Custom React UI Component - "Image Slider with Parallax Effect"

Project Description: Your task is to create a unique and visually captivating React UI component - an "Image Slider with Parallax Effect". The component should allow users to navigate through a series of images while experiencing a parallax scrolling effect that creates depth and dynamism. This task aims to assess your creativity and ability to create complex UI interactions.

Requirements:

1. **Image Slider:** Design and implement a React component that displays a series of images. Users should be able to navigate through the images using navigation arrows or dots.
2. **Parallax Effect:** Implement a parallax scrolling effect that provides a sense of depth as the user interacts with the slider. The effect should involve the images moving at different speeds relative to the user's scrolling.
3. **Smooth Transitions:** Ensure smooth and visually appealing transitions between images as the user navigates.
4. **Configuration Options:** Include configuration options for adjusting the parallax intensity and transition speed.
5. **Responsive Design:** Create a responsive UI that works well on different screen sizes.
6. **Custom Styles:** Apply custom CSS styles to enhance the overall look of the component and the parallax effect.
7. **Code Reusability:** Design the component to be reusable and easy to integrate into different projects.

Evaluation Criteria:

1. **Creativity:** Demonstration of creativity in creating a unique and engaging UI component.
2. **React Proficiency:** Effective use of React concepts and components.
3. **Parallax Effect:** Successful implementation of the parallax scrolling effect.
4. **Transitions:** Smooth and visually pleasing transitions between images.
5. **Configuration Options:** Implementation of configuration options for customization.
6. **Responsive Design:** UI that adapts well to different screen sizes.
7. **CSS Styling:** Skillful application of custom CSS for aesthetics and visual effects.
8. **Code Quality:** Well-structured and maintainable code.

9. **Documentation:** Clear instructions for using the component and adjusting configurations.

Submission:

1. Create a new repository for the task.
2. Commit your code regularly with descriptive commit messages.
3. Push your code to your chosen Git repository.
4. Provide clear instructions on how to use the component and customize parallax effects.
5. Share the repository link with us when you're done.

Please note that this task requires a creative approach and may not have direct answers available on the internet. It aims to assess your problem-solving skills and your ability to implement complex UI interactions in React. Good luck!

Bonus task:

Provided below is a bonus task which is not mandatory. You can attend this for some extra points.

Complex React UI Component - "Interactive Data Visualization Chart"

Project Description: As a bonus challenge, you are invited to create a highly interactive and visually appealing React UI component - an "Interactive Data Visualization Chart". The goal of this bonus task is to assess your ability to work with complex data visualization libraries and create dynamic user experiences.

Requirements:

1. **Data Visualization:** Choose a data visualization library (e.g., D3.js, Chart.js, Plotly) and implement a dynamic chart that visualizes a dataset of your choice. The chart should be interactive and capable of handling user interactions like hovering, clicking, and zooming.
2. **Multiple Chart Types:** Implement at least two different types of charts within the same component, allowing users to toggle between them (e.g., bar chart and line chart).
3. **Data Integration:** Fetch data from a public API or generate a mock dataset to showcase the chart's capabilities.

4. **Interactive Features:** Implement tooltips, highlighting, and animations to enhance the user experience. Users should be able to interact with the chart elements and see relevant information.
5. **Customization Options:** Provide options to customize chart parameters such as color schemes, labels, and axes.
6. **Responsive Design:** Ensure that the chart responds effectively to different screen sizes.
7. **Code Reusability:** Design the component to be reusable and versatile for integration into diverse projects.

Bonus Evaluation Criteria:

1. **Complexity:** Successful implementation of a sophisticated and interactive data visualization.
2. **Library Integration:** Effective utilization of a data visualization library.
3. **Interactivity:** Implementation of interactive features like tooltips, highlighting, and zooming.
4. **Chart Types:** Integration of at least two different types of charts.
5. **Data Handling:** Proper handling and integration of data from an external source.
6. **Customization:** Implementation of options for customization.
7. **Responsive Design:** UI that adapts well to different screen sizes.
8. **Code Quality:** Structured and well-documented code.
9. **Documentation:** Clear instructions for using and customizing the complex component.

Submission:

1. Create a new repository for this bonus task and mention "bonus-task in repo name.
2. Commit your code regularly with descriptive commit messages.
3. Push your code to your chosen Git repository.
4. Provide clear instructions on how to use and customize the interactive chart.
5. Share the repository link with us when you're done.

Please note that this bonus task involves a higher level of complexity and is meant to showcase your skills in working with intricate UI components and data visualization libraries. Good luck with this challenging task!