- ## Description of the Approach :

The primary objective was to build a React application that dynamically generates a timeline based on user input, utilizing the **Tldraw** library for rendering the shapes and text.The approach involves managing two separate states in the **App** component: one for tracking the user's input (**inputValue**) and another for controlling the rendering of timeline items (**itemCount**). Initially, the inputValue reflects the number the user types in the input field, but it doesn't immediately trigger a re-render of the **TldrawComponent**. The application consists of two main components:

1. **App Component**: Manages the user interface, including the input field for the number of timeline items and the "Generate" button. It controls the state of the application, ensuring that the timeline is only updated when the "Generate" button is clicked.
2. **TldrawComponent**: Handles the rendering of the timeline in the form of shapes and text using the **Tldraw** library. It dynamically creates shapes based on the items prop received from the **App** component.

- ## Challenges Faced :

1. **State Management**: Ensuring that the timeline only updates when the "Generate" button is clicked required a clear separation between the input field value and the state controlling the rendering. Initially, the **'itemCount'** was tied directly to the input value, causing the timeline to update immediately, which wasn't the desired behavior.
2. **Rendering Control**: It was essential to ensure that the **TldrawComponent** correctly re-renders based on the updated state while also cleaning up any previously created shapes to avoid duplication or memory leaks. This was managed by using the **useEffect** hook with a cleanup function in **TldrawComponent**.

These adjustments provided a more user-controlled interaction, allowing the timeline to only update when the user explicitly triggers it by clicking the "Generate" button.