**Lab 7 React Unit Testing with Jest**

1. Import dependencies.
   1. Add the following import statements at the top of your test file:

import React from 'react';

import Greeter from './Greeter';

1. Write the test cases.
   1. Use the `test` or `it` function provided by your testing framework to define individual test cases.
   2. Write a test case to verify that the `Greeter` class provides a greeting for a given name:

('provides a greeting', () => {

const greeter = new Greeter('Ada Lovelace');

const greeting = greeter.getGreeting();

expect(greeting).toBe('Hello, Ada Lovelace');

});

* 1. Write a test case to verify that the `Greeter` class provides a default greeting for an anonymous user:

test('provides a default greeting', () => {

const greeter = new Greeter();

const greeting = greeter.getGreeting();

expect(greeting).toBe('Hello, Anonymous');

});

* 1. Write a test case to verify that the `Greeter` class can generate a greeting asynchronously:

test('Can generate a greeting asynchronously', async () => {

const greeter = new Greeter('Ada Lovelace');

const greeting = await greeter.getGreetingAsync();

expect(greeting).toBe('Hello, Ada Lovelace');

});

1. Run tests:
   1. In your terminal or command prompt, navigate to the project directory. Execute the command to run your tests using your chosen testing framework (e.g., `test`).
   2. Observe the test results in the terminal or command prompt, which will indicate whether the tests passed or failed.
2. For App.test.js, add the following import statements at the top of your test file:

import React from 'react';

import { render, fireEvent } from '@testing-library/react';

import App from './App';

1. Write test cases.
   1. Use the `test` or `it` function provided by your testing library to define individual test cases.
   2. Write a test case to verify that the `App` component renders two message elements:

test('renders two message elements', () => {

const app = render(<App />);

const inputHeadings = app.getAllByText(/message:/i);

expect(inputHeadings.length).toBe(2);

});

1. Write a test case to verify that the `App` component has a `GreetingComponent` displaying "Anonymous":

test('has a GreetingComponent', () => {

const app = render(<App />);

const greetingComponent = app.getByText(/Anonymous/i);

expect(greetingComponent).toBeInTheDocument();

});

1. Write a test case to verify that changing the input updates the displayed message:

test('changing the input updates the message', () => {

const app = render(<App />);

const input = app.getByLabelText(/user name/i);

fireEvent.change(input, { target: { value: 'Ada Lovelace' } });

const message1 = app.getByText(/Hello, Ada Lovelace/i);

expect(message1).toBeInTheDocument();

const clearButton = app.getByText(/clear/i);

fireEvent.click(clearButton);

const message2 = app.getByText(/Hello, Anonymous/i);

expect(message2).toBeInTheDocument();

});

1. Run the tests.