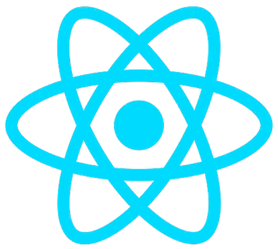
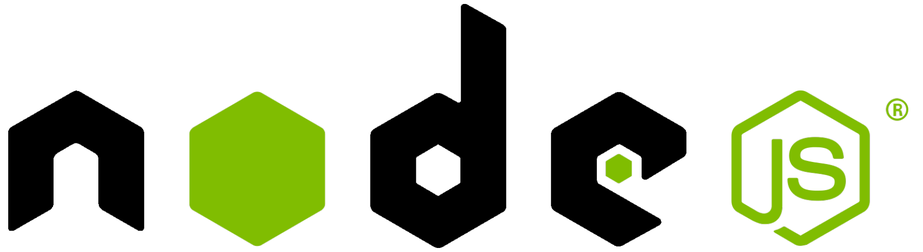
JAVASCRIPT

REACT FRONT-END PART 1

Anirach Mingkhwan

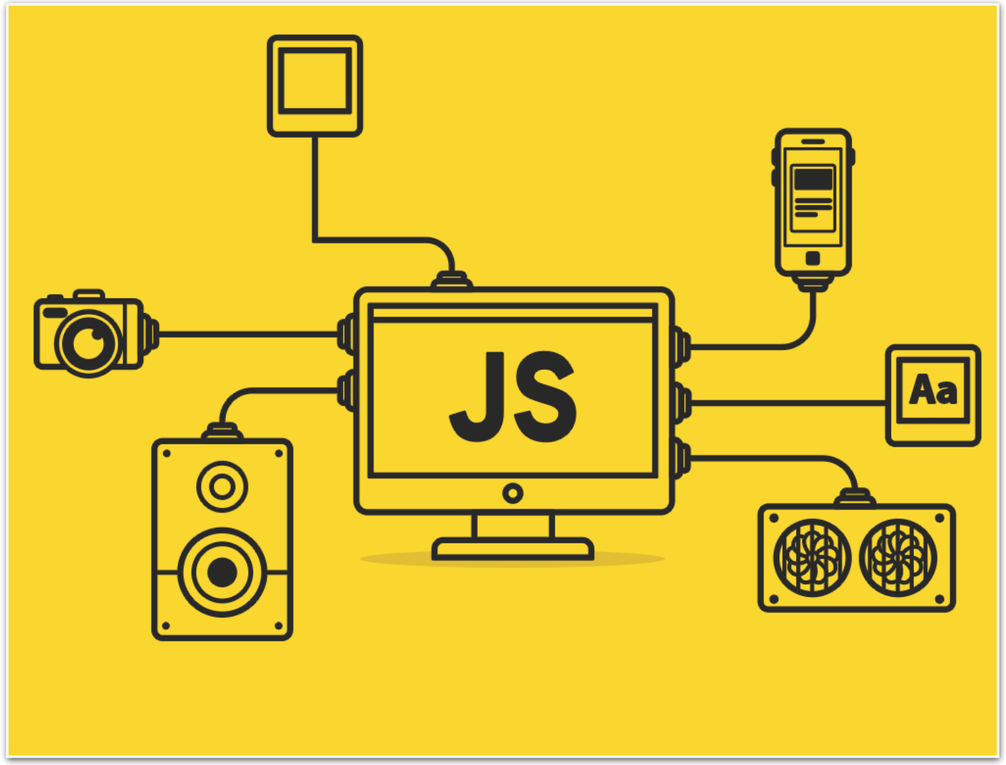
Anirach.m@fitm.kmutnb.ac.th





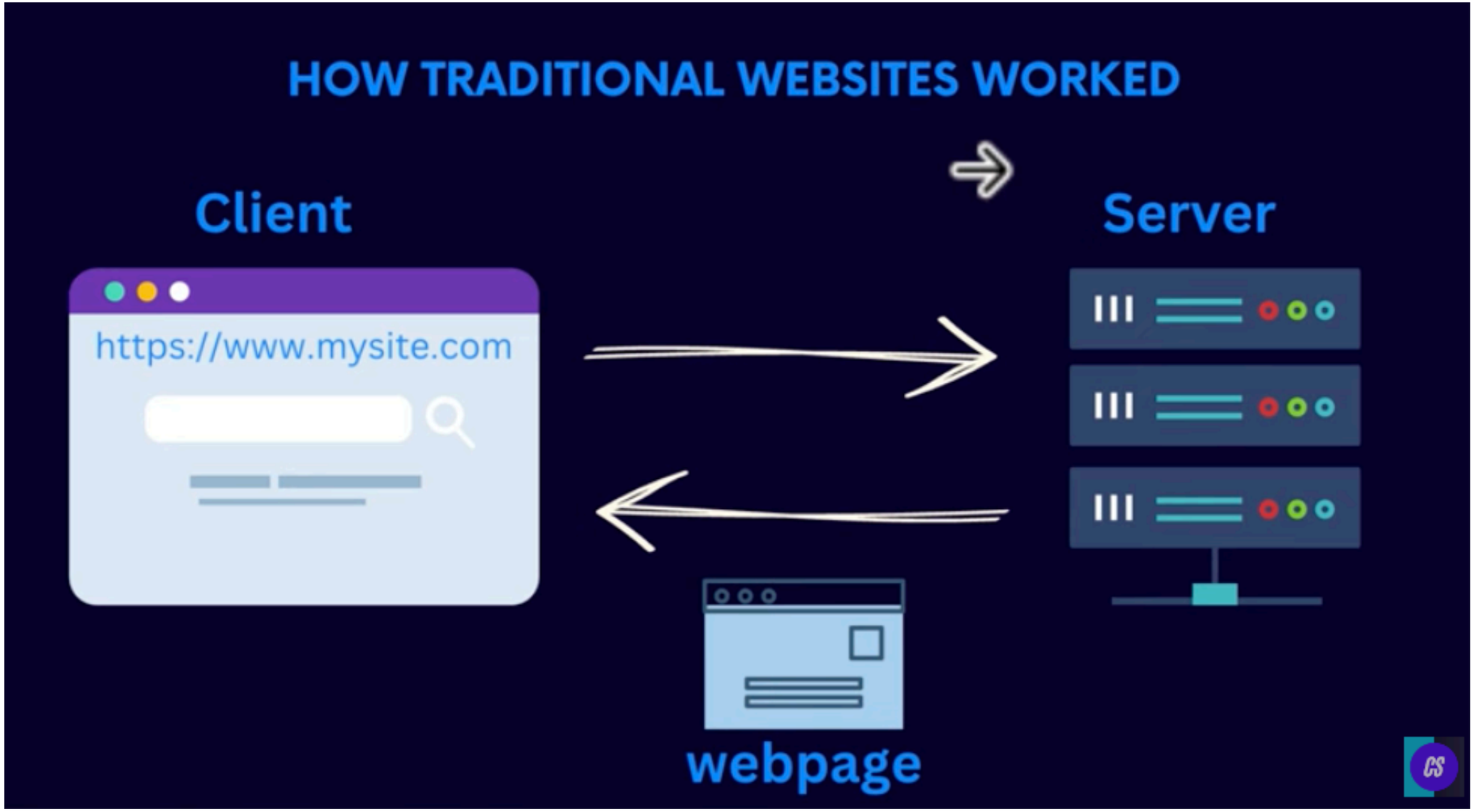
OUTLINE

- React Introduction
- Fundamentals of React
- Create React Project
- React props/state
- Thinking React



REACT INTRO

Serverside Rendering



React

The library for web and native user interfaces

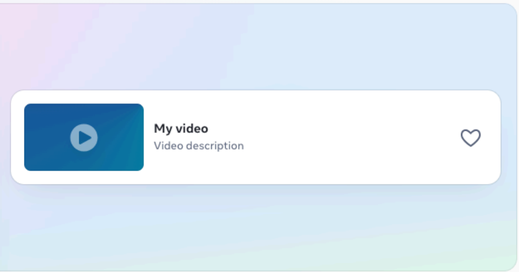
Learn React | API Reference

Create user interfaces from components

React lets you build user interfaces out of individual pieces called components. Create your own React components like `Thumbnail`, `LikeButton`, and `Video`. Then combine them into entire screens, pages, and apps.

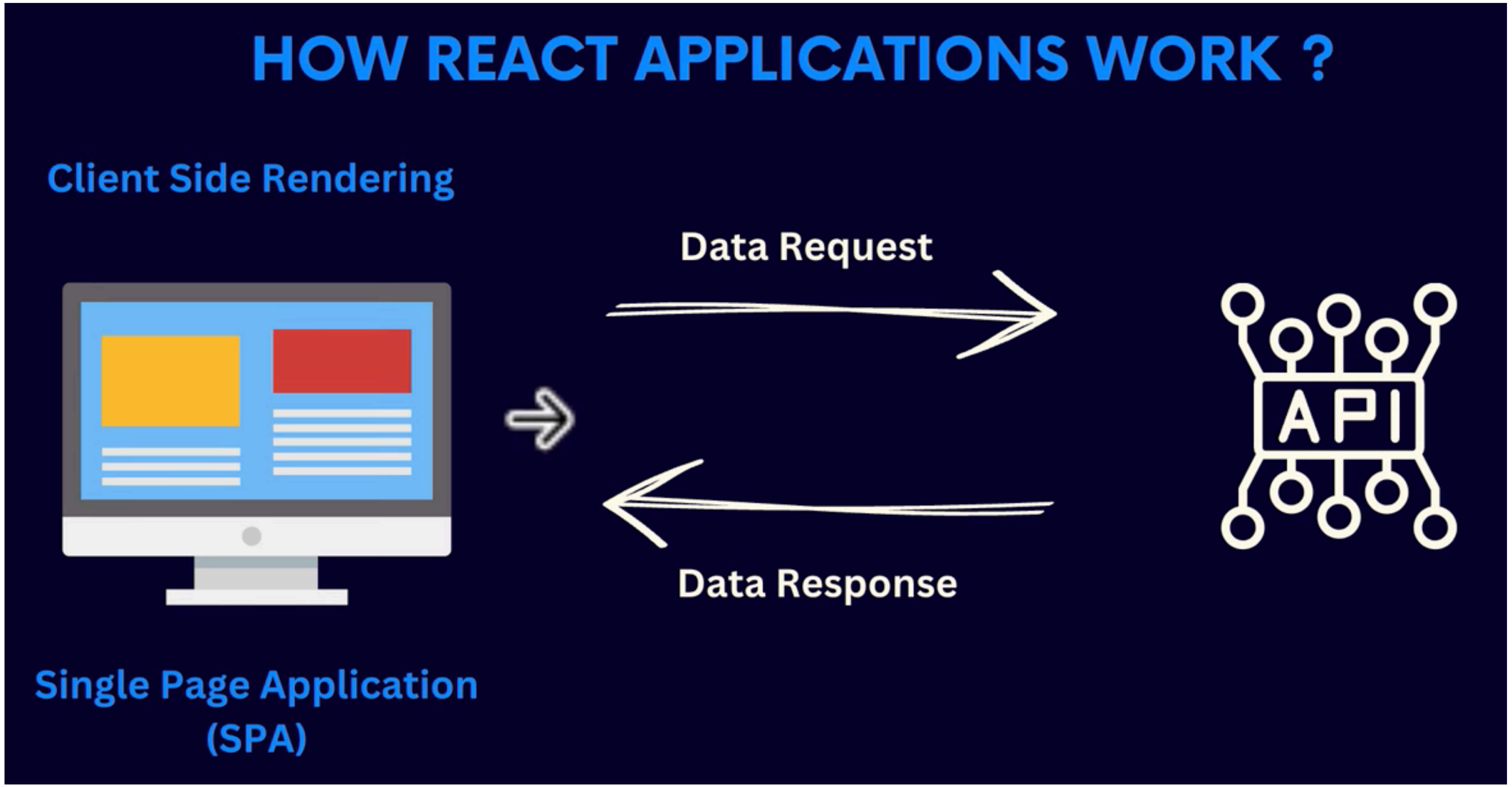
Video.js

```
function Video({ video }) {
  return (
    <div>
      <Thumbnail video={video} />
      <a href={video.url}>
        <h3>{video.title}</h3>
        <p>{video.description}</p>
      </a>
      <LikeButton video={video} />
    </div>
  );
}
```



