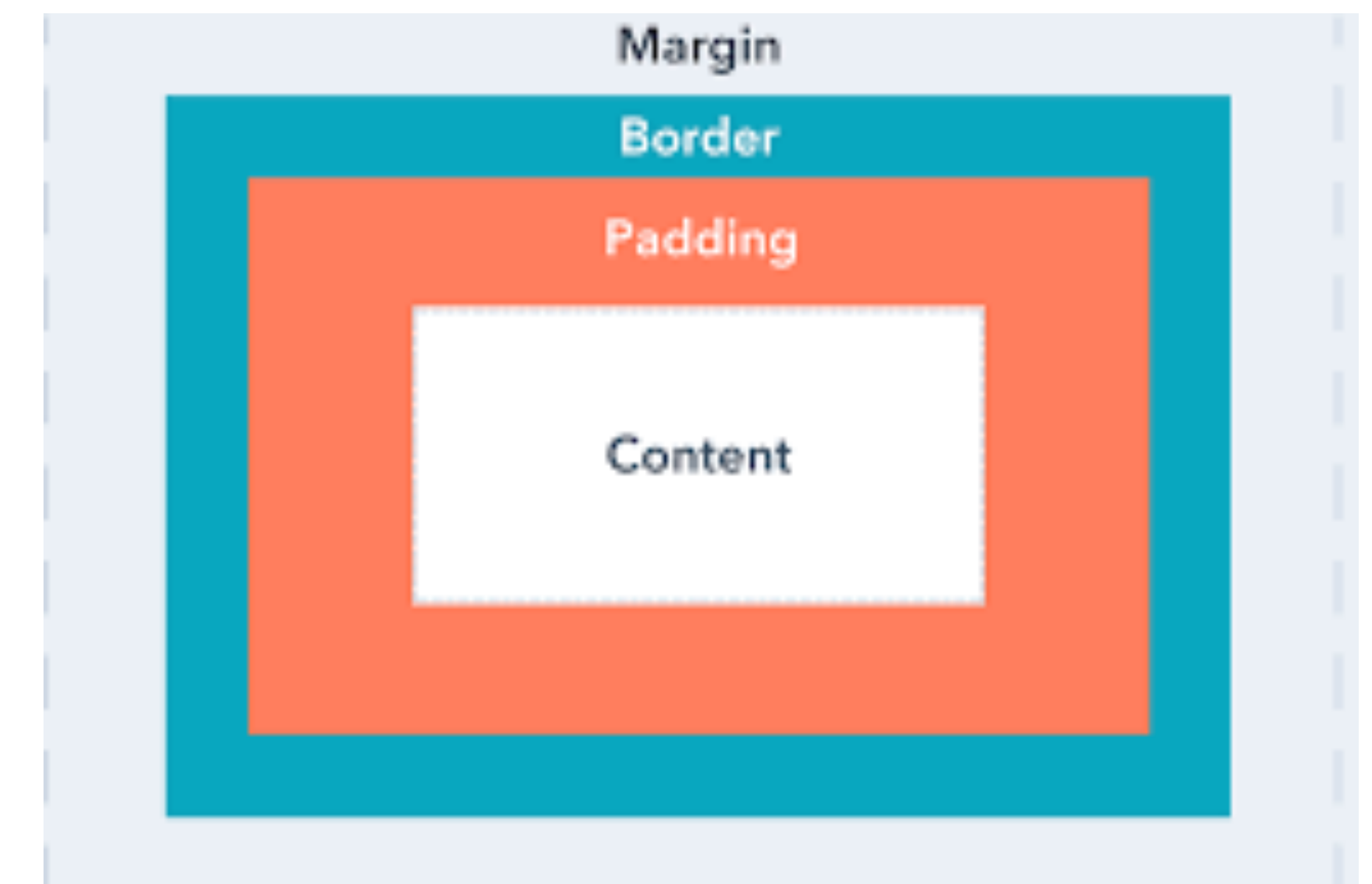
Week 1 Review - commonly used HTML tags & CSS properties

CSS

- `color: "green";`
- `background-color: "#00ff00";`
- `text-align: center;`
- `width: 50%`

CSS border, Margin, Padding

- `margin: 20px`
- `margin: 20px 10px 5px 2px;` **top right bottom left**
- `border: 2px solid black;`
- `padding: 20px;`



JavaScript Basic Syntax

JS Basic Syntax

What is JavaScript (JS)?

Think of JS as a **programming language** (like C/Python/Java) used to add logic to your frontend site.

