Test Automation using:

- Java
- React
- Node.JS
- Auth0
- Visual Studio Code
- Vitest (framework)
- Libraries (ie., @testing-library, jest-dom)
- Git and GitHub repository

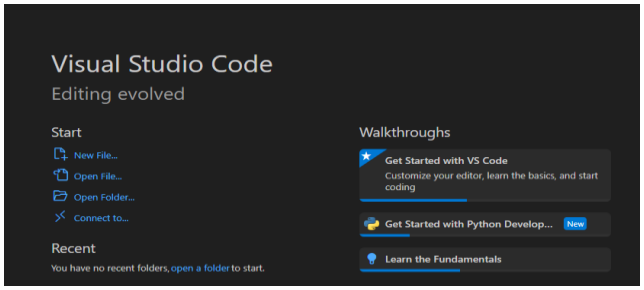
Document highlight:

- Tools, packages, and libraries installation and setup
- Create and run test cases
  - Greet
  - User Account
  - User List
  - Product Image Gallery
  - Terms And Conditions
  - Expandable Text
- Simplifying the Tests
- Final end result

1. Base installation and setup

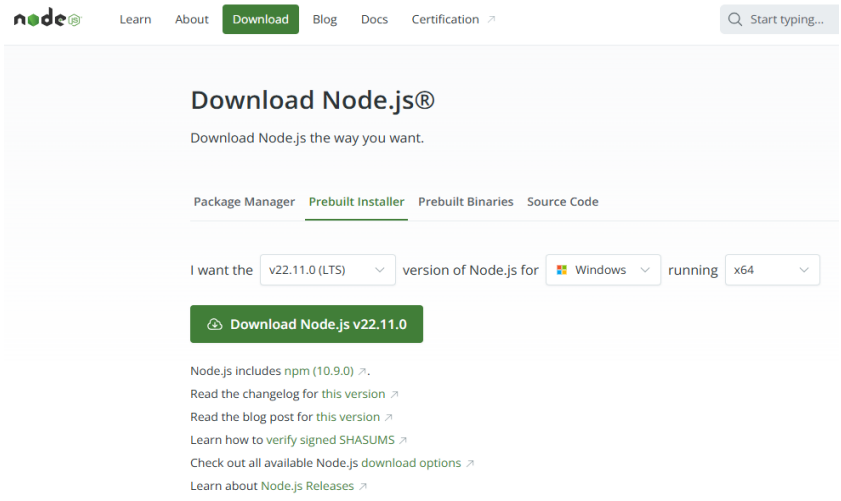
Ensure that the Visual Studio Code is setup…

Download:
https://code.visualstudio.com/download

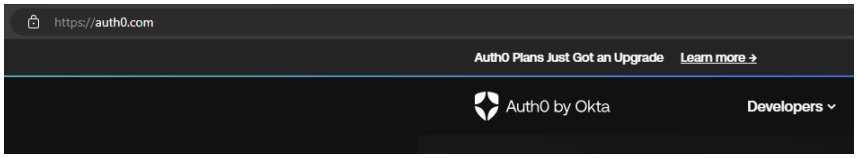


Ensure that the Node.JS is setup

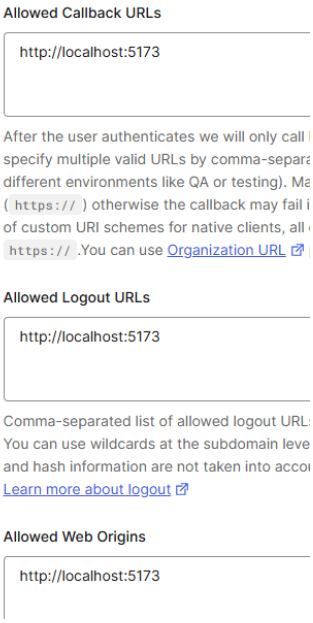
Download:
https://nodejs.org/en/download/prebuilt-installer



Ensure to have an Auth0 account:
https://auth0.com/



Update Auth0 config details under settings



Install node components:

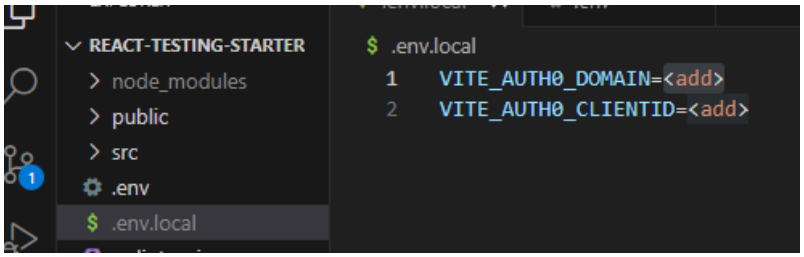
```
PS C:\Users\njmlo\react-testing-starter> npm i

added 538 packages, and audited 539 packages in 28s
```

```
PS C:\Users\njmlo\react-testing-starter> npm -v
10.9.0
```

Open in VS Code:

```
PS C:\Users\njmlo\react-testing-starter> code .
```

Make a copy of the “env” file, in the copied file fill in the details off of the Auth0 app:

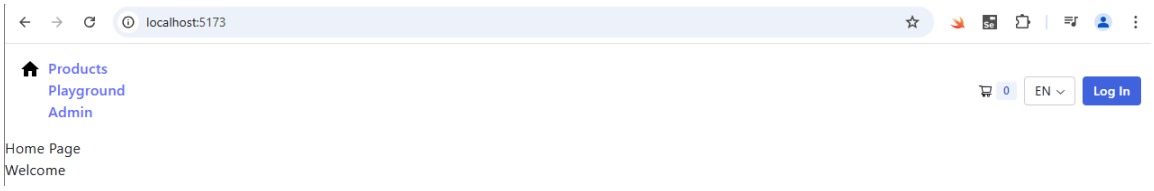


Start npm:

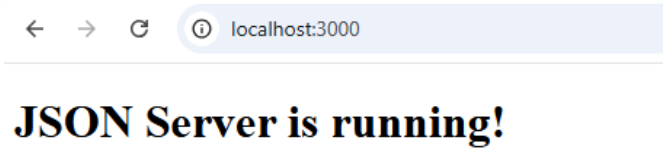
```
PS C:\Users\njmlo\react-testing-starter> npm start
```

```
> react-testing-vite@0.0.0 start
> concurrently "npm run server" "npm run dev"

(node:15388) [DEP0060] DeprecationWarning: The `util._extend` API is deprecated. Please use Object.assign() instead.
(Use `node --trace-deprecation ...` to show where the warning was created)
[1]
[1] > react-testing-vite@0.0.0 dev
[1] > vite
[1]
[0]
[0] > react-testing-vite@0.0.0 server
[0] > json-server --watch src/data/db.json --delay 500
[0]
[0]
[0]   \{^_^}/ hi!
[0]
[0]   Loading src/data/db.json
[0]   Done
[1]
[1]   VITE v5.4.11  ready in 515 ms
[1]
[1]   ➜  Local:   http://localhost:5173/
[1]   ➜  Network: use --host to expose
[0]
[0]   Resources
[0]   http://localhost:3000/categories
[0]   http://localhost:3000/products
[0]
[0]   Home
[0]   http://localhost:3000
[0]
[0]   Type s + enter at any time to create a snapshot of the database
[0]   Watching...
[0]
```

Front-end url:
http://localhost:5173



Back-end url:
http://localhost:3000

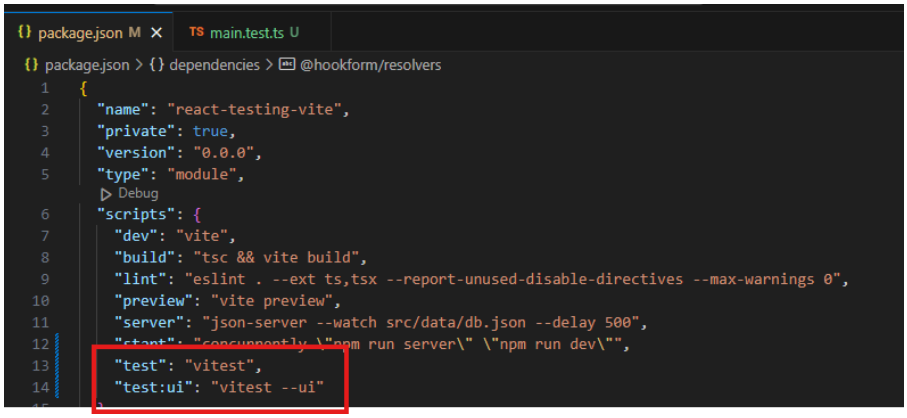


2. Install the testing framework Vitest:

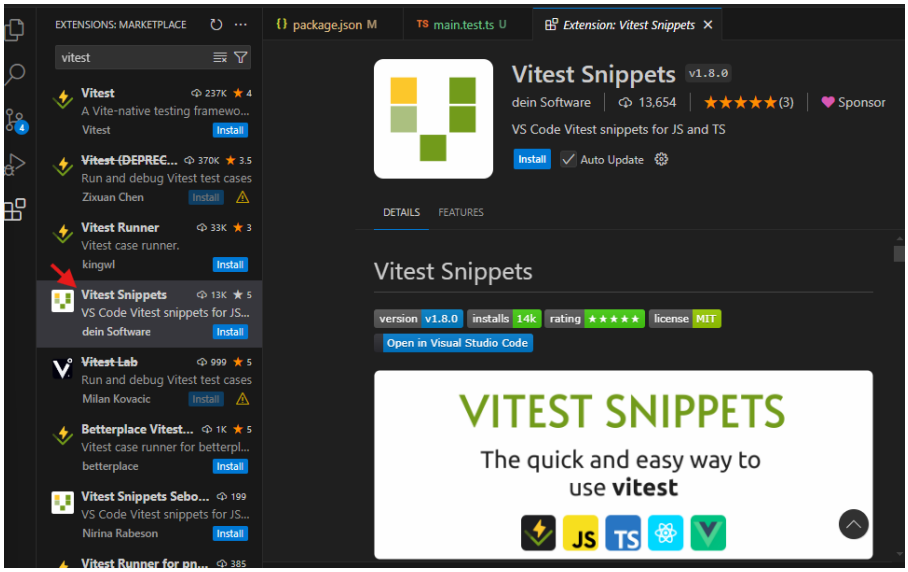
```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

Microsoft Windows [Version 10.0.22631.4460]
(c) Microsoft Corporation. All rights reserved.

C:\Users\njmlo\react-testing-starter>npm i -D vitest

added 30 packages, removed 1 package, changed 3 packages, and audited 570 packages in 8s

found 0 vulnerabilities

C:\Users\njmlo\react-testing-starter>
```

```
C:\Users\njmlo\react-testing-starter>npm vitest -v
10.9.0

C:\Users\njmlo\react-testing-starter>
```

Update the package.json file and include (under the section of scripts objects) the new dependency:



Install the needed extension to be able to use the essential functions of Vitest:

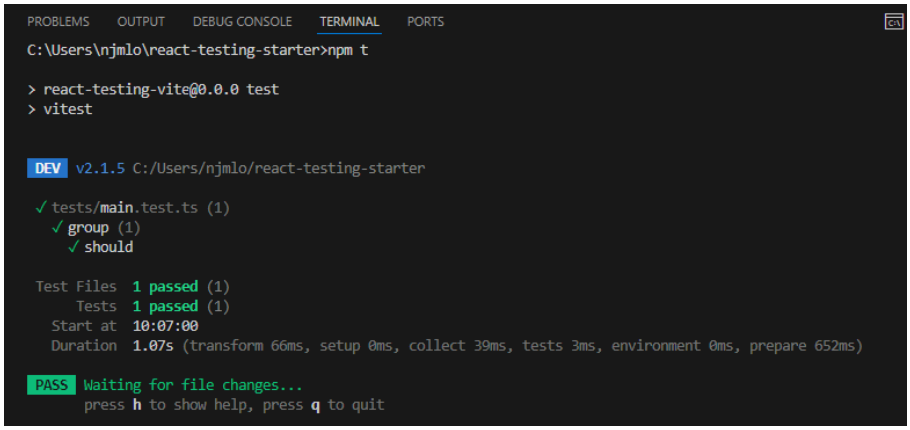


To test that the setup was set properly and runs as expected:

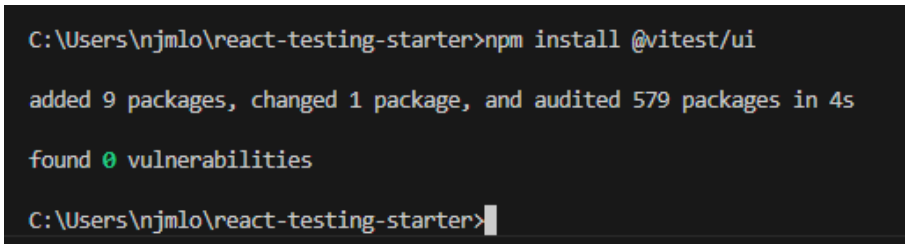
Create a folder and file

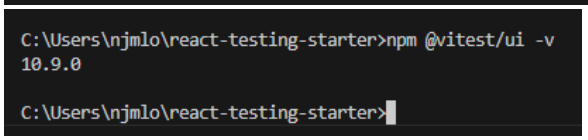
Code a test suite and run it:

In the terminal window, run test npm t:



In the terminal window, install @vitest/ui:



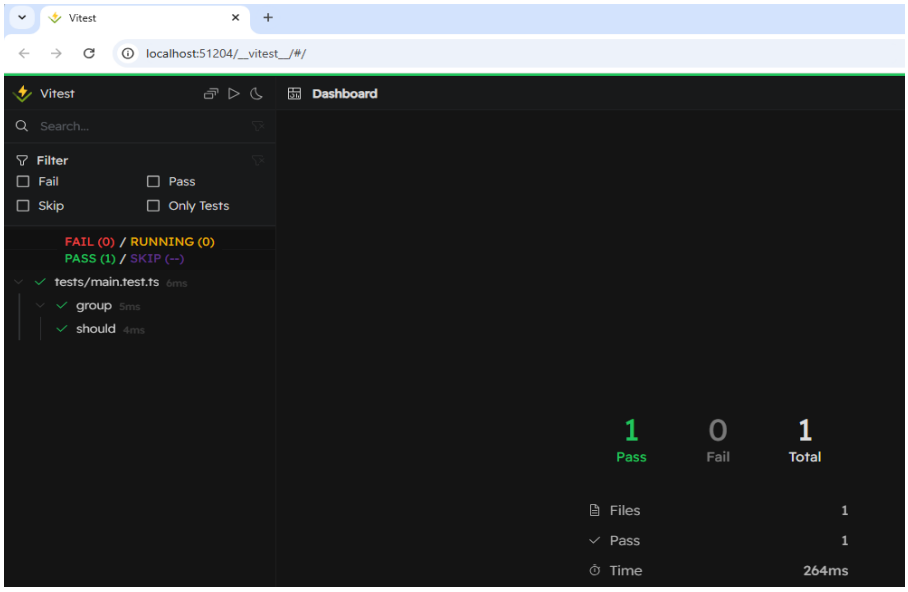


In the terminal window, run test npm test ui:

```
C:\Users\njmlo\react-testing-starter>npm run test:ui

> react-testing-vite@0.0.0 test:ui
> vitest --ui


 DEV  v2.1.5 C:/Users/njmlo/react-testing-starter
      UI started at http://localhost:51204/__vitest__/

 ✓ tests/main.test.ts (1)
   ✓ group (1)
     ✓ should

 Test Files  1 passed (1)
      Tests  1 passed (1)
   Start at  10:23:10
   Duration  1.35s (transform 146ms, setup 0ms, collect 73ms, tests 6ms, environment 0ms, prepare 262ms)

 PASS  Waiting for file changes...
       press h to show help, press q to quit
```

```
C:\Users\njmlo\react-testing-starter>npm test:ui -v
10.9.0

C:\Users\njmlo\react-testing-starter>
```

Test UI:



3. Setup React testing libraries

Install libraries @testing-library/react and @testing-library/dom :

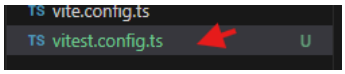
```
C:\Users\njmlo\react-testing-starter>npm i -D @testing-library/react @testing-library/dom

added 16 packages, and audited 595 packages in 5s

found 0 vulnerabilities

C:\Users\njmlo\react-testing-starter>
```

```
C:\Users\njmlo\react-testing-starter>npm @testing-library/react -v
10.9.0

C:\Users\njmlo\react-testing-starter>
```

Install library jsdom:

```
C:\Users\njmlo\react-testing-starter>npm i -D jsdom

added 28 packages, and audited 623 packages in 5s

found 0 vulnerabilities

C:\Users\njmlo\react-testing-starter>npm jsdom -v
10.9.0

C:\Users\njmlo\react-testing-starter>
```

Create a new config file from the root directory:



TS vitest.config.ts U

TS vitest.config.ts > default > test > environment

```ts
import {defineConfig} from 'vitest/config';

export default defineConfig({
    test:{
        environment: 'jsdom'
    }
});
```

To test that all installed are working at this point:

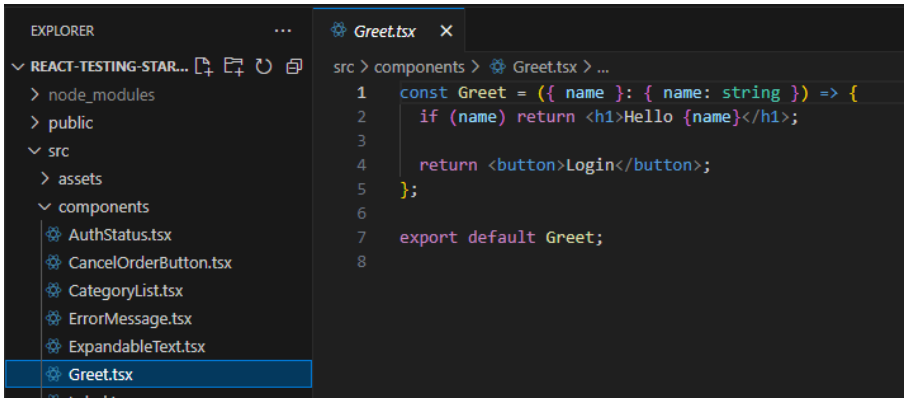
```
C:\Users\njmlo\react-testing-starter>npm run test:ui

> react-testing-vite@0.0.0 test:ui
> vitest --ui


 DEV  v2.1.5 C:/Users/njmlo/react-testing-starter
      UI started at http://localhost:51204/__vitest__/

 ✓ tests/main.test.ts (1)
   ✓ group (1)
     ✓ should

 Test Files  1 passed (1)
      Tests  1 passed (1)
   Start at  11:25:43
   Duration  2.69s (transform 74ms, setup 0ms, collect 50ms, tests 4ms, environment 1.18s, prepare 207ms)

 PASS  Waiting for file changes...
       press h to show help, press q to quit
```

Install library jest-dom:

```
C:\Users\njmlo\react-testing-starter>npm i -D @testing-library/jest-dom

added 14 packages, and audited 637 packages in 3s

found 0 vulnerabilities

C:\Users\njmlo\react-testing-starter>npm @testing-library/jest-dom -v
10.9.0

C:\Users\njmlo\react-testing-starter>
```