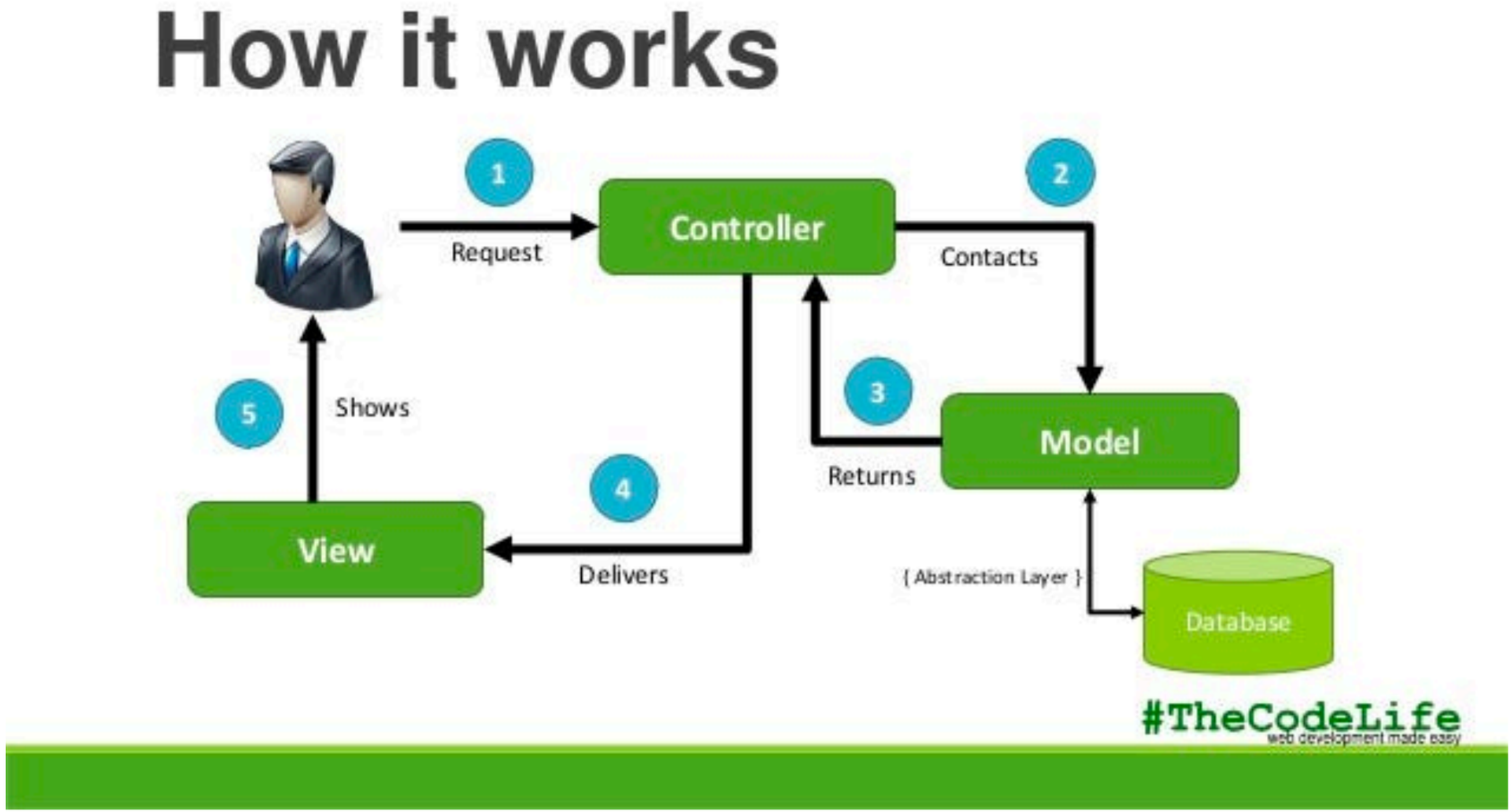
Whether you work on your own or with thousands of other developers, using React feels the same. It is designed to let you seamlessly combine components written by independent people, teams, and organizations.

https://react.dev/


HOW REACT APPLICATIONS WORK ?
Client Side Rendering
Data Request
API
Data Response
Single Page Application
(SPA)

What is React?

- React is an open-source JavaScript library for creating user interfaces.
- It is a library, not a framework.
- Used for handling the "view" layer of the application.
- Allow us to create large web applications, that change data without reloading the webpage.


How it works
1
Request
Controller
2
Contacts
3
Returns
Model
4
Delivers
View
5
Shows
{ Abstraction Layer }
Database
#TheCodeLife
web development made easy

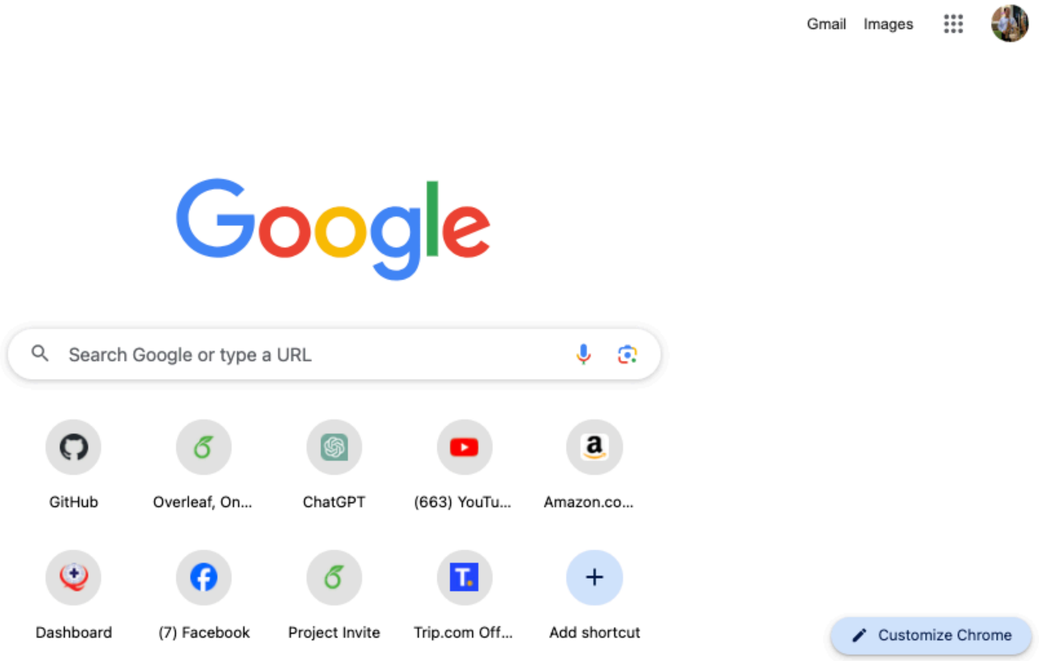
Why you should learn React?

- Developed by Facebook.
- JSX.
- In-demand skill.
- Could be combined with any tech stack.
- Mobile application development with React Native.

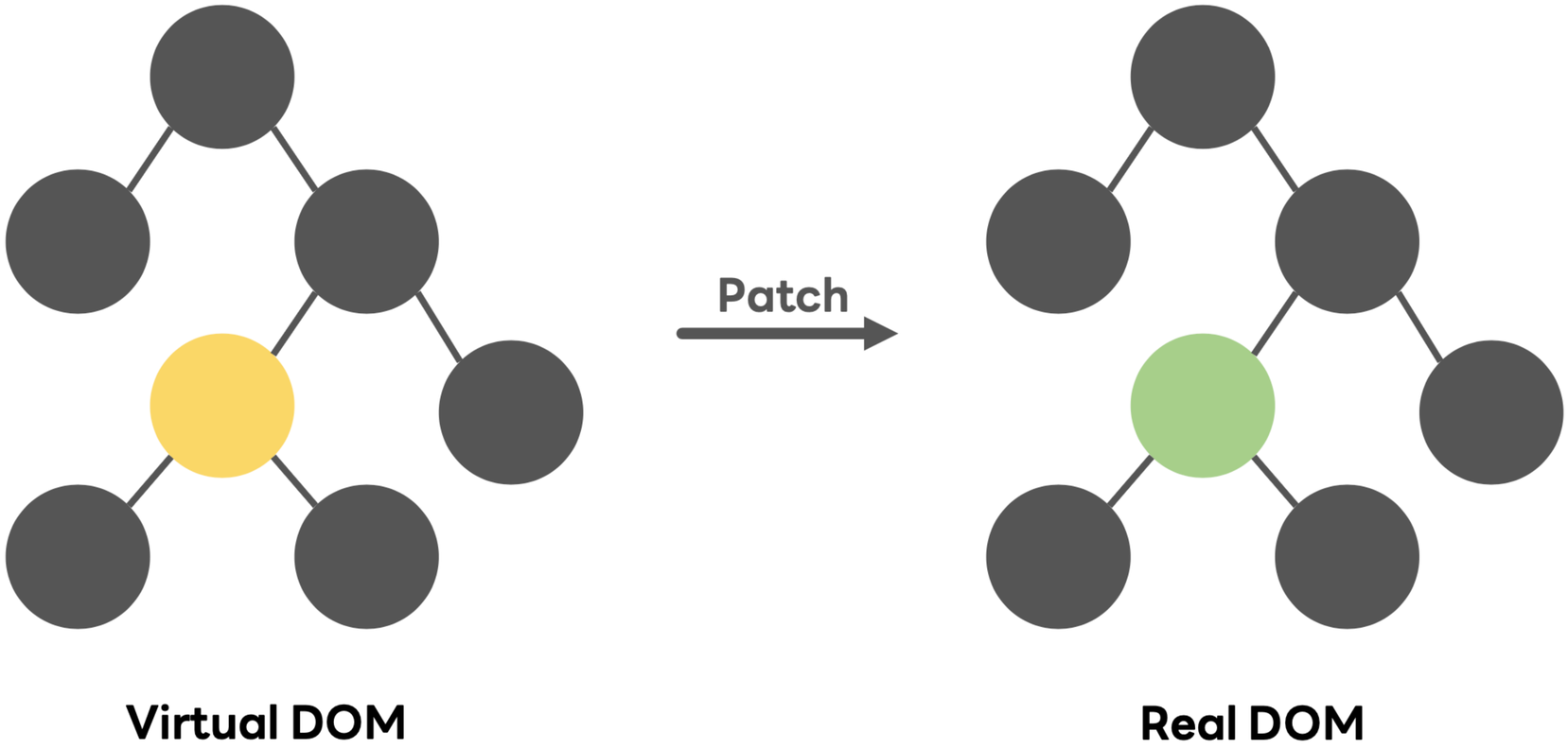
FUNDAMENTALS OF REACT

Component-based

- React is component-based.
- React application is a combination of multiple components.
- Components are re-useable.