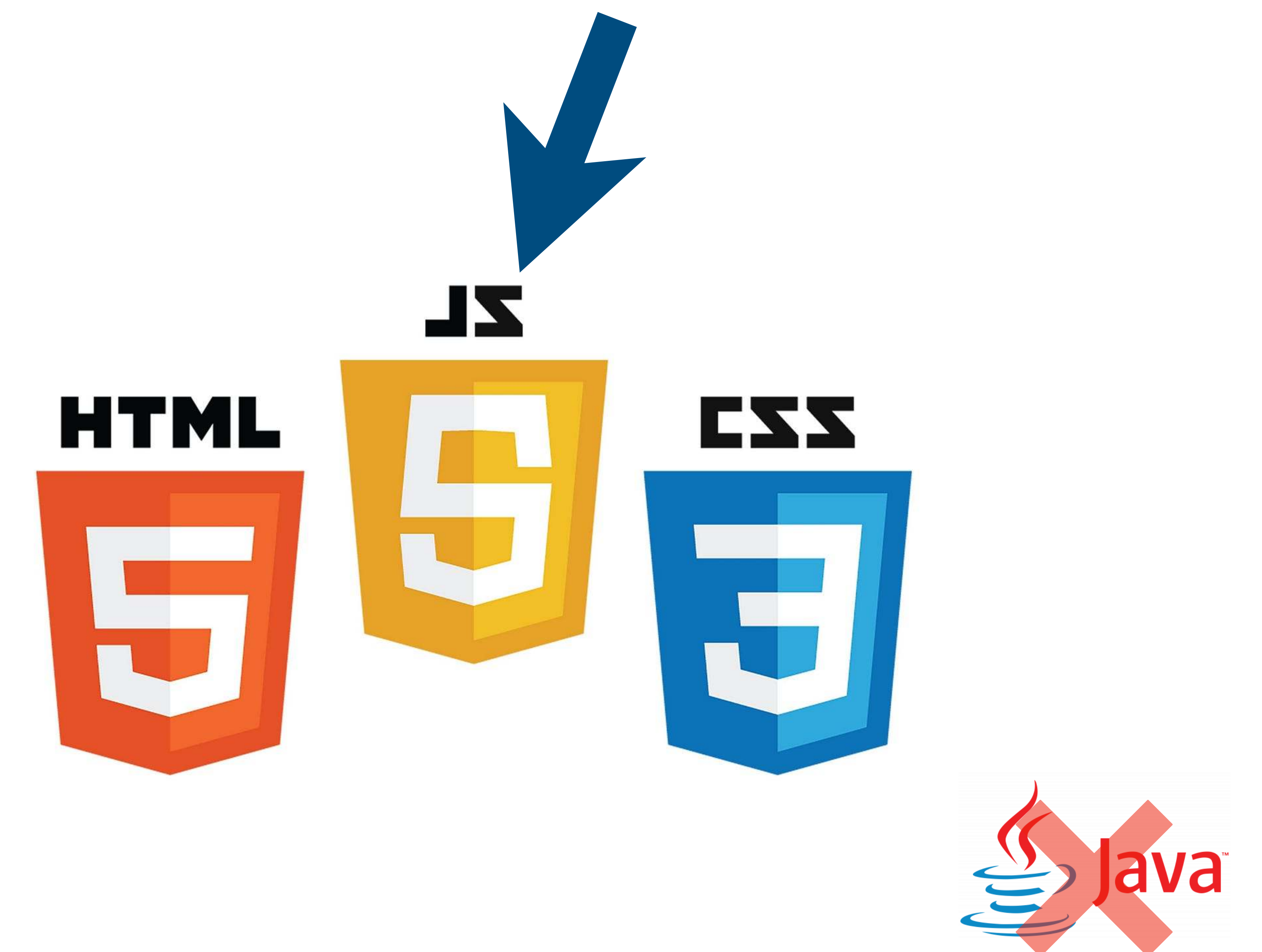
These include...

- What do we do once a **button is clicked**?
- **Sending/receiving data** from backend (via API)
- etc.

While **HTML & CSS** are mainly for **displays**

Other notes

- Don't confuse **JavaScript** with **Java**! They're different languages



JS Basic Syntax

Reviewing setup from the last lecture

If not done already:

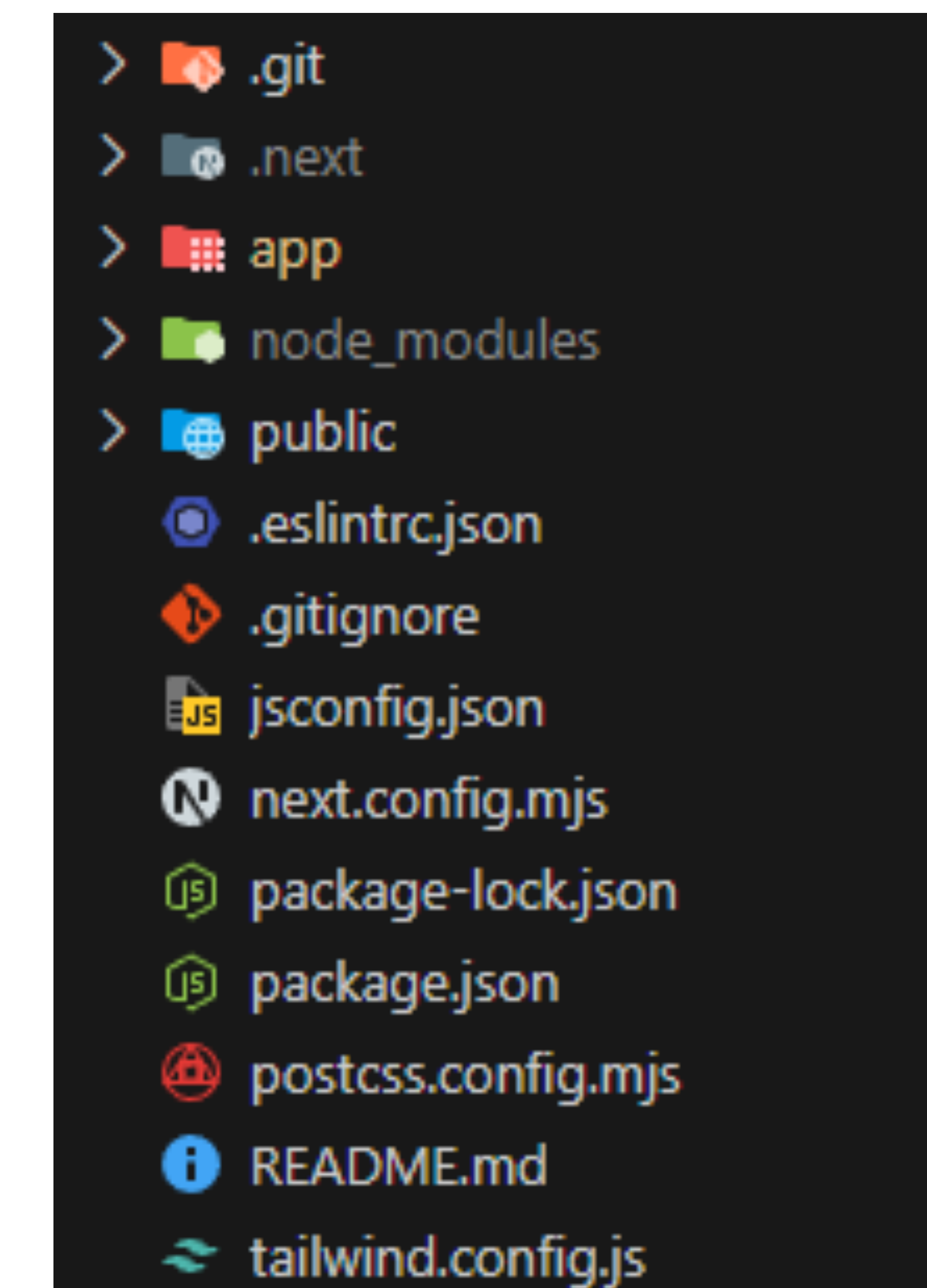
- Run "*npm install -g react react-dom next*" to install these packages globally **once**
- Download and install NodeJS (google)

