Project Instructions: Building and Testing a React Application

**Objective**

Your task is to create a simple React application that interacts with a 3rd party API to fetch and display data. The application should include at least three components and demonstrate basic interactivity. You will also write test cases for your components to ensure they work as expected.

There is a document on how to get a React App started with the proper libraries to interact with Typescript and Jest.

**Requirements**

1. **React Components**:
   * Create at least three React components in your application.
     + **SearchForm**: A component that handles user input and form submission.
     + **ResultList**: A component that displays a list of results fetched from the API.
     + **ResultItem**: A component that displays individual result details.
   * Ensure that your components demonstrate basic interactivity. For example, the SearchForm component should handle user input and form submission, and the ResultList should update based on the search results.
2. **API Integration**:
   * Use a 3rd party API to fetch data and display it in your components.
     + Examples of APIs you can use:
       - [OMDB API](http://www.omdbapi.com/)
       - OpenWeatherMap API
       - Google Books API
       - Spoonacular API
       - [NewsAPI](https://newsapi.org/)
   * Implement data fetching in response to user actions (e.g., submitting the search form).
   * Display loading states while fetching data is optional but nice.
   * Handle and display error messages if the API request fails.
   * You may hard code some data for development and testing purposes, but the application must demonstrate fetching data from an API.
3. **Testing**:
   * Write at least four test cases for your components using React Testing Library and Jest.
     + **Component Rendering**: Test if each component renders correctly with given props.
     + **User Interactions**: Test if user interactions (e.g., form submission) trigger the correct functions and state updates.
     + **Conditional Rendering**: Test if components render different content based on state or props (e.g., loading state, error messages, results).
   * Ensure your tests cover different aspects of your application, including rendering, user interactions, and conditional rendering.