# Internship Report – Frontend Dev

# Week 7:  React.js Advanced

|  |
|---|
| **Name:  Zainab** |
| **Father Name:  Assad Qayyum** |
| **Date: 4th Aug, 2025** |
| **Internship Domain:  Front-end Intern** |
| **Task:  React Project Folder, Component Libraries(Tailwind, Bootstrap)** |

## Task Overview: (Day1)

In today's task, I learned about how a React project is organized and how component libraries like Tailwind and Bootstrap are used for styling. I explored the default folder structure, the purpose of each file, and how to organize components for scalability. I also learned about using Tailwind CSS to style components with utility-first classes.

## Content Covered:

The content covered is:

- React Project Folder Structure
- Understanding Core Files (src, public, node_modules, package.json)
- Organizing components in folders
- Component libraries (Tailwind CSS and Bootstrap

## Introduction:

React is a popular JavaScript library used to build user interfaces. A well-organized folder structure is important for maintaining and scaling projects easily. Along with this, using component libraries like Tailwind or Bootstrap helps in quickly styling applications and maintaining consistency.

## React Project Folder Structure:

When we create a React project (using Create React App or Vite), a default folder structure is generated:

### Main Folders and Files

- **src/**: Main coding area where all React components and logic are written.
- **public/**: Contains static assets like index.html and favicon.
- **node_modules/**: Stores installed packages and libraries.
- **package.json**: Lists project dependencies and scripts for running and building the project.

### Organizing Components

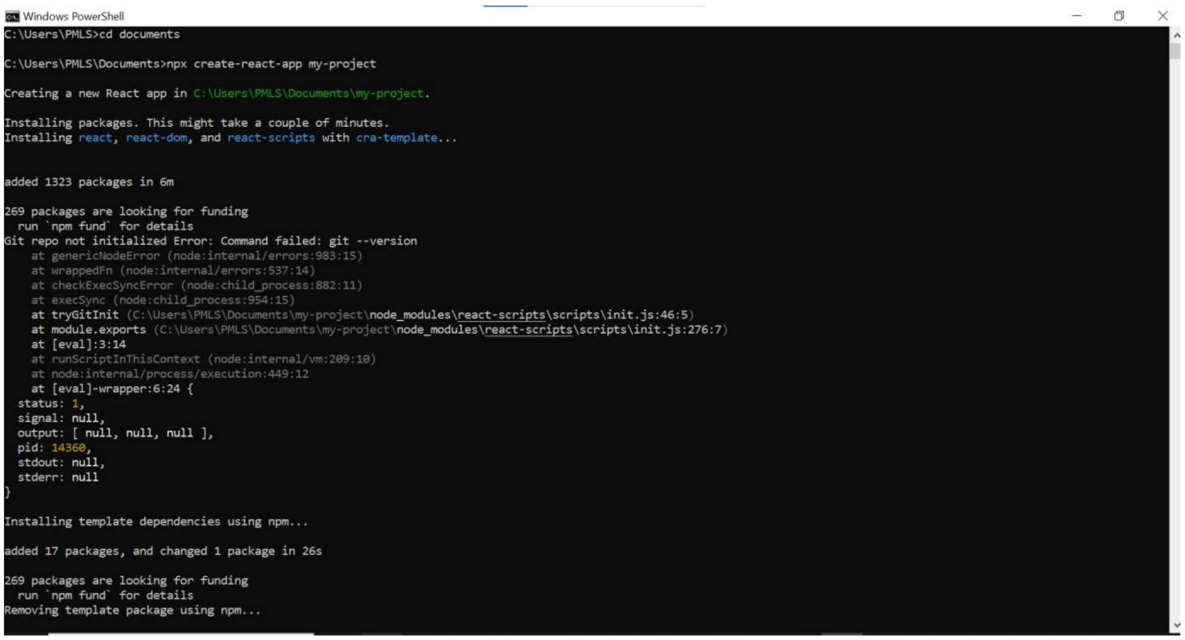
- Inside src/, we create a components/ folder
- Each component should ideally have its own folder containing .jsx and .css files for better organization.