Virtual DOM & Real DOM



DOM = Document Object Model

Core concepts of React

Component

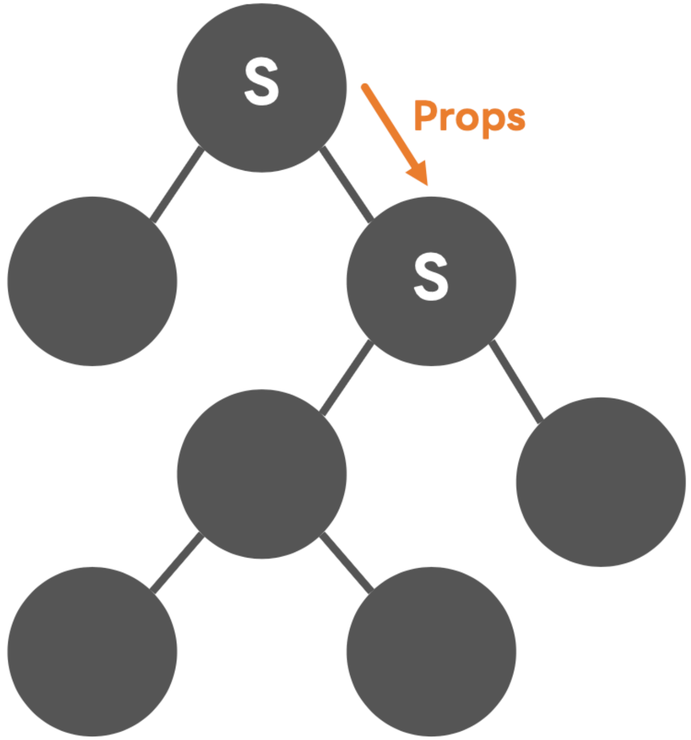
ส่วนประกอบต่างๆ ภายในเว็บไซต์
เราจะมองทุกสิ่งเป็น Component
ถ้าอ้างอิงตามภาพคือ ลูกวงกลม

State

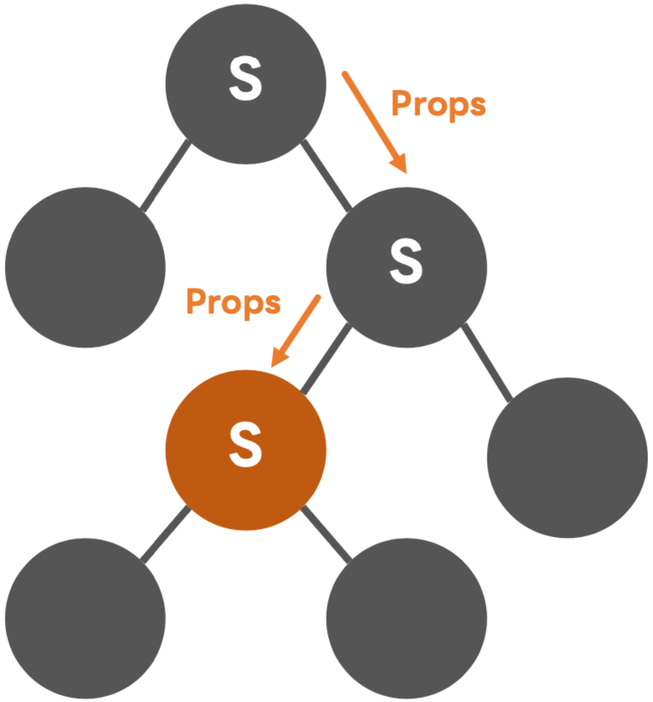
ข้อมูลที่อยู่ใน Component โดย
ข้อมูลจะถูกดำเนินการตามลำดับ

Props

ข้อมูลที่ถูกส่งระหว่าง Components
ซึ่งข้อมูลที่ถูกส่งคือ State ของ
Component จากต้นทางซึ่งจะ
ถูกถ่ายทอดจนกว่าจะถึงจุดจบ



