Purwadhika
Digital Technology School

Full Stack Web Development

CSS Framework

Job Connector Program

Outline

- CSS Framework
- Tailwind CSS, Chakra-UI & Kuma-UI
- Installation & Usage

What is CSS Framework?

CSS framework are tools used by UI developers to make their work easier. Instead of having to manually style every time a new project comes up

Frameworks give developers the convenience of quickly setting up user interfaces that can be changed and repeated across projects instead of wasting time starting from a blank document.

Why using CSS Framework?

Advantages of using CSS Framework

- Developers and designers can use CSS framework to implement various advanced features and visual elements on a website – forms, different buttons, navbars, breadcrumbs, and even clean symmetrical layouts.
- CSS framework make it simple to create websites compatible with multiple browsers and browser versions. This reduces the likelihood of bugs popping up during cross browser testing.
- Since these framework have ready-to-use stylesheets in place, using them allows faster and more convenient web development. Users don’t have to dive deep into CSS code to accomplish required tasks.
- Developers can quickly generate a user-friendly and visually appealing UI that can be modified throughout a project without starting from scratch.

CSS Frameworks

- Tailwind CSS
- Chakra-UI
- Kuma-UI
- Material-UI
- Reactstrap
- Etc ...

Tailwind CSS

What is **Tailwind CSS**?

Tailwind CSS is a utility-first CSS framework that provides low-level utility classes to build designs directly in your markup. Unlike traditional frameworks, Tailwind does not impose design decisions but provides building blocks for customization.

Tailwind follows a utility-first approach, where small utility classes are used directly in the HTML.

Benefits:

- Faster development with less custom CSS.
- Highly customizable and modular.

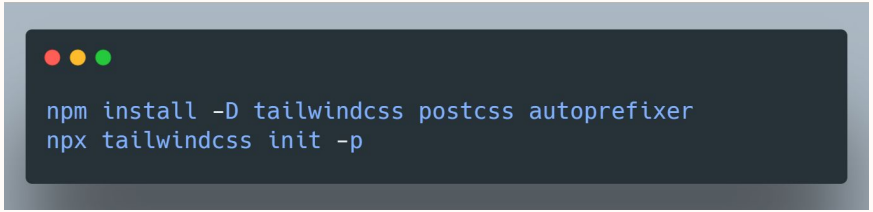
Reference : https://tailwindcss.com/docs/guides/vite

Tailwind - Installation & Configuration

The simplest and fastest way to get up and running with Tailwind CSS from scratch is with the Tailwind CLI tool.

- **Install Tailwind CSS**.
  Install tailwindcss via npm, and create your tailwind.config.js file.

- **Configure your template paths.**
  Add the paths to all of your template files in your tailwind.config.js file.



Tailwind - Usage

- Add the **@tailwind** directives for each of tailwind's layers to your main CSS file.
- Now you can start using Tailwind CSS classes in your React components

Tailwind - Usage

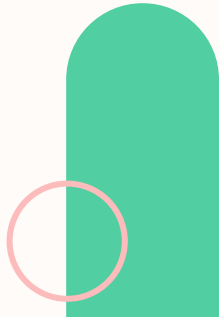
```jsx
import React from 'react';
import './styles.css';

const ExampleComponent = () => {
  return (
    <div className="bg-blue-500 p-4 rounded-md">
      <p className="text-white">Hello, Tailwind CSS in React!</p>
    </div>
  );
};

export default ExampleComponent;
```

Tailwind - Documentation

Find out more about Tailwind CSS : https://tailwindcss.com/docs/utility-first



Chakra UI

Chakra UI is a simple, modular and accessible component library that gives you the building blocks you need to build your React applications.

Reference : https://chakra-ui.com/

Chakra UI - Installation

Inside your project directory, install Chakra UI by running :

Chakra UI - Provider Setup

After installing Chakra UI, you need to set up the ChakraProvider at the root of your application. This can be either in your index.jsx or index.tsx

Put in the following code:

```jsx
import React from 'react'
import ReactDOM from 'react-dom/client'
import App from './App.jsx'
import './index.css'
// 1. import chakra provider
import { ChakraProvider } from '@chakra-ui/react'

ReactDOM.createRoot(document.getElementById('root')).ren
der(
  <React.StrictMode>
    {/* 2. wrap chakra provider onto App Component */}
    <ChakraProvider>
      <App />
    </ChakraProvider>
  </React.StrictMode>,
)
```

Chakra UI - Provider Setup

ChakraProvider Props:

Name	Type	Default	Description
resetCSS	`boolean`	`true`	automatically includes `<CSSReset />`
theme	`Theme`	`@chakra-ui/theme`	optional custom theme
colorModeManager	`StorageManager`	`localStorageManager`	manager to persist a users color mode preference in
portalZIndex	`number`	`undefined`	common z-index to use for `Portal`

Chakra UI - Usage

For example we want to use button component from Chakara UI, first import the button component like this:

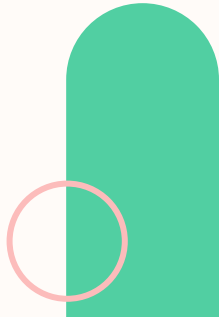
```
import { Button, ButtonGroup } from "@chakra-ui/react";
```

Then use it in our component like this:

```
<Button colorScheme="lightblue">Button</Button>
```

Chakra UI - Documentation

Find out more about Chakra UI : https://chakra-ui.com/getting-started



Kuma UI

Kuma UI is a headless, utility-first, and zero-runtime UI component library. Kuma UI empower your web with ultimate performance and flexibility.

Reference : https://www.kuma-ui.com/docs

Kuma UI

Kuma UI - Installation & Configuration

To install Kuma UI in your project, run one of the following commands in your terminal:

```
npm install @kuma-ui/core
npm install -D @kuma-ui/vite
```

If you use Vite, update vite.config.js :

```
import { defineConfig } from "vite";
import react from "@vitejs/plugin-react";
import KumaUI from "@kuma-ui/vite";

export default defineConfig({
  plugins: [
    react(),
    KumaUI(),
  ],
});
```

Kuma UI - Usage

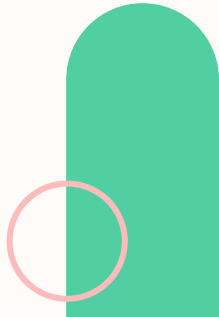
```jsx
import { Box } from "@kuma-ui/core";

function App() {
  return (
    <Box p={8} bg="blue" color="white">
      Hello world
    </Box>
  );
}

export default App;
```

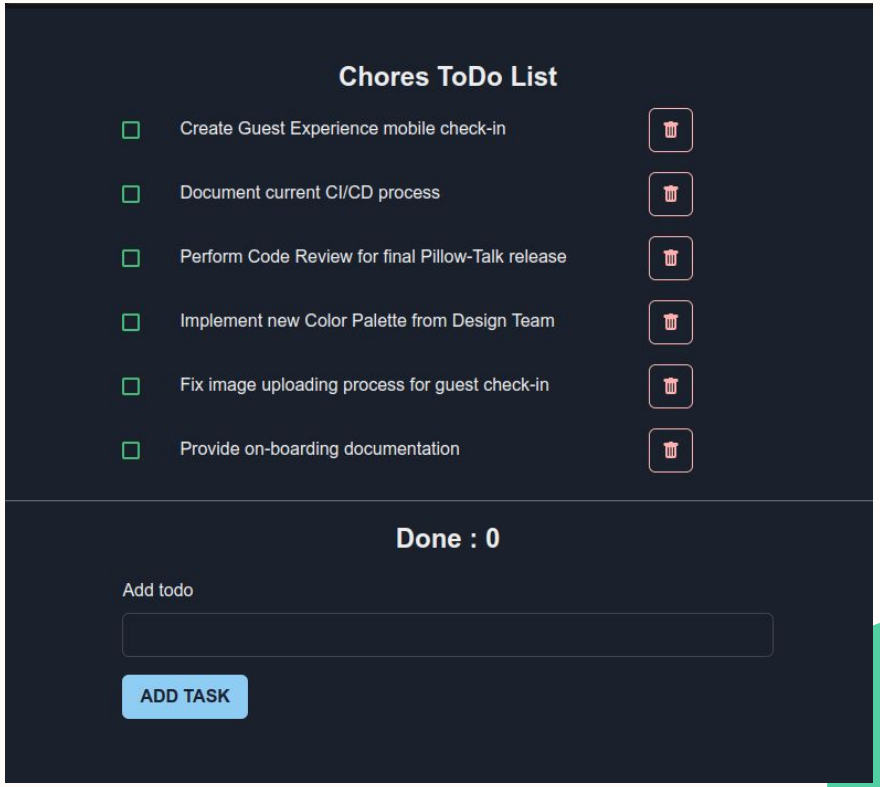
Kuma UI - Documentation

Find out more about Kuma UI : https://www.kuma-ui.com/docs



Exercise

Create to do list with React + CSS framework you like, like the example below.



Thank You!