Testing means like writing some code to test your code
There are two types of testing normally

1. Software enginner Testing
2. QA engineer Testing

Software Testing :

- `Unit Testing` : Unit Testing refers to the mechanism in which every small unit of code (means every class, functions etc..) separatly testing.
- `Integration Testing` : you have to combine multiple parts of your application to work together and then test them.
- `E2E(End to end testing)` : You have to test entire application including user interaction also.
- `Load Testing` : Whether your applications are able to take large amount of load or not. example you have server that server can take 1000 requests are not like that.
- `Snapshot Testing` :Snapshot testing is a type of output comparison testing. This snapshot acts as a reference or "expected result." In future tests, your code's current output is compared with this saved snapshot, the snapshot test will fail because the result doesn't match the original snapshot.

So there are some dedicated libraries that can help you to do all testings easily

(REACT-JEST-TESTING PROJECT IS THERE CHECK)

Jest

```
npm i jest
```

React-testing library :

it will give couple of more functionalities

```
npm i @testing-library/react @testing-library/dom

```

some of the functionalities provided by react testing library
`render` : what ever you render it will render in actual screen kind of like testing replacement of actual render (main.jsx);

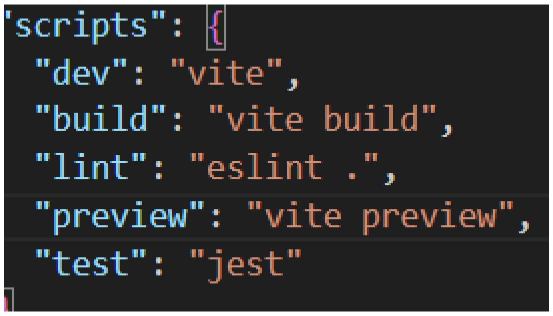
`screen` : find something on screen or actually shown in the screen
`fireEvent` : manually firing an event or programatically.

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

** Along with just we can use react testing library or jest dom our wish

How to setup jest with vite :

[https://zaferayan.medium.com/how-to-setup-jest-and-react-testing-library-in-vite-project-2600f2d04bdd]

- add script in package.json file
  

  ```
  "scripts": {
    "dev": "vite",
    "build": "vite build",
    "lint": "eslint .",
    "preview": "vite preview",
    "test": "jest"
  ```
  -

```
npm i @babel/preset-env @babel/preset-react
```

install above one then create .babelrc file everything is on above link
contiune all steps

```
//App.test.js
// this code is for testing button will click or not using jest
 import { render, screen, fireEvent } from '@testing-library/react';

 import Button from './Components/Button';



 test('render the button and handle click event',()=>{

    // write logic of testing
     const handleClick = jest.fn();

   render(<Button onClick={handleClick}>Click</Button>);

    const renderedButton = screen.getByText('Click');
```

```
    fireEvent.click(renderedButton); //this will actually simulate the
clicking of the button



     expect(handleClick).toHaveBeenCalledTimes(1);
 });


// use case of test function this test()actually contain one

// logical unit for your code

//jest.fn() is the mock function means dummy implementation

//getByText method means this allows you to find element by its text content
```

Playwright :

https://github.com/singhsanket143/react-jest-testing

```
npm i @playwright/test

we have to install this one it supported browsers for playwright.
npx playwright install

if you want to see output ig you use playwright.
npx playwright test
```

you can take screenshots of any page using class name like this

```
await page.locator('.header').screenshot({ path: 'screenshot.png'});
```