Purwadhika
Digital Technology School

Full Stack AI Software Development

Using CSS Frameworks

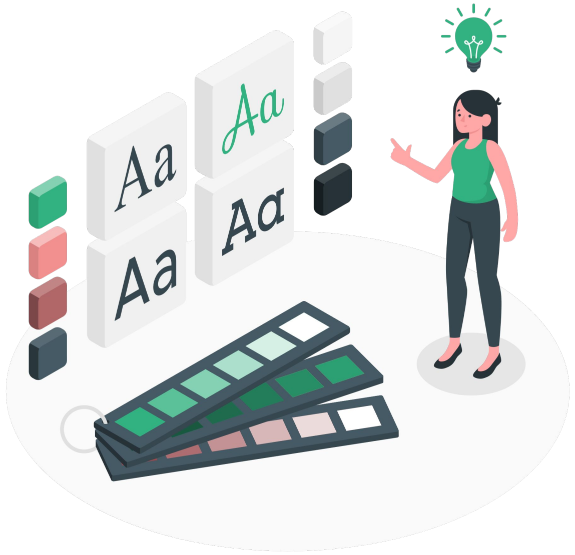
Job Connector Program

CSS Frameworks in React

CSS frameworks and component libraries are essential for building consistent, accessible, and responsive user interfaces (UI) in React. They abstract away repetitive styling tasks.

Why Use Them?

- **Speed:** Accelerate development with pre-built styles or components.
- **Consistency:** Maintain a unified look and feel across the application.
- **Accessibility:** Many frameworks include built-in features for better ARIA and keyboard navigation.



CSS Frameworks in React

The Three Approaches We'll Compare:

- **Tailwind CSS**: Utility-First Framework.
- **Material UI (MUI)**: Opinionated Component Library.
- **shadcn/ui**: Component Blueprint built with Radix and Tailwind.

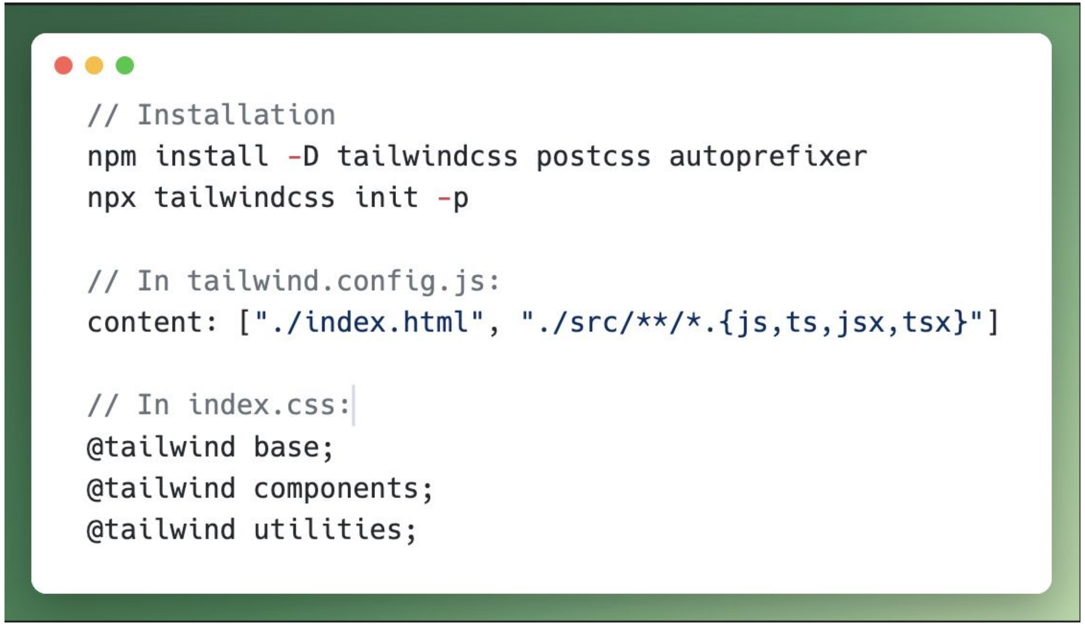
Tailwind CSS

Tailwind CSS is **not a component library**; it's a **utility-first CSS framework**. It provides thousands of low-level classes that you compose directly in your HTML (JSX) to build any design. Instead of pre-designed buttons, you apply classes like ***bg-blue-500***, ***text-white***, and ***rounded-lg*** to create your own button.

