A FULLSTACK REACT *MASTERCLASS*
TINYHOUSE

Quiz Questions | Answers

Module 9 | Lesson 9.2 - Styling with Ant Design

1. Which of the following best describes how we've handled the loading and error states of our GraphQL query?

A: We have no data to display to the user when the query is loading or has errored, so we display the skeleton UI when the query is loading and we display the skeleton UI + an error alert if an error has occurred.
B: We don't display anything to the user when the query is loading or has errored.
C: When the query is loading or has failed, we resort to displaying an alert above the already existing list.
D: We only display the skeleton UI when the query is either loading or has errored.

Answer: A - We have no data to display to the user when the query is loading or has errored, so we display the skeleton UI when the query is loading and we display the skeleton UI + an error alert if an error has occurred.

2. What would we expect be displayed to the user when the mutation has errored, in the code example below?

```
export const HelloWorld = () => {
  const [request, { data, loading, error }] = useMutation(
    MUTATION
  );

  if (loading) {
    return <h2>Loading...</h2>;
  }

  return (
    <div>
      <h2>Hello World!</h2>
      <button onClick={request}>Request mutation</button>
    </div>
  );
};
```

A: The "Loading..." header element.
B: A blank page.
C: An error alert above the "Hello World!" header message.
D: No UI change will occur when an error is made. The "Hello World!" header message and button will continue to be displayed after loading is complete.

Answer: D - No UI change will occur when an error is made. The "Hello World!" header message and button will continue to be displayed after loading is complete.