### Key Files

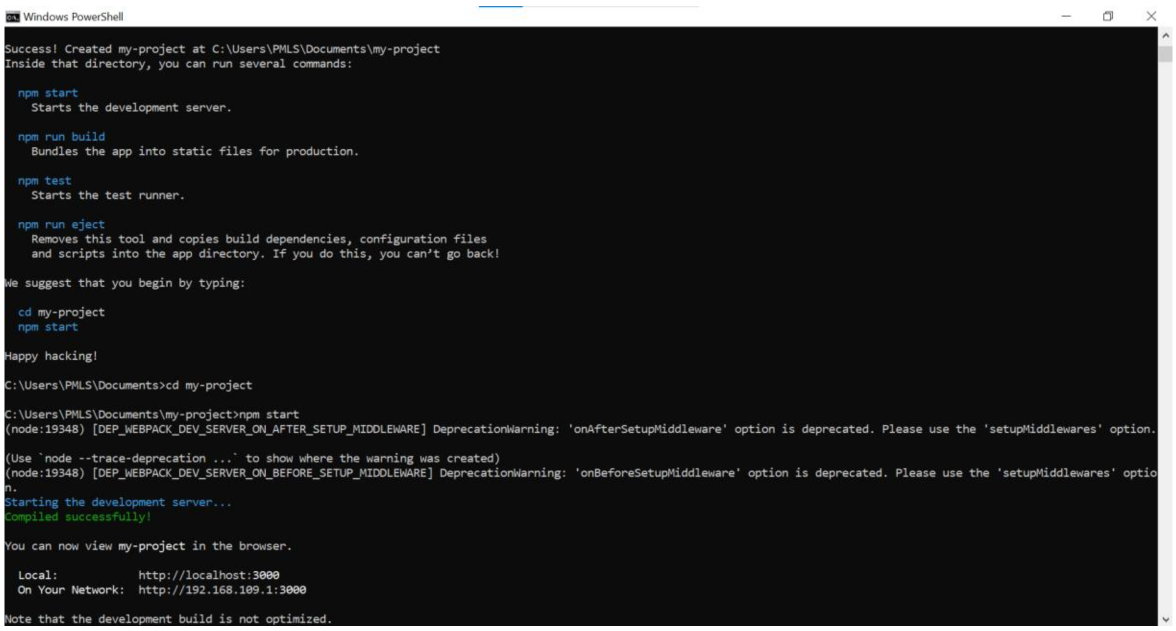
- **index.jsx:** The entry point of the app that renders the main component (App.jsx) into the index.html file.
- **App.jsx:** The root component that combines and displays other components.