Flow of props

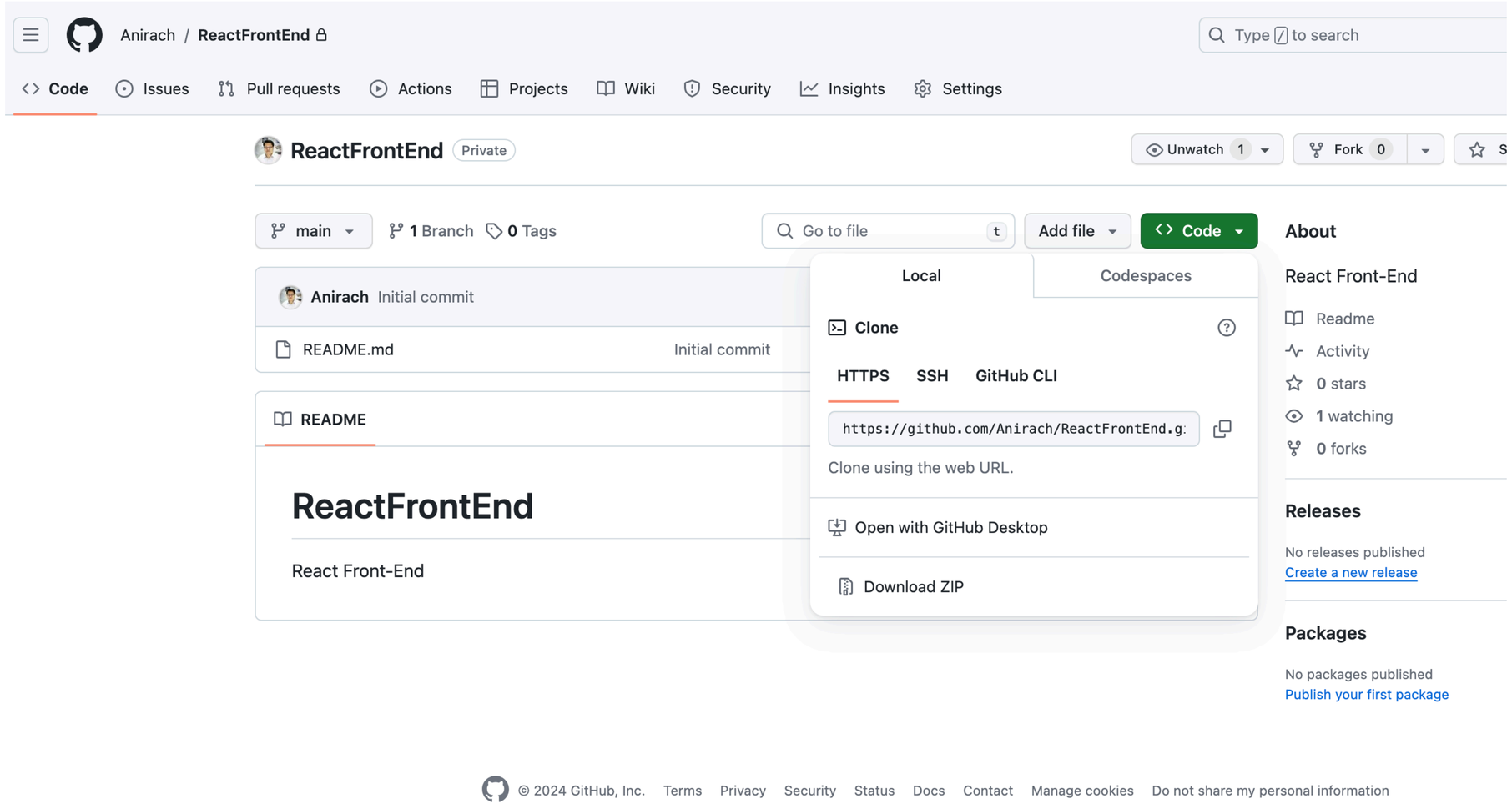


Passing values between components

ส่งผ่านค่าระหว่างส่วนประกอบ

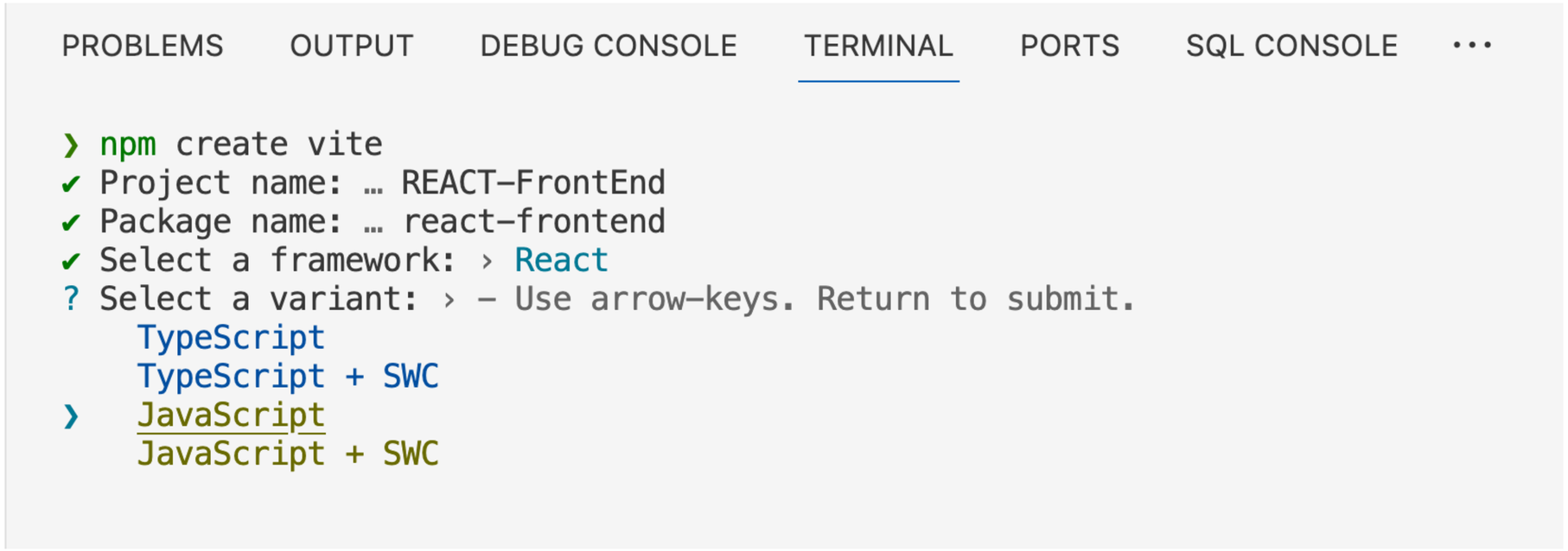
CREATE REACT PROJECT

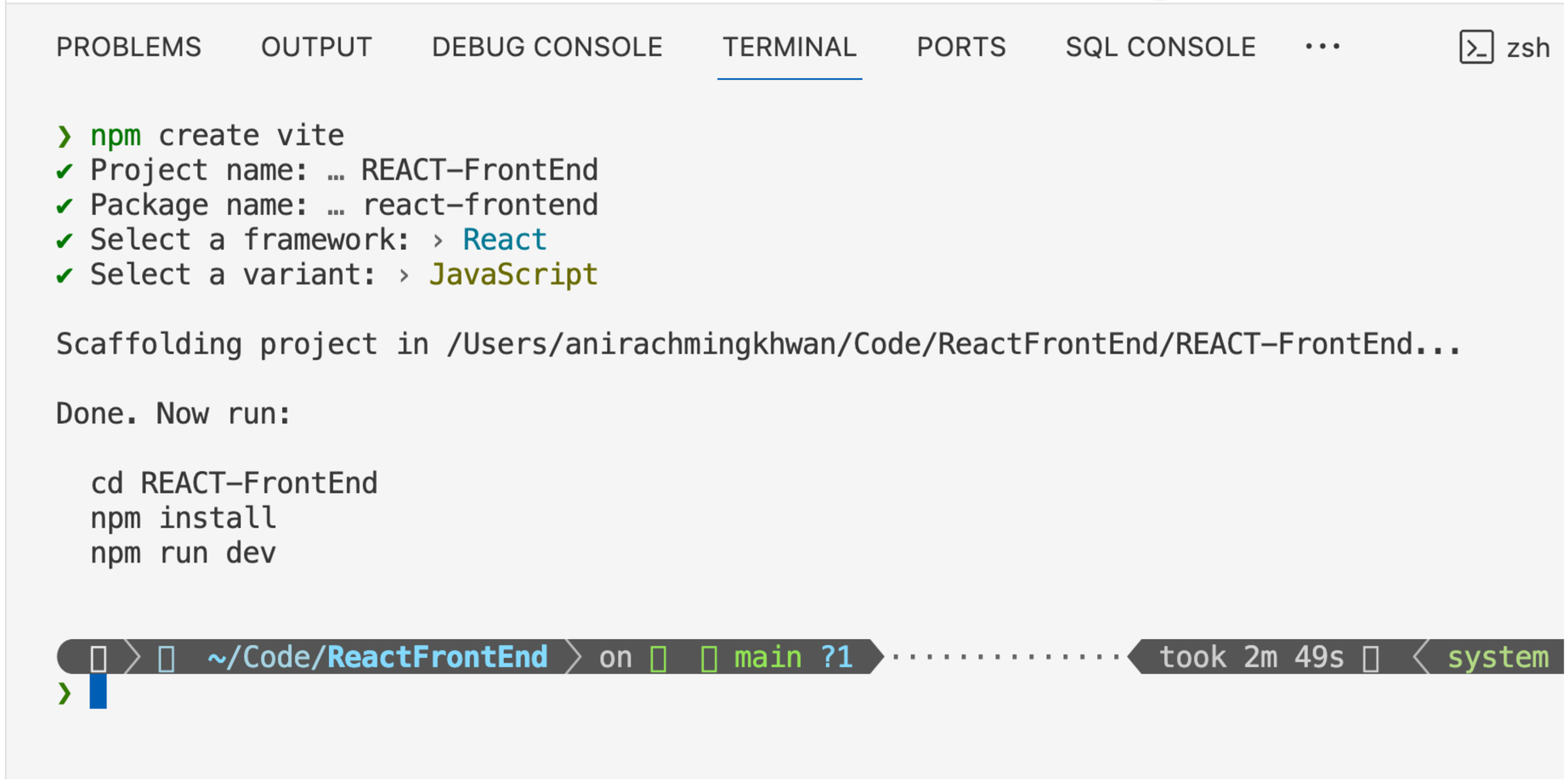
Create GitHUB Repository

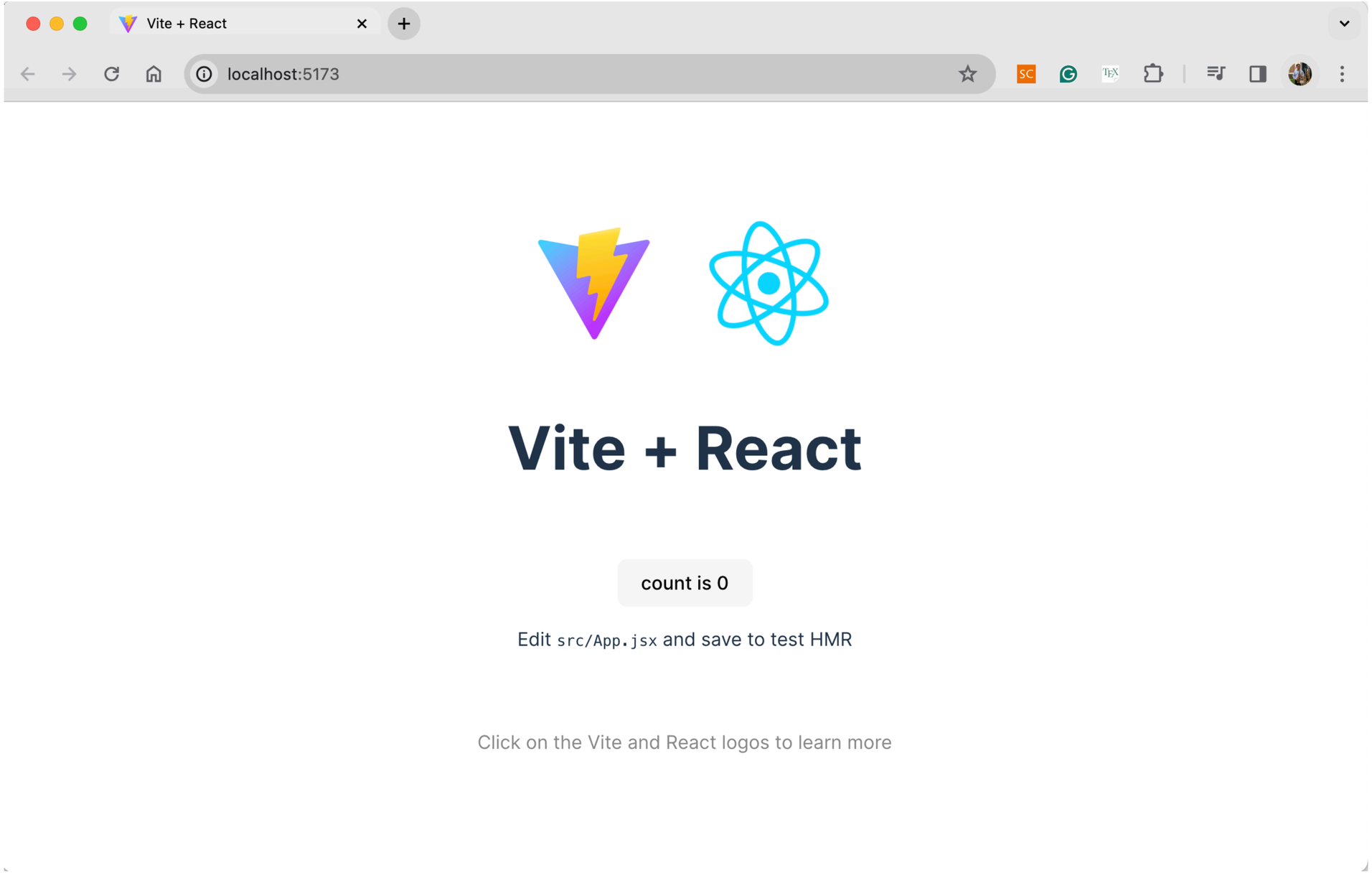


PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SQL CONSOLE

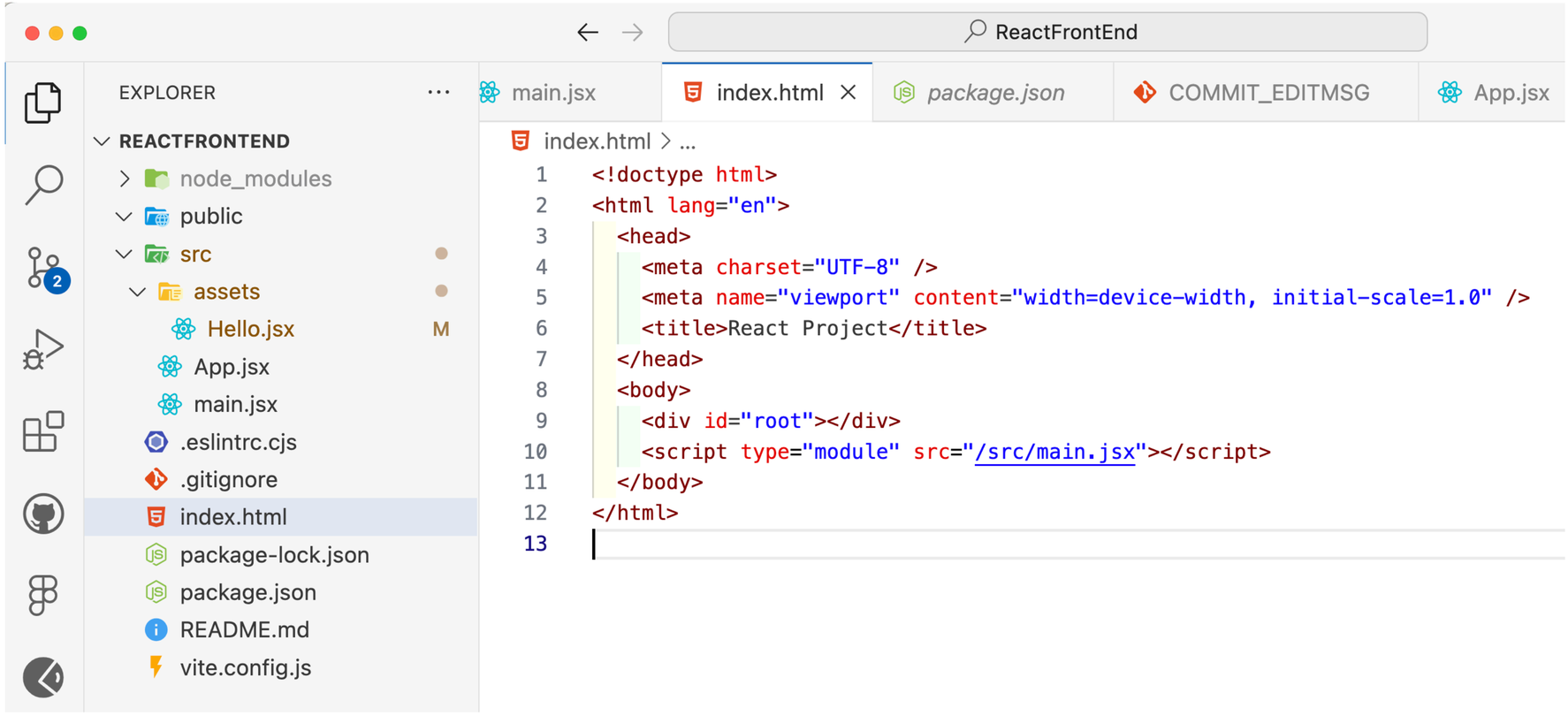
```
❯ npm create vite
✔ Project name: … REACT-FrontEnd
✔ Package name: … react-frontend
? Select a framework: › - Use arrow-keys. Return to submit.
    Vanilla
    Vue
❯   React
    Preact
    Lit
    Svelte
    Solid
    Qwik
    Others
```