Key Characteristics:

- Customization: Offers maximum flexibility. You have complete control over every pixel.
- Bundle Size: Optimized using PostCSS tools like PurgeCSS to remove unused styles, resulting in a small production CSS file.
- Learning Curve: Requires learning the naming convention for its vast utility classes.

Tailwind CSS

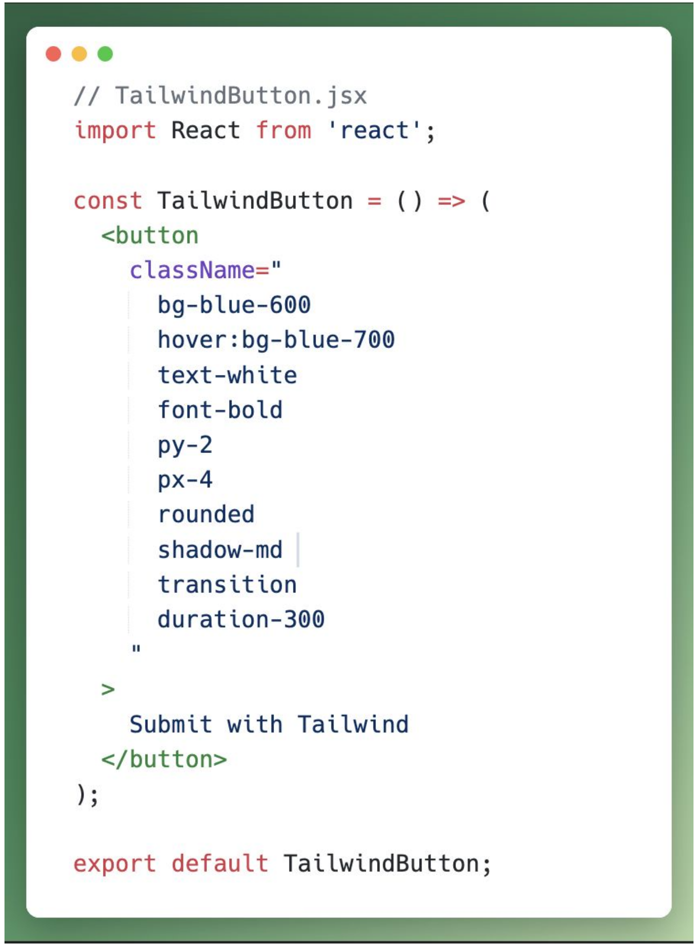


```
// Installation
npm install -D tailwindcss postcss autoprefixer
npx tailwindcss init -p

// In tailwind.config.js:
content: ["./index.html", "./src/**/*.{js,ts,jsx,tsx}"]

// In index.css:
@tailwind base;
@tailwind components;
@tailwind utilities;
```

[Get started with Tailwind CSS]()

Tailwind CSS

Code Example:



```jsx
// TailwindButton.jsx
import React from 'react';

const TailwindButton = () => (
  <button
    className="
      bg-blue-600
      hover:bg-blue-700
      text-white
      font-bold
      py-2
      px-4
      rounded
      shadow-md
      transition
      duration-300
    "
  >
    Submit with Tailwind
  </button>
);

export default TailwindButton;
```

shadcn/ui

shadcn/ui is a new, growing approach. It is not a traditional component library you install as a dependency. Instead, **it provides a collection of beautifully designed, accessible components** built using Radix UI primitives and styled with Tailwind CSS.