### Steps Practiced

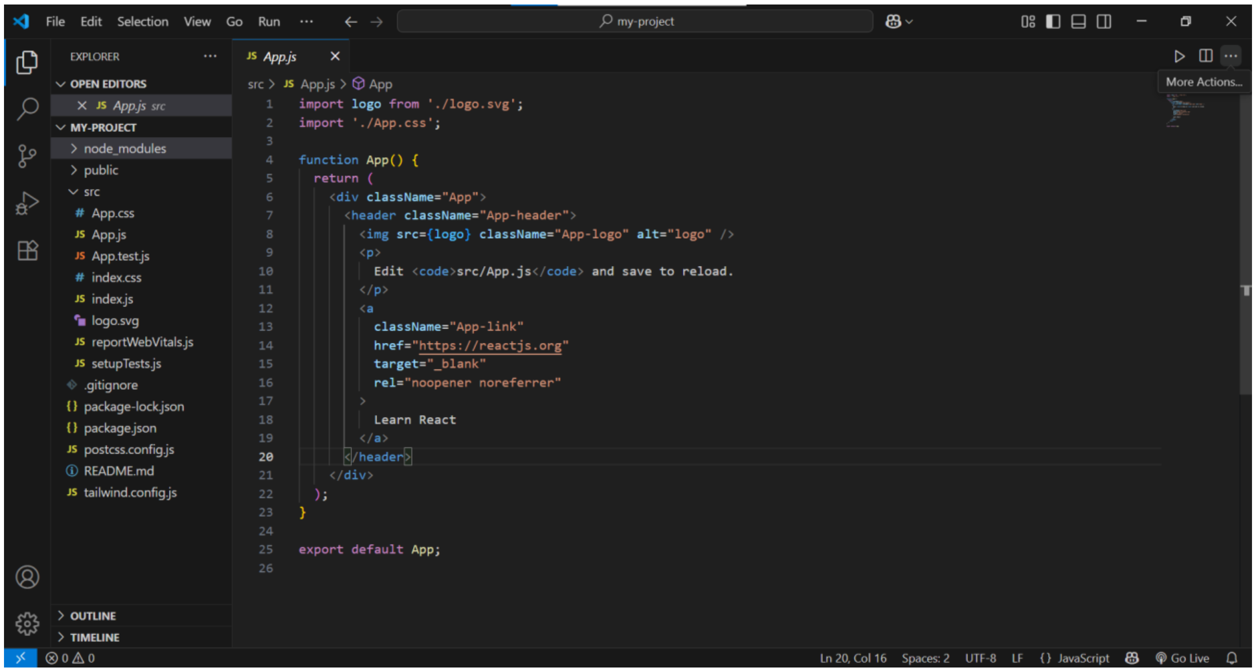
1. Created a new React project.
2. Explored and understood each folder and file.
3. Created a reusable Button component inside src/components/Button/ and used it inside App.jsx.



```
C:\Users\PMLS>cd documents

C:\Users\PMLS\Documents>npx create-react-app my-project

Creating a new React app in C:\Users\PMLS\Documents\my-project.

Installing packages. This might take a couple of minutes.
Installing react, react-dom, and react-scripts with cra-template...


added 1323 packages in 6m

269 packages are looking for funding
  run `npm fund` for details
Git repo not initialized Error: Command failed: git --version
    at genericNodeError (node:internal/errors:983:15)
    at wrappedFn (node:internal/errors:537:14)
    at checkExecSyncError (node:child_process:882:11)
    at execSync (node:child_process:954:15)
    at tryGitInit (C:\Users\PMLS\Documents\my-project\node_modules\react-scripts\scripts\init.js:46:5)
    at module.exports (C:\Users\PMLS\Documents\my-project\node_modules\react-scripts\scripts\init.js:276:7)
    at [eval]:3:14
    at runScriptInThisContext (node:internal/vm:209:10)
    at node:internal/process/execution:449:12
    at [eval]-wrapper:6:24 {
  status: 1,
  signal: null,
  output: [ null, null, null ],
  pid: 14360,
  stdout: null,
  stderr: null
}

Installing template dependencies using npm...

added 17 packages, and changed 1 package in 26s

269 packages are looking for funding
  run `npm fund` for details
Removing template package using npm...
```



```
Success! Created my-project at C:\Users\PMLS\Documents\my-project
Inside that directory, you can run several commands:

  npm start
    Starts the development server.

  npm run build
    Bundles the app into static files for production.

  npm test
    Starts the test runner.

  npm run eject
    Removes this tool and copies build dependencies, configuration files
    and scripts into the app directory. If you do this, you can’t go back!

We suggest that you begin by typing:

  cd my-project
  npm start

Happy hacking!

C:\Users\PMLS\Documents>cd my-project

C:\Users\PMLS\Documents\my-project>npm start
(node:19348) [DEP_WEBPACK_DEV_SERVER_ON_AFTER_SETUP_MIDDLEWARE] DeprecationWarning: 'onAfterSetupMiddleware' option is deprecated. Please use the 'setupMiddlewares' option.

(Use `node --trace-deprecation ...` to show where the warning was created)
(node:19348) [DEP_WEBPACK_DEV_SERVER_ON_BEFORE_SETUP_MIDDLEWARE] DeprecationWarning: 'onBeforeSetupMiddleware' option is deprecated. Please use the 'setupMiddlewares' optio
n.
Starting the development server...
Compiled successfully!

You can now view my-project in the browser.

  Local:            http://localhost:3000
  On Your Network:  http://192.168.109.1:3000

Note that the development build is not optimized.
```



```
import logo from './logo.svg';
import './App.css';

function App() {
  return (
    <div className="App">
      <header className="App-header">
        <img src={logo} className="App-logo" alt="logo" />
        <p>
          Edit <code>src/App.js</code> and save to reload.
        </p>
        <a
          className="App-link"
          href="https://reactjs.org"
          target="_blank"
          rel="noopener noreferrer"
        >
          Learn React
        </a>
      </header>
    </div>
  );
}

export default App;
```

## Component Libraries (Tailwind, Bootstrap):

Component libraries provide **pre-written CSS and components** that help in building stylish applications faster.

### a) Bootstrap(Component-based Framework)

Provides ready-made UI components like buttons, navbars, and modals.

Good for quick setups but less customizable.

Uses predefined classes like btn btn-primary.

**Example:**

```
<button className="btn btn-primary">Click Me</button>
```

### b) Tailwind CSS (Utility-first Framework)

A utility-first CSS framework that allows applying styles directly in JSX using classes.

Highly customizable and encourage component reusability.

