ASSIGNMENT-10

1. Explore all the ways of writing css.

We can write css by 3 ways

- inline css
- class based css -> where we create class names and then have css files to hold styles in that class
- using styles tag -> where we can write css as an object

2. How do we configure tailwind?

We can set up tailwind with parcel by following below

Link: https://tailwindcss.com/docs/guides/parcel

Install Tailwind CSS with Parcel - Tailwind CSS

once we set it up we will have tailwind.config.js file where we can configure tailwind.

We first have to Install Tailwind CSS in our project. npm install -D tailwindcss postcss npx tailwindcss init

here we install postcss because post css is a tool for transforming css with javascript.

`Configure PostCSS`

Create a .postcssrc file in your project root, and enable the tailwindcss plugin.

```
{
"plugins": {
"tailwindcss": {}
}
}
```

Configure your template paths - init command will create a tailwind.config.js file where we have to write the code below

```
content: [
"./src/**/*.{html,js,ts,jsx,tsx}",
],
```

Add the Tailwind directives to your CSS

```
@tailwind base;
@tailwind components;
```

```
@tailwind utilities;
```

Start your build process Start using Tailwind in your project .

3. In tailwind.config.js, what does all the keys mean (content, theme, extend, plugins)?

By default, Tailwind will look for an optional tailwind.config.js file at the root of your project where you can define any customizations.

Content – The `content` section is where you configure the paths to all of your HTML templates, JS components, and any other files that contain Tailwind class names.

```
module.exports = {
content: [
    './pages/**/*.{html,js}',
    './components/**/*.{html,js}',
    ],
    // ...
}
```

Theme – The `theme` section is where you define your color palette, fonts, type scale, border sizes, breakpoints — anything related to the visual design of your site.

module.exports = { // ... theme: { colors: { 'blue': '#1fb6ff', 'purple': '#7e5bef', 'pink': '#ff49db', 'orange': '#ff7849', 'green': '#13ce66', 'yellow': '#ffc82c', 'gray-dark': '#273444', 'gray': '#8492a6', 'gray-light': '#d3dce6', }, fontFamily: { sans: ['Graphik', 'sans-serif'], serif: ['Merriweather', 'serif'], }, extend: { spacing: { '8xl': '96rem', '9xl': '128rem', }, borderRadius: { '4xl': '2rem', } } } }

Plugins – The `plugins` section allows you to register plugins with Tailwind that can be used to generate extra utilities, components, base styles, or custom variants.

```
module.exports = {
// ...
plugins: [
    require('@tailwindcss/forms'),
    require('@tailwindcss/aspect-ratio'),
    require('@tailwindcss/typography'),
    require('tailwindcss-children'),
],
}
```

Extending the default theme - If you'd like to preserve the default values for a theme option but also add new values, add your extensions under the extend key in the theme section of your configuration file. For example, if you wanted to add an extra breakpoint but preserve the existing ones, you could extend the screens property:

```
module.exports = {
     theme: {
          extend: {
          // Adds a new breakpoint in addition to the default breakpoints
               screens: {
                    '3xl': '1600px',
               }
          }
     }
}
```

4. Why do we have .postcssrc file?

PostCSS is a build tool that can take a CSS file and convert it to an abstract syntax tree. PostCSS transforms your styles using JavaScript plugins. It is the CSS post processor that will compile the Tailwind assets into CSS. We are to create a `.postcssrc` file, which will contain the PostCSS settings. PostCSS will read all tailwind css files, scan for classes that are actually in use and only compile those. All unused classes will not be compiled, which will give you a well optimized CSS file.

Basically we need to tell our parcel that we are using tailwind class so there is need to convert them into normal css so that browser can read it.

In the .postcssrc file we need to configure tailwindcss, to tell our bundler (parcel) that while bundling things up we will be using tailwind, so need to compile the tailwindcss into normal css.