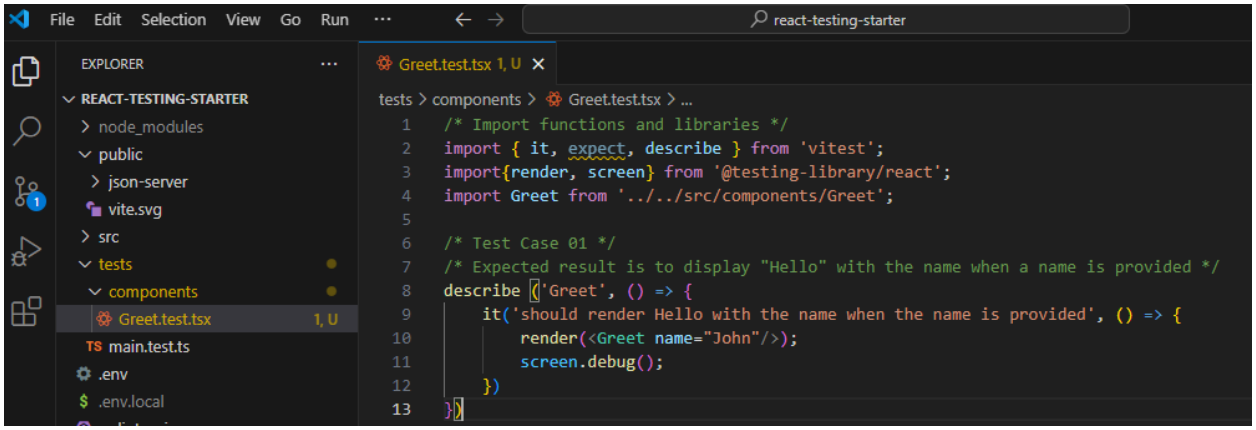
4. Create and run test cases

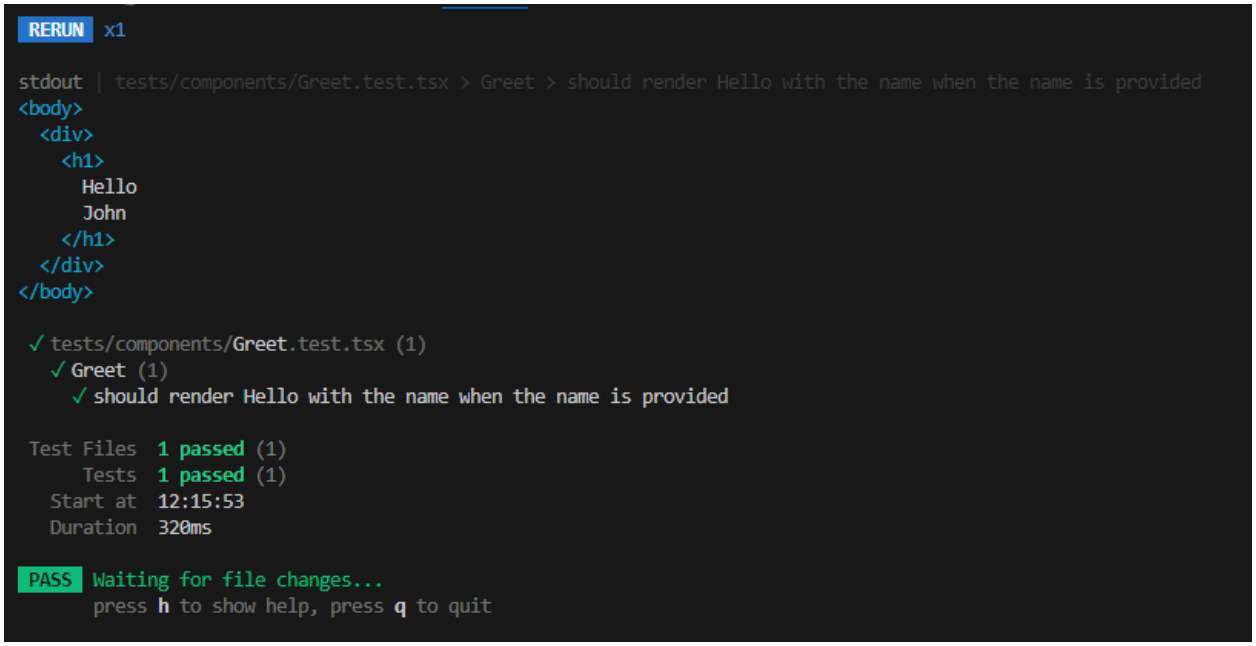
Component to test: Greet.tsx



Test case 01:

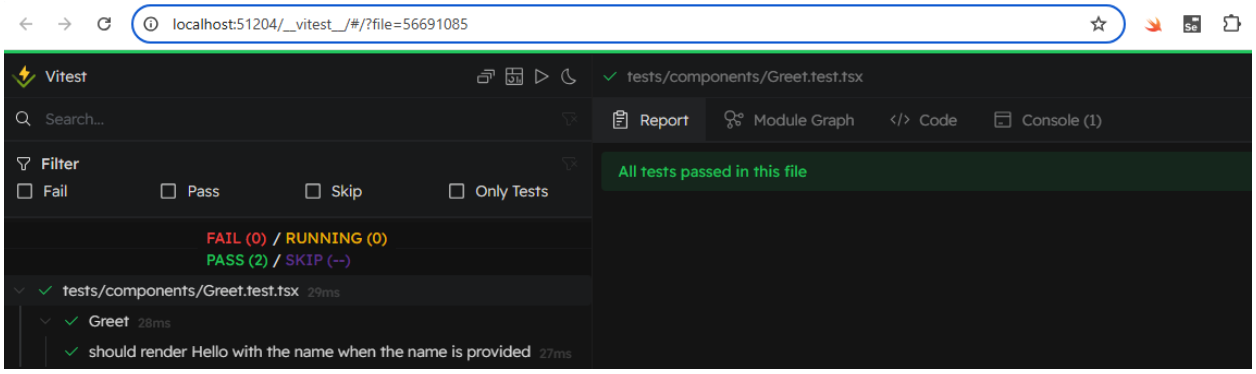
Create a test file and test case…

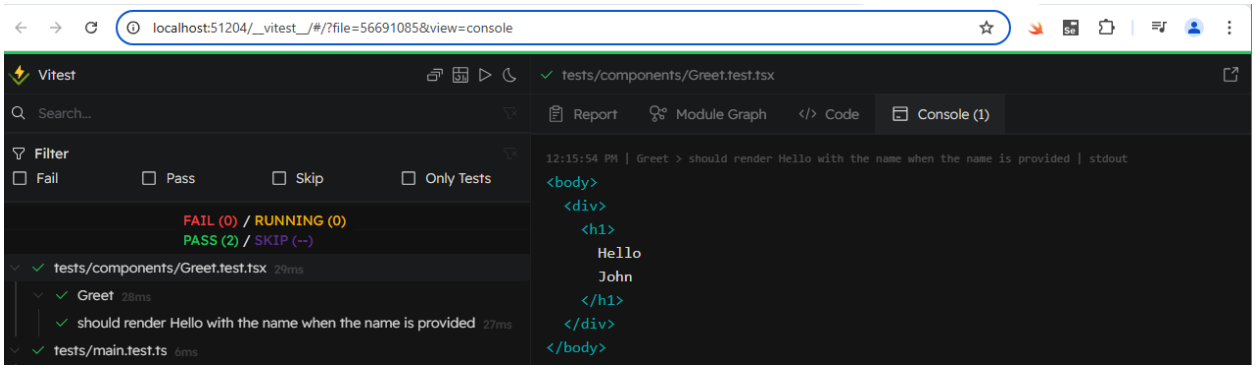



RERUN x1
stdout | tests/components/Greet.test.tsx > Greet > should render Hello with the name when the name is provided
<body>

<h1>
Hello
John
</h1>

</body>
✓ tests/components/Greet.test.tsx (1)
✓ Greet (1)
✓ should render Hello with the name when the name is provided
Test Files 1 passed (1)
Tests 1 passed (1)
Start at 12:15:53
Duration 320ms
PASS Waiting for file changes...
press h to show help, press q to quit

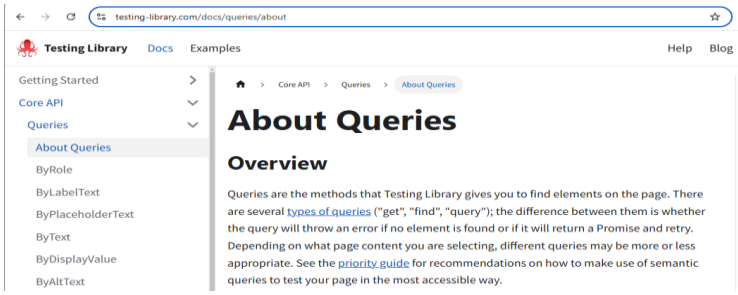

localhost:51204/__vitest__/#/?file=56691085
Vitest
Search...
Filter
Fail
Pass
Skip
Only Tests
FAIL (0) / RUNNING (0)
PASS (2) / SKIP (--)
tests/components/Greet.test.tsx 29ms
Greet 28ms
should render Hello with the name when the name is provided 27ms
tests/components/Greet.test.tsx
Report
Module Graph
Code
Console (1)
All tests passed in this file


localhost:51204/__vitest__/#/?file=56691085&view=console
Vitest
Search...
Filter
Fail
Pass
Skip
Only Tests
FAIL (0) / RUNNING (0)
PASS (2) / SKIP (--)
tests/components/Greet.test.tsx 29ms
Greet 28ms
should render Hello with the name when the name is provided 27ms
tests/main.test.ts 6ms
tests/components/Greet.test.tsx
Report
Module Graph
Code
Console (1)
12:15:54 PM | Greet > should render Hello with the name when the name is provided | stdout
<body>

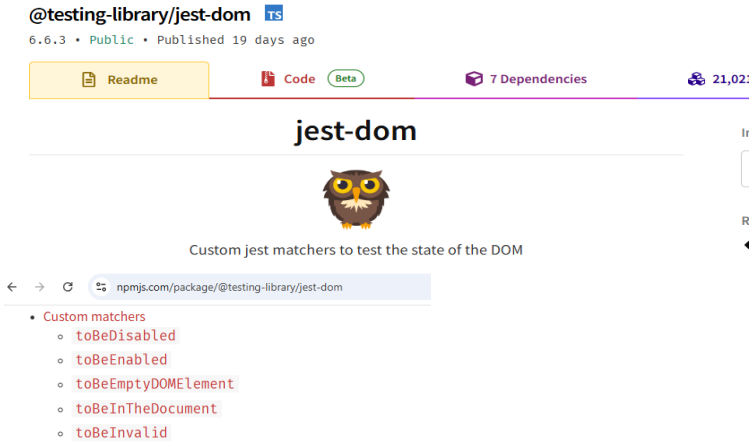
<h1>
Hello
John
</h1>

</body>

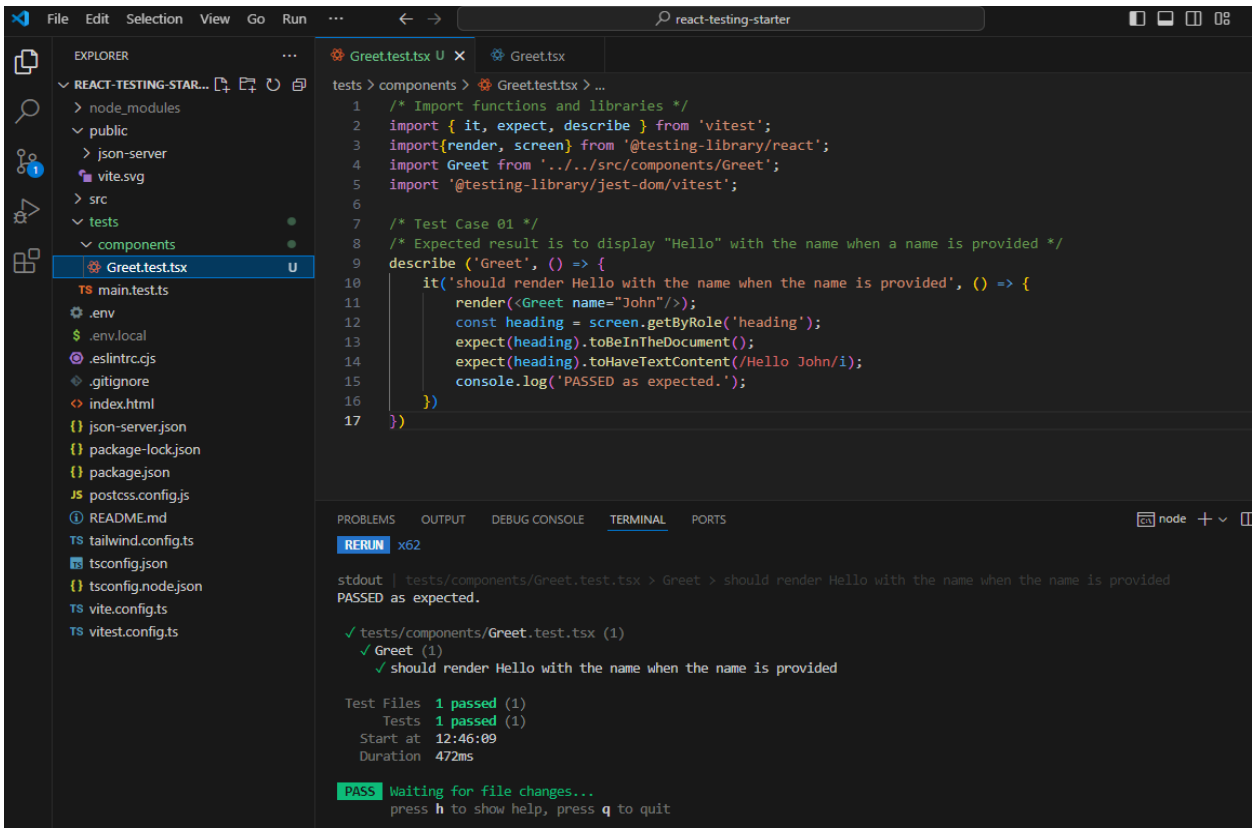
To find the <body> <h1> in the dom and make accession, use testing library query methods:

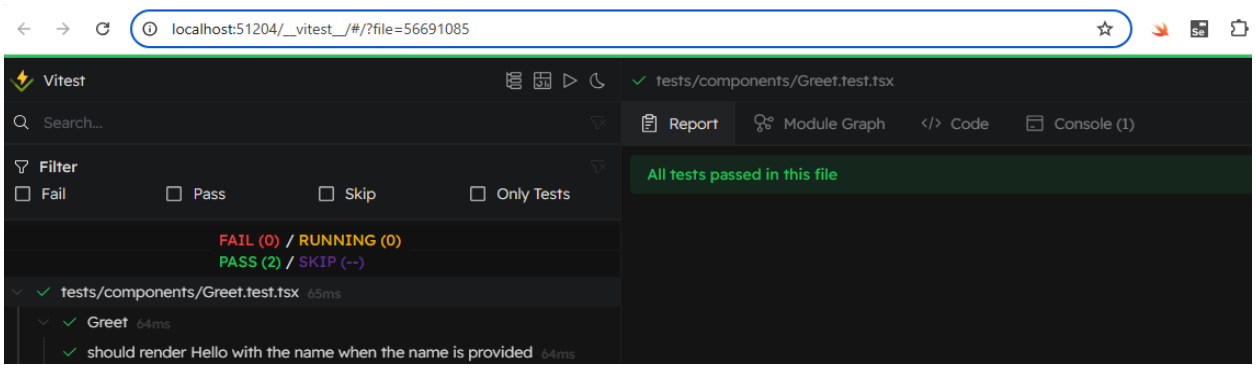


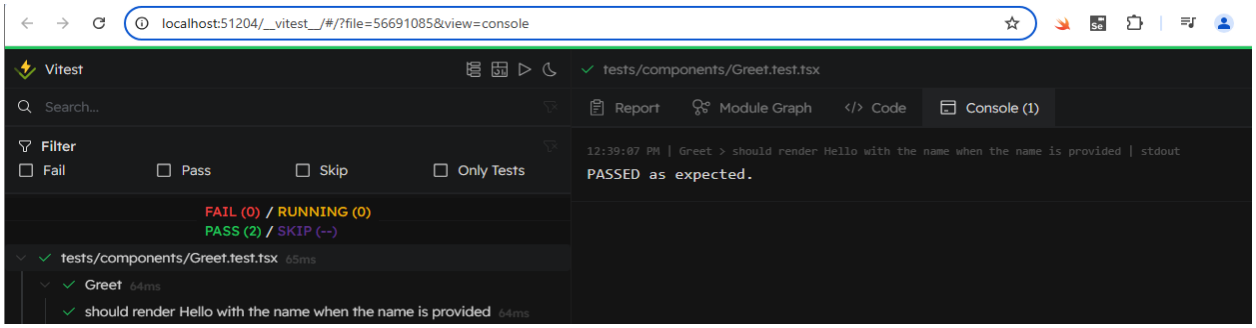
To use jest-dom custom matchers:



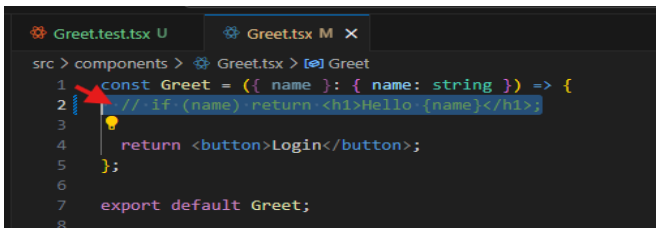
After modifying the test case, rerun the test:

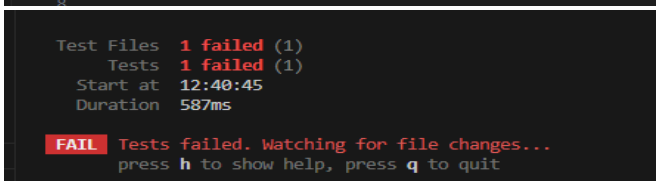


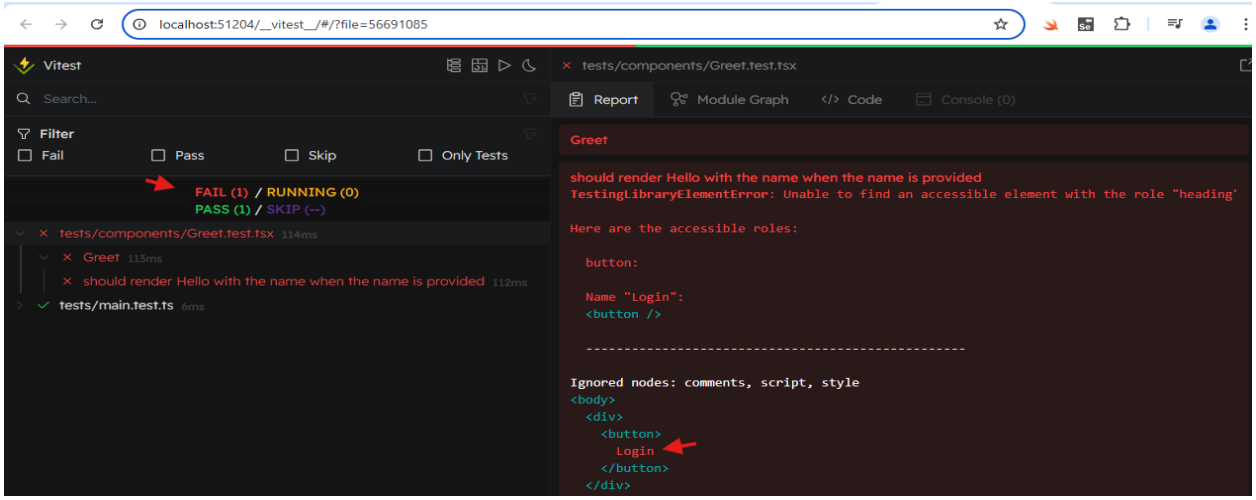




Running a negative test (fail the test case):





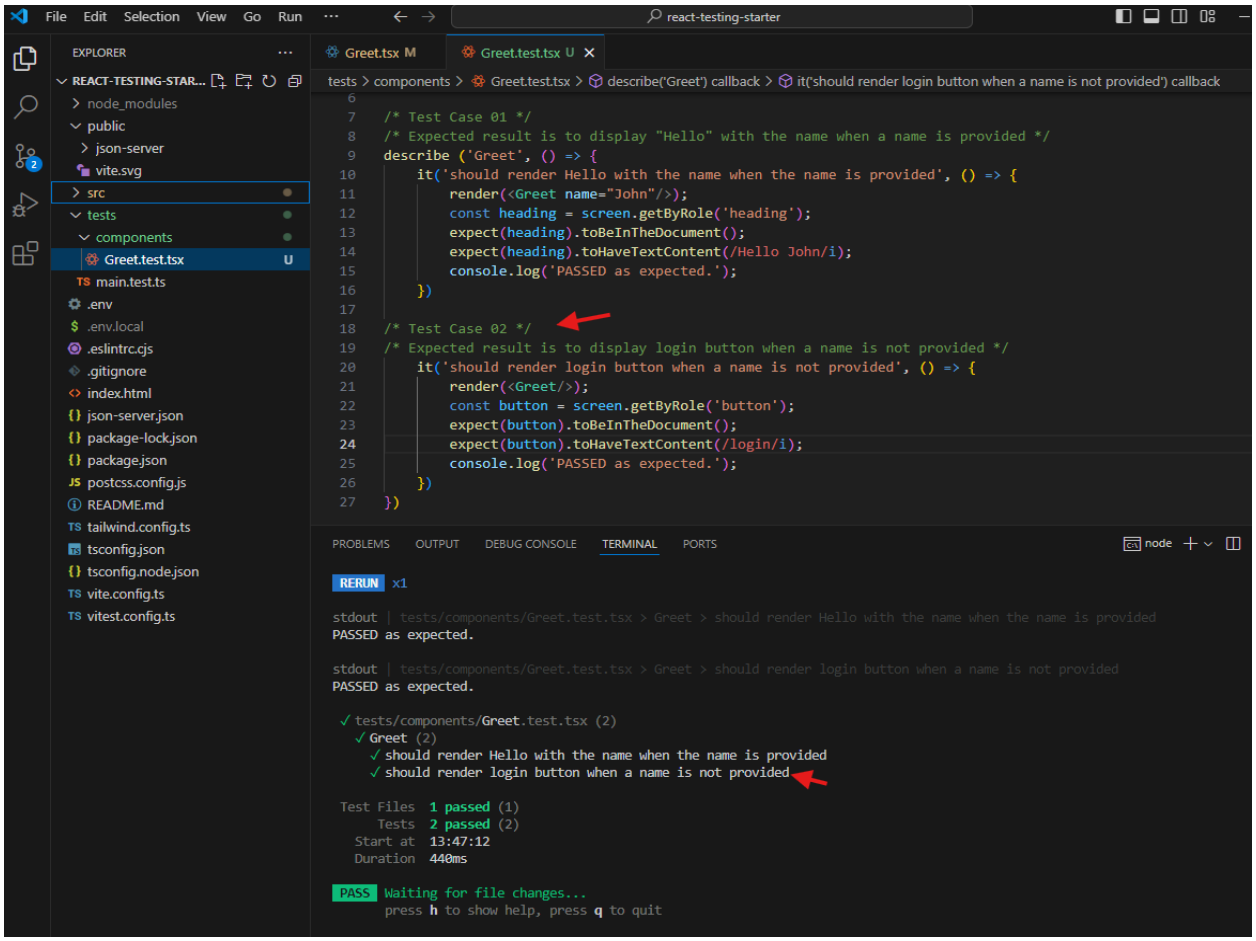


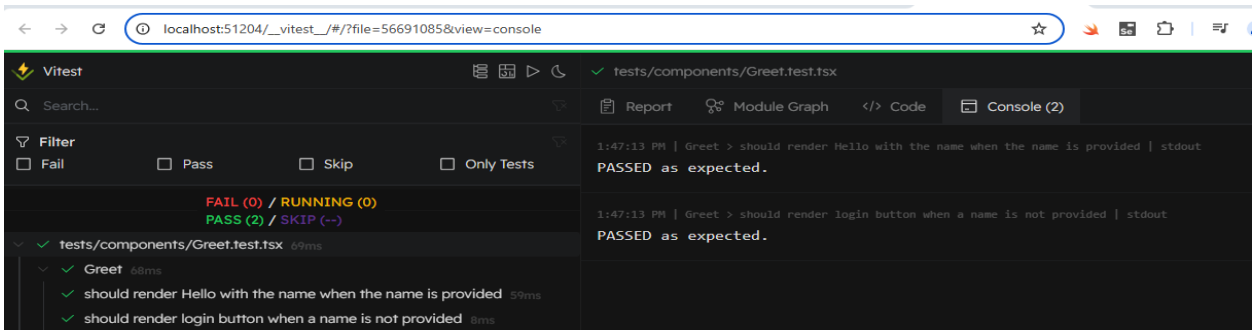
Test case 02:

What we are testing…

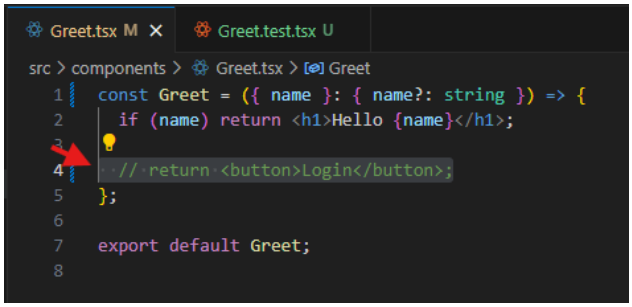
```
Greet.tsx M   Greet.test.tsx U
src > components > Greet.tsx > Greet > name
1  const Greet = ({ name }: { name: string }) => {
2    if (name) return <h1>Hello {name}</h1>;
3
4    return <button>Login</button>;
5  };
6
7  export default Greet;
8
```

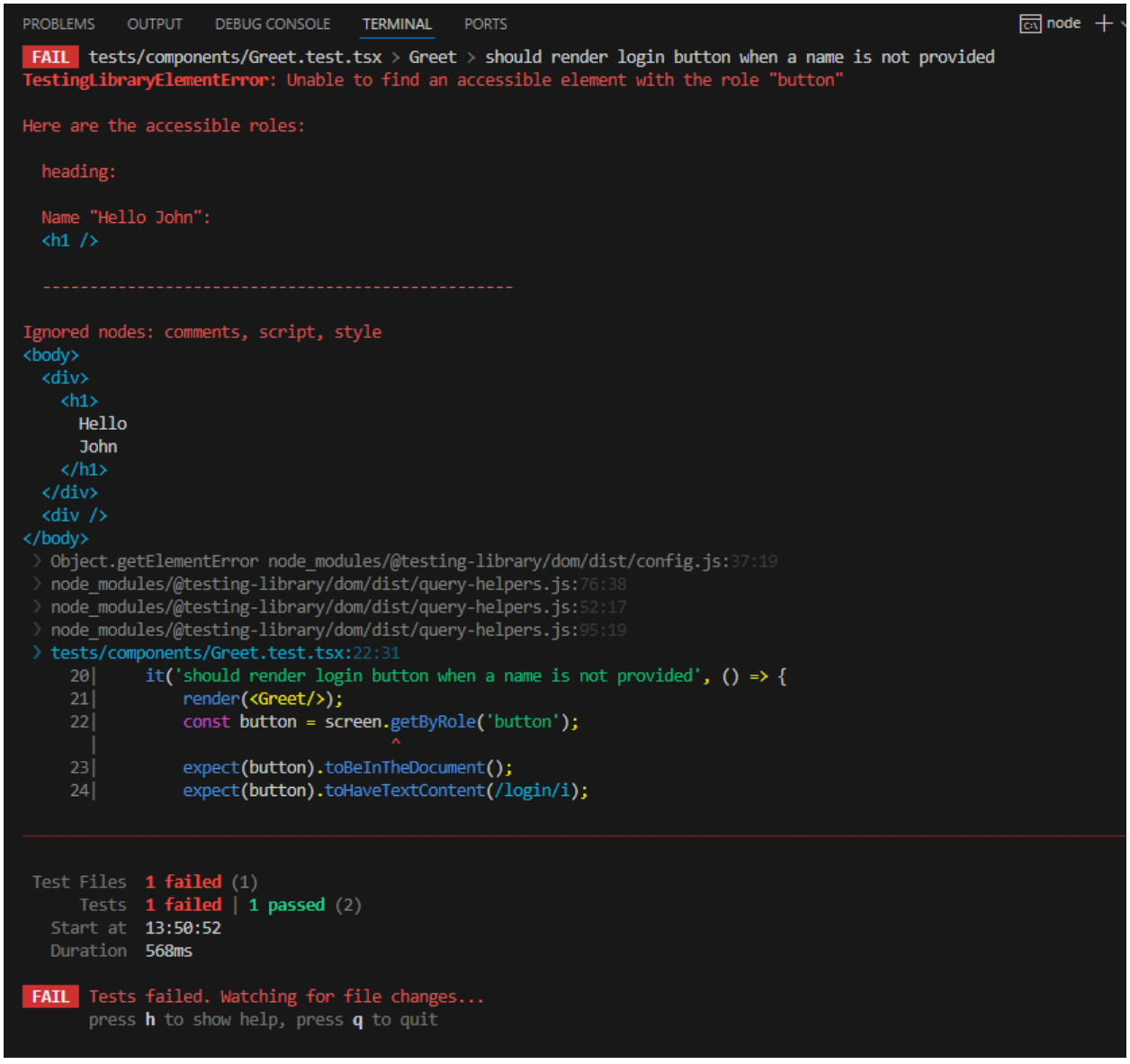
Test case and outcome…

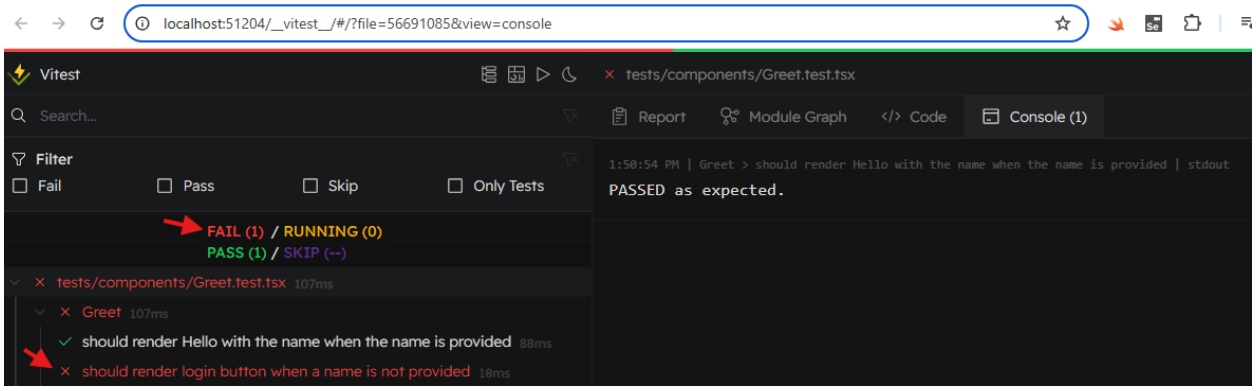




Running a negative test (fail the test case):



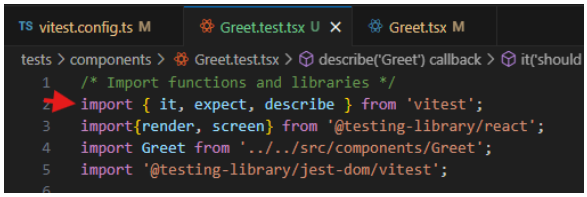




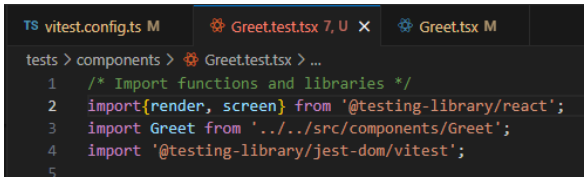
5. Simplifying the test setup

In the vitest.config.ts file, setting the globals for Vitest to true…

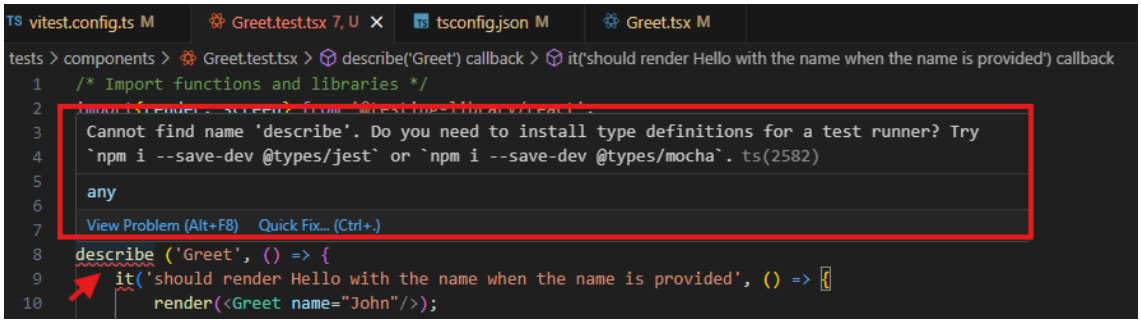
So no need to have this line in every test file…



After removing…

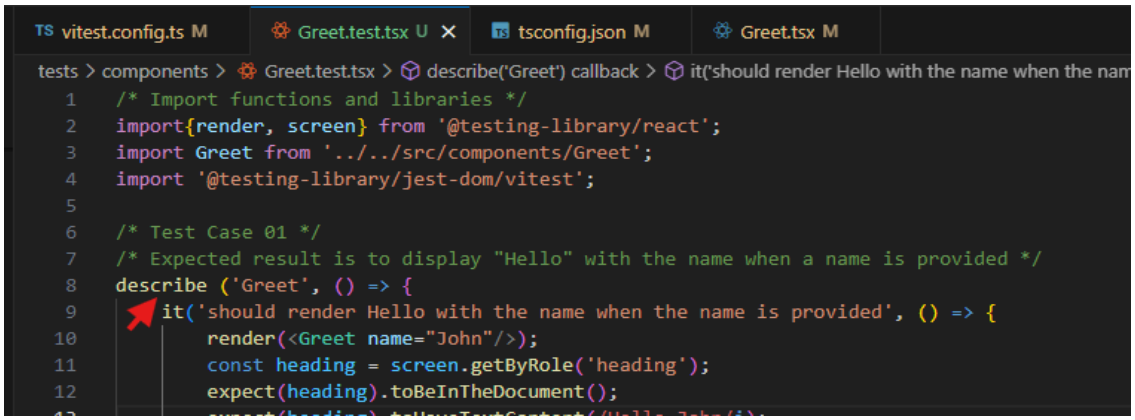


Another item or error to address…

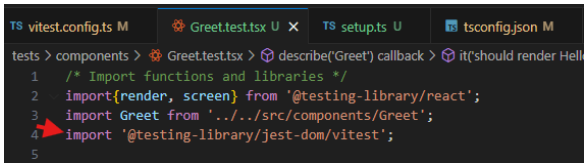


Update tsconfig.json file…

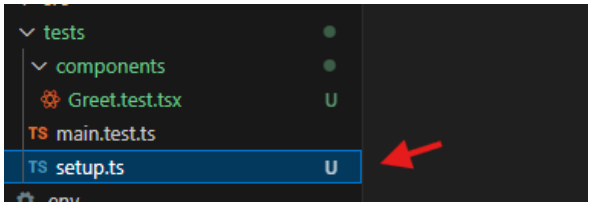
After updating the config file…



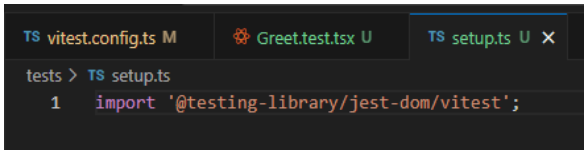
Another import line to move into global setting…



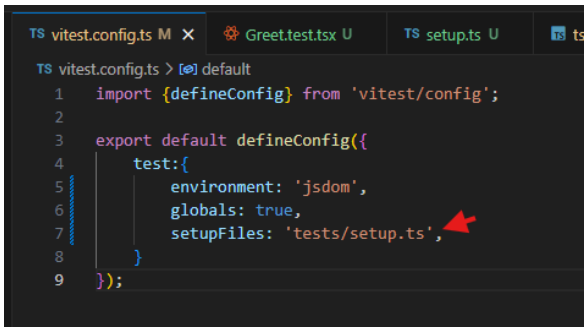
So, in test folder, create a setup.ts file…



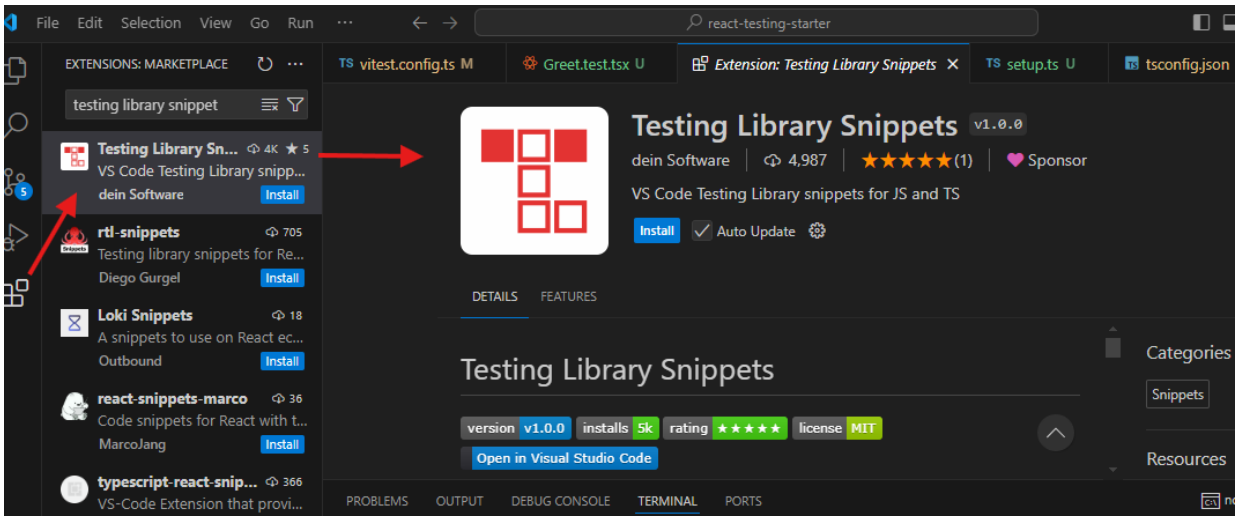
Move the import line into the setup file…



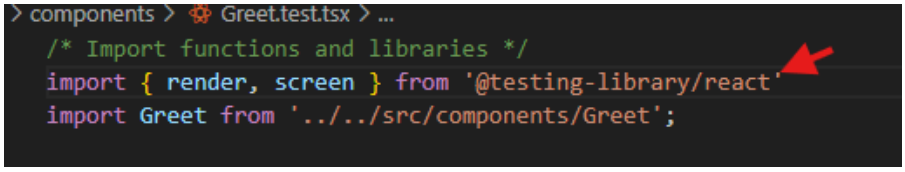
Reference the file from the config file…



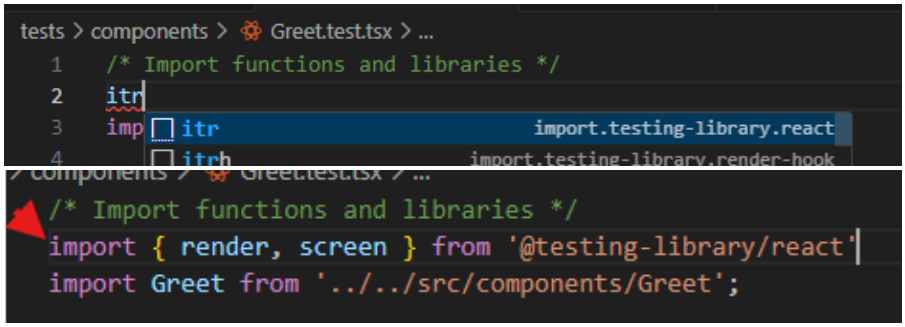
Install Testing Library Snippets extension…



Now, instead of typing this long line…

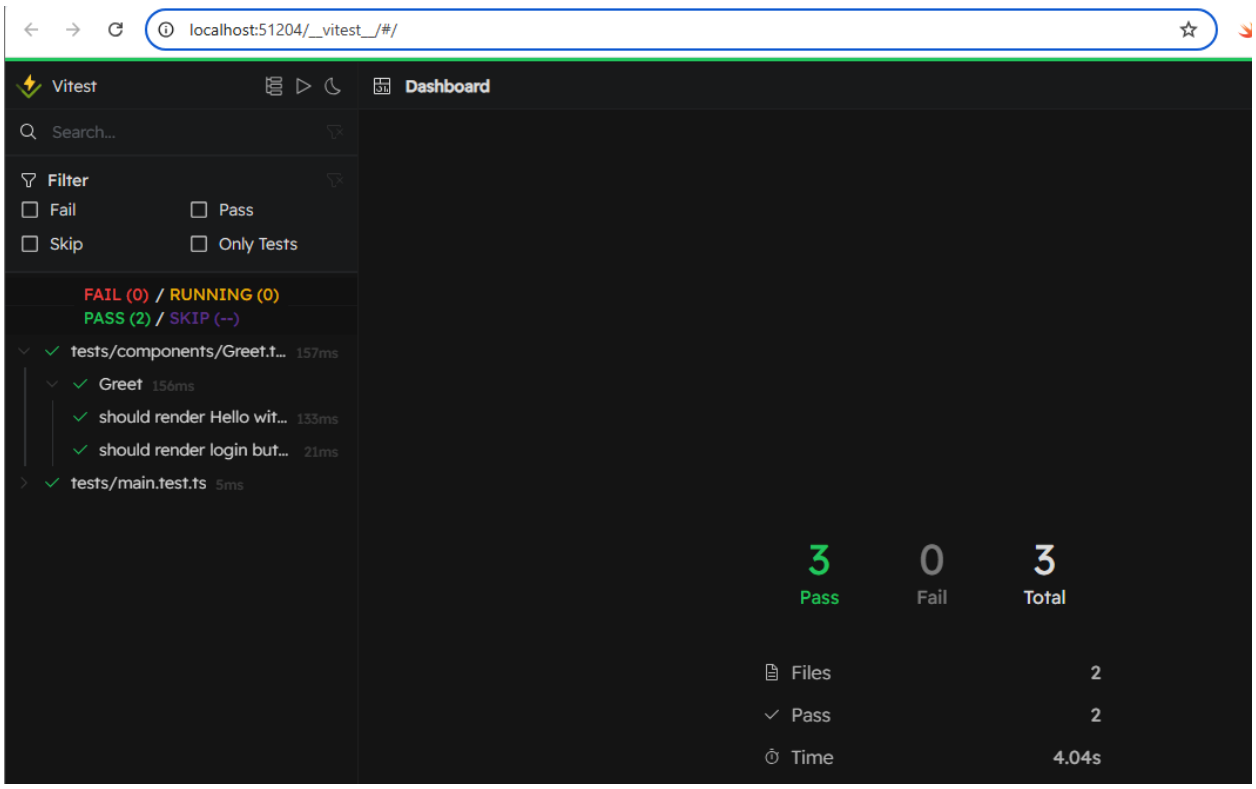


Simply key in “itr” and hit enter…



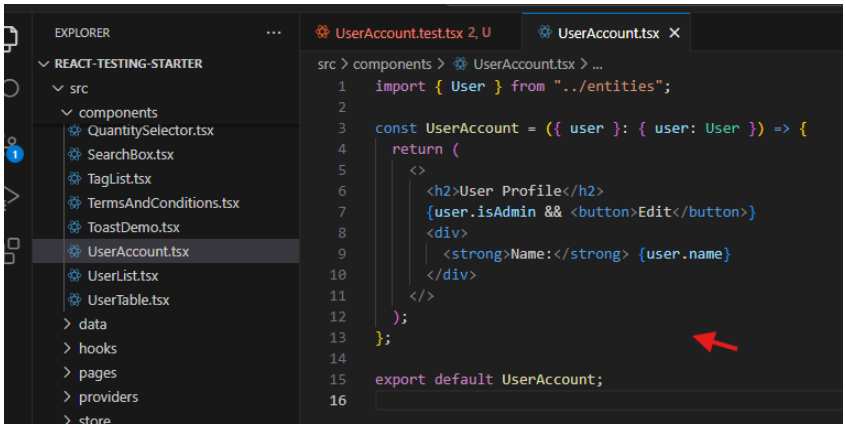
To ensure that all the changes are properly setup and test cases still passed, run the test:

```
C:\Users\njmlo\react-testing-starter>npm run test:ui

> react-testing-vite@0.0.0 test:ui
> vitest --ui


 DEV  v2.1.5 C:/Users/njmlo/react-testing-starter
      UI started at http://localhost:51204/__vitest__/

stdout | tests/components/Greet.test.tsx > Greet > should render Hello with the name when the name is provided
PASSED as expected.

stdout | tests/components/Greet.test.tsx > Greet > should render login button when a name is not provided
PASSED as expected.

 ✓ tests/main.test.ts (1)
 ✓ tests/components/Greet.test.tsx (2)

 Test Files  2 passed (2)
      Tests  3 passed (3)
   Start at  14:27:12
   Duration  3.49s (transform 135ms, setup 649ms, collect 376ms, tests 162ms, environment 2.42s, prepare 431ms)

 PASS  Waiting for file changes...
       press h to show help, press q to quit
```