1. To setup a new nextjs project: `npx create-next-app@latest`

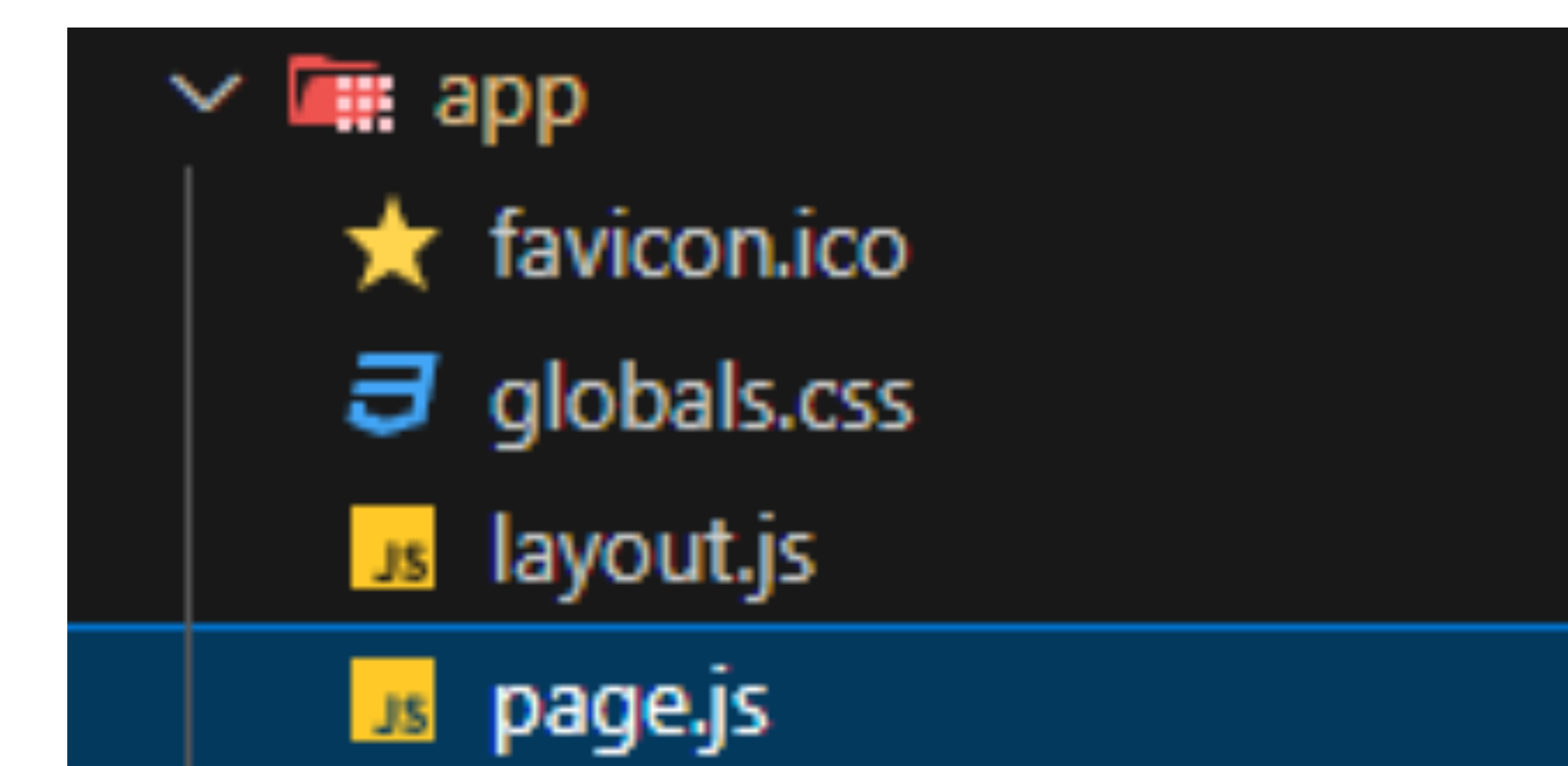
```
> npx create-next-app@latest
✓ What is your project named? ... week1-review
✓ Would you like to use TypeScript? ... No / Yes
✓ Would you like to use ESLint? ... No / Yes
✓ Would you like to use Tailwind CSS? ... No / Yes
✓ Would you like to use `src/` directory? ... No / Yes
✓ Would you like to use App Router? (recommended) ... No / Yes
✓ Would you like to customize the default import alias (@/*)? ... No / Yes
```


Week 1 Review - Setup NextJS project

2. Navigate to the project directory and open VSCode in that folder.



3. Open the file **app/page.js**, this is where we'll be coding today mainly



JS Basic Syntax

Reviewing setup from the last lecture

4. Open terminal, and run "npm run dev"
5. And open <http://localhost:3000> to open your default page.

```
> npm run dev
```

Extra Notes:

- To stop running, press **CTRL+C**
- Every time you save a file, changes will be made **automatically**

JS Basic Syntax (cont.)

Basic Syntax - variables & data types

Try playing with these (starting around line 3-4, don't erase other parts tho):

```
3  export default function Home() {  
4      let firstname = "john";  
5      var lastname = "mario";  
6      const age = 21;  
7  
8      if (age >= 18) {  
9          console.log("You are an adult");  
10     }  
11     else {  
12         console.log("You are not an adult");  
13     }  
14 }
```


JS Basic Syntax (cont.)

Basic Syntax - variables & data types

```
3  export default function Home() {  
4    let firstname = "john";  
5    var lastname = "mario";  
6    const age = 21;  
7  
8    if (age >= 18) {  
9      console.log("You are an adult");  
10   }  
11   else {  
12     console.log("You are not an adult");  
13   }  
14 }
```

JS is a dynamic language, so you don't need to declare int/char/float...and just use **let/var/const**

- let/var - can be modified
- const - **cannot** be modified

JS Basic Syntax (cont.)

