React-19

**Title:** Unit Testing and Mocking in React using Jest

**Objective:**  
To create and test a React application that fetches and displays GitHub repository names using mocked API calls, demonstrating testing in isolation with Jest and Axios.

**Learning Outcomes:**

* Understand the importance of isolation in unit testing.
* Learn to use mocking and spying effectively.
* Use Jest to implement unit testing with mocked dependencies.

**Prerequisites:**

* Node.js
* NPM
* Visual Studio Code

**Estimated Time to Complete:**

60 minutes

**Scenario:**

As part of an internship project at OpenAI, you are tasked with building and testing a React application that fetches a list of public repositories from GitHub for a specific user. The goal is to ensure that the module can be tested independently using Jest and mocks.

**Implementation Instructions:**

**1. Create the React Application**

Use the following command in the terminal to create the app:

npx create-react-app gitclientapp

Open the project in Visual Studio Code:

cd gitclientapp

code .

**2. Install Axios**

Install the Axios library to make HTTP requests:

npm install axios

**3. Create GitClient Module**

Create a file named GitClient.js in the src folder and define the following class:

import axios from 'axios';

class GitClient {

async getRepositories(user) {

const response = await axios.get(`https://api.github.com/users/${user}/repos`);

return response.data.map(repo => repo.name);

}

}

export default GitClient;

**4. Modify App Component**

Update App.js to use the GitClient module:

import React, { useEffect, useState } from 'react';

import GitClient from './GitClient';

function App() {

const [repos, setRepos] = useState([]);

useEffect(() => {

const client = new GitClient();

client.getRepositories('techiesyed').then(setRepos);

}, []);

return (

<div>

<h1>Repository List</h1>

<ul>

{repos.map((repo, index) => <li key={index}>{repo}</li>)}

</ul>

</div>

);

}

export default App;

**5. Run the Application**

Build and run the application using:

npm start

**6. Create Unit Test File**

Create a new test file named GitClient.test.js in the src folder. Add the following test suite:

import axios from 'axios';

import GitClient from './GitClient';

jest.mock('axios');

describe('Git Client Tests', () => {

test('should return repository names for techiesyed', async () => {

const dummyData = [

{ name: 'repo1' },

{ name: 'repo2' }

];

axios.get.mockResolvedValue({ data: dummyData });

const client = new GitClient();

const repos = await client.getRepositories('techiesyed');

expect(repos).toEqual(['repo1', 'repo2']);

expect(axios.get).toHaveBeenCalledWith('https://api.github.com/users/techiesyed/repos');

});

});

**7. Run the Tests**

Use the following command to execute unit tests:

npm test

**Conclusion:**

By completing this lab, you will have successfully created a React application and implemented unit tests that isolate external API calls using Jest and Axios mocking. This approach ensures reliable and independent testing of application logic.