In a Selenium test for a React application, once you have validated that the login page works and the user is successfully logged in, you can verify that you are on the app's home page by checking for specific elements, URLs, or other indicators unique to the home page. Here are a few approaches you can use:

1. **Check for Unique Elements**: Identify one or more elements that are present on the home page but not on other pages. These elements could be part of the navigation bar, header, or content that's specific to the home page.
2. **Verify URL**: If the URL changes when the user is redirected to the home page after successful login, you can verify the current URL to ensure it matches the expected URL of the home page.
3. **Page Title**: Verify the page title to make sure it matches the expected title of the home page.
4. **Custom Attributes**: If your home page has unique custom attributes or data attributes, you can use them to identify the page.

Here's an example of how you might use some of these approaches in your Selenium test:

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.chrome.ChromeDriver;

public class SampleSeleniumTest {

public static void main(String[] args) {

WebDriver driver = new ChromeDriver();

driver.get("your\_login\_page\_url");

// Log in and navigate to the home page

// ...

// Check for a unique element on the home page

WebElement homePageElement = driver.findElement(By.id("unique-element-id"));

if (homePageElement.isDisplayed()) {

System.out.println("Successfully on the home page.");

} else {

System.out.println("Not on the home page.");

}

// Or check the URL to verify the home page

String expectedHomePageUrl = "https://example.com/home";

if (driver.getCurrentUrl().equals(expectedHomePageUrl)) {

System.out.println("Successfully on the home page.");

} else {

System.out.println("Not on the home page.");

}

driver.quit();

}

}