Basic Syntax - loops (1/2)

Try adding these:

```
// While loop
let count = 0;
while (count < 5) {
  console.log("While loop count: " + count);
  count++;
}

// For loop
for (let i = 0; i < 5; i++) {
  console.log("For loop iteration: " + i);
}
```

Commonly used loops in JS:

- while loop

while (end condition)

- for loop

**for (define counter, end condition;
counter increment)**

JS Basic Syntax (cont.)

Basic Syntax - loops (2/2)

Try writing these:

```
let myArray = [10, 20, 30, 40];

myArray.forEach(number => {
  console.log(" " + number);
});

// 10 20 30 40
```


JS Basic Syntax (cont.)

Basic Syntax - loops (2/2)

Try writing these:

```
let myArray = [10, 20, 30, 40];

myArray.forEach(number => {
  console.log(" " + number);
});

// 10 20 30 40
```

An array is basically a group of data (in this case: 10, 20, 30, 40).

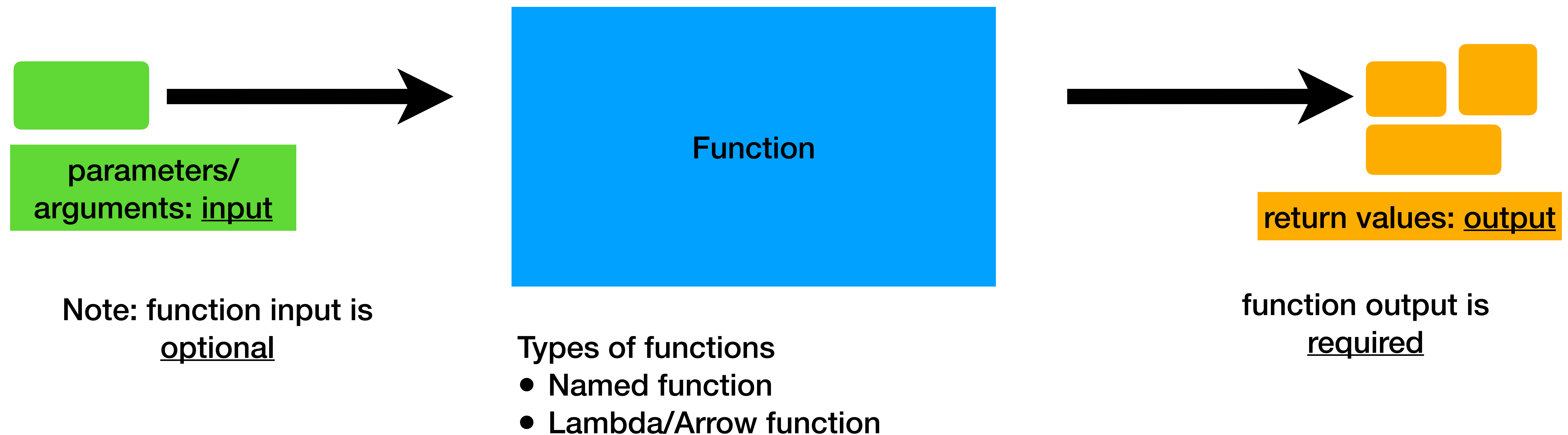
- **for-each loop**

for-each loop to go through the array's **element one by one**.

Here, it'll print 10 20 30 40

JS Basic Syntax (cont.)

Basic Syntax - functions



JS Basic Syntax (cont.)

Basic Syntax - named function

Try writing these:

```
function greet(person) {  
  ... return "Hello, " + person + "!";  
}  
  
console.log(greet("john mario"));  
// john mario
```

Function name

parameters/arguments:
input

return values: output

JS Basic Syntax (cont.)

Basic Syntax - lambda/arrow function

Try writing these:

Lambda/arrow functions don't have name

parameters/arguments:
input

```
let greet = (person) => {  
  return "Hello, " + person + "!";  
}  
  
console.log(greet("john mario"))
```

return values: output

Useful for passing **temporary unnamed** functions, "greet" in this case is just a variable name

JS Basic Syntax (cont.)

Basic Syntax - lambda/arrow function, more examples

Try writing these:

Lambda/arrow functions don't have name

```
// double each number in the array
```

```
const numbers = [1, 2, 3, 4, 5];
```

```
const doubledNumbers = numbers.map((x) => x * 2);
```

```
// now will be 2, 4, 6, 8, 10
```

return values: output

parameters/arguments:
input

React (NextJS) Intro: Components & Props

React (NextJS) Intro: Components & Props

What are React & NextJS?

