WEB APP DEVELOPMENT WITH REACTJS (INT252)

Lecture 9: Styling in React (CSS & Tailwind Basics)

Unit II – Component Design & Styling

Syllabus Mapping:

- Styling React components
- CSS in React
- Introduction to Tailwind CSS

Target Learner: Beginner (HTML, CSS, JavaScript knowledge)

Why Styling in React Is Different

In normal HTML:

- CSS is global

In React:

- Components are independent
- Styling must be controlled carefully

Ways to Style React Components

React supports multiple styling approaches:

1. External CSS files
2. Inline styles
3. CSS Modules
4. Utility frameworks (Tailwind CSS)

Method 1: External CSS File

Same as HTML, but imported into component

External CSS Example

App.css

```
.title {
  color: blue;
  text-align: center;
}
```

Using CSS in Component

```
import './App.css';

function App() {
  return <h1 className="title">Hello React</h1>;
}
```

Why `className` Instead of `class`

- `class` is reserved in JavaScript
- React uses `className`

Method 2: Inline Styling

Styles written as JavaScript objects

Inline Style Example

```
function App() {
  const style = {
    color: 'red',
    fontSize: '24px'
  };

  return <h1 style={style}>Inline Style</h1>;
}
```

Inline Styling Rules

- Properties are camelCase
- Values are strings or numbers

Pros & Cons of Inline Styles

✅ Dynamic styles

❌ No hover, media queries

Method 3: CSS Modules (Concept)

CSS scoped to one component

```
/* Button.module.css */
.btn { color: green; }
```

Using CSS Module

```
import styles from './Button.module.css';

<button className={styles.btn}>Click</button>
```

Why CSS Modules

- Avoid class conflicts
- Cleaner large projects

Method 4: Tailwind CSS (Modern Approach)

Tailwind = Utility-first CSS framework

Why Tailwind Is Popular

- No custom CSS files
- Fast UI building
- Consistent design

Tailwind Setup (Vite)

Steps:

1. Install Tailwind

```
npm install tailwindcss @tailwindcss/vite
```

2. Configure files update respective line

```
my-react-app\vite.config.js
import tailwindcss from '@tailwindcss/vite'
plugins: [react(),tailwindcss(),],
```

3. Add Tailwind directives

(Setup covered conceptually here)

```
src\index.css

@import "tailwindcss";
```

4. Now use it in your code

```
  <h1 class="text-3xl font-bold underline">
    Hello world!
  </h1>
```

Tailwind Example

```
function App() {
  return (
    <h1 className="text-blue-500 text-2xl font-bold">
      Tailwind Styled Text
    </h1>
  );
}
```

Reading Tailwind Classes

```
text-blue-500 → color
text-2xl      → font size
font-bold     → font weight
```

Comparing Styling Methods

Method	Use Case
CSS File	Small apps
Inline	Dynamic styles
Modules	Medium apps
Tailwind	Modern scalable apps

Common Beginner Mistakes

❌ Using `class` instead of `className`
❌ Forgetting CSS import
❌ Mixing inline and CSS badly

Practice Exercises

1. Style a heading using CSS file
2. Style a button using inline style
3. Use Tailwind to center text

✅ Answers – Practice Exercises

1. CSS File

```
.heading { color: purple; }
```

```
<h1 className="heading">Hello</h1>
```

✅ Answers – Continued

2. Inline Style

```
<button style={{ backgroundColor: 'green', color: 'white' }}>
  Click
</button>
```

✅ Answers – Continued

3. Tailwind Center Text

```
<h1 className="text-center">Centered</h1>
```

Key Takeaways

- React supports multiple styling methods
- `className` replaces `class`
- Inline styles are JS objects
- Tailwind speeds up UI development

Next Lecture

Lecture 10: State & useState Hook

Unit III – State & Hooks