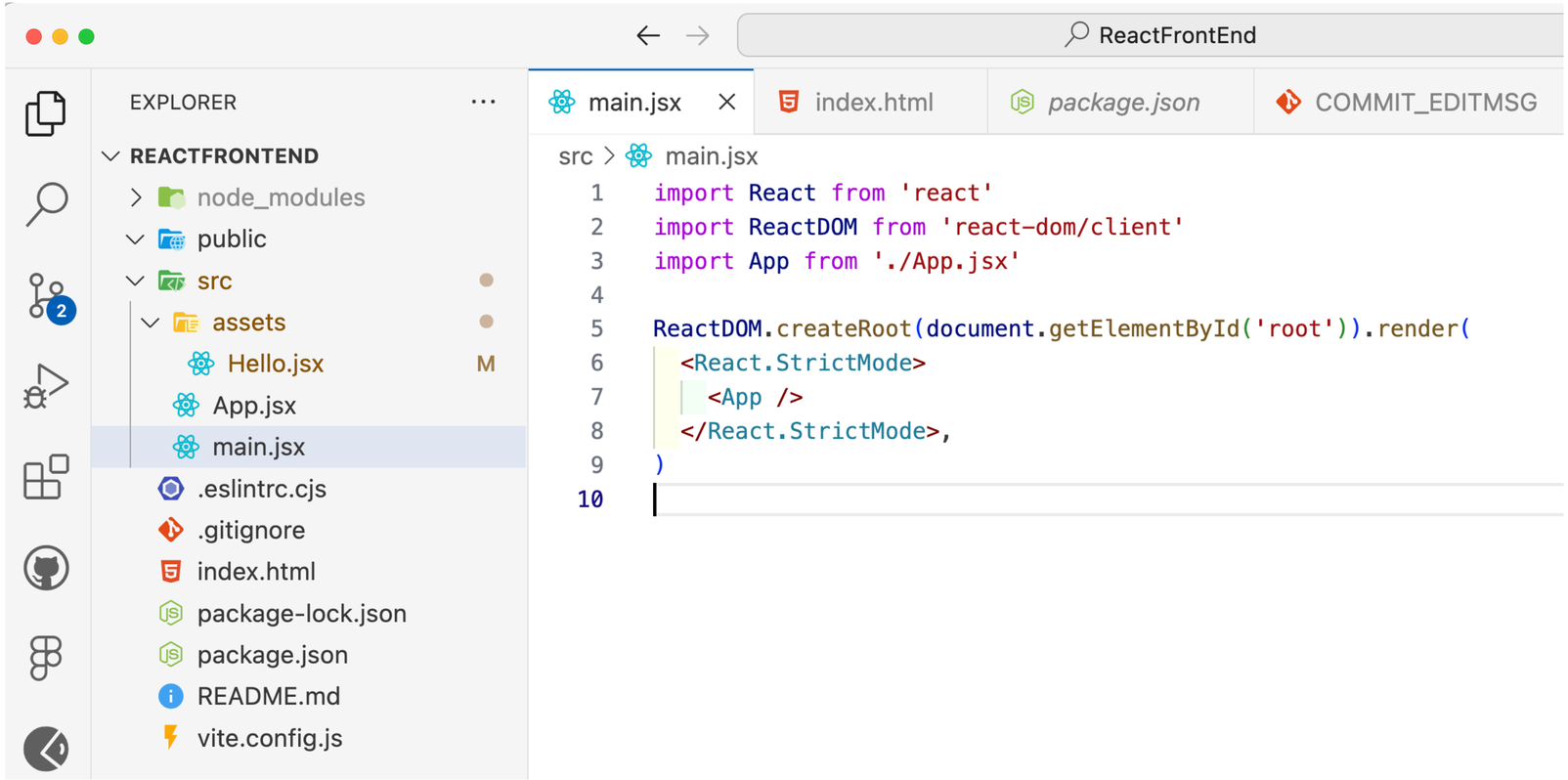

PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
SQL CONSOLE
❯ npm create vite
✔ Project name: … REACT-FrontEnd
✔ Package name: … react-frontend
✔ Select a framework: › React
? Select a variant: › - Use arrow-keys. Return to submit.
TypeScript
TypeScript + SWC
❯ JavaScript
JavaScript + SWC

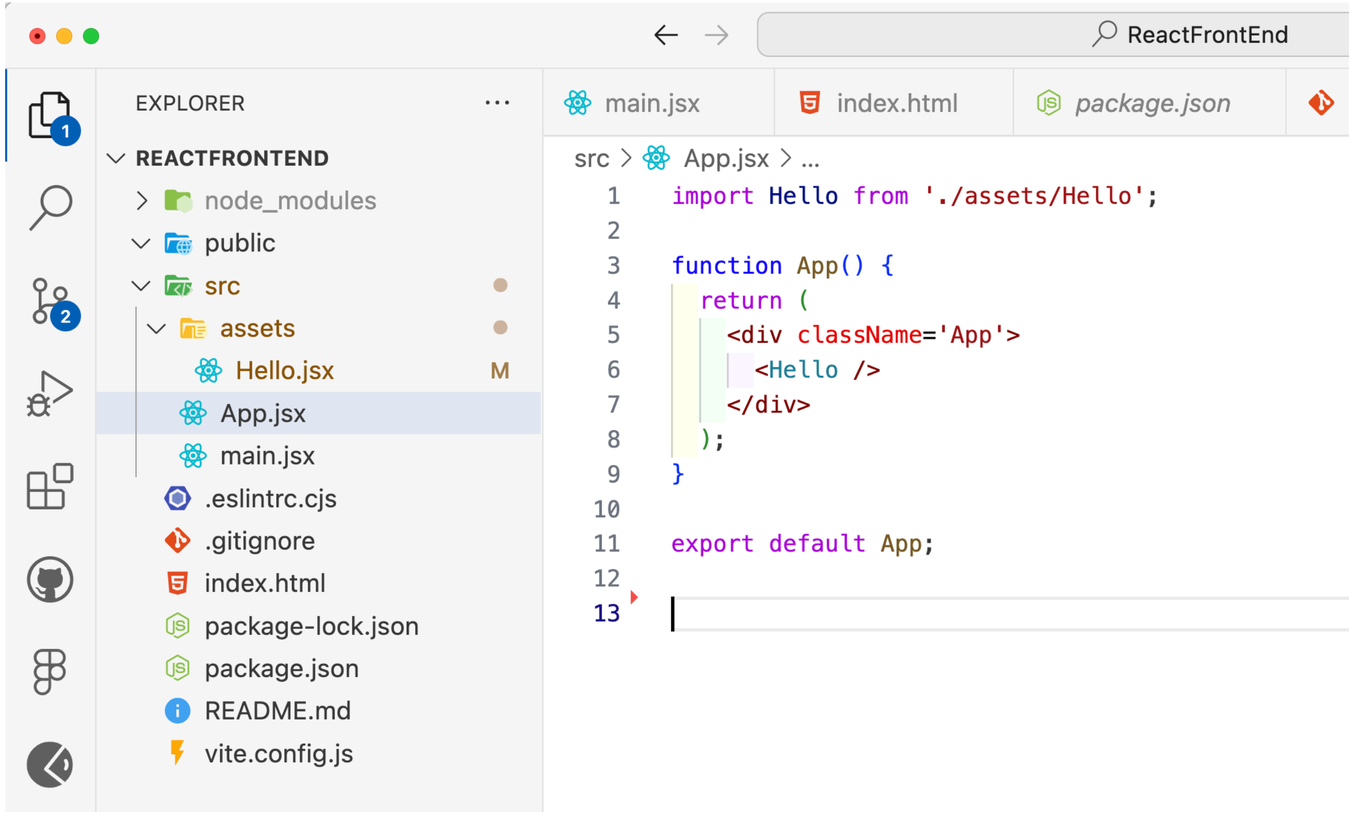

PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
SQL CONSOLE
zsh
❯ npm create vite
✔ Project name: … REACT-FrontEnd
✔ Package name: … react-frontend
✔ Select a framework: › React
✔ Select a variant: › JavaScript
Scaffolding project in /Users/anirachmingkhwan/Code/ReactFrontEnd/REACT-FrontEnd...
Done. Now run:
cd REACT-FrontEnd
npm install
npm run dev
~/Code/ReactFrontEnd
on
main ?1
took 2m 49s
system
❯

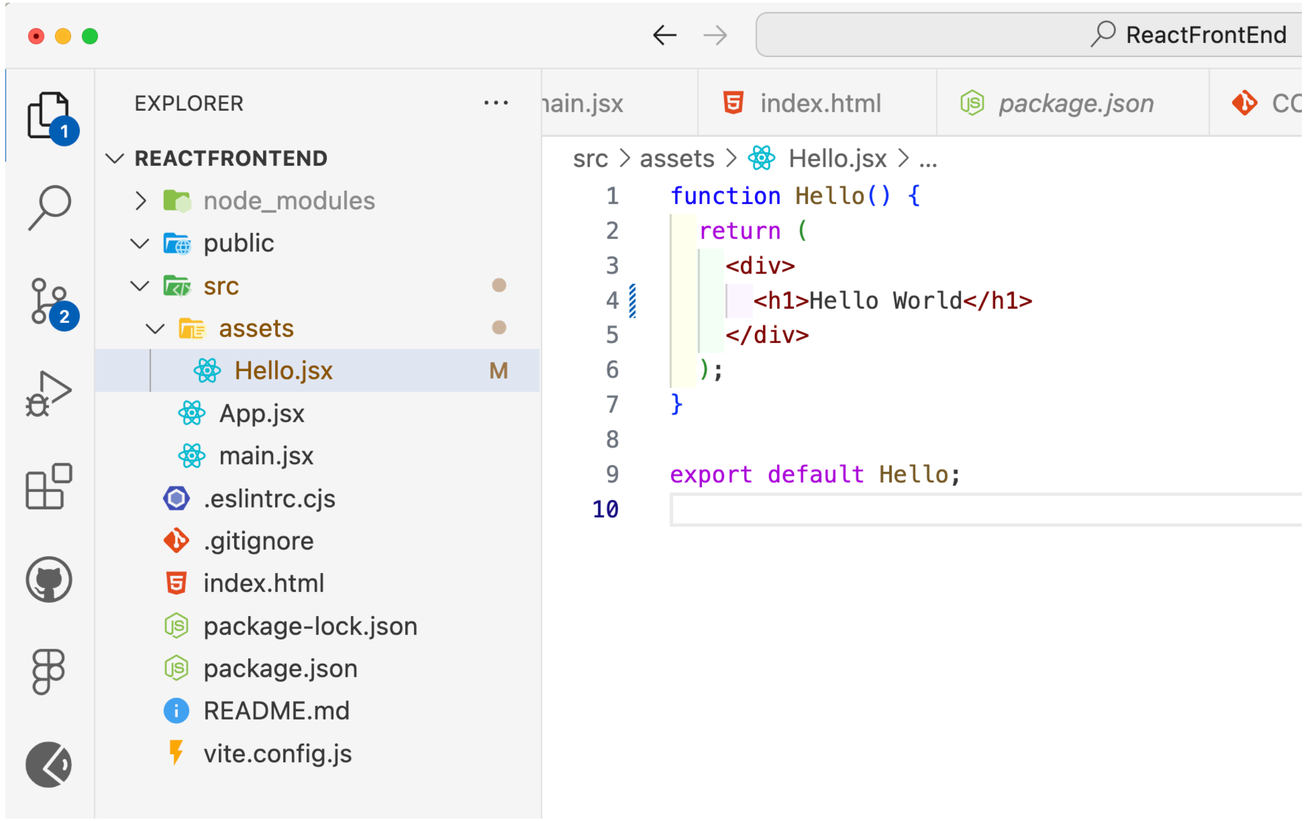

Vite + React
localhost:5173
Vite + React
count is 0
Edit src/App.jsx and save to test HMR
Click on the Vite and React logos to learn more

STRIP TO BASIC


ReactFrontEnd
EXPLORER
REACTFRONTEND
node_modules
public
src
assets
Hello.jsx
M
App.jsx
main.jsx
.eslintrc.cjs
.gitignore
index.html
package-lock.json
package.json
README.md
vite.config.js
main.jsx
index.html
package.json
COMMIT_EDITMSG
App.jsx
index.html > ...
1 <!doctype html>
2 <html lang="en">
3 <head>
4 <meta charset="UTF-8" />
5 <meta name="viewport" content="width=device-width, initial-scale=1.0" />
6 <title>React Project</title>
7 </head>
8 <body>
9
10 <script type="module" src="/src/main.jsx"></script>
11 </body>
12 </html>
13


ReactFrontEnd
EXPLORER
REACTFRONTEND
node_modules
public
src
assets
Hello.jsx
M
App.jsx
main.jsx
.eslintrc.cjs
.gitignore
index.html
package-lock.json
package.json
README.md
vite.config.js
main.jsx
index.html
package.json
COMMIT_EDITMSG
src > main.jsx
1 import React from 'react'
2 import ReactDOM from 'react-dom/client'
3 import App from './App.jsx'
4
5 ReactDOM.createRoot(document.getElementById('root')).render(
6 <React.StrictMode>
7 <App />
8 </React.StrictMode>,
9)
10