React

- Frontend framework
- You'll likely learn this in Y2

NextJS

- Fullstack framework (both frontend & backend), based on react
- We'll mainly be using it for frontend here



React (NextJS) Intro: Components & Props

Try replacing all the HTML tags inside return (...)

```
3  export default function Home() {  
4    return (  
5      <>  
6      <h1>Hello fellow humans</h1>  
7      <p>lorem ipsum</p>  
8      </>  
9    );  
10 }
```

Hello fellow humans
lorem ipsum

The return statement starts around line 4, and ends around line 111 on the default template.

The return statement **only accepts 1 HTML tag**, so make sure to cover them like this:

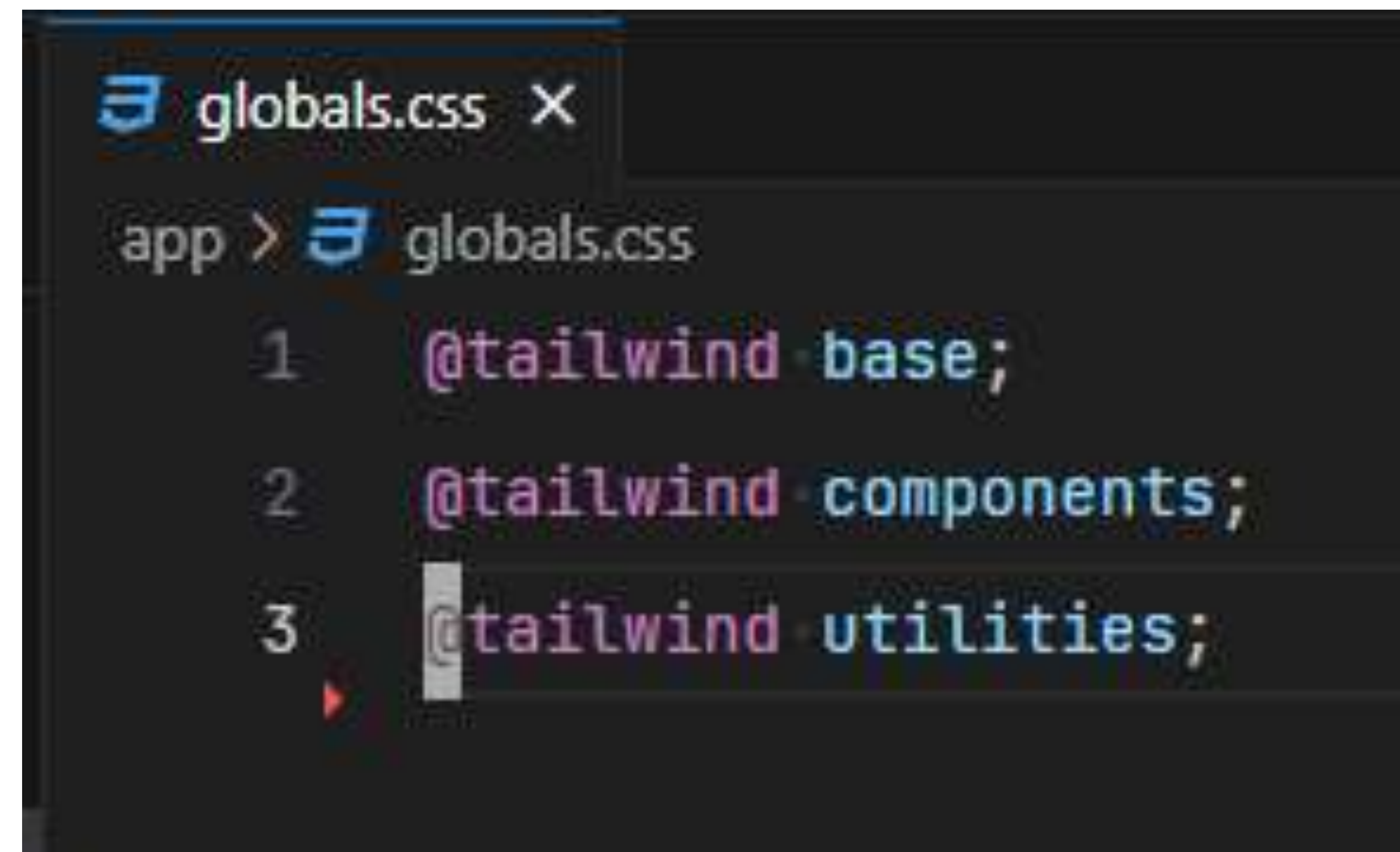
<>

all your tags here

</>

React (NextJS) Intro: Components & Props

Changing default global CSS styles



```
globals.css X
app > globals.css
1  @tailwind base;
2  @tailwind components;
3  @tailwind utilities;
```

Hello fellow humans
lorem ipsum

1. Go to "**app/global.css**" file
2. Clear everything in the file except the first three lines (image)
 - These are **TailwindCSS** directives, we'll explore more in later lectures
3. Save and now the black background should be gone.