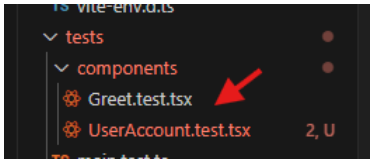
6. Testing User Account

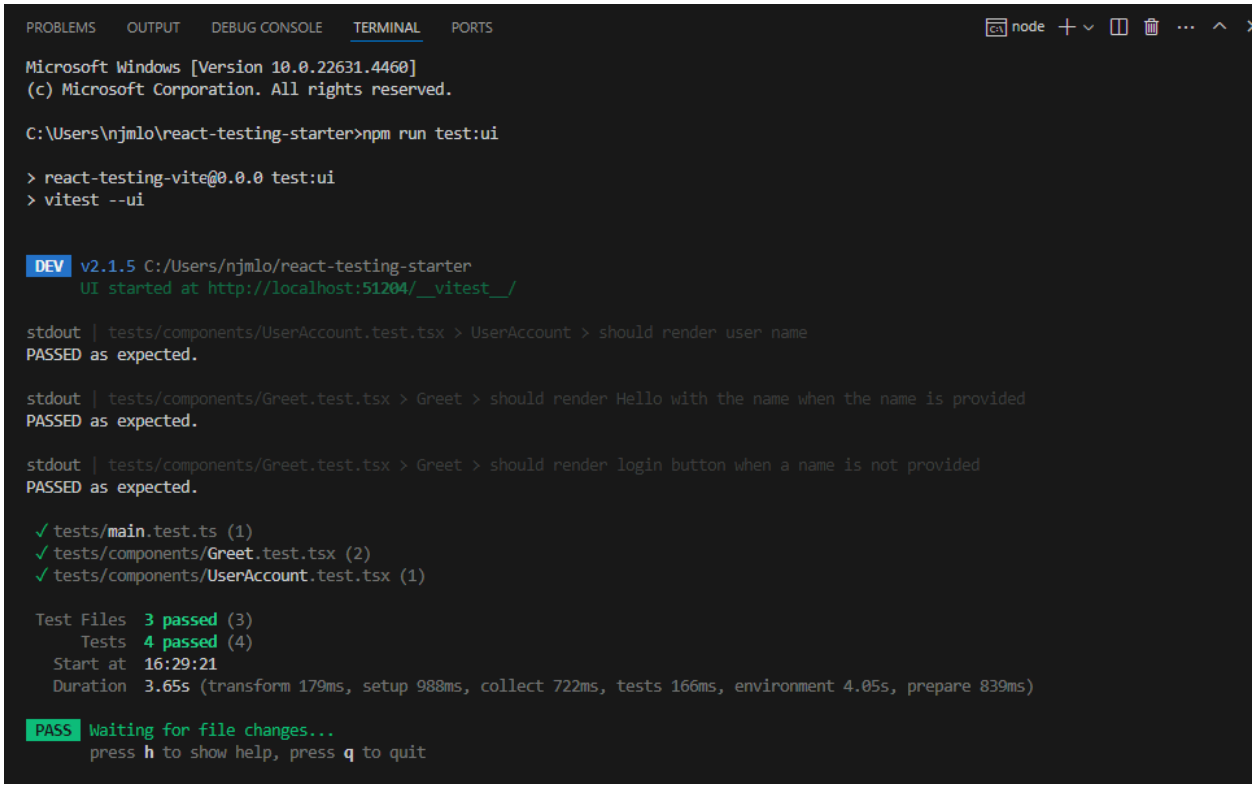
Component to test: UserAccount.tsx

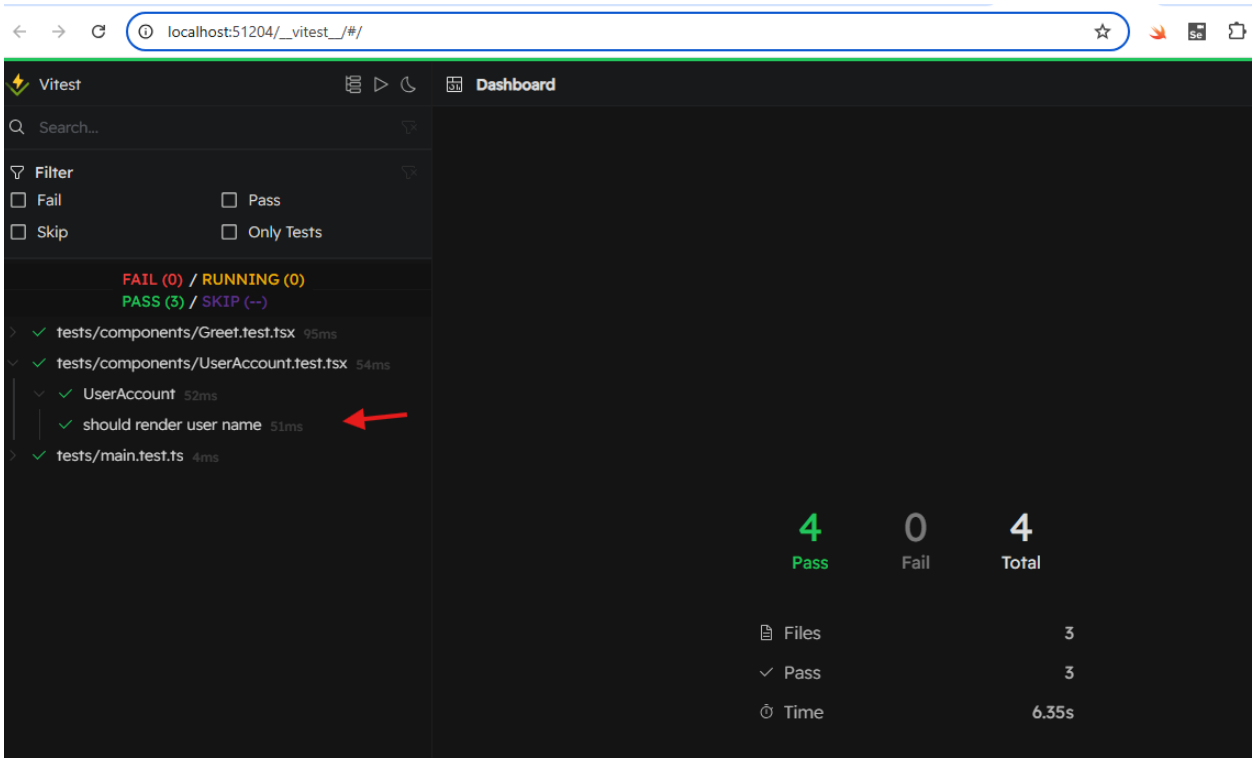


Test case 01:

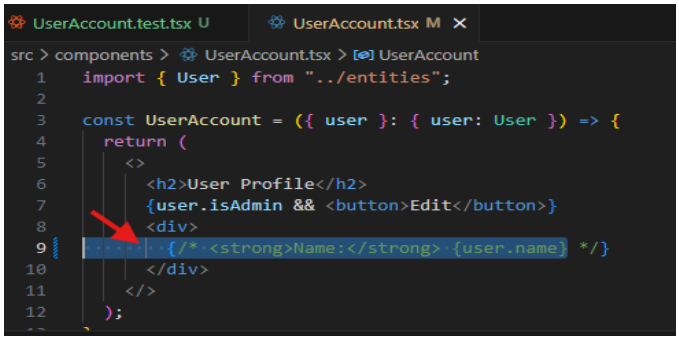
Create a test file…

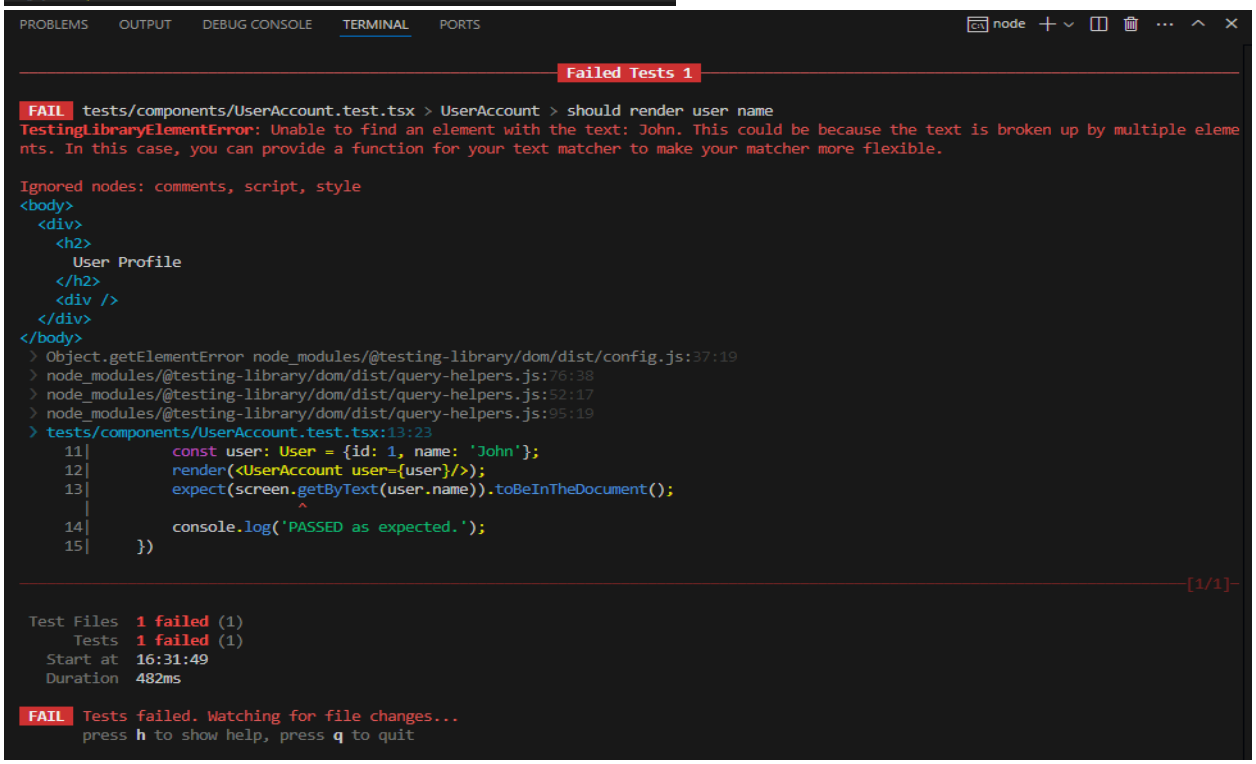


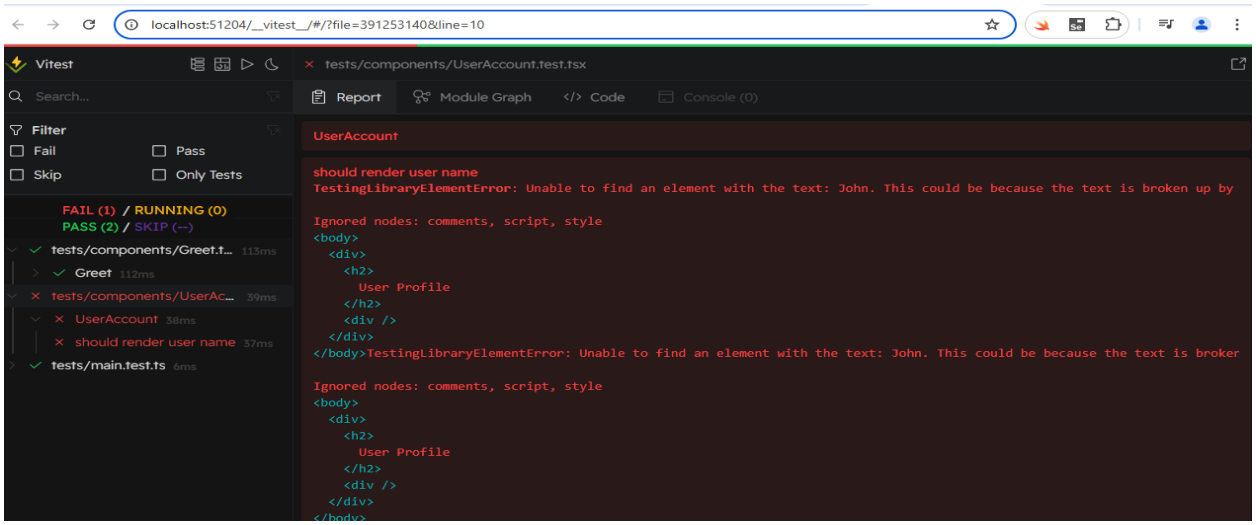

PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
node
Microsoft Windows [Version 10.0.22631.4460]
(c) Microsoft Corporation. All rights reserved.
C:\Users\njmlo\react-testing-starter>npm run test:ui
> react-testing-vite@0.0.0 test:ui
> vitest --ui
DEV v2.1.5 C:/Users/njmlo/react-testing-starter
UI started at http://localhost:51204/__vitest__/
stdout | tests/components/UserAccount.test.tsx > UserAccount > should render user name
PASSED as expected.
stdout | tests/components/Greet.test.tsx > Greet > should render Hello with the name when the name is provided
PASSED as expected.
stdout | tests/components/Greet.test.tsx > Greet > should render login button when a name is not provided
PASSED as expected.
✓ tests/main.test.ts (1)
✓ tests/components/Greet.test.tsx (2)
✓ tests/components/UserAccount.test.tsx (1)
Test Files 3 passed (3)
Tests 4 passed (4)
Start at 16:29:21
Duration 3.65s (transform 179ms, setup 988ms, collect 722ms, tests 166ms, environment 4.05s, prepare 839ms)
PASS Waiting for file changes...
press h to show help, press q to quit


localhost:51204/__vitest__/#/
Vitest
Dashboard
Search...
Filter
Fail
Pass
Skip
Only Tests
FAIL (0) / RUNNING (0)
PASS (3) / SKIP (--)
tests/components/Greet.test.tsx 95ms
tests/components/UserAccount.test.tsx 54ms
UserAccount 52ms
should render user name 51ms
tests/main.test.ts 4ms
4
Pass
0
Fail
4
Total
Files 3
Pass 3
Time 6.35s

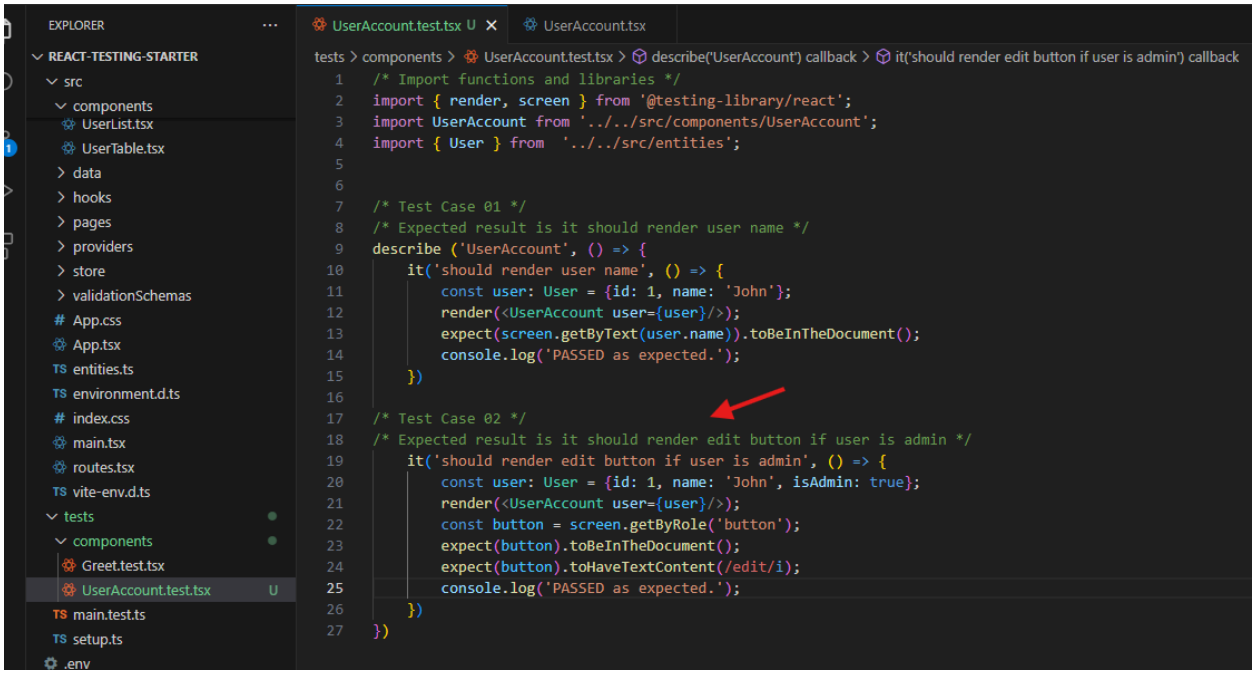
Running a negative test (fail the test case):