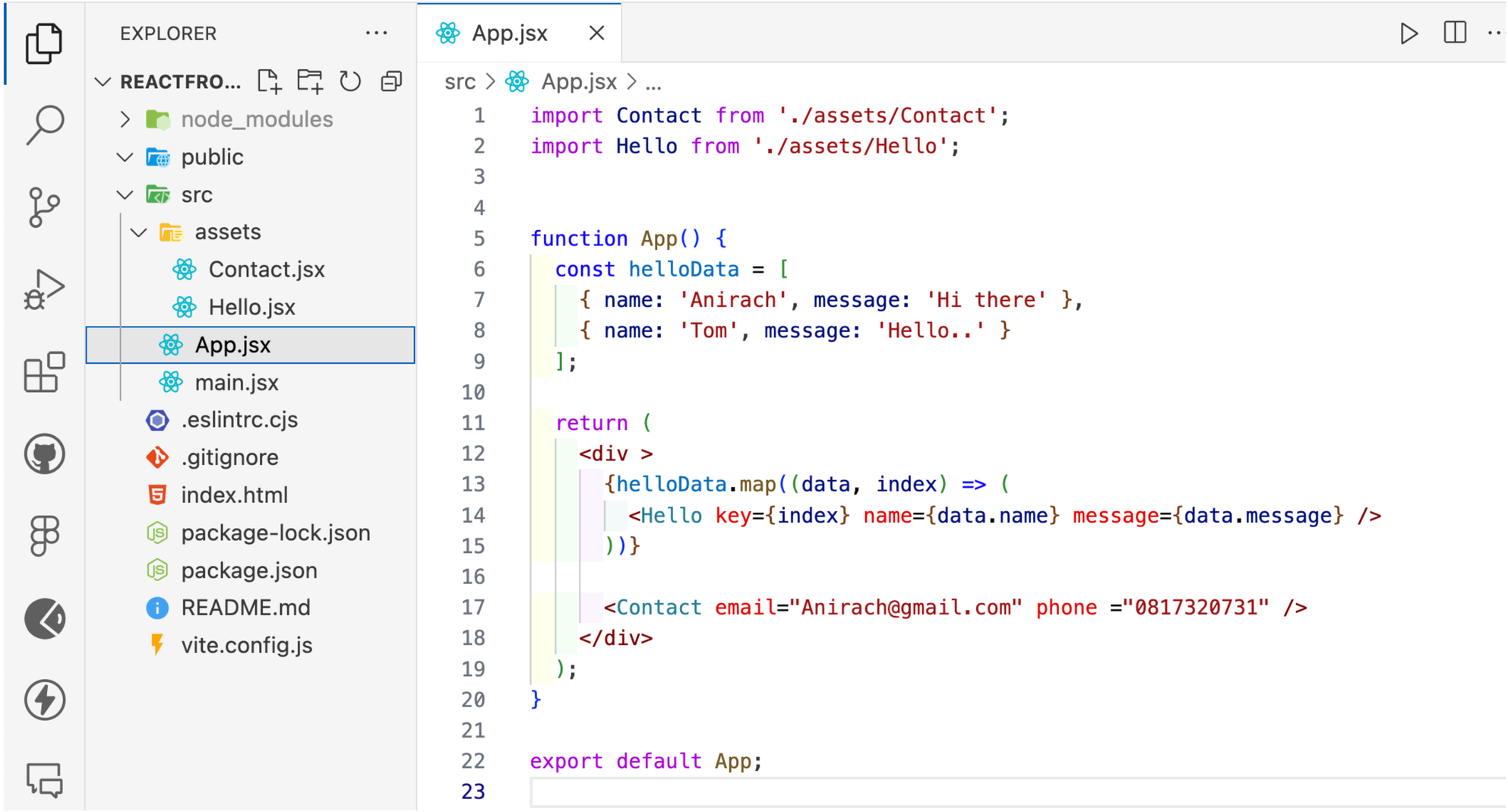

ReactFrontEnd
EXPLORER
REACTFRONTEND
node_modules
public
src
assets
Hello.jsx
M
App.jsx
main.jsx
.eslintrc.cjs
.gitignore
index.html
package-lock.json
package.json
README.md
vite.config.js
main.jsx
index.html
package.json
src > App.jsx > ...
1 import Hello from './assets/Hello';
2
3 function App() {
4 return (
5
6 <Hello />
7
8);
9 }
10
11 export default App;
12
13

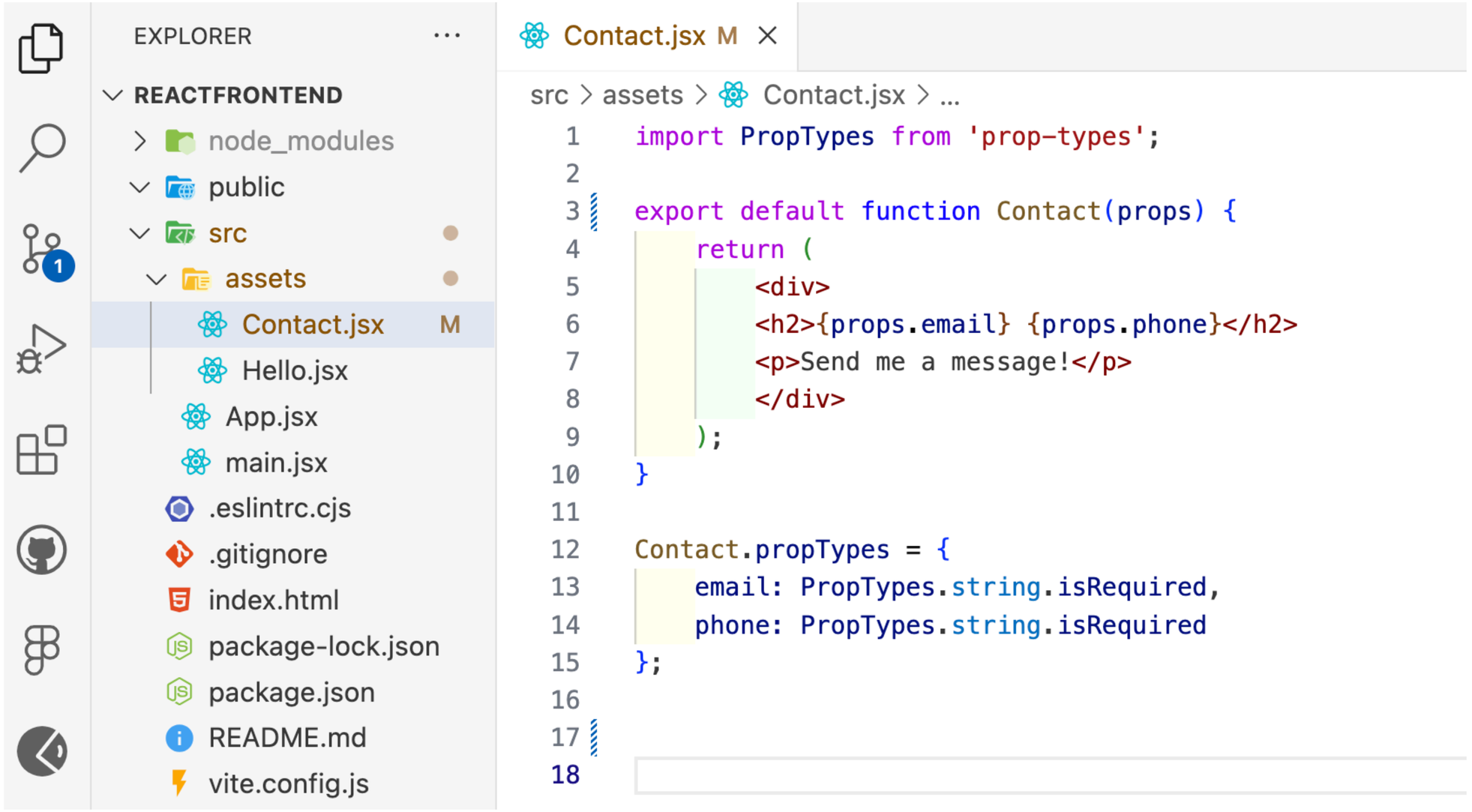

ReactFrontEnd
EXPLORER
REACTFRONTEND
node_modules
public
src
assets
Hello.jsx
M
App.jsx
main.jsx
.eslintrc.cjs
.gitignore
index.html
package-lock.json
package.json
README.md
vite.config.js
index.html
package.json
src > assets > Hello.jsx > ...
function Hello() {
return (

