🎨 Week 3: React.js, JSX, and Tailwind CSS – Mastering Front-End Development

🚀 Objective

Build a responsive React application using JSX and Tailwind CSS that demonstrates component architecture, state management, hooks usage, and API integration.

📂 Tasks

Task 1: Project Setup

- Create a new React application using Vite
- Install and configure Tailwind CSS
- Set up the project structure with components, pages, and utility folders
- Configure basic routing using React Router

Task 2: Component Architecture

- Create reusable UI components:
  - `Button` component with different variants (primary, secondary, danger)
  - `Card` component for displaying content in a boxed layout
  - `Navbar` component for site navigation
  - `Footer` component with links and copyright information
- Implement a layout component that includes the Navbar and Footer
- Use props to make components customizable and reusable

Task 3: State Management and Hooks

- Implement a `TaskManager` component that allows users to:
  - Add new tasks
  - Mark tasks as completed
  - Delete tasks
  - Filter tasks (All, Active, Completed)
- Use the following hooks:
  - `useState` for managing component state
  - `useEffect` for side effects (e.g., loading saved tasks)
  - `useContext` for theme management (light/dark mode)
  - Create a custom hook (e.g., `useLocalStorage`) for persisting tasks

Task 4: API Integration

- Fetch data from a public API (e.g., JSONPlaceholder)
- Display the fetched data in a list or grid layout
- Implement loading and error states
- Add pagination or infinite scrolling
- Create a search feature to filter the API results

Task 5: Styling with Tailwind CSS

- Create a responsive design that works on mobile, tablet, and desktop
- Implement a theme switcher (light/dark mode) using Tailwind's dark mode
- Use Tailwind's utility classes for layout, spacing, typography, and colors
- Create custom animations or transitions for interactive elements

🧪 Expected Outcome

- A fully functional React application with multiple components
- Proper state management using React hooks
- API integration with loading and error handling
- Responsive design implemented with Tailwind CSS
- Clean, well-organized code following React best practices

🛠️ Setup

1. Make sure you have Node.js installed (v18 or higher recommended)
2. Use the provided starter files in this repository
3. Install the required dependencies:

```
npm install
```

4. Start the development server:

```
npm run dev
```

✅ Submission Instructions

1. Accept the GitHub Classroom assignment invitation
2. Clone your personal repository that was created by GitHub Classroom
3. Complete all the tasks in the assignment
4. Commit and push your code regularly to show progress
5. Include in your repository:
    - All project files with proper organization
    - A comprehensive README.md with setup instructions
    - Screenshots of your application in the README.md
6. Deploy your application to Vercel, Netlify, or GitHub Pages
7. Add the deployed URL to your README.md

8. Your submission will be automatically graded based on the criteria in the autograding configuration
9. The instructor will review your submission after the autograding is complete