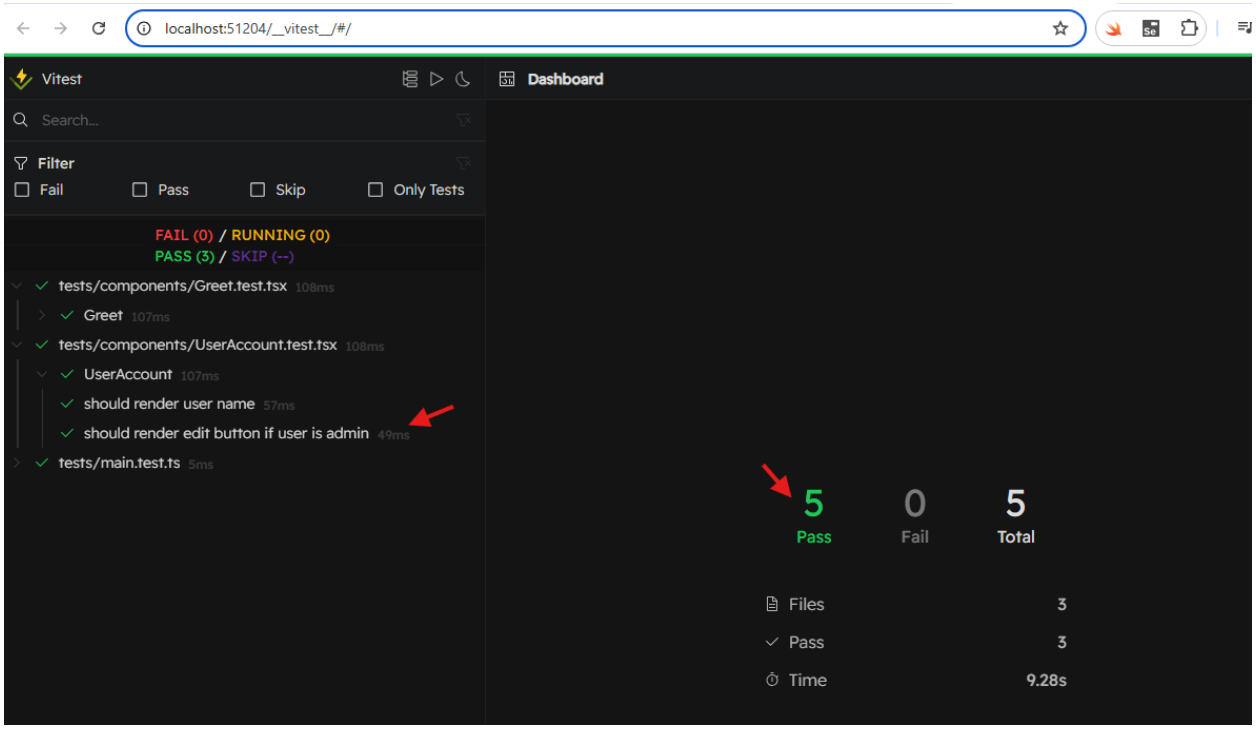






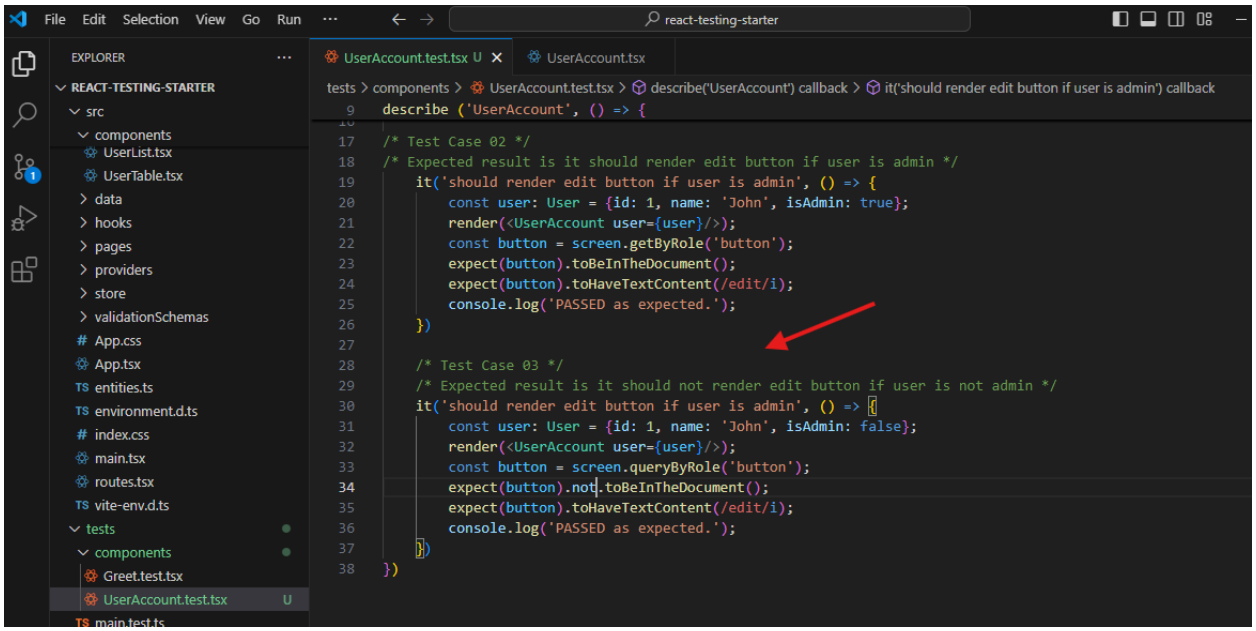
Test case 02:

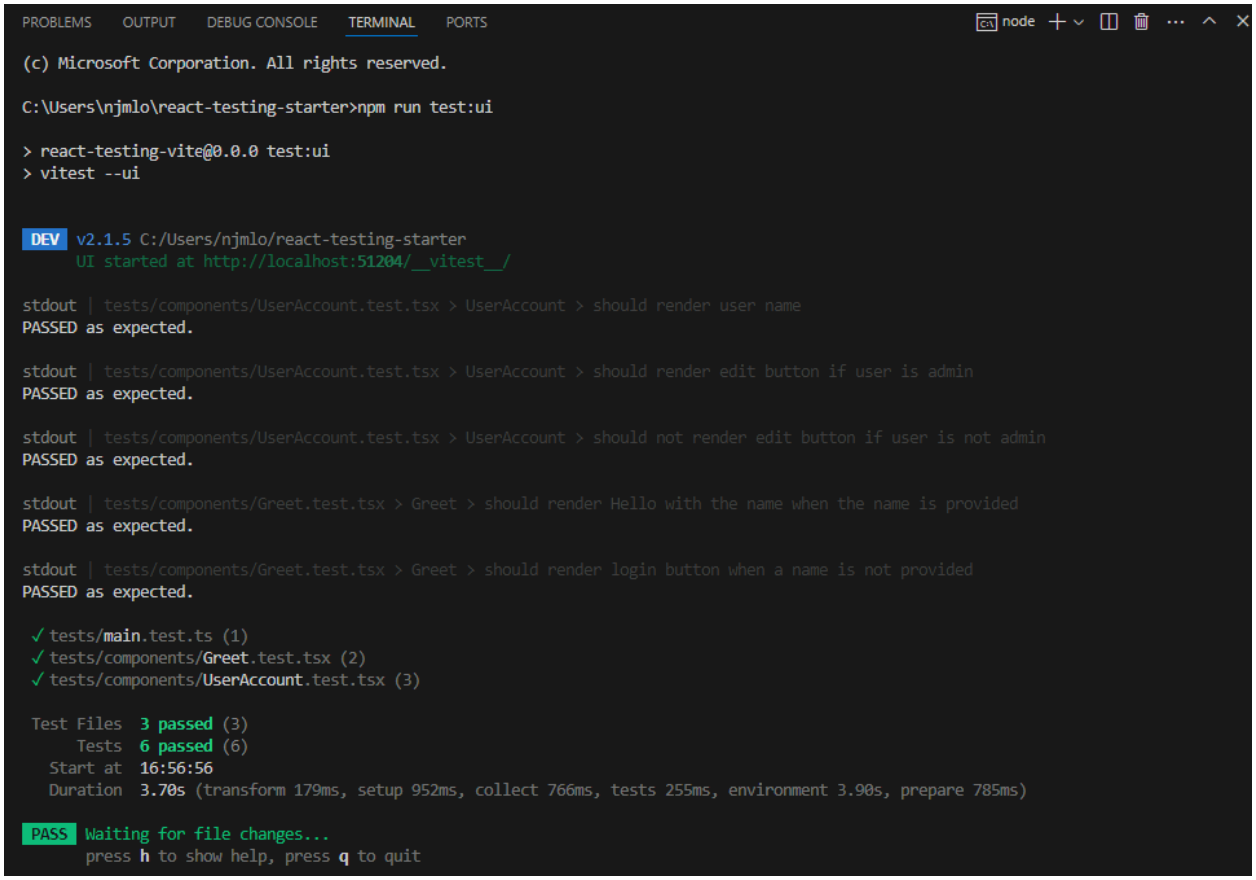

EXPLORER
REACT-TESTING-STARTER
src
components
UserList.tsx
UserTable.tsx
data
hooks
pages
providers
store
validationSchemas
App.css
App.tsx
entities.ts
environment.d.ts
index.css
main.tsx
routes.tsx
vite-env.d.ts
tests
components
Greet.test.tsx
UserAccount.test.tsx
main.test.ts
setup.ts
.env
UserAccount.test.tsx U
UserAccount.tsx
tests > components > UserAccount.test.tsx > describe('UserAccount') callback > it('should render edit button if user is admin') callback
/* Import functions and libraries */
import { render, screen } from '@testing-library/react';
import UserAccount from '../../src/components/UserAccount';
import { User } from '../../src/entities';
/* Test Case 01 */
/* Expected result is it should render user name */
describe ('UserAccount', () => {
it('should render user name', () => {
const user: User = {id: 1, name: 'John'};
render(<UserAccount user={user}/>);
expect(screen.getByText(user.name)).toBeInTheDocument();
console.log('PASSED as expected.');
})
/* Test Case 02 */
/* Expected result is it should render edit button if user is admin */
it('should render edit button if user is admin', () => {
const user: User = {id: 1, name: 'John', isAdmin: true};
render(<UserAccount user={user}/>);
const button = screen.getByRole('button');
expect(button).toBeInTheDocument();
expect(button).toHaveTextContent(/edit/i);
console.log('PASSED as expected.');
})
})

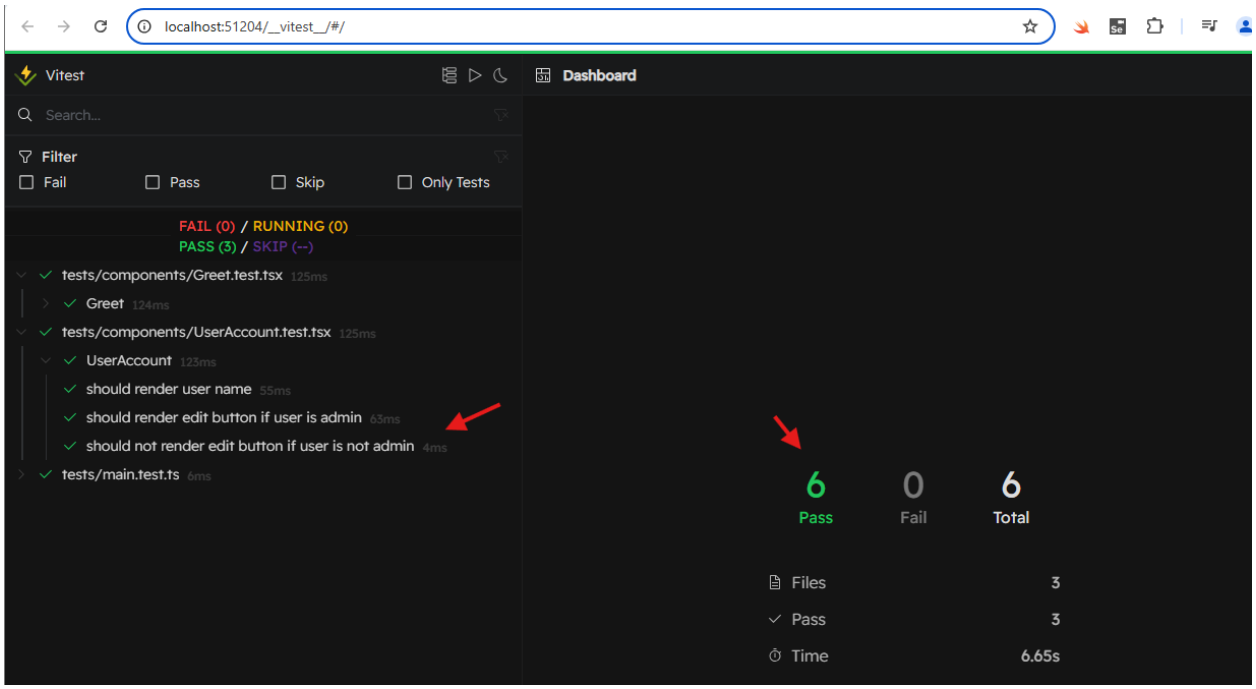

localhost:51204/__vitest__/#/
Vitest
Dashboard
Search...
Filter
Fail
Pass
Skip
Only Tests
FAIL (0) / RUNNING (0)
PASS (3) / SKIP (--)
tests/components/Greet.test.tsx 108ms
Greet 107ms
tests/components/UserAccount.test.tsx 108ms
UserAccount 107ms
should render user name 57ms
should render edit button if user is admin 49ms
tests/main.test.ts 5ms
5
Pass
0
Fail
5
Total
Files 3
Pass 3
Time 9.28s

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS                    node

      press h to show help, press q to quit

C:\Users\njmlo\react-testing-starter>npm run test:ui

> react-testing-vite@0.0.0 test:ui
> vitest --ui


 DEV  v2.1.5 C:/Users/njmlo/react-testing-starter
      UI started at http://localhost:51204/__vitest__/

stdout | tests/components/Greet.test.tsx > Greet > should render Hello with the name when the name is provided
PASSED as expected.

stdout | tests/components/Greet.test.tsx > Greet > should render login button when a name is not provided
PASSED as expected.

stdout | tests/components/UserAccount.test.tsx > UserAccount > should render user name
PASSED as expected.

stdout | tests/components/UserAccount.test.tsx > UserAccount > should render edit button if user is admin
PASSED as expected.

 ✓ tests/main.test.ts (1)
 ✓ tests/components/Greet.test.tsx (2)
 ✓ tests/components/UserAccount.test.tsx (2)

 Test Files  3 passed (3)
      Tests  5 passed (5)
   Start at  16:44:01
   Duration  3.75s (transform 260ms, setup 921ms, collect 723ms, tests 221ms, environment 4.21s, prepare 3.20s)

 PASS  Waiting for file changes...
       press h to show help, press q to quit
```

Test case 03:




PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
node
(c) Microsoft Corporation. All rights reserved.
C:\Users\njmlo\react-testing-starter>npm run test:ui
> react-testing-vite@0.0.0 test:ui
> vitest --ui
DEV v2.1.5 C:/Users/njmlo/react-testing-starter
UI started at http://localhost:51204/__vitest__/
stdout | tests/components/UserAccount.test.tsx > UserAccount > should render user name
PASSED as expected.
stdout | tests/components/UserAccount.test.tsx > UserAccount > should render edit button if user is admin
PASSED as expected.
stdout | tests/components/UserAccount.test.tsx > UserAccount > should not render edit button if user is not admin
PASSED as expected.
stdout | tests/components/Greet.test.tsx > Greet > should render Hello with the name when the name is provided
PASSED as expected.
stdout | tests/components/Greet.test.tsx > Greet > should render login button when a name is not provided
PASSED as expected.
✓ tests/main.test.ts (1)
✓ tests/components/Greet.test.tsx (2)
✓ tests/components/UserAccount.test.tsx (3)
Test Files 3 passed (3)
Tests 6 passed (6)
Start at 16:56:56
Duration 3.70s (transform 179ms, setup 952ms, collect 766ms, tests 255ms, environment 3.90s, prepare 785ms)
PASS Waiting for file changes...
press h to show help, press q to quit


localhost:51204/__vitest__/#/
Vitest
Search...
Filter
Fail
Pass
Skip
Only Tests
FAIL (0) / RUNNING (0)
PASS (3) / SKIP (--)
tests/components/Greet.test.tsx 125ms
Greet 124ms
tests/components/UserAccount.test.tsx 125ms
UserAccount 123ms
should render user name 55ms
should render edit button if user is admin 63ms
should not render edit button if user is not admin 4ms
tests/main.test.ts 6ms
Dashboard
6
Pass
0
Fail
6
Total
Files 3
Pass 3
Time 6.65s

7. Testing Lists

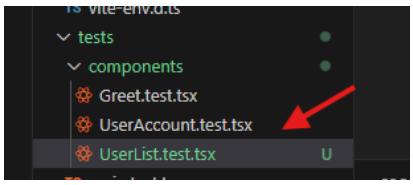
Component to test: UserList.tsx

UserList.tsx

```tsx
import { User } from "../entities";

const UserList = ({ users }: { users: User[] }) => {
  if (users.length === 0) return <p>No users available.</p>;

  return (
    <ul>
      {users.map((user) => (
        <li key={user.id}>
          <a href={`/users/${user.id}`}>{user.name}</a>
        </li>
      ))}
    </ul>
  );
};

export default UserList;
```

Test case 01:

Create a test file…

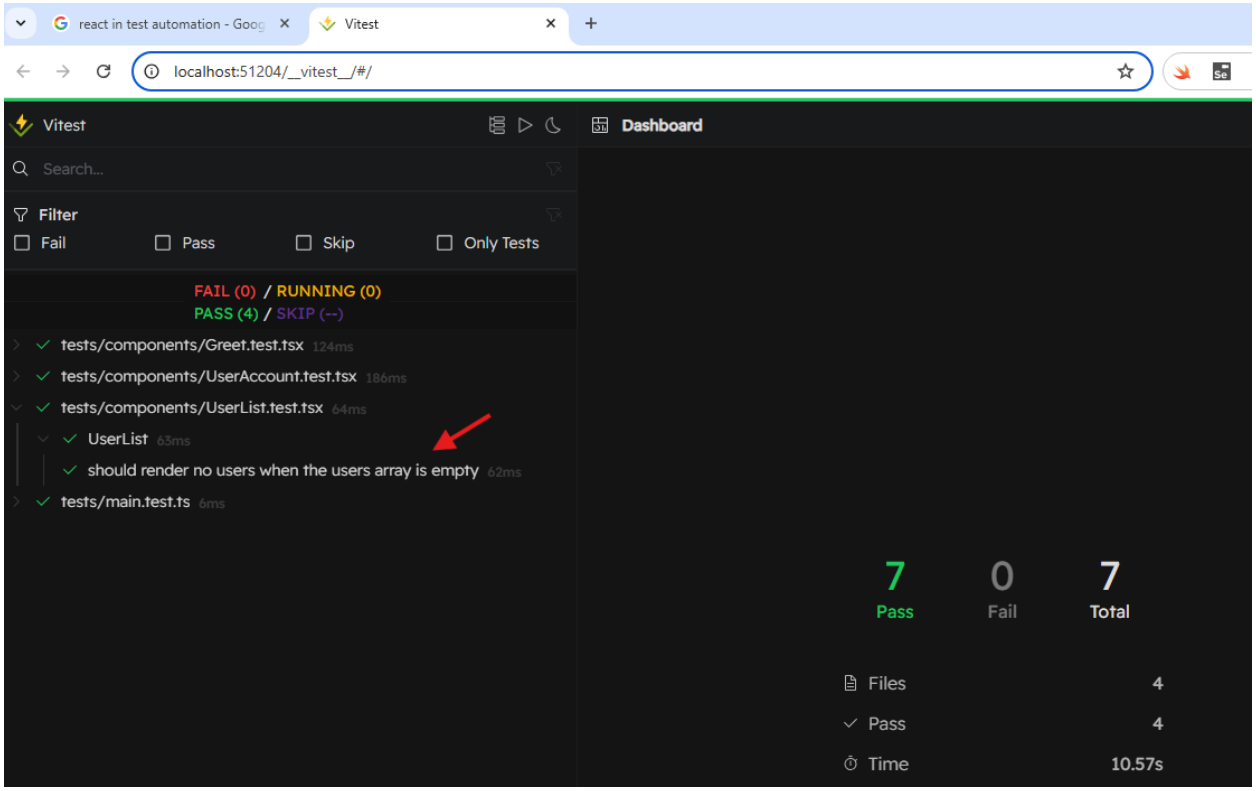


UserList.test.tsx U

tests > components > UserList.test.tsx > ...

```tsx
import { render, screen } from '@testing-library/react';
import UserList from '../../src/components/UserList';

/* Test Case 01 */
/* Expected result is it should render no users when the users array is empty */
describe ('UserList', () => {
    it('should render no users when the users array is empty', () => {
        render(<UserList users={[]}/>);
        expect(screen.getByText(/No users/)).toBeInTheDocument();
        console.log('PASSED as expected.');
    })
})
```

```
> react-testing-vite@0.0.0 test:ui
> vitest --ui


 DEV  v2.1.5 C:/Users/njmlo/react-testing-starter
      UI started at http://localhost:51204/__vitest__/

stdout | tests/components/UserList.test.tsx > UserList > should render no users when the users array is empty
PASSED as expected.

stdout | tests/components/Greet.test.tsx > Greet > should render Hello with the name when the name is provided
PASSED as expected.

stdout | tests/components/Greet.test.tsx > Greet > should render login button when a name is not provided
PASSED as expected.

stdout | tests/components/UserAccount.test.tsx > UserAccount > should render user name
PASSED as expected.

stdout | tests/components/UserAccount.test.tsx > UserAccount > should render edit button if user is admin
PASSED as expected.

stdout | tests/components/UserAccount.test.tsx > UserAccount > should not render edit button if user is not admin
PASSED as expected.