React (NextJS) Intro: Components & Props

Changing default global CSS styles

```
globals.css M X
app > globals.css
1  @tailwind base;
2  @tailwind components;
3  @tailwind utilities;
4
5  h1 {
6    font-size: 50px;
7  }
8
9  p {
10   background-color: #555555;
11   color: white;
12   padding: 15px;
13   margin: 15px;
14   border-radius: 20px;
15 }
```

4. Feel free to customize styles like in the previous lectures in this **global.css** file (or inside the html tags as well)

Hello fellow humans

lorem ipsum

Components & Props

React (NextJS) Intro: Components & Props

Functional Components

You can define components once, and **reuse** them everywhere (similar to Java class or Figma components).

Every **changes will affect all** the other instances of the components too.

```
... <MyButton></MyButton>  
... <MyButton></MyButton>  
... <MyButton></MyButton>
```

We'll be writing these
next

React (NextJS) Intro: Components & Props

Functional Components

```
3 function MyButton() {  
4   return (  
5     <>  
6     <button type="button" style={{  
7       backgroundColor: "red",  
8       color: "white",  
9       padding: "20px",  
10      borderRadius: "20px",  
11    }}>Some Button Text</button>  
12   </>  
13   );  
14 }  
15
```

Here we defined MyButton component. Think of this like a blueprint.

Same Home "root" component from the previous slides.

```
16 export default function Home() {  
17   return (  
18     <>  
19     <h1>Hello fellow humans</h1>  
20     <p>lorem ipsum</p>  
21     <MyButton></MyButton>  
22     <MyButton></MyButton>  
23     <MyButton></MyButton>  
24   </>  
25   );  
26 }
```

Note

Notice that **style** has two brackets **{{...}}**. This is because we need to pass an object inside - JSON in this case.

Hello fellow humans

lorem ipsum

Some Button Text

Some Button Text

Some Button Text

React (NextJS) Intro: Components & Props

Functional Components

Clean things up by moving your component into another file.

Copy into a new file inside "app/" folder

```
MyButton.js U X
app > JS MyButton.js > ...
Codeium: Refactor | Explain | Generate JSDoc | X
1 export function MyButton() {
2   return (
3     <>
4     <button type="button" style={{
5       backgroundColor: "#008877",
6       color: "white",
7       padding: "20px",
8       borderRadius: "20px",
9     }}>Some Button Text</button>
10   </>
11 );
12 }
```

Import your component into your Home page

```
> JS page.js > ...
1 import { MyButton } from "../MyButton";
```


React (NextJS) Intro: Components & Props

Props

Similar to function arguments/parameters inputs.

Try to add this prop

```
1 export function MyButton({ buttonText }) {  
2   return (  
3     <>  
4       <button type="button" style={{  
5         backgroundColor: "#008877",  
6         color: "white",  
7         padding: "20px",  
8         borderRadius: "20px",  
9       }}>{buttonText}</button>  
10    </>  
11  );  
12 }
```

And pass in props to the components

```
<MyButton buttonText={"Login"}></MyButton>  
<MyButton buttonText={"Register"}></MyButton>  
<MyButton buttonText={"Forgot Password?"}></MyButton>
```

Hello fellow humans

lorem ipsum

Login

Register

Forgot Password?

React (NextJS) Intro: Components & Props

Props, more examples

Similar to function arguments/parameters inputs.

New textColor prop

```

1  export function MyButton({ buttonText, textColor }) {
2    return (
3      <>
4        <button type="button" style={{
5          backgroundColor: "#008877",
6          color: textColor,
7          padding: "20px",
8          borderRadius: "20px",
9        }}>{buttonText}</button>
10     </>
11   );
12 }
  
```

And pass in props to the components

```

<MyButton textColor={"red"} buttonText={"Login"}></MyButton>
<MyButton textColor={"green"} buttonText={"Register"}></MyButton>
<MyButton textColor={"#0000ff"} buttonText={"Forgot Password?"}></MyButton>
  
```

Hello fellow humans

lorem ipsum

Login

Register

Forgot Password?

**Hosting on Vercel
Demo
(if enough time left)**

In-class Assignments

ICT Mahidol Devclub

Please enter your basic information

Please enter your credentials

Register

Already have an account?

Login

Forgot Password?

