React Web Development Assignments

Assignment 1: Personal Blog Platform

Time Required: 6-8 hours

Objectives:

- Learn component creation and JSX
- Implement state management with hooks
- Practice routing
- Understand basic form handling

Problem Statement:

Develop a Personal Blog Platform that allows users to:

1. Create and view blog posts
2. Navigate between different views using React Router
3. Manage post state using React hooks
4. Implement a simple, responsive design

Key Requirements:

- Create multiple components:
  - HomePage (displays blog post list)
  - PostCreationForm
  - PostDetailView
  - Navigation component
- Implement routing between views
- Use useState for managing blog posts
- Use useEffect to persist posts in local storage
- Add basic form validation for post creation

Learning Outcomes:

- Understand JSX component structure
- Learn state management with useState
- Implement React Router for navigation
- Practice handling form inputs and state

Hints:

- Start by creating a basic component structure
- Use react-router-dom for navigation
- Implement local storage to save posts
- Break down the application into smaller, manageable components
- Focus on clean, readable code

- Add simple error handling for form submissions

Assignment 2: Theme-Based Task Management App

Time Required: 6-8 hours

Objectives:

- Master Context API for state management
- Implement advanced React hooks
- Create a dynamic, interactive application
- Learn about global state and theme switching

Problem Statement:

Create a Task Management Application with Theme Switching that includes:

1. Full task management functionality
2. Global theme context (light/dark modes)
3. Persistent storage of tasks and theme preference
4. Responsive design

Key Requirements:

- Implement task features:
  - Add new tasks
  - Mark tasks as complete
  - Delete tasks
  - Filter tasks
- Create a theme switcher using Context API
- Use useContext for theme management
- Persist tasks and theme preference in local storage
- Create reusable components
- Implement basic animations or transitions

Learning Outcomes:

- Understand Context API and global state management
- Advanced usage of React hooks (useState, useEffect, useContext)
- Learn about component composition
- Practice creating and consuming context

Hints:

- Create a separate ThemeContext for managing themes
- Use a ThemeProvider component
- Implement a custom hook for theme management
- Store theme and task data in local storage
- Use conditional rendering for theme-based styling

- Break down the application into smaller, focused components
- Add basic error handling
- Focus on creating a clean, intuitive user interface