Multi-Step Form using React.js **(time: 90 min)**

Task:

The task is to create a multi-step form in React that allows users to progress through multiple steps.
Step 1: Enter Name,
Step 2: Enter Email,
Step 3: Enter Date,
Step 4: Enter Password.
Each step should have its own input fields and the form should meet specific requirements.

Requirements:

Navigation: Users should be able to navigate forward and backward through the form's steps, if the user is on first step backward functionality should not be accessible.

Input Validation: Input fields in each step are validated based on regular expressions. Users are unable to proceed to the next step until their input meets the validation criteria. Additionally, there's an option to display an error message to users, though this is not mandatory.

Submission: Upon reaching the final step, the form should be successfully submitted with a “Form submitted successfully” screen.

Extras>>
User Experience: Enhance the user experience by providing clear feedback and navigation options.

Notes:
The primary focus of this question is on achieving seamless functionality. The objective is to ensure that users can smoothly complete all the steps of the form.

Styling is not mandatory; however, it may be implemented to enhance proper navigation and accessibility. Notably, incorporating styling for these purposes can earn you bonus credit in interviews.

Solution

The solution involves implementing a React component that handles the multi-step form. It utilizes React state management to keep track of the current step, manage input validation, and store form data.

State Management:

The state of the multipage form is managed using the `useState` hook. It maintains the following state variables:

`step`: is of type number,
Tracks the current form progress.

`disable`: is of type boolean
Controls the "Next" button's disabled state.

`formData`: is an object
Each object of formData consists of a key that is used to refer to the current step.
The corresponding value to key is an object that
has form data for each step, including label, input type, placeholder, pattern for validation, and stores the input value for that step.

The `useEffect` hook is used to keep track of step changes to update the UI accordingly, it checks if input value is exist or empty and validates if it exists to set the button state on the current step.

Input Validation:

Input validation is performed for each step's input fields. The validation is based on the `pattern` specified in the `formData`. The "Next" button is disabled when input does not meet the validation criteria.

Step Navigation:

Users can navigate between steps by clicking the "Next" or "Back" buttons. The component handles step progression and regression, updating the `step` state and managing the form data accordingly.

Success Message:

Upon reaching the final step (Step 4) and submitting, a success message is displayed to indicate a successful form submission.

Accessibility:

The component incorporates accessibility features to ensure a better user experience. It adheres to ARIA (Accessible Rich Internet Applications) guidelines for roles, states, and properties.

Test Cases:

1. Verify that all provided items are displayed on each step.
2. Test the navigation to ensure that the input is correctly set in the corresponding state objects.
4. Check for data persistence on navigating backwards.
3. State management and re-renders should be handled correctly.
5. For styling, test for correct styles to be displayed at corresponding input or events.