Try making this :D

Make sure that you used **at least 2 components with props**

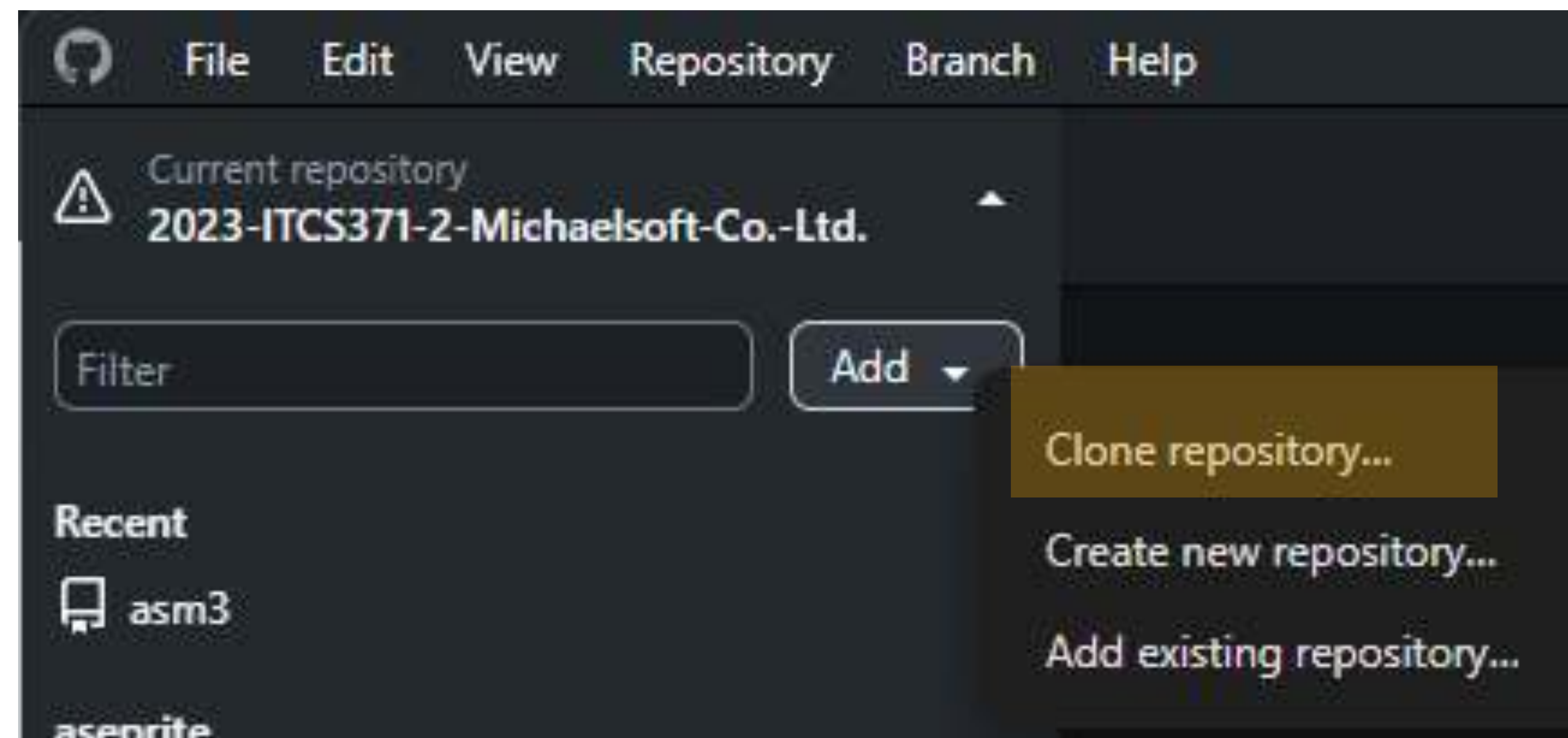
- **Submit URL from Vercel or your own domain!**

Appendix - Git & Github setup

1. Download git into your PC/laptop (<https://git-scm.com/downloads>)
2. Sign up on GitHub <https://github.com/>. GitHub is like a place where you **store your project (repository) and collaborate** based on git, while git is more of a tool to **locally** track your project's code version history.
3. Configure git in the terminal:
git config --global user.name "Your Name"
git config --global user.email "your.email@example.com"
4. Now try cloning a repository by typing into the terminal
git clone <https://github.com/CogWorksBWSI/GitPracticeRepo.git>

Appendix - Github Desktop setup (if you prefer GUI)

1. Download GitHub desktop from <https://github.com/apps/desktop>
2. Sign in to GitHub
3. To clone, click "clone repository"



Appendix - Github Desktop setup (if you prefer GUI)

4. You can find online public GitHub repositories and copy their HTTPS link, then paste them into GitHub desktop