 ✓ tests/main.test.ts (1)
 ✓ tests/components/Greet.test.tsx (2)
 ✓ tests/components/UserAccount.test.tsx (3)
 ✓ tests/components/UserList.test.tsx (1)

 Test Files  4 passed (4)
      Tests  7 passed (7)
   Start at  11:30:58
   Duration  4.04s (transform 202ms, setup 1.51s, collect 1.36s, tests 381ms, environment 6.14s, prepare 1.18s)

 PASS  Waiting for file changes...
       press h to show help, press q to quit
```



Test case 02:

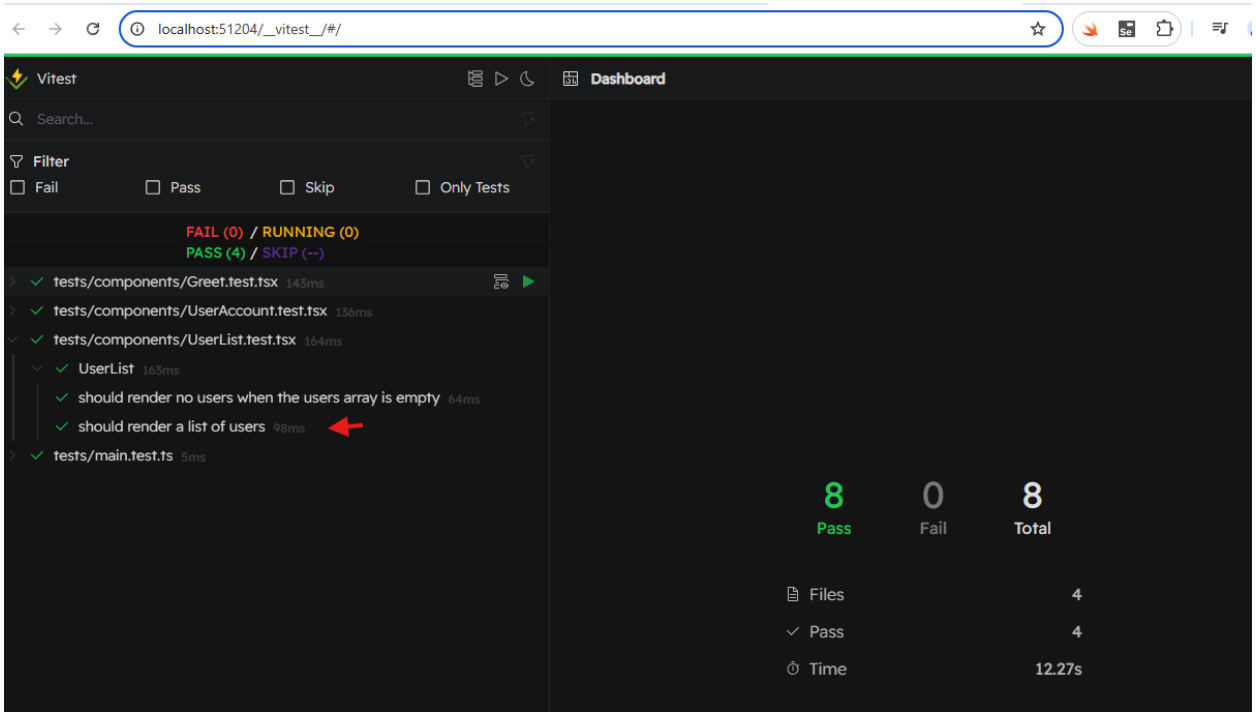
UserList.test.tsx U | UserList.tsx

tests > components > UserList.test.tsx > describe('UserList') callback

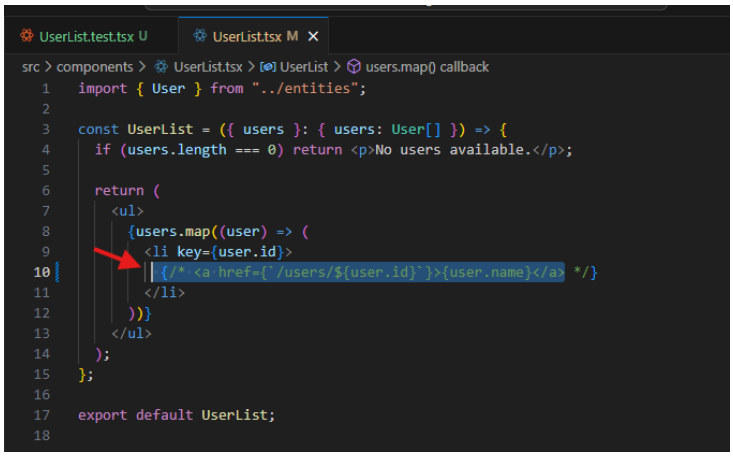
```tsx
describe ('UserList', () => {

    /* Test Case 02 */
/* Expected result is it should render a list of users */
    it('should render a list of users', () => {
        const users: User [] = [
        // two array objects...
            {id: 1, name: 'John'},
            {id: 2, name: 'Sam'},
        ]
        // render code...
        render(<UserList users={users}/>);
        // assertion code...
        users.forEach(user=>{
            const link = screen.getByRole('link', {name: user.name});
            expect(link).toBeInTheDocument();
            expect (link).toHaveAttribute('href', `/users/${user.id}`);
            console.log('PASSED as expected.');
        })
    })
})
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS node

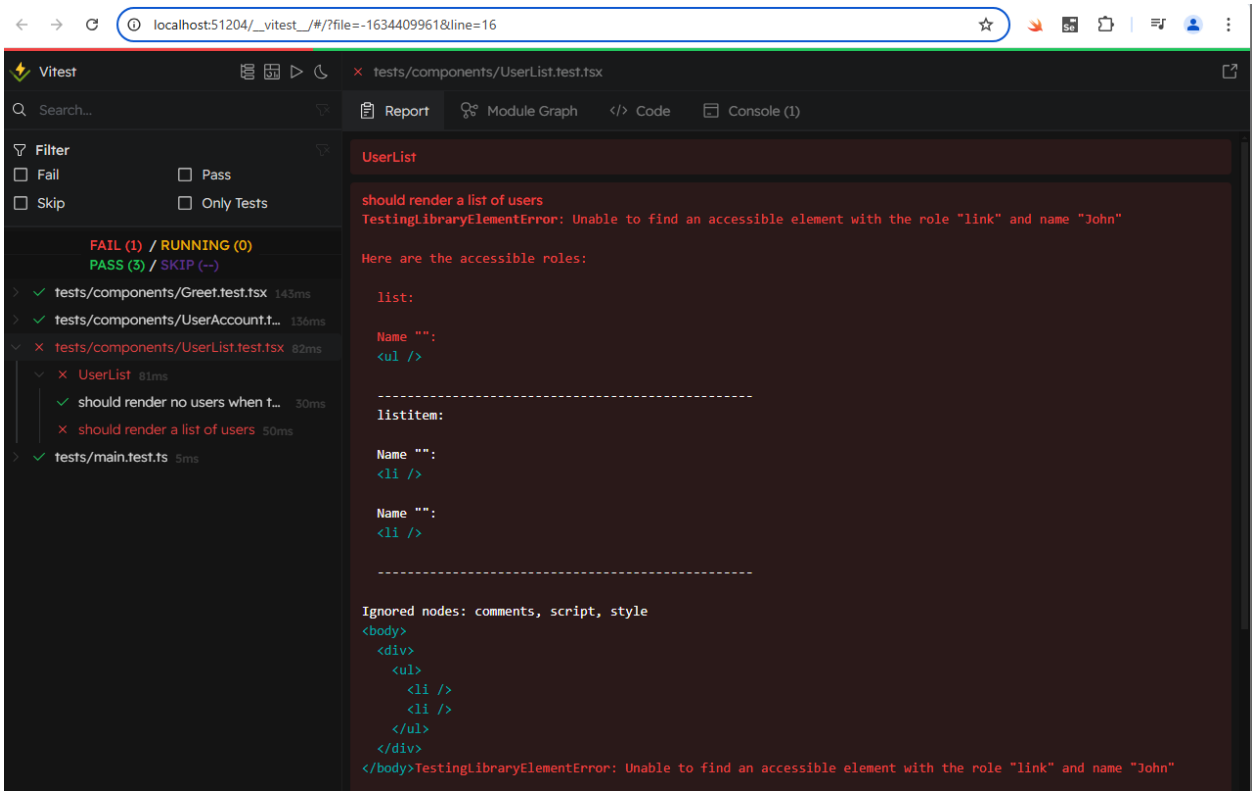
```
C:\Users\njmlo\react-testing-starter>npm run test:ui

> react-testing-vite@0.0.0 test:ui
> vitest --ui


 DEV  v2.1.5 C:/Users/njmlo/react-testing-starter
      UI started at http://localhost:51204/__vitest__/

stdout | tests/components/UserAccount.test.tsx > UserAccount > should render user name
PASSED as expected.

stdout | tests/components/UserAccount.test.tsx > UserAccount > should render edit button if user is admin
PASSED as expected.

stdout | tests/components/UserAccount.test.tsx > UserAccount > should not render edit button if user is not admin
PASSED as expected.

stdout | tests/components/Greet.test.tsx > Greet > should render Hello with the name when the name is provided
PASSED as expected.

stdout | tests/components/Greet.test.tsx > Greet > should render login button when a name is not provided
PASSED as expected.

stdout | tests/components/UserList.test.tsx > UserList > should render no users when the users array is empty
PASSED as expected.

stdout | tests/components/UserList.test.tsx > UserList > should render a list of users
PASSED as expected.
PASSED as expected.

 ✓ tests/main.test.ts (1)
 ✓ tests/components/Greet.test.tsx (2)
 ✓ tests/components/UserAccount.test.tsx (3)
 ✓ tests/components/UserList.test.tsx (2)

 Test Files  4 passed (4)
      Tests  8 passed (8)
   Start at  12:06:36
   Duration  4.56s (transform 264ms, setup 1.63s, collect 1.33s, tests 448ms, environment 7.36s, prepare 1.49s)

 PASS  Waiting for file changes...
       press h to show help, press q to quit
```



Running a negative test (fail the test case):

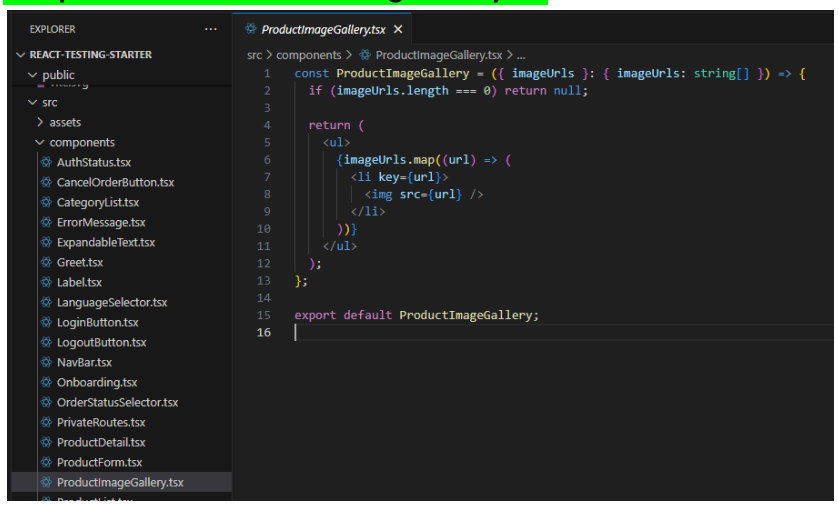


```
Failed Tests 1

FAIL  tests/components/UserList.test.tsx > UserList > should render a list of users
TestingLibraryElementError: Unable to find an accessible element with the role "link" and name "John"

Here are the accessible roles:

  list:

  Name "":
  <ul />

  --------------------------------------------------
  listitem:

  Name "":
  <li />

  Name "":
  <li />

  --------------------------------------------------

Ignored nodes: comments, script, style
<body>
  <div>
    <ul>
      <li />
      <li />
    </ul>
  </div>
</body>
 ❯ Object.getElementError node_modules/@testing-library/dom/dist/config.js:37:19
 ❯ node_modules/@testing-library/dom/dist/query-helpers.js:76:38
 ❯ node_modules/@testing-library/dom/dist/query-helpers.js:52:17
 ❯ node_modules/@testing-library/dom/dist/query-helpers.js:95:19
 ❯ tests/components/UserList.test.tsx:26:33
     24|         // assertion code...
     25|         users.forEach(user=>{
     26|             const link = screen.getByRole('link', {name: user.name});
       |                                 ^
     27|             expect(link).toBeInTheDocument();
     28|             expect (link).toHaveAttribute('href', `/users/${user.id}`);
 ❯ tests/components/UserList.test.tsx:25:15


 Test Files  1 failed (1)
      Tests  1 failed | 1 passed (2)
   Start at  12:08:52
   Duration  520ms

FAIL  Tests failed. Watching for file changes...
      press h to show help, press q to quit
```



8. Testing ProductImageGallery

Component to test: ProductImageGallery.tsx



Test case 01:

Create a test file…

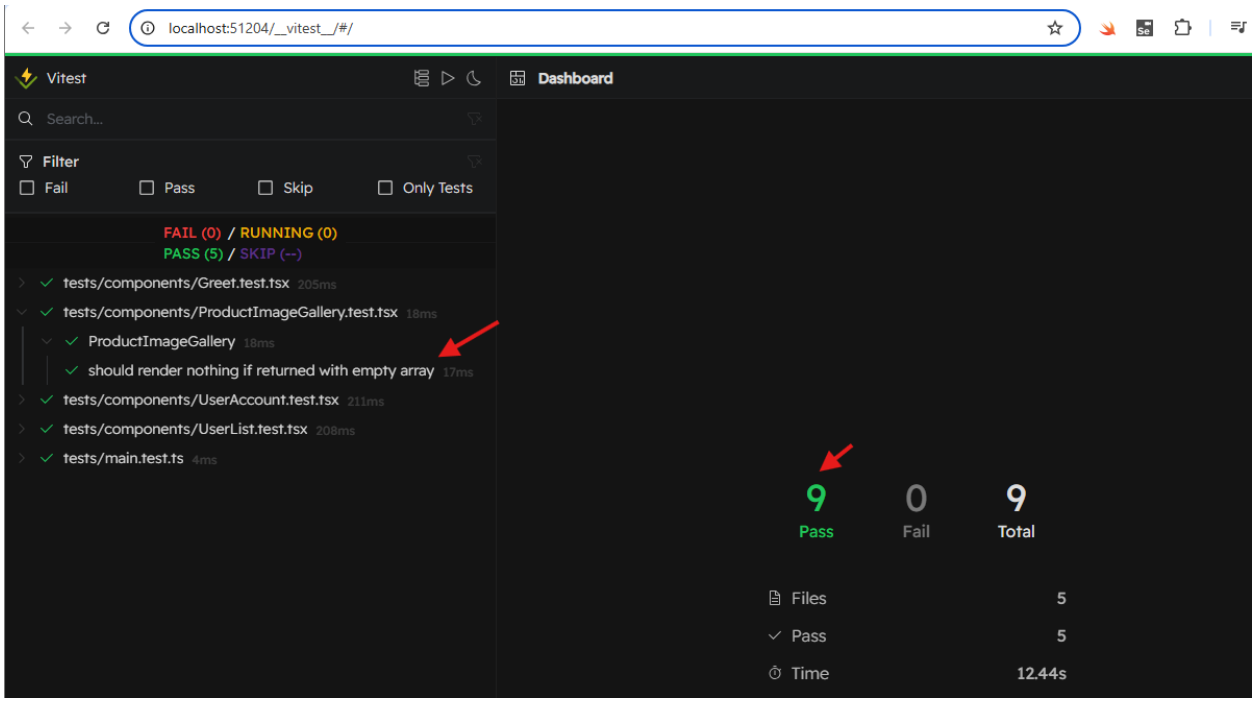
ProductImageGallery.test.tsx 1, U | ProductImageGallery.tsx

tests > components > ProductImageGallery.test.tsx > describe('ProductImageGallery') callback

```tsx
import { render, screen } from '@testing-library/react'
import ProductImageGallery from '../../src/components/ProductImageGallery';


/* Test Case 01 */
/* Expected result is it should render nothing if returned with empty array */
describe ('ProductImageGallery', () => {
    it('should render nothing if returned with empty array', () => {
        const{container} = render(<ProductImageGallery  imageUrls={[]}/>);
        expect(container).toBeEmptyDOMElement();
        console.log('PASSED as expected.');
    })

```

PROBLEMS 1 | OUTPUT | DEBUG CONSOLE | TERMINAL | PORTS

```
PASSED as expected.

 ✓ tests/components/ProductImageGallery.test.tsx (1)
   ✓ ProductImageGallery (1)
     ✓ should render nothing if returned with empty array

 Test Files  1 passed (1)
      Tests  1 passed (1)
   Start at  12:59:35
   Duration  401ms

 PASS  Waiting for file changes...
       press h to show help, press q to quit
```



Test case 02:

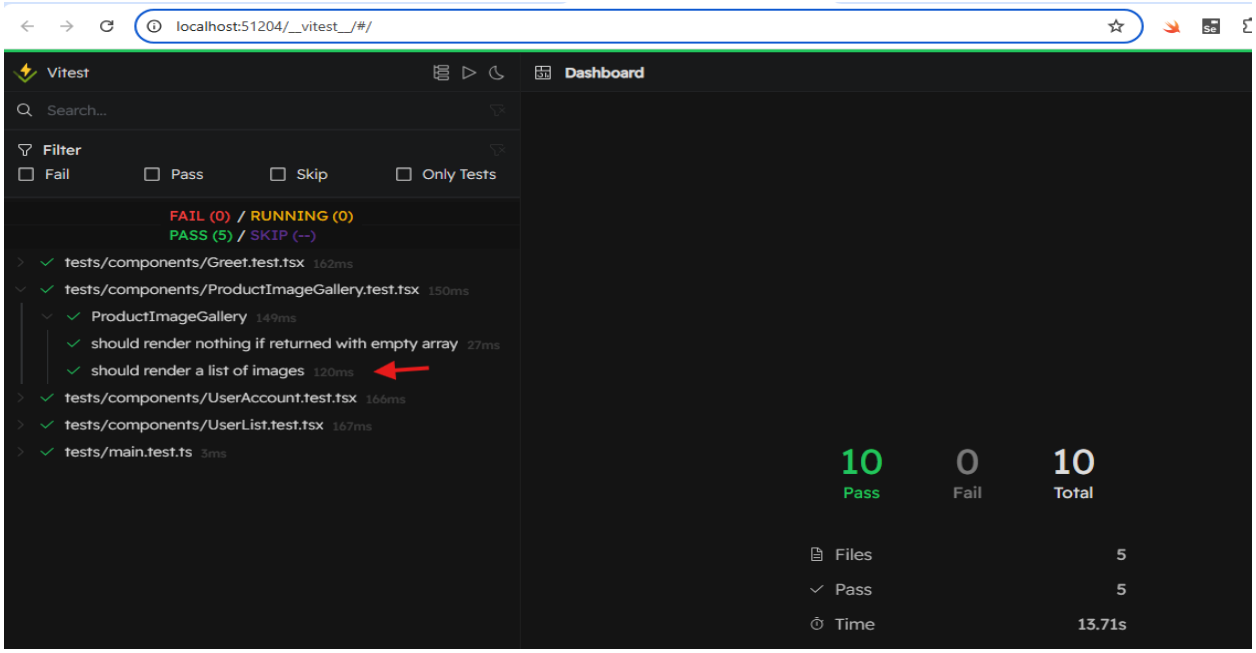
ProductImageGallery.test.tsx U | ProductImageGallery.tsx

tests > components > ProductImageGallery.test.tsx > describe('ProductImageGallery') callback > it('should render a list of images') callback

```
6   describe ('ProductImageGallery', () => {
13      /* Test Case 02 */
14  /* Expected result is it should render a list of images */
15      it('should render a list of images', () => {
16          // two array objects...
17          const imageUrls = ['url1','url2'];
18          // render code...
19          render(<ProductImageGallery imageUrls={imageUrls}/>);
20          // assertion code...
21          const images = screen.getAllByRole('img');
22          expect(images).toHaveLength(2);
23          imageUrls.forEach((url, index) => {
24              expect (images[index]).toHaveAttribute('src', url);
25          });
26          console.log('PASSED as expected.');
27      })
28  })
```

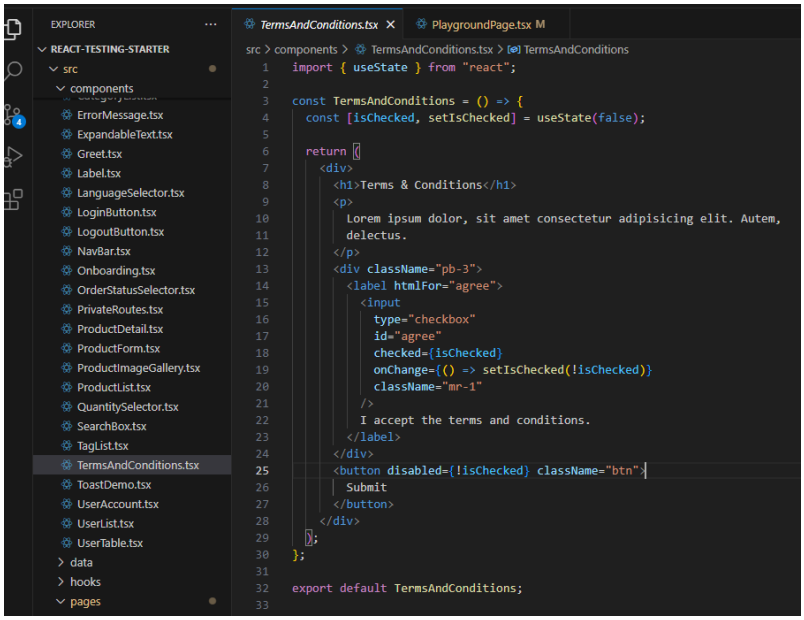
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS node

```
stdout | tests/components/ProductImageGallery.test.tsx > ProductImageGallery > should render a list of images
PASSED as expected.

 ✓ tests/components/ProductImageGallery.test.tsx (2)
   ✓ ProductImageGallery (2)
     ✓ should render nothing if returned with empty array
     ✓ should render a list of images

 Test Files  1 passed (1)
      Tests  2 passed (2)
   Start at  13:36:03
   Duration  465ms

 PASS  Waiting for file changes...
       press h to show help, press q to quit
```

