### 🎨 Understanding the Need for Styling React Components

Styling is essential to make React components:

* **Visually appealing** and user-friendly
* **Responsive** across devices and screen sizes
* **Consistent** with branding and design systems
* **Interactive**, using hover effects, transitions, and animations
* **Modular**, allowing styles to be scoped to individual components

Without styling, components are just functional blocks with no visual hierarchy or user engagement.

### 🧵 Working with CSS Modules

CSS Modules allow **locally scoped styles**, avoiding global conflicts.

#### ✅ Benefits:

* Class names are **automatically scoped** to the component
* Prevents **naming collisions**
* Easier to **maintain and reuse styles**

#### 🛠️ Steps:

1. Create a CSS file with .module.css extension:

/\* Button.module.css \*/

.button {

background-color: #3498db;

color: white;

padding: 10px 20px;

border: none;

border-radius: 5px;

}

1. Import and use in your component:

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

function Button() {

return <button className={styles.button}>Click Me</button>;

}

### 🖍️ Working with Inline Styles

Inline styles are defined as **JavaScript objects** and applied directly to JSX elements.

#### Benefits:

* Styles are **component-scoped**
* Easy to apply **dynamic styles** using props or state
* No need to manage external CSS files

#### 🛠️ Example:

function Button() {

const buttonStyle = {

backgroundColor: '#e74c3c',

color: 'white',

padding: '10px 20px',

border: 'none',

borderRadius: '5px'

};

return <button style={buttonStyle}>Click Me</button>;

}