**Example:**

```
<button className="bg-blue-500 text-white p-2 rounded">Click</button>
```

Utility classes like:

- text-red-500 → Red text
- m-4 / p-2 → Margin and padding
- flex, items-center → Flexbox styling

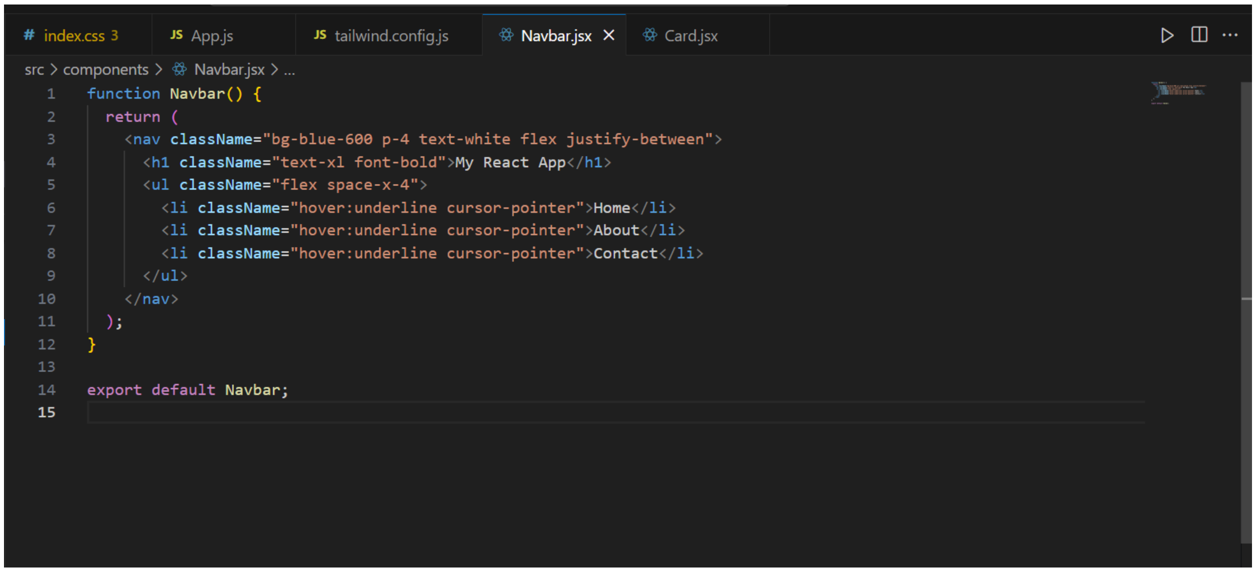
**When to Use**

- **Tailwind:** When yo want full design control and lightweight styling.
- **Bootstrap:** When you want a ready-made consistent design fast.

## Practice Code:

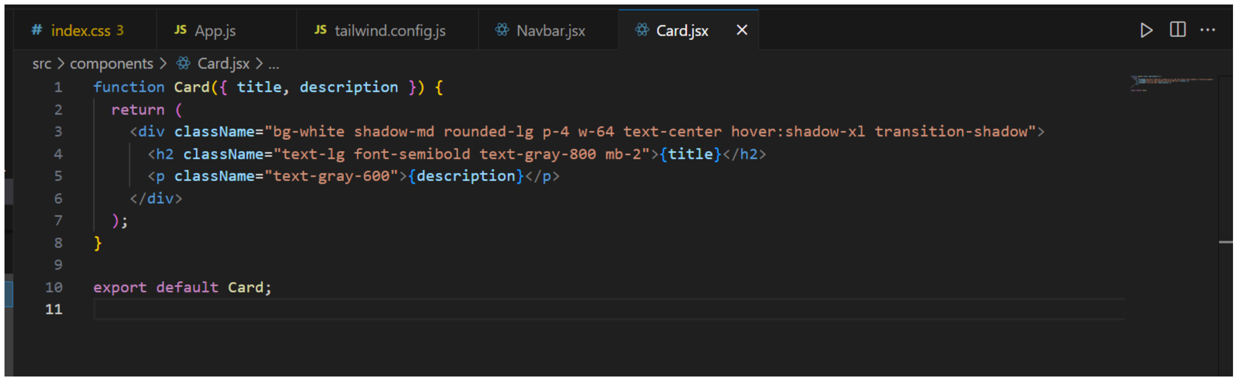
### Components:

### Navbar.jsx:



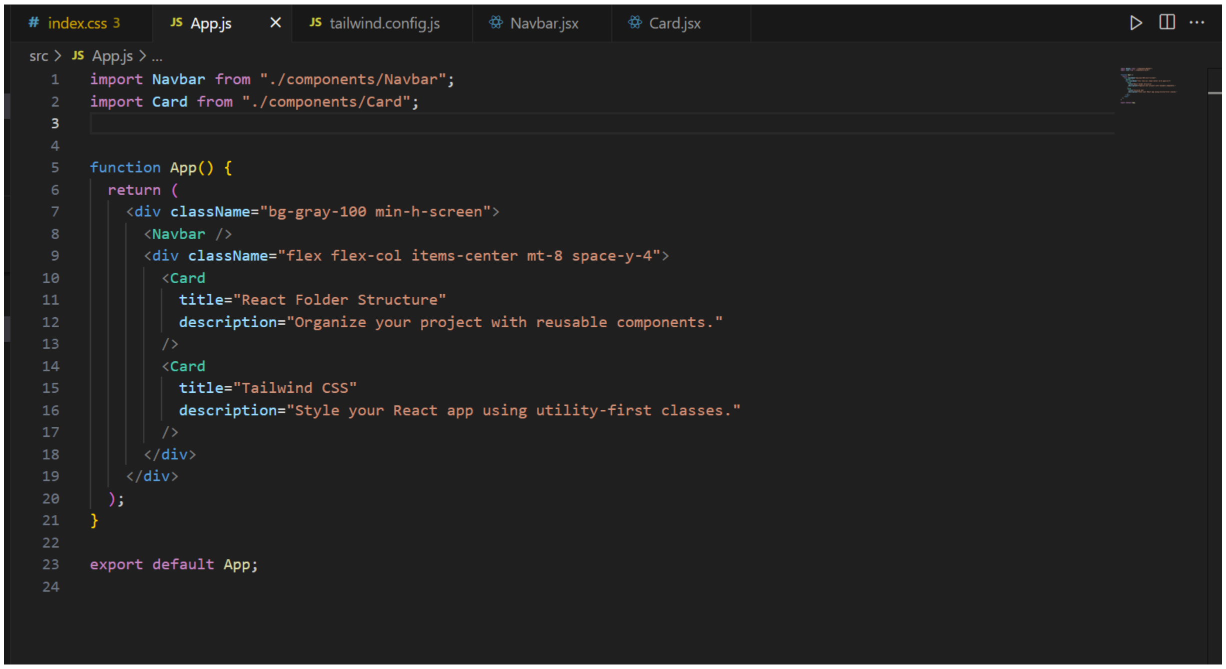
```
function Navbar() {
  return (
    <nav className="bg-blue-600 p-4 text-white flex justify-between">
      <h1 className="text-xl font-bold">My React App</h1>
      <ul className="flex space-x-4">
        <li className="hover:underline cursor-pointer">Home</li>
        <li className="hover:underline cursor-pointer">About</li>
        <li className="hover:underline cursor-pointer">Contact</li>
      </ul>
    </nav>
  );
}

export default Navbar;
```

## Card.jsx:



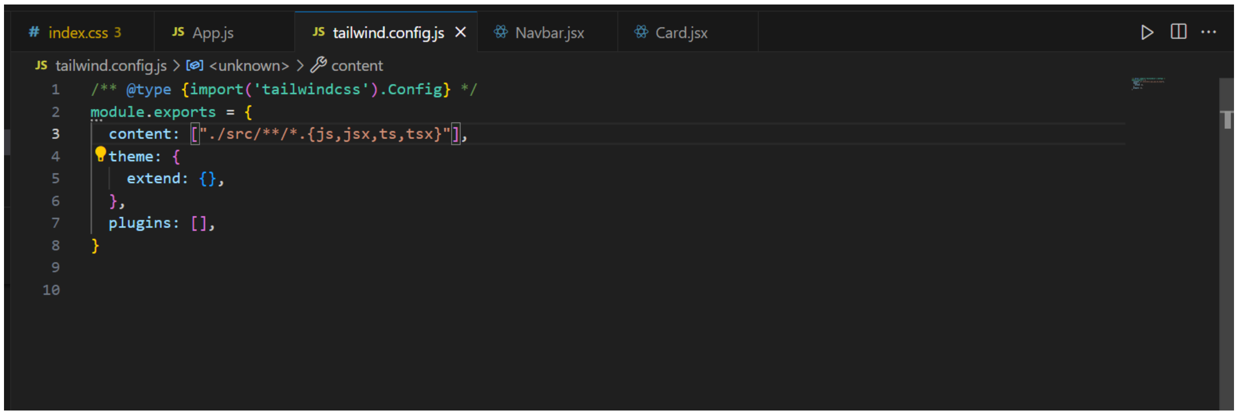
```jsx
function Card({ title, description }) {
  return (
    <div className="bg-white shadow-md rounded-lg p-4 w-64 text-center hover:shadow-xl transition-shadow">
      <h2 className="text-lg font-semibold text-gray-800 mb-2">{title}</h2>
      <p className="text-gray-600">{description}</p>
    </div>
  );
}

export default Card;
```

## App.js:

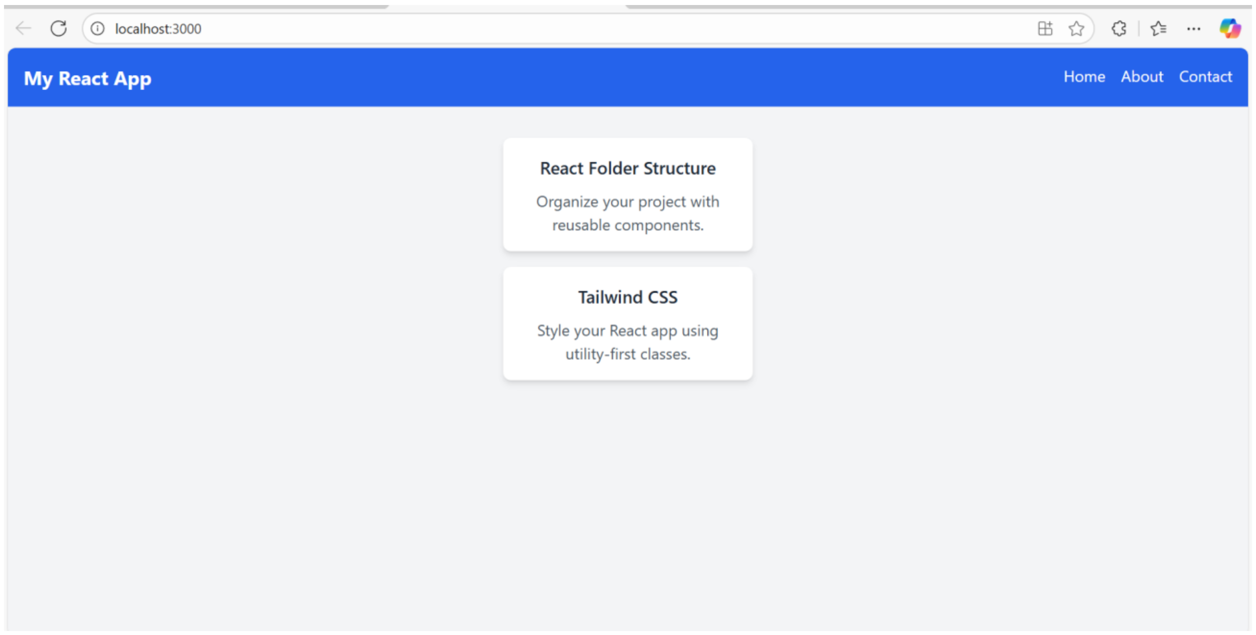


```jsx
import Navbar from "./components/Navbar";
import Card from "./components/Card";


function App() {
  return (
    <div className="bg-gray-100 min-h-screen">
      <Navbar />
      <div className="flex flex-col items-center mt-8 space-y-4">
        <Card
          title="React Folder Structure"
          description="Organize your project with reusable components."
        />
        <Card
          title="Tailwind CSS"
          description="Style your React app using utility-first classes."
        />
      </div>
    </div>
  );
}

export default App;
```

## Tailwind.config.js:



```js
/** @type {import('tailwindcss').Config} */
module.exports = {
  content: ["./src/**/*.{js,jsx,ts,tsx}"],
  theme: {
    extend: {},
  },
  plugins: [],
}
```



React Folder Structure

Organize your project with reusable components.

Tailwind CSS

Style your React app using utility-first classes.

**Conclusion:**

Today, I learned how React projects are structured and why organizing components is important. I also explored component libraries like Bootstrap and Tailwind CSS and understood their uses. Tailwind's utility-first approach made styling components simple and fast.