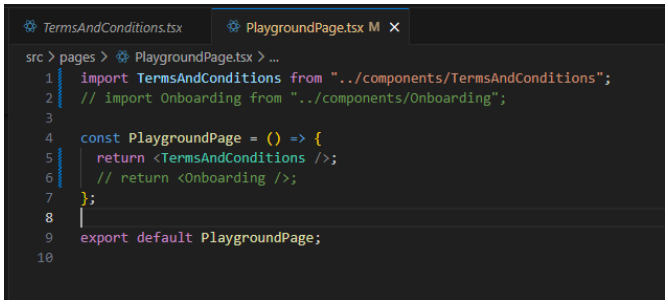


9. Testing User Interactions (TermsAndConditions)

Component to test: TermsAndConditions.tsx

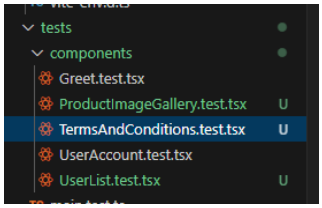


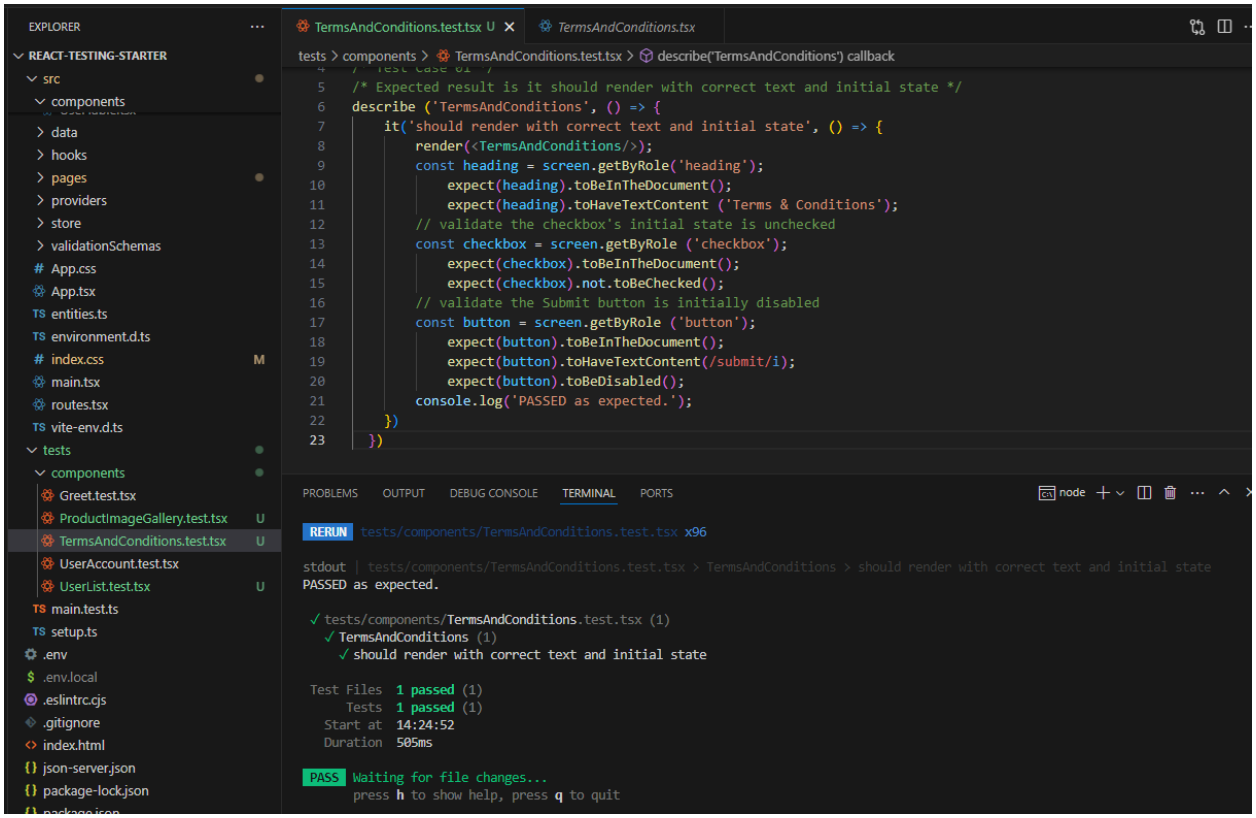
To display component in UI…

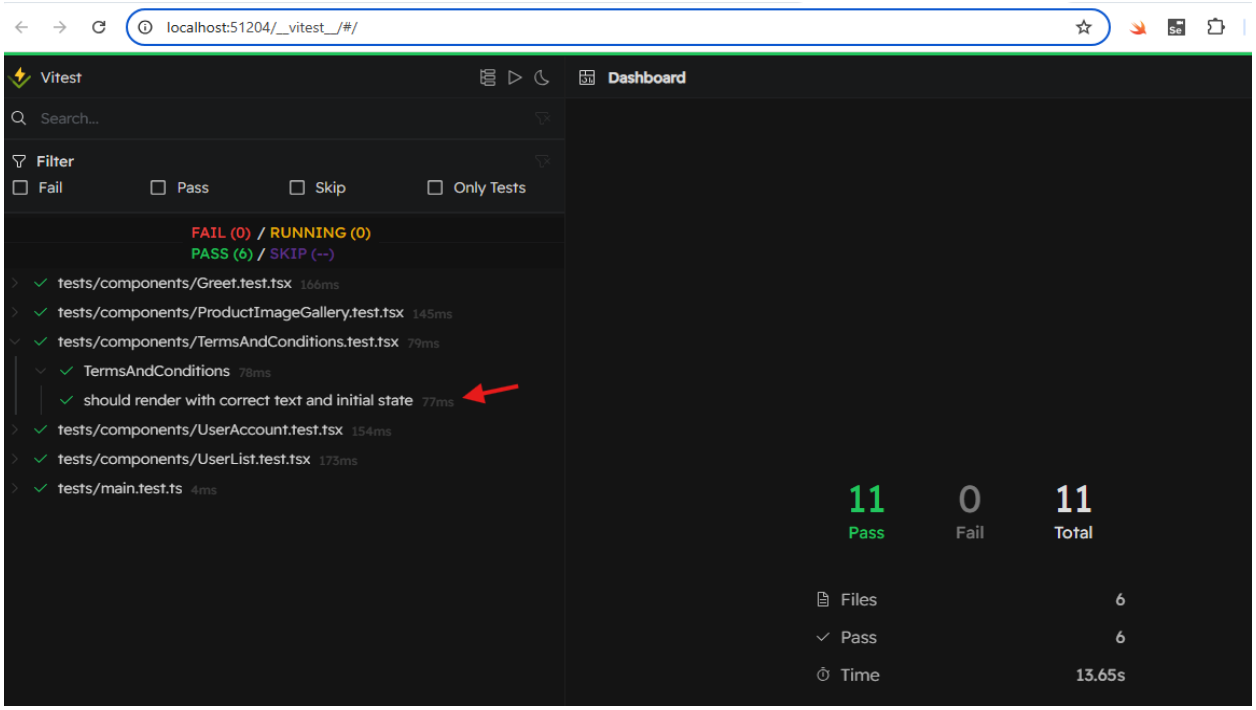


Test case 01:

Create a test file…

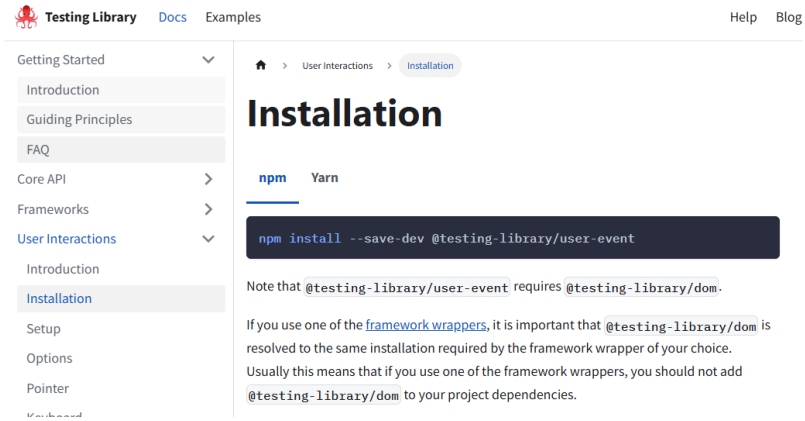



EXPLORER
REACT-TESTING-STARTER
src
components
data
hooks
pages
providers
store
validationSchemas
App.css
App.tsx
entities.ts
environment.d.ts
index.css
main.tsx
routes.tsx
vite-env.d.ts
tests
components
Greet.test.tsx
ProductImageGallery.test.tsx
TermsAndConditions.test.tsx
UserAccount.test.tsx
UserList.test.tsx
main.test.ts
setup.ts
.env
.env.local
.eslintrc.cjs
.gitignore
index.html
json-server.json
package-lock.json
TermsAndConditions.test.tsx U
TermsAndConditions.tsx
tests > components > TermsAndConditions.test.tsx > describe('TermsAndConditions') callback
/* Expected result is it should render with correct text and initial state */
describe ('TermsAndConditions', () => {
it('should render with correct text and initial state', () => {
render(<TermsAndConditions/>);
const heading = screen.getByRole('heading');
expect(heading).toBeInTheDocument();
expect(heading).toHaveTextContent ('Terms & Conditions');
// validate the checkbox's initial state is unchecked
const checkbox = screen.getByRole ('checkbox');
expect(checkbox).toBeInTheDocument();
expect(checkbox).not.toBeChecked();
// validate the Submit button is initially disabled
const button = screen.getByRole ('button');
expect(button).toBeInTheDocument();
expect(button).toHaveTextContent(/submit/i);
expect(button).toBeDisabled();
console.log('PASSED as expected.');
})
})
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
node
RERUN tests/components/TermsAndConditions.test.tsx x96
stdout | tests/components/TermsAndConditions.test.tsx > TermsAndConditions > should render with correct text and initial state
PASSED as expected.
✓ tests/components/TermsAndConditions.test.tsx (1)
✓ TermsAndConditions (1)
✓ should render with correct text and initial state
Test Files 1 passed (1)
Tests 1 passed (1)
Start at 14:24:52
Duration 505ms
PASS Waiting for file changes...
press h to show help, press q to quit


localhost:51204/__vitest__/#/
Vitest
Dashboard
Search...
Filter
Fail
Pass
Skip
Only Tests
FAIL (0) / RUNNING (0)
PASS (6) / SKIP (--)
tests/components/Greet.test.tsx 166ms
tests/components/ProductImageGallery.test.tsx 145ms
tests/components/TermsAndConditions.test.tsx 79ms
TermsAndConditions 78ms
should render with correct text and initial state 77ms
tests/components/UserAccount.test.tsx 154ms
tests/components/UserList.test.tsx 173ms
tests/main.test.ts 4ms
11
Pass
0
Fail
11
Total
Files 6
Pass 6
Time 13.65s

Test case 02:

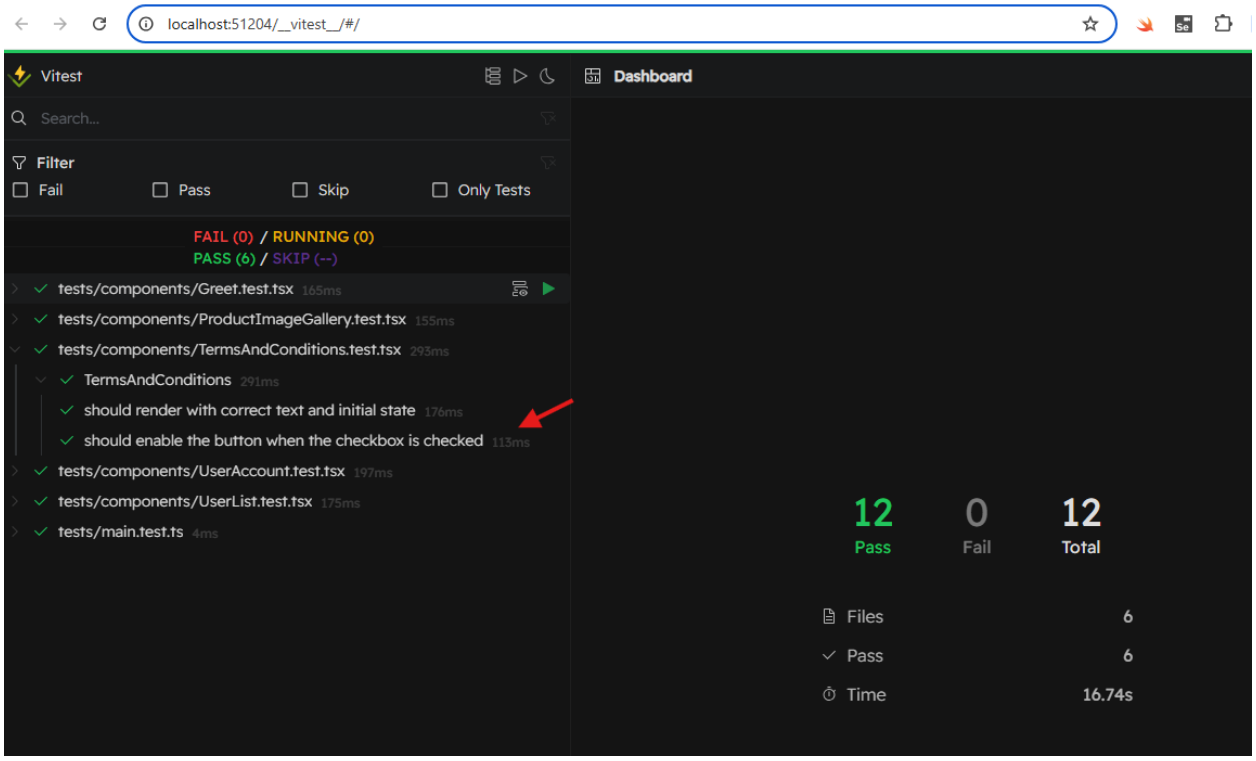
First is to install the user-event library…




```
C:\Users\njmlo\react-testing-starter>npm install --save-dev @testing-library/user-event

added 1 package, and audited 638 packages in 4s

found 0 vulnerabilities

C:\Users\njmlo\react-testing-starter>
```

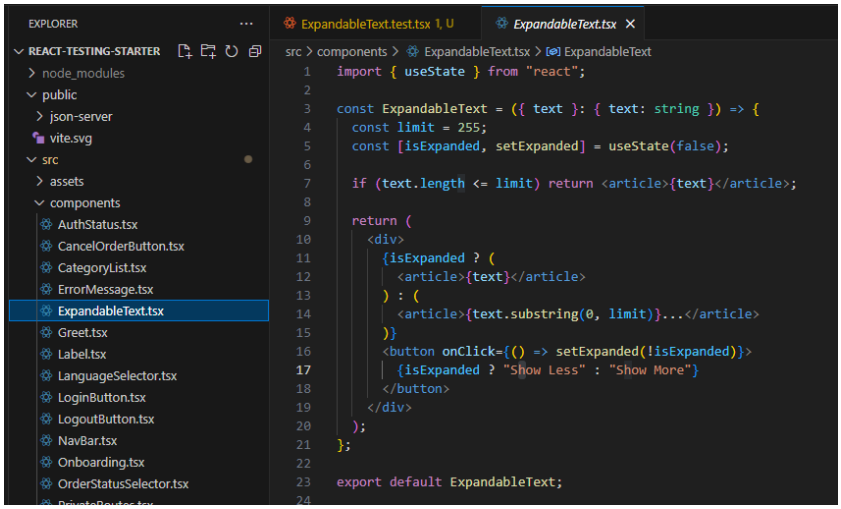
```
25  /* Test Case 02 */
26  /* Expected result is it should enable the button when the checkbox is checked */
27  // Note: install user-event library from @testing-library (this is to simullate a user behavior from the UI)
28      it('should enable the button when the checkbox is checked', async () => {
29          //Arrange
30          render(<TermsAndConditions/>);
31          //Act
32          const checkbox = screen.getByRole ('checkbox');
33          const user = userEvent.setup();
34              await user.click(checkbox);
35          //Assert
36              expect(screen.getByRole('button')).toBeEnabled();
37          console.log('PASSED as expected.');
38      })
39  })
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS                                                    node

stdout | tests/components/TermsAndConditions.test.tsx > TermsAndConditions > should enable the button when the checkbox is checked
PASSED as expected.

stdout | tests/components/Greet.test.tsx > Greet > should render Hello with the name when the name is provided
PASSED as expected.

stdout | tests/components/Greet.test.tsx > Greet > should render login button when a name is not provided
PASSED as expected.

 ✓ tests/main.test.ts (1)
 ✓ tests/components/Greet.test.tsx (2)
 ✓ tests/components/ProductImageGallery.test.tsx (2)
 ✓ tests/components/TermsAndConditions.test.tsx (2)
 ✓ tests/components/UserAccount.test.tsx (3)
 ✓ tests/components/UserList.test.tsx (2)

 Test Files  6 passed (6)
      Tests  12 passed (12)
   Start at  14:40:21
   Duration  6.80s (transform 371ms, setup 2.19s, collect 2.52s, tests 990ms, environment 9.34s, prepare 1.70s)

 PASS  Waiting for file changes...
       press h to show help, press q to quit
```



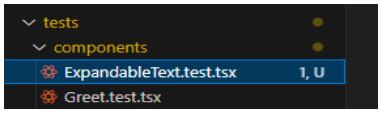
10. Testing Expandable Test

Component to test: ExpandableText.tsx



Test case 01:

Create a test file…



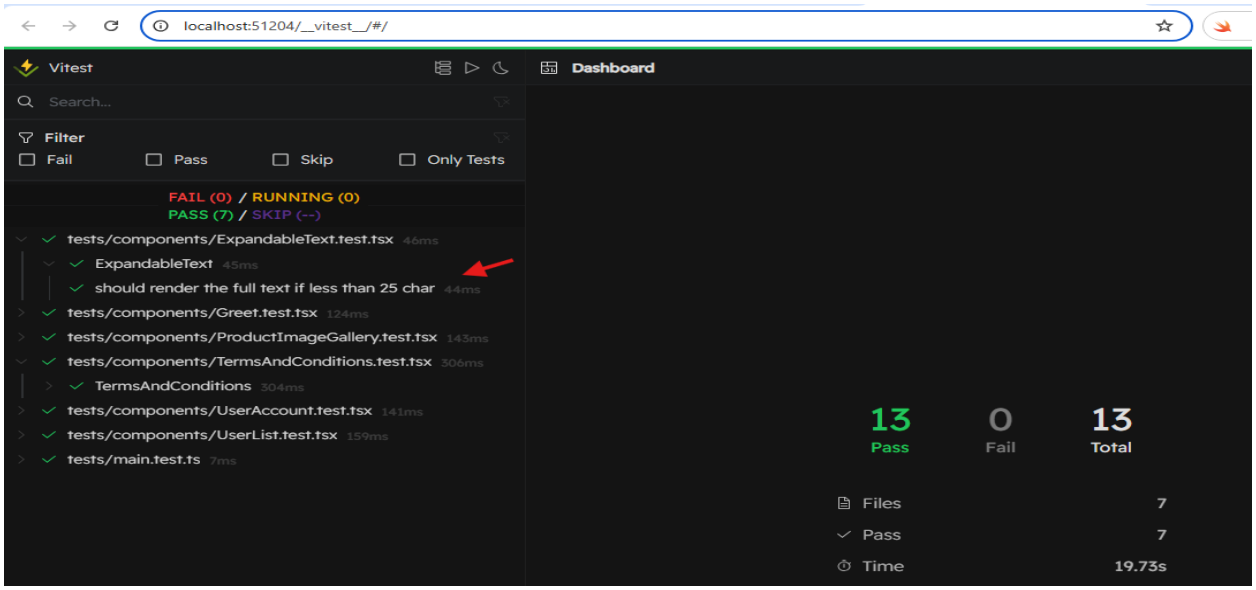
ExpandableText.test.tsx M | ExpandableText.tsx M | PlaygroundPage.tsx M

tests > components > ExpandableText.test.tsx > describe('ExpandableText') callback

```tsx
/* Import functions and libraries */
import { render, screen } from '@testing-library/react';
import ExpandableText from '../../src/components/ExpandableText';

/* Test Case 01 */
/* Expected result is it should render the full text if less than 25 char */
describe ('ExpandableText', () => {
    it('should render the full text if less than 25 char', () => {
        const text = "Short text";
            render(<ExpandableText text={text}/>);
            expect(screen.getByText(text)).toBeInTheDocument();
        console.log('PASSED as expected.');
    })

})
```