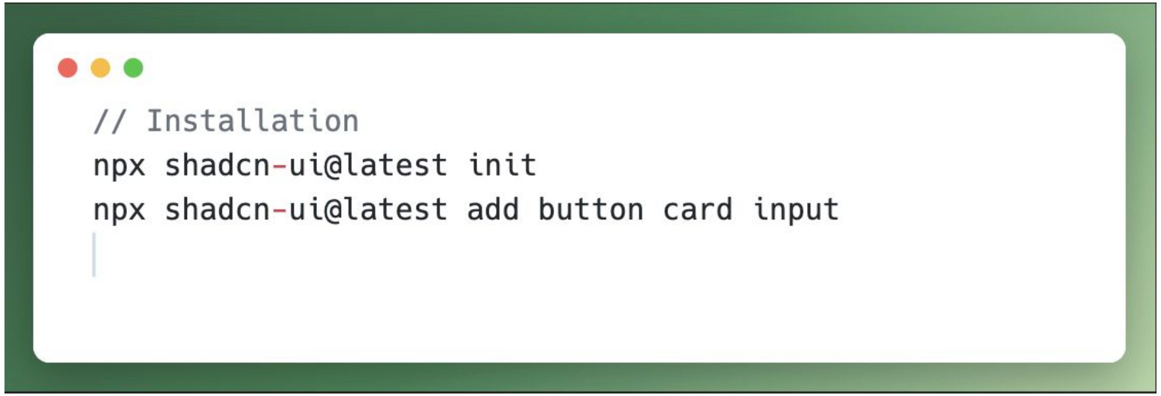
Core Philosophy
The components are blueprints. You use a CLI to copy the component code (e.g., a Button, a Dialog) directly into your project's source code. This gives you full ownership.

Key Characteristics

- Ownership: You are never locked into a dependency version. Since the code is yours, you can modify it completely to fit your project's needs.
- Accessibility: Built on Radix UI, which focuses on excellent un-styled, accessible primitives.
- Styling: Uses the power and flexibility of Tailwind CSS for easy customization.

shadcn/ui

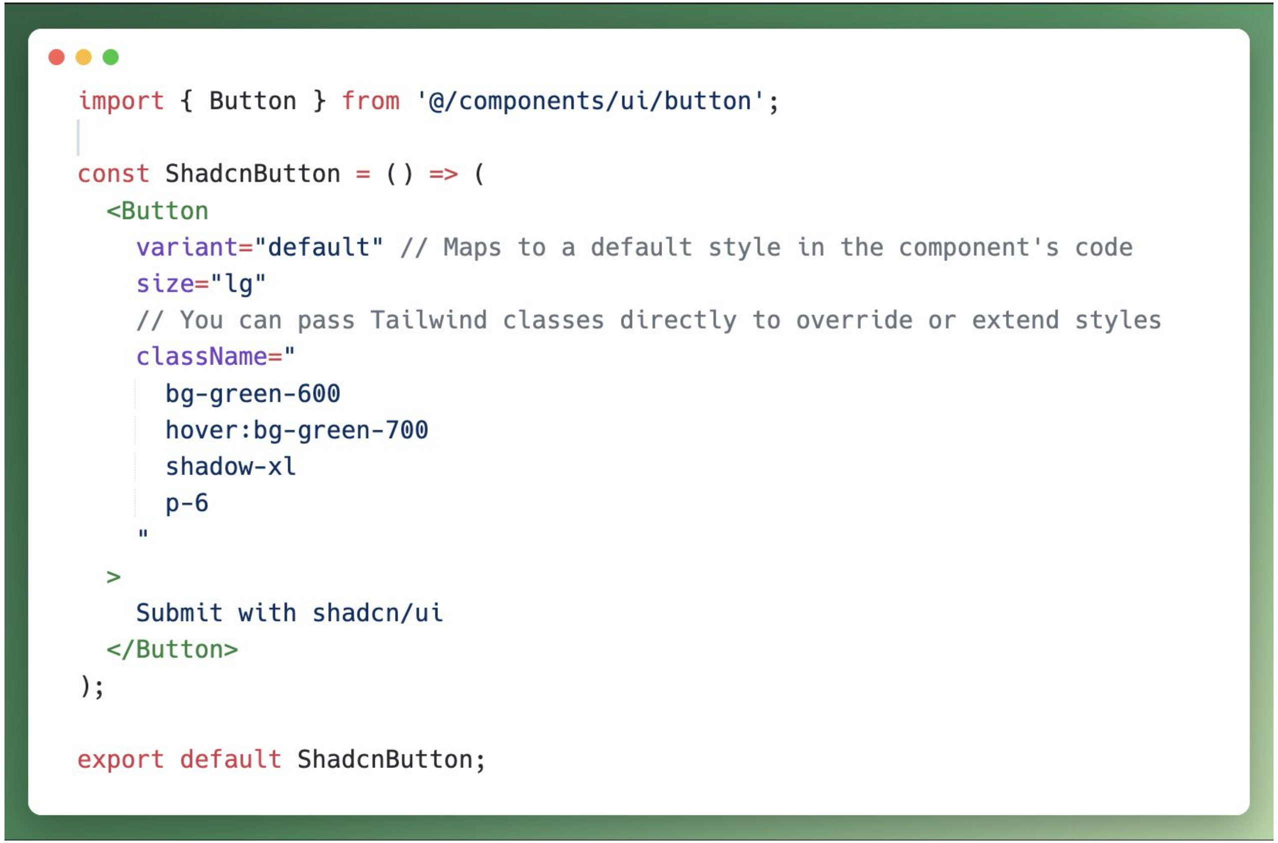
Getting Started with shadcn/ui



```
// Installation
npx shadcn-ui@latest init
npx shadcn-ui@latest add button card input
```