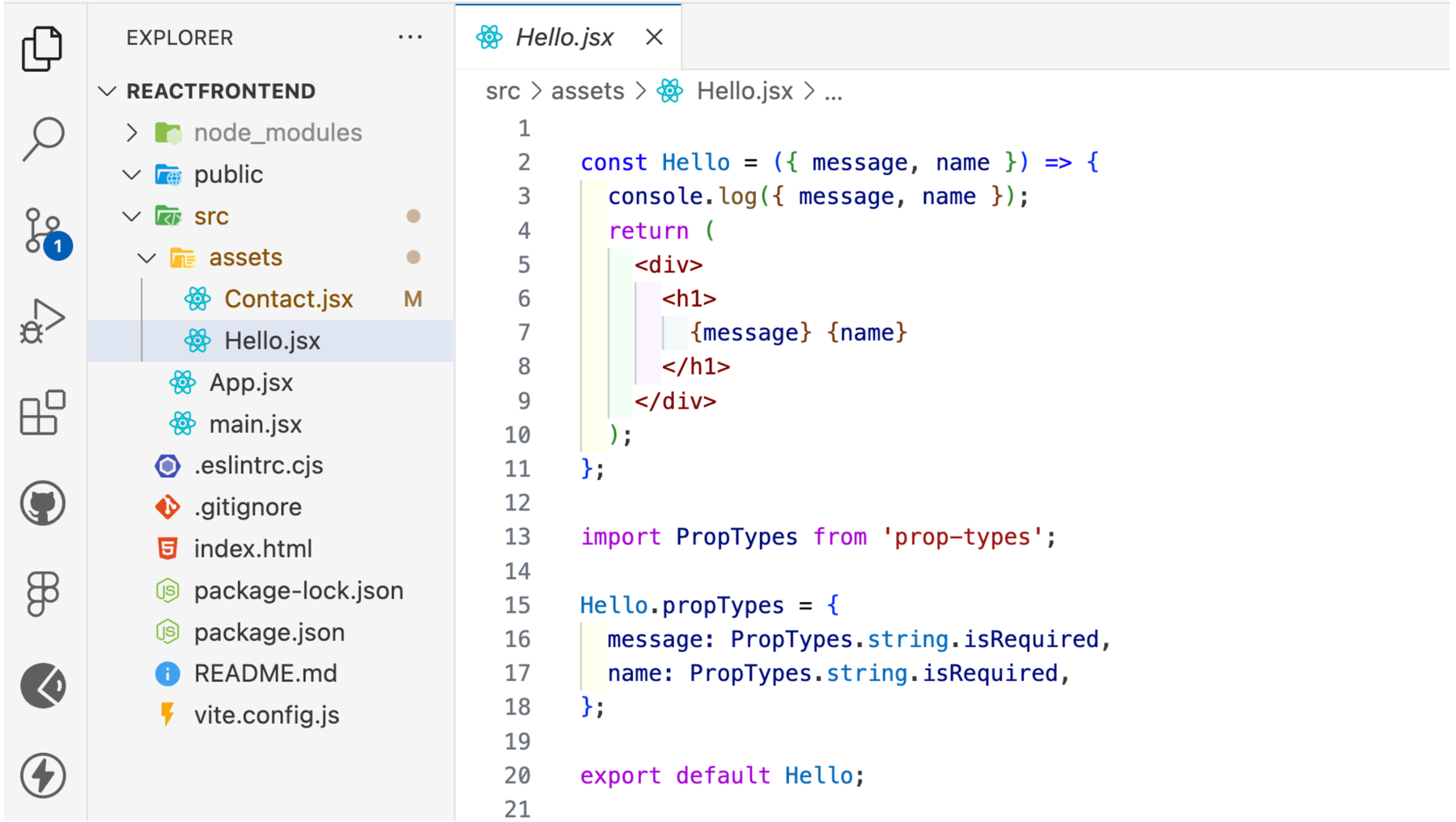
<h1>Hello World</h1>

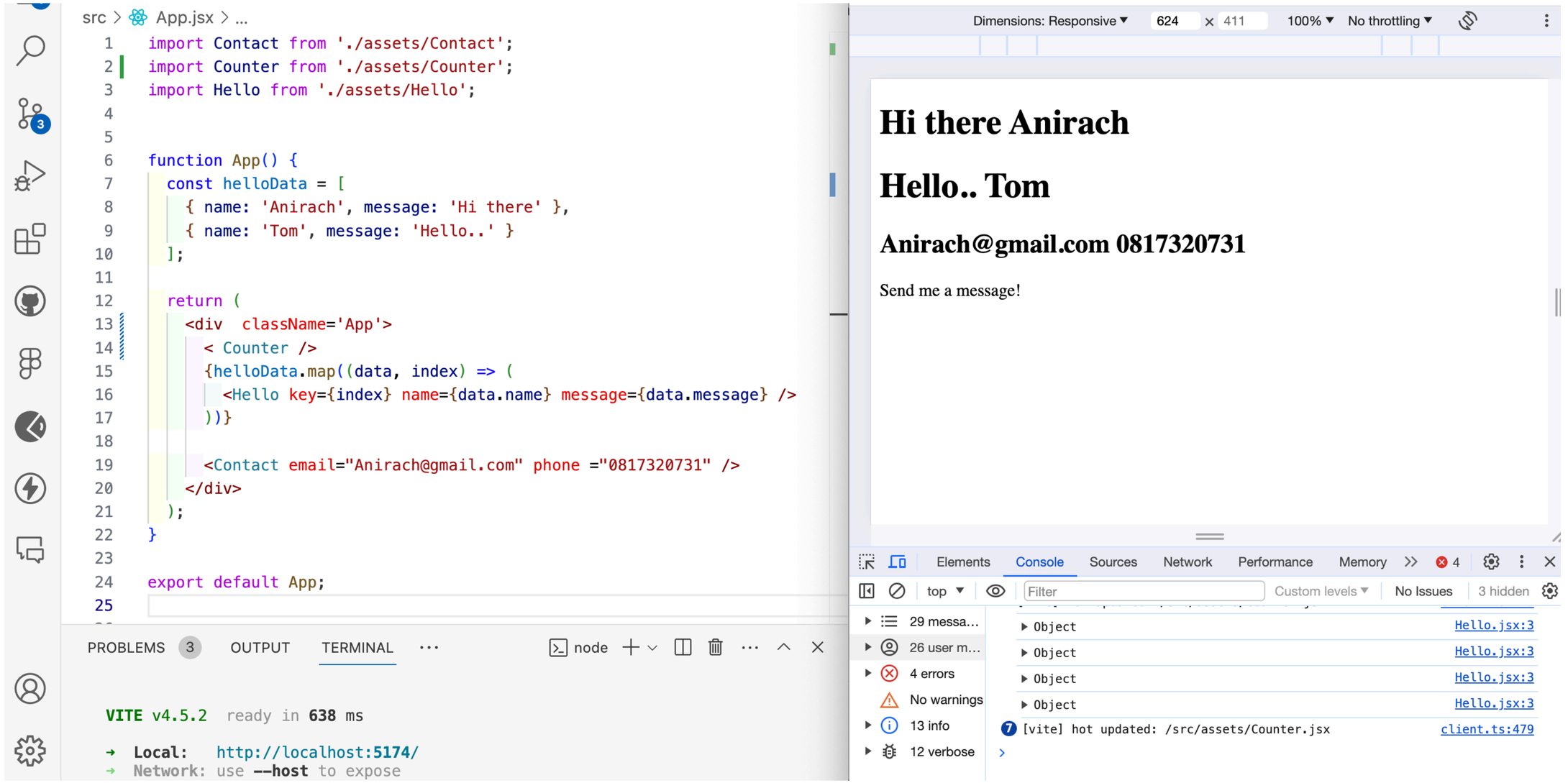
);
}
export default Hello;

REACT - PROPS


EXPLORER
REACTFRO...
node_modules
public
src
assets
Contact.jsx
Hello.jsx
App.jsx
main.jsx
.eslintrc.cjs
.gitignore
index.html
package-lock.json
package.json
README.md
vite.config.js
App.jsx
src > App.jsx > ...
1 import Contact from './assets/Contact';
2 import Hello from './assets/Hello';
3
4
5 function App() {
6 const helloData = [
7 { name: 'Anirach', message: 'Hi there' },
8 { name: 'Tom', message: 'Hello..' }
9];
10
11 return (
12
13 {helloData.map((data, index) => (
14 <Hello key={index} name={data.name} message={data.message} />
15))}
16
17 <Contact email="Anirach@gmail.com" phone ="0817320731" />
18
19);
20 }
21
22 export default App;
23


EXPLORER
REACTFRONTEND
node_modules
public
src
assets
Contact.jsx M
Hello.jsx
App.jsx
main.jsx
.eslintrc.cjs
.gitignore
index.html
package-lock.json
package.json
README.md
vite.config.js
Contact.jsx M
src > assets > Contact.jsx > ...
1 import PropTypes from 'prop-types';
2
3 export default function Contact(props) {
4 return (
5
6 <h2>{props.email} {props.phone}</h2>
7 Send me a message!
8
9);
10 }
11
12 Contact.propTypes = {
13 email: PropTypes.string.isRequired,
14 phone: PropTypes.string.isRequired
15 };
16
17
18


EXPLORER
REACTFRONTEND
node_modules
public
src
assets
Contact.jsx M
Hello.jsx
App.jsx
main.jsx
.eslintrc.cjs
.gitignore
index.html
package-lock.json
package.json
README.md
vite.config.js
Hello.jsx
src > assets > Hello.jsx > ...
1
2 const Hello = ({ message, name }) => {
3 console.log({ message, name });
4 return (
5
6 <h1>
7 {message} {name}
8 </h1>
9
10);
11 };
12
13 import PropTypes from 'prop-types';
14
15 Hello.propTypes = {
16 message: PropTypes.string.isRequired,
17 name: PropTypes.string.isRequired,
18 };
19
20 export default Hello;
21


src > App.jsx > ...
1 import Contact from './assets/Contact';
2 import Counter from './assets/Counter';
3 import Hello from './assets/Hello';
4
5
6 function App() {
7 const helloData = [
8 { name: 'Anirach', message: 'Hi there' },
9 { name: 'Tom', message: 'Hello..' }
10];
11
12 return (
13
14 < Counter />
15 {helloData.map((data, index) => (
16 <Hello key={index} name={data.name} message={data.message} />
17))}
18
19 <Contact email="Anirach@gmail.com" phone ="0817320731" />
20
21);
22 }
23
24 export default App;
25
PROBLEMS 3
OUTPUT
TERMINAL
node
VITE v4.5.2 ready in 638 ms
Local: http://localhost:5174/
Network: use --host to expose
Dimensions: Responsive
624
411
100%
No throttling
Hi there Anirach
Hello.. Tom
Anirach@gmail.com 0817320731
Send me a message!
Elements
Console
Sources
Network
Performance
Memory
top
Filter
Custom levels
No Issues
3 hidden
29 messa...
26 user m...
4 errors
No warnings
13 info
12 verbose
Object
Hello.jsx:3
Object
Hello.jsx:3
Object
Hello.jsx:3
Object
Hello.jsx:3
[vite] hot updated: /src/assets/Counter.jsx
client.ts:479

REACT - STATE

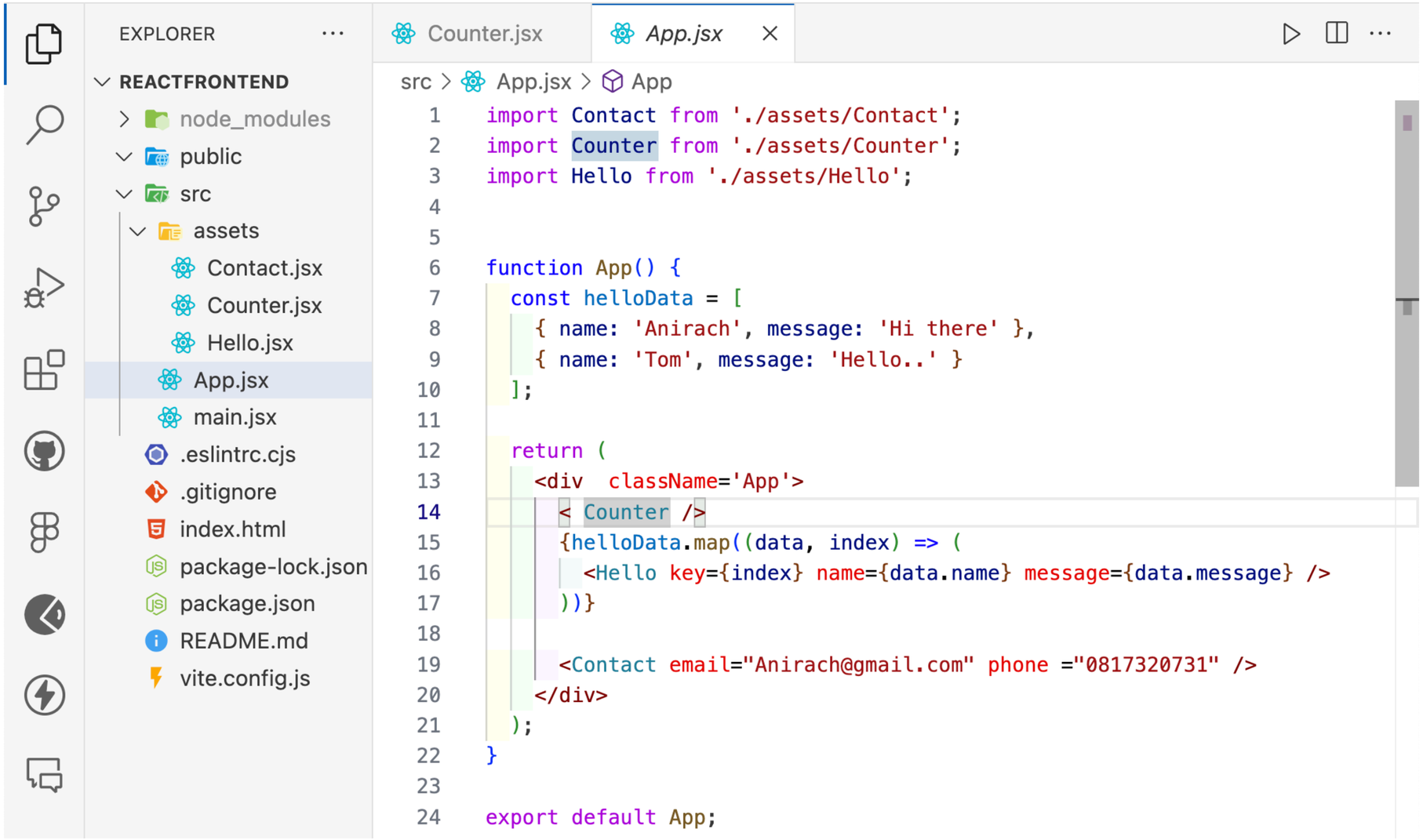
Props VS State

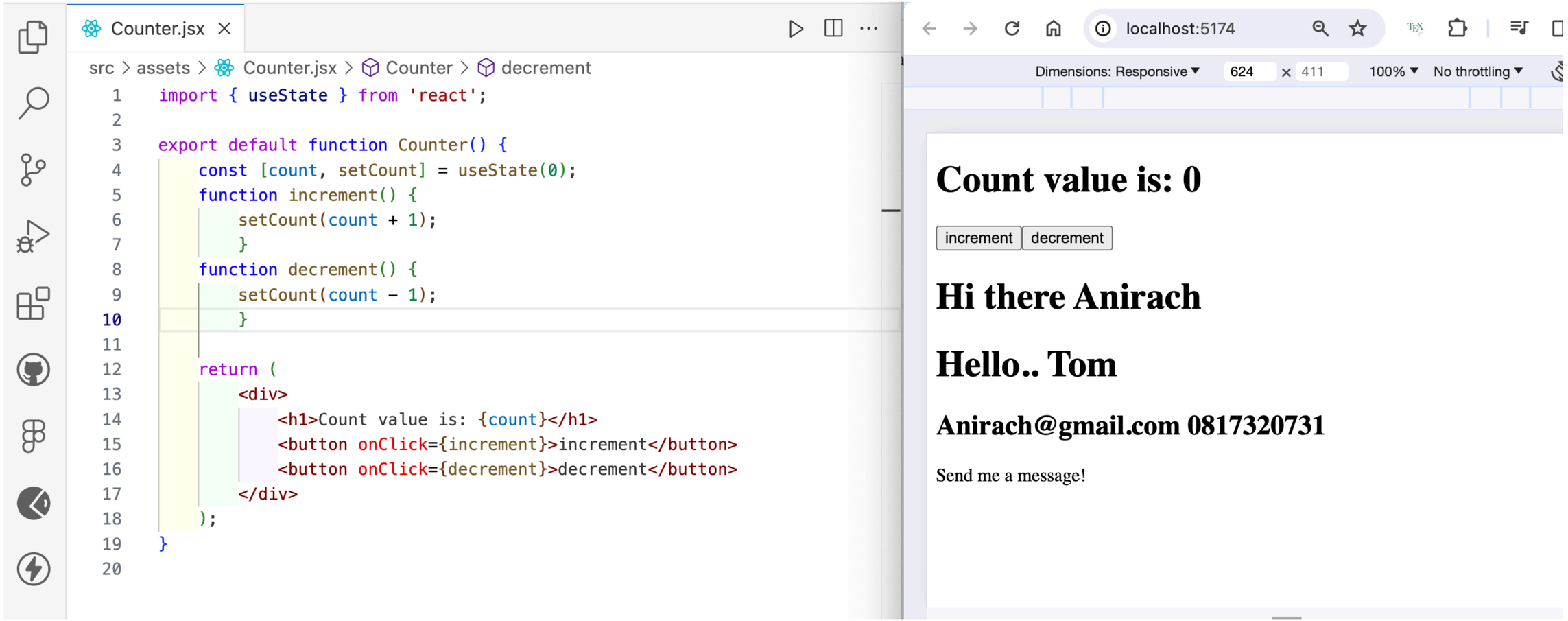
- **Props are passed to the component.**
- **State is contained inside the component.**
- **Props are immutable.**
- **State can be changed.**
- **Props are like parameters passed to a function.**
- **State is like a variable declared in function body.**


EXPLORER
REACTFRONTEND
node_modules
public
src
assets
Contact.jsx
Counter.jsx
Hello.jsx
App.jsx
main.jsx
.eslintrc.cjs
.gitignore
index.html
package-lock.json
package.json
README.md
vite.config.js
Counter.jsx
src > assets > Counter.jsx > ...
import { useState } from 'react';
export default function Counter() {
const [count, setCount] = useState(0);
function increment() {
setCount(count + 1);
}
function decrement() {
setCount(count - 1);
}
return (

<h1>Count value is: {count}</h1>
<button onClick={increment}>increment</button>
<button onClick={decrement}>decrement</button>

);
}


EXPLORER
REACTFRONTEND
node_modules
public
src
assets
Contact.jsx
Counter.jsx
Hello.jsx
App.jsx
main.jsx
.eslintrc.cjs
.gitignore
index.html
package-lock.json
package.json
README.md
vite.config.js
Counter.jsx
App.jsx
src > App.jsx > App
1 import Contact from './assets/Contact';
2 import Counter from './assets/Counter';
3 import Hello from './assets/Hello';
4
5
6 function App() {
7 const helloData = [
8 { name: 'Anirach', message: 'Hi there' },
9 { name: 'Tom', message: 'Hello..' }
10];
11
12 return (
13
14 < Counter />
15 {helloData.map((data, index) => (
16 <Hello key={index} name={data.name} message={data.message} />
17))}
18
19 <Contact email="Anirach@gmail.com" phone ="0817320731" />
20
21);
22 }
23
24 export default App;


Counter.jsx
src > assets > Counter.jsx > Counter > decrement
1 import { useState } from 'react';
2
3 export default function Counter() {
4 const [count, setCount] = useState(0);
5 function increment() {
6 setCount(count + 1);
7 }
8 function decrement() {
9 setCount(count - 1);
10 }
11
12 return (
13
14 <h1>Count value is: {count}</h1>
15 <button onClick={increment}>increment</button>
16 <button onClick={decrement}>decrement</button>
17
18);
19 }
20
localhost:5174
Dimensions: Responsive 624 x 411 100% No throttling
Count value is: 0
increment decrement
Hi there Anirach
Hello.. Tom
Anirach@gmail.com 0817320731
Send me a message!

THINKING REACT

https://react.dev/learn/thinking-in-react

Start with the mockup

Imagine that you already have a JSON API and a mockup from a designer.

The JSON API returns some data that looks like this:

```
[
  { category: "Fruits", price: "$1", stocked: true, name: "Apple" },
  { category: "Fruits", price: "$1", stocked: true, name: "Dragonfruit" },
  { category: "Fruits", price: "$2", stocked: false, name: "Passionfruit" },
  { category: "Vegetables", price: "$2", stocked: true, name: "Spinach" },
  { category: "Vegetables", price: "$4", stocked: false, name: "Pumpkin" },
  { category: "Vegetables", price: "$1", stocked: true, name: "Peas" }
]
```

The mockup look like this:

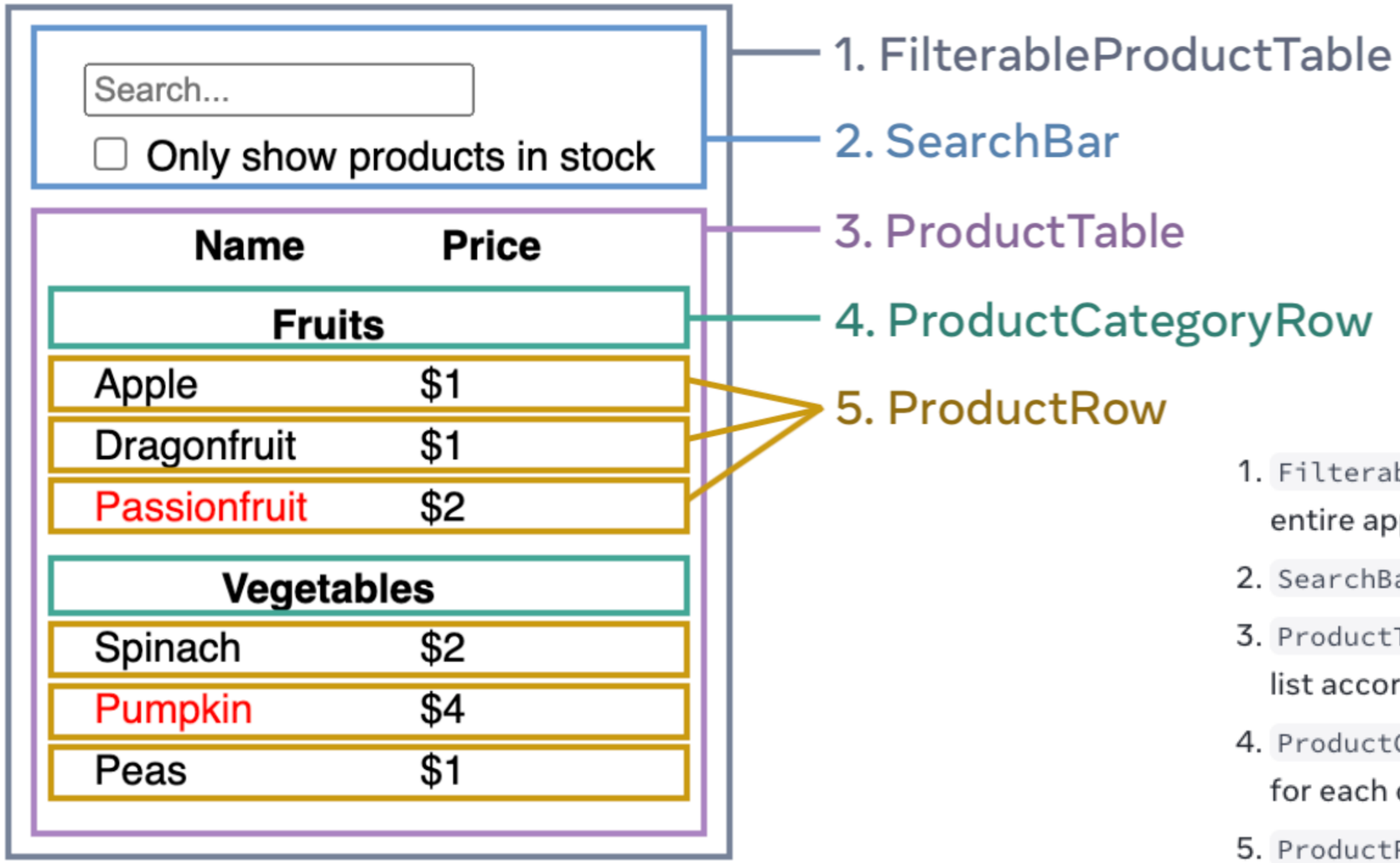

```
[
  { category: "Fruits", price: "$1", stocked: true, name: "Apple" },
  { category: "Fruits", price: "$1", stocked: true, name: "Dragonfruit" },
  { category: "Fruits", price: "$2", stocked: false, name: "Passionfruit" },
  { category: "Vegetables", price: "$2", stocked: true, name: "Spinach" },
  { category: "Vegetables", price: "$4", stocked: false, name: "Pumpkin" },
  { category: "Vegetables", price: "$1", stocked: true, name: "Peas" }
]
```

Step 1: Break the UI into a component hierarchy

- **Programming** -- use the same techniques for deciding if you should create a new function or object. One such technique is the single responsibility principle.
- **CSS** -- consider what you would make class selectors for.
- **Design** -- consider how you would organize the design's layers.

Step 1: Break the UI into a component hierarchy

There are five components on this screen:



1. `FilterableProductTable` (grey) contains the entire app.
2. `SearchBar` (blue) receives the user input.
3. `ProductTable` (lavender) displays and filters the list according to the user input.
4. `ProductCategoryRow` (green) displays a heading for each category.
5. `ProductRow` (yellow) displays a row for each product.

Step 2: Build a static version of React

App.js