PROBLEMS | OUTPUT | DEBUG CONSOLE | TERMINAL | PORTS

```
PASSED as expected.

stdout | tests/components/ExpandableText.test.tsx > ExpandableText > should render the full text if less than 25 char
PASSED as expected.

stdout | tests/components/Greet.test.tsx > Greet > should render Hello with the name when the name is provided
PASSED as expected.

stdout | tests/components/Greet.test.tsx > Greet > should render login button when a name is not provided
PASSED as expected.

 ✓ tests/main.test.ts (1)
 ✓ tests/components/ExpandableText.test.tsx (1)
 ✓ tests/components/Greet.test.tsx (2)
 ✓ tests/components/ProductImageGallery.test.tsx (2)
 ✓ tests/components/TermsAndConditions.test.tsx (2) 306ms
 ✓ tests/components/UserAccount.test.tsx (3)
 ✓ tests/components/UserList.test.tsx (2)

 Test Files  7 passed (7)
      Tests  13 passed (13)
   Start at  15:32:41
   Duration  7.07s (transform 378ms, setup 2.85s, collect 2.86s, tests 928ms, environment 10.88s, prepare 2.20s)

 PASS  Waiting for file changes...
       press h to show help, press q to quit
```

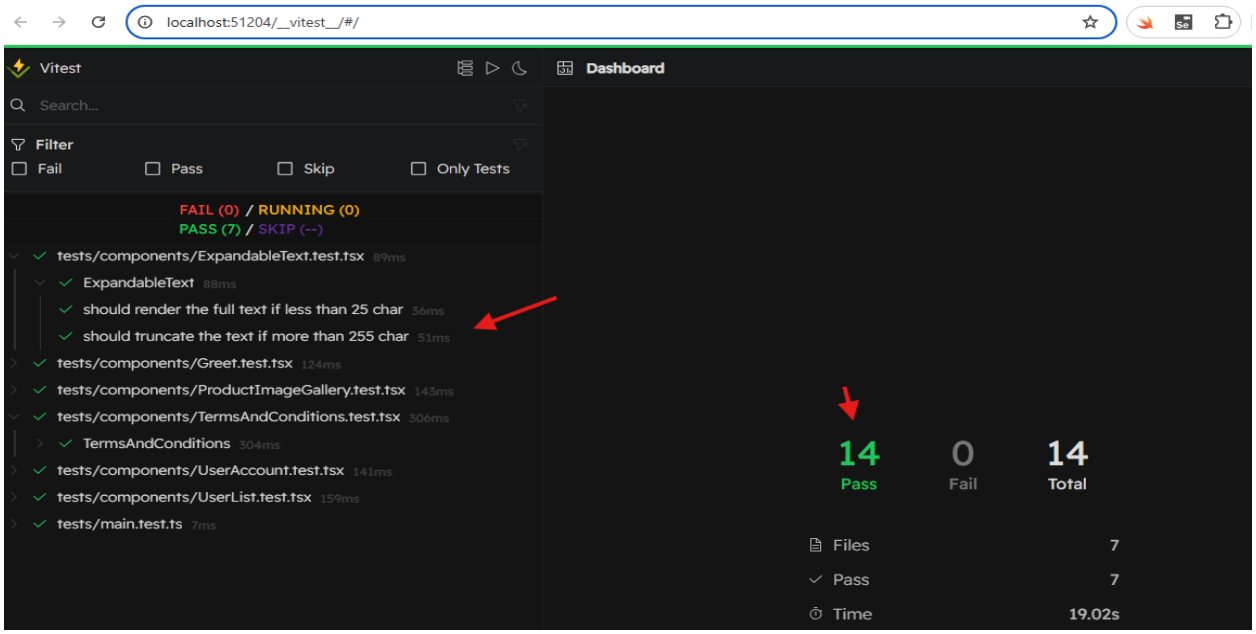


Test case 02:

ExpandableText.test.tsx M

tests > components > ExpandableText.test.tsx > ...

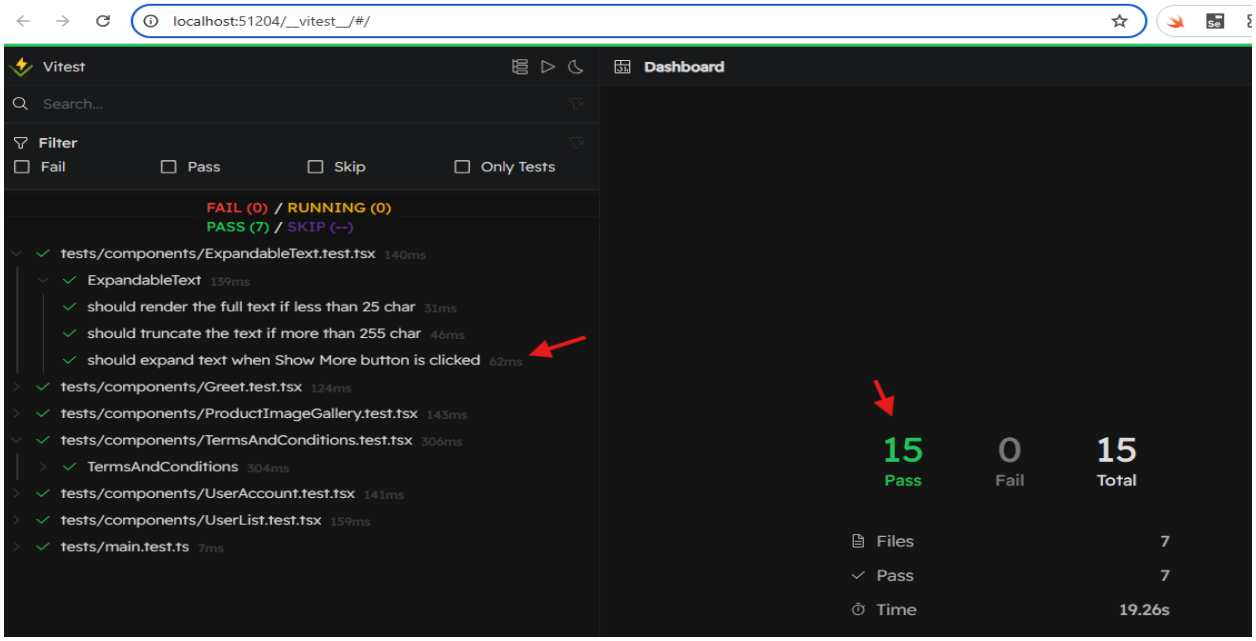
```tsx
/* Import functions and libraries */
import { render, screen } from '@testing-library/react';
import ExpandableText from '../../src/components/ExpandableText';
// import userEvent from '@testing-library/user-event';

describe ('ExpandableText', () => {
    const limit = 255;
    const longText = 'a'.repeat (limit + 1);
    const truncatedText = longText.substring (0, limit) + "...";

    /* Test Case 01 */
    /* Expected result is it should render the full text if less than 25 char */
    it('should render the full text if less than 25 char', () => {
        const text = "Short text";
            render(<ExpandableText text={text}/>);
            expect(screen.getByText(text)).toBeInTheDocument();
        console.log('PASSED as expected.');
    })

    /* Test Case 02 */
    /* Expected result is it should truncate the text if more than 255 char */
    it('should truncate the text if more than 255 char', () => {
            // this is to check the text if more than 255 char
            render(<ExpandableText text={longText} />);
            expect(screen.getByText(truncatedText)).toBeInTheDocument();
            // this is to check the expected buttons
            const button = screen.getByRole('button');
            expect(button).toHaveTextContent(/more/i);
        console.log('PASSED as expected.');
    })
```



Test case 03:

```
32      /* Test Case 03 */
33      /* Expected result is it should expand text when Show More button is clicked */
34      it('should expand text when Show More button is clicked', async () => {
35          // this is to check the text if more than 255 char...
36          render(<ExpandableText text={longText} />);
37          // this is to check when button is clicked...
38          const button = screen.getByRole('button');
39          const user = userEvent.setup();
40          await user.click(button);
41          // this is to check the expected buttons...
42          expect(screen.getByText(longText)).toBeInTheDocument();
43                  expect(button).toHaveTextContent(/less/i);
44      console.log('PASSED as expected.');
45      })
46  })
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS                                   node

stdout | tests/components/ExpandableText.test.tsx > ExpandableText > should expand text when Show More button is clicked
PASSED as expected.

 ✓ tests/components/ExpandableText.test.tsx (3)
   ✓ ExpandableText (3)
     ✓ should render the full text if less than 25 char
     ✓ should truncate the text if more than 255 char
     ✓ should expand text when Show More button is clicked

 Test Files  1 passed (1)
      Tests  3 passed (3)
   Start at  16:44:32
   Duration  667ms

 PASS  Waiting for file changes...
       press h to show help, press q to quit
```



Test case 04:

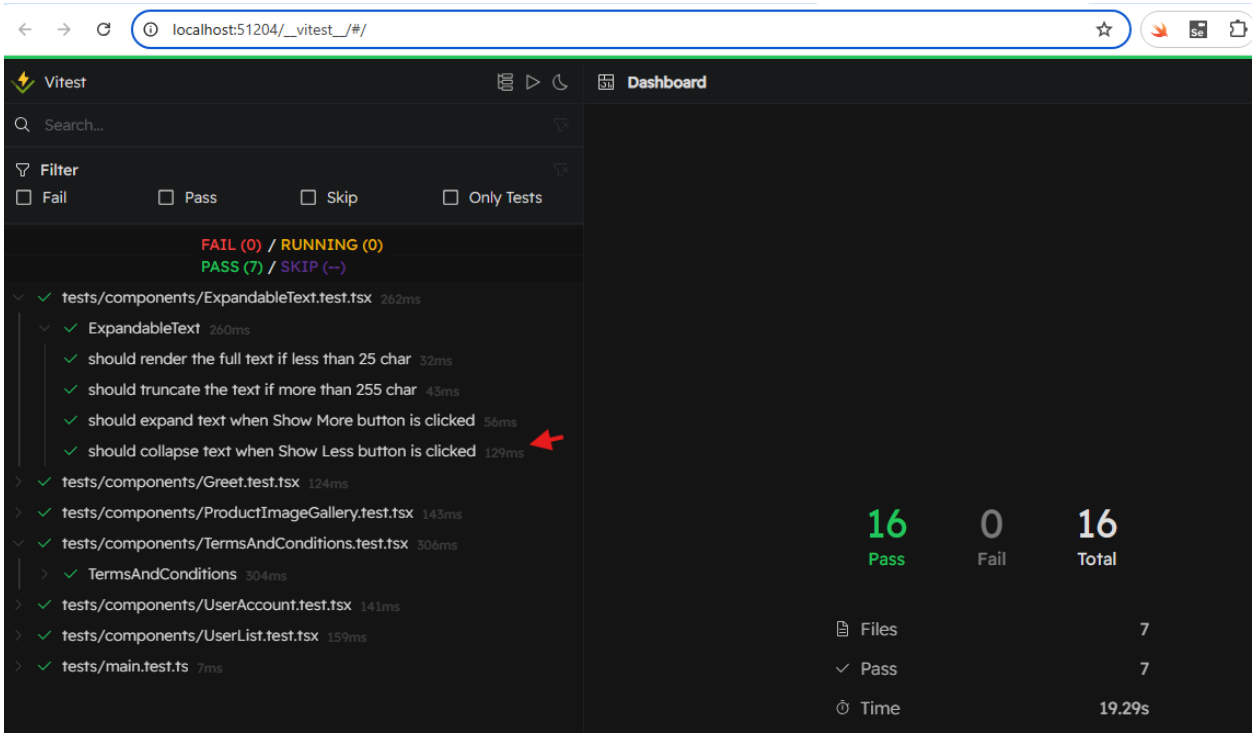
ExpandableText.test.tsx M

tests > components > ExpandableText.test.tsx > describe('ExpandableText') callback > it('should collapse text when Show Less button is clicked') callback

```tsx
describe ('ExpandableText', () => {
    /* Test Case 04 */
    /* Expected result is it should collapse text when Show Less button is clicked */
    it('should collapse text when Show Less button is clicked', async () => {
        // this is to check the text if more than 255 char...
        render(<ExpandableText text={longText} />);

        // this is to check when button is clicked the first time (button is from Show More to Show Less)...
        const showMoreButton = screen.getByRole('button', {name: /more/i});
        const user = userEvent.setup();
        await user.click(showMoreButton);
        // this is to check when button is clicked the after full text is displayed and button is from Show Less to Show More again)...
        const showLessButton = screen.getByRole('button', {name: /less/i});
        await user.click(showLessButton);

        // this is to check the expected buttons...
        expect(screen.getByText(truncatedText)).toBeInTheDocument();
        expect(showMoreButton).toHaveTextContent(/more/i);
    console.log('PASSED as expected.');
    })
})
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS node

```
PASSED as expected.

stdout | tests/components/ExpandableText.test.tsx > ExpandableText > should collapse text when Show Less button is clicked
PASSED as expected.

 ✓ tests/components/ExpandableText.test.tsx (4)
   ✓ ExpandableText (4)
     ✓ should render the full text if less than 25 char
     ✓ should truncate the text if more than 255 char
     ✓ should expand text when Show More button is clicked
     ✓ should collapse text when Show Less button is clicked

 Test Files  1 passed (1)
      Tests  4 passed (4)
   Start at  16:54:56
   Duration  761ms

 PASS  Waiting for file changes...
       press h to show help, press q to quit
```



11. Simplifying the Tests

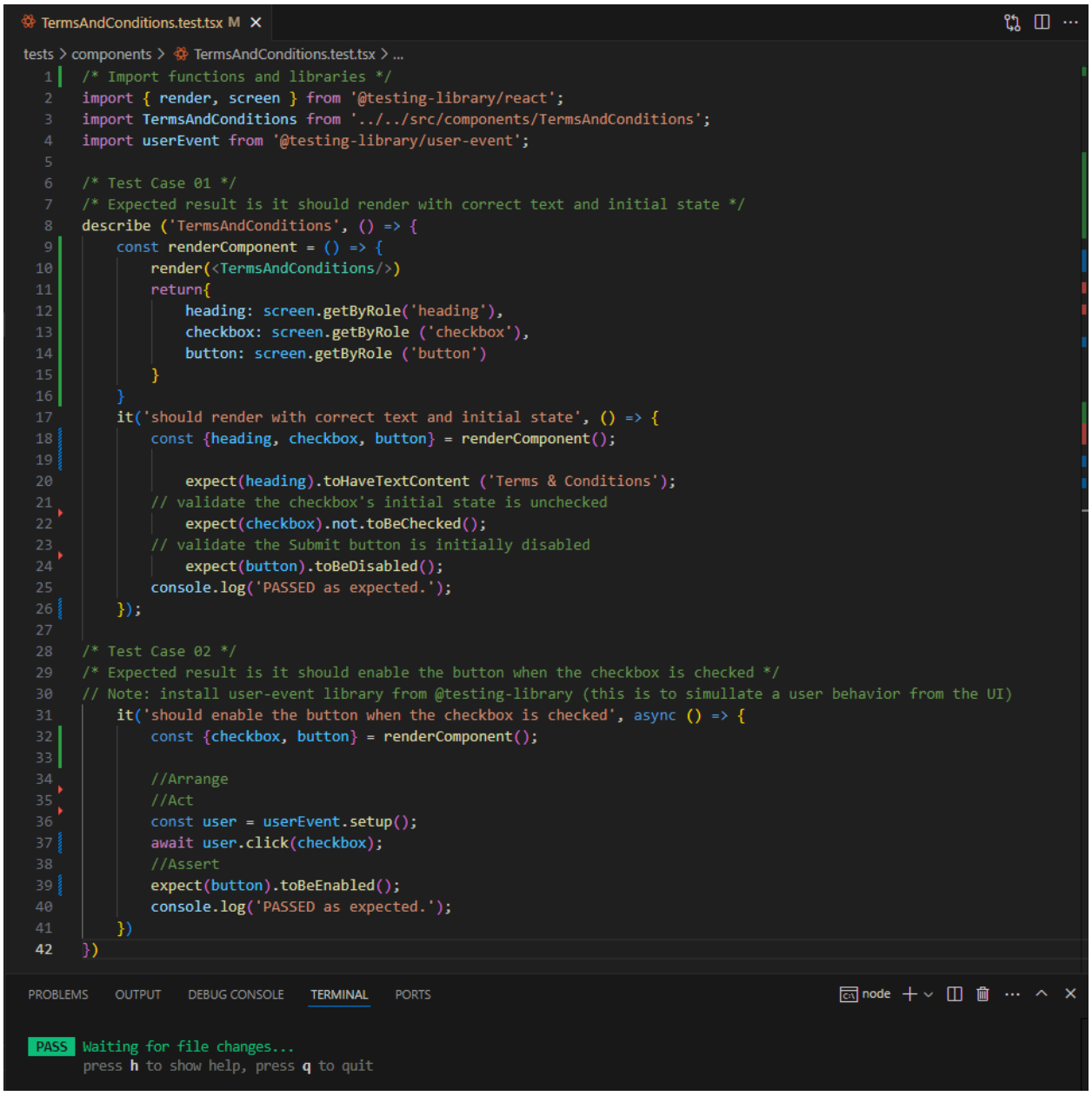
Before cleanup…

TermsAndConditions.test.tsx M

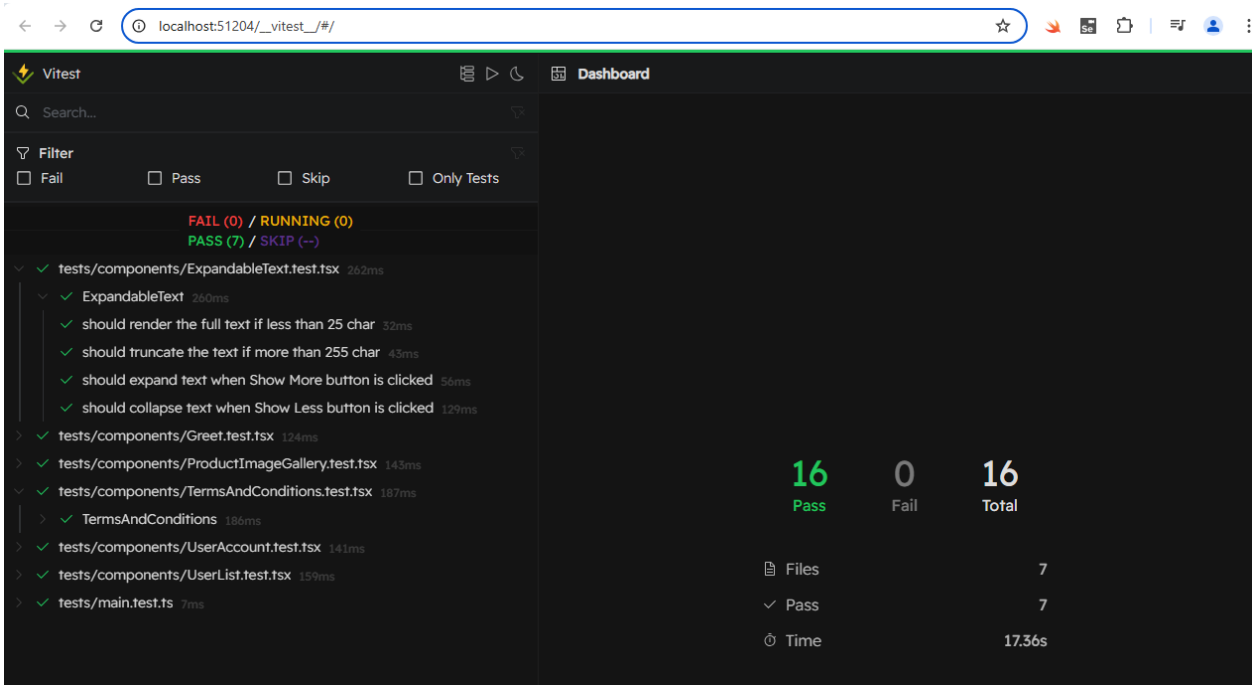
tests > components > TermsAndConditions.test.tsx > describe('TermsAndConditions') callback

```
/* Import functions and libraries */
import { render, screen } from '@testing-library/react';
import TermsAndConditions from '../../src/components/TermsAndConditions';
import userEvent from '@testing-library/user-event';

/* Test Case 01 */
/* Expected result is it should render with correct text and initial state */
describe ('TermsAndConditions', () => {
    it('should render with correct text and initial state', () => {
        render(<TermsAndConditions/>);
        const heading = screen.getByRole('heading');
            expect(heading).toBeInTheDocument();
            expect(heading).toHaveTextContent ('Terms & Conditions');
        // validate the checkbox's initial state is unchecked
        const checkbox = screen.getByRole ('checkbox');
            expect(checkbox).toBeInTheDocument();
            expect(checkbox).not.toBeChecked();
        // validate the Submit button is initially disabled
        const button = screen.getByRole ('button');
            expect(button).toBeInTheDocument();
            expect(button).toHaveTextContent(/submit/i);
            expect(button).toBeDisabled();
        console.log('PASSED as expected.');
    })

/* Test Case 02 */
/* Expected result is it should enable the button when the checkbox is checked */
// Note: install user-event library from @testing-library (this is to simullate a user behavior from the UI)
    it('should enable the button when the checkbox is checked', async () => {
        //Arrange
        render(<TermsAndConditions/>);
        //Act
        const checkbox = screen.getByRole ('checkbox');
        const user = userEvent.setup();
            await user.click(checkbox);
        //Assert
            expect(screen.getByRole('button')).toBeEnabled();
        console.log('PASSED as expected.');
    })
})
```

After cleanup…



12. Final end result - all test cases PASSED



**********END**********