shadcn/ui

Code Example:



```
import { Button } from '@/components/ui/button';

const ShadcnButton = () => (
  <Button
    variant="default" // Maps to a default style in the component's code
    size="lg"
    // You can pass Tailwind classes directly to override or extend styles
    className="
      bg-green-600
      hover:bg-green-700
      shadow-xl
      p-6
    "
  >
    Submit with shadcn/ui
  </Button>
);

export default ShadcnButton;
```

Material UI

Material UI (MUI) is a popular Component Library that implements Google's Material Design principles. It offers a huge collection of ready-to-use, fully functional React components.

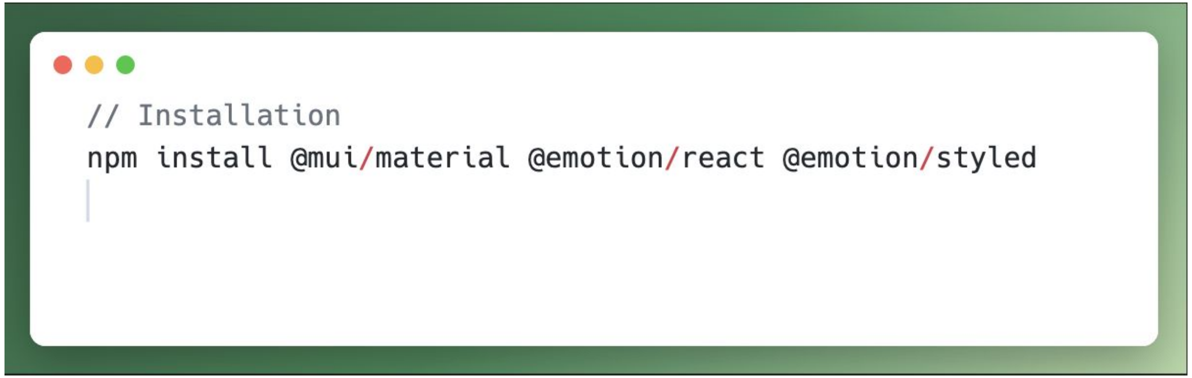
Core Philosophy
Focus on rapid development with highly consistent, well-documented components that adhere to a specific design system.

Key Characteristics

- Consistency: All components follow the Material Design guidelines, giving a polished look out-of-the-box.
- Feature Rich: Offers everything from basic buttons and forms to complex components like data grids (MUI X).
- Styling: Primarily customized through its Theming system (e.g., changing primary color) and the powerful sx prop for local overrides using CSS-in-JS.

