UserList Component

Framework

React

Goal

Test asynchronous behaviour by simulating a fetch request and verifying UI updates during loading, success, and error states.

Instructions

1. Create a userService module with a method fetchUsers() that returns a Promise of user data.
2. Use jest.mock() to stub the service in your test file and control the returned data or errors.
3. Write a test that checks the component shows “Loading...” initially.
4. Once the data resolves, use findByText to verify that at least one user’s name appears in the DOM.
5. Add a test that simulates a rejected Promise and verifies that an error message or fallback UI is shown.
6. Ensure async handling uses useEffect and useState, and avoid using real network calls.

Tech & Tools

- Jest mocks (jest.mock(), mockResolvedValue, mockRejectedValue)
- React Testing Library: findByText, queryByText, waitFor
- useEffect, useState

What to Test

- "Loading..." displays initially
- Fetched data renders after promise resolves
- Error message or fallback appears if fetch fails
- No unhandled promise rejections