INNOMATİCS
RESEARCH LABS

React Project: Static Pages with Routing

Project Overview:

In this project, you will create a React application that consists of multiple static pages. The key focus of this project is to understand and implement routing using `react-router-dom`. You will also practice nested routing by creating at least four pages with a nested structure. This will help you understand how to set up different routes and manage page navigation within a React application.

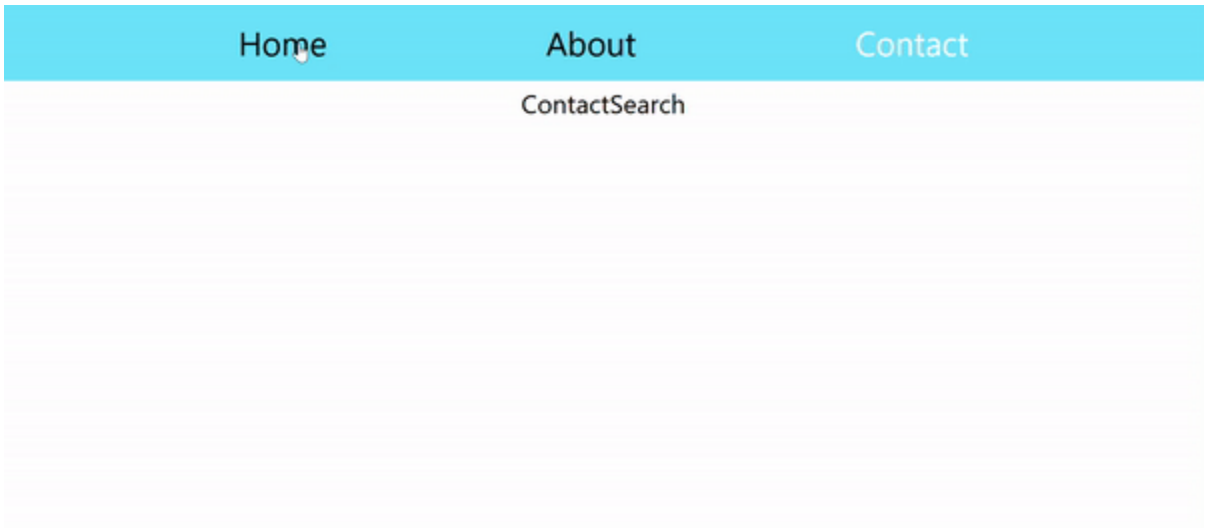
Project Requirements:

- Create a React app with a minimum of four static pages.
- Implement basic routing using `react-router-dom`.
- Create nested routes for at least one of the pages.
- Use functional components and React hooks where necessary.
- Ensure the app is responsive for different screen sizes.

Example Pages:

1. Home: A simple introduction page with navigation links.
2. About: Information about the project or company.
3. Services: List of services offered, with nested routes for individual service details.
4. Contact: A page with a contact form or information.
5. Nested Example (e.g., Services): Create sub-routes like `/services/design`, `/services/development`, etc.

Image Reference :



Key Point: **Output shouldn't be same it should follow this format and remaining content should related to Innomatics Reserach Labs**

Testing Your Application:

- Run the app using `npm start` and ensure the following:
  - All pages are accessible through their respective routes.
  - The `Services` page allows navigation to nested routes like `/services/design` and `/services/development`.
  - The page content updates based on the route.

Additional Tips:

- Make sure to add basic styling to improve the look and feel of your application.
- Use `NavLink` instead of `Link` for active class styling.
- Ensure that the nested routes display correctly within the `Services` page.

Learning Outcomes:

- Understanding of how to set up and manage routing in a React application.
- Experience with nested routing and dynamic route rendering.
- Improved ability to structure React applications using components.
- Practice using React hooks like `useState` and `useEffect` where necessary.

Optional Enhancements:

- Add animations when navigating between pages.
- Implement form validation on the Contact page using `useState` and `onChange` handlers.
- Create a 404 page to handle undefined routes using a wildcard `*` route.