Material UI

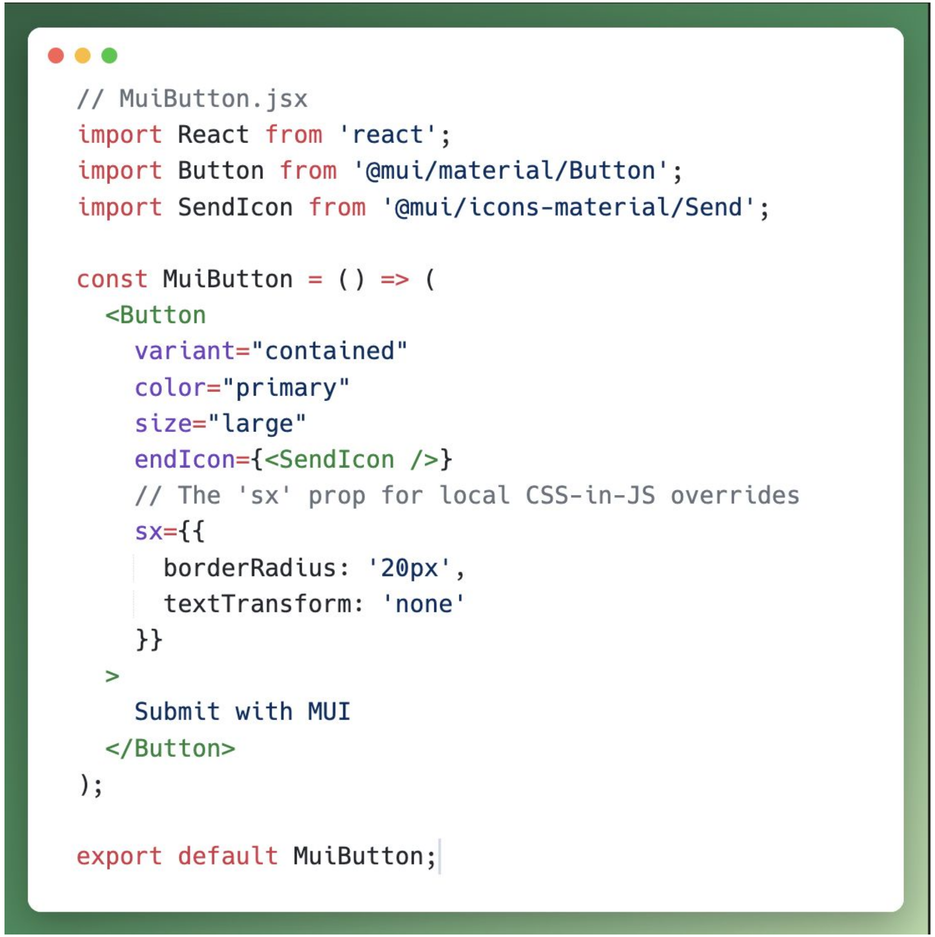
Getting Started with Material UI



```
// Installation
npm install @mui/material @emotion/react @emotion/styled
```

Material UI

Code Example:



```jsx
// MuiButton.jsx
import React from 'react';
import Button from '@mui/material/Button';
import SendIcon from '@mui/icons-material/Send';

const MuiButton = () => (
  <Button
    variant="contained"
    color="primary"
    size="large"
    endIcon={<SendIcon />}
    // The 'sx' prop for local CSS-in-JS overrides
    sx={{
      borderRadius: '20px',
      textTransform: 'none'
    }}
  >
    Submit with MUI
  </Button>
);

export default MuiButton;
```

Comparison Summary - Choosing Your Tool

	Tailwind CSS	**shadcn/ui**	**Material UI (MUI)**
Design	Zero Opinion (Build Anything)	Modern, Minimalist Design	Opinionated (Material Design)
Learning	Learn Utility Classes	Learn Tailwind + Radix Concepts	Learn Component Props
Component Source	Build from Scratch	Copy/Paste into Project	Install as Dependency
Best For	Completely unique designs, experienced CSS developers.	Projects prioritizing a modern look, accessibility, and full code control.	Large-scale apps needing speed, consistency, and a massive component catalog.
Maintenance	Manage many classes.	Manage local component files.	Manage library updates/versions.

Exercise

Create a to-do list app with Next.js and a CSS framework of your choice that matches this Figma design.

https://www.figma.com/design/CxBIpLb2atLLuILkIrtrwb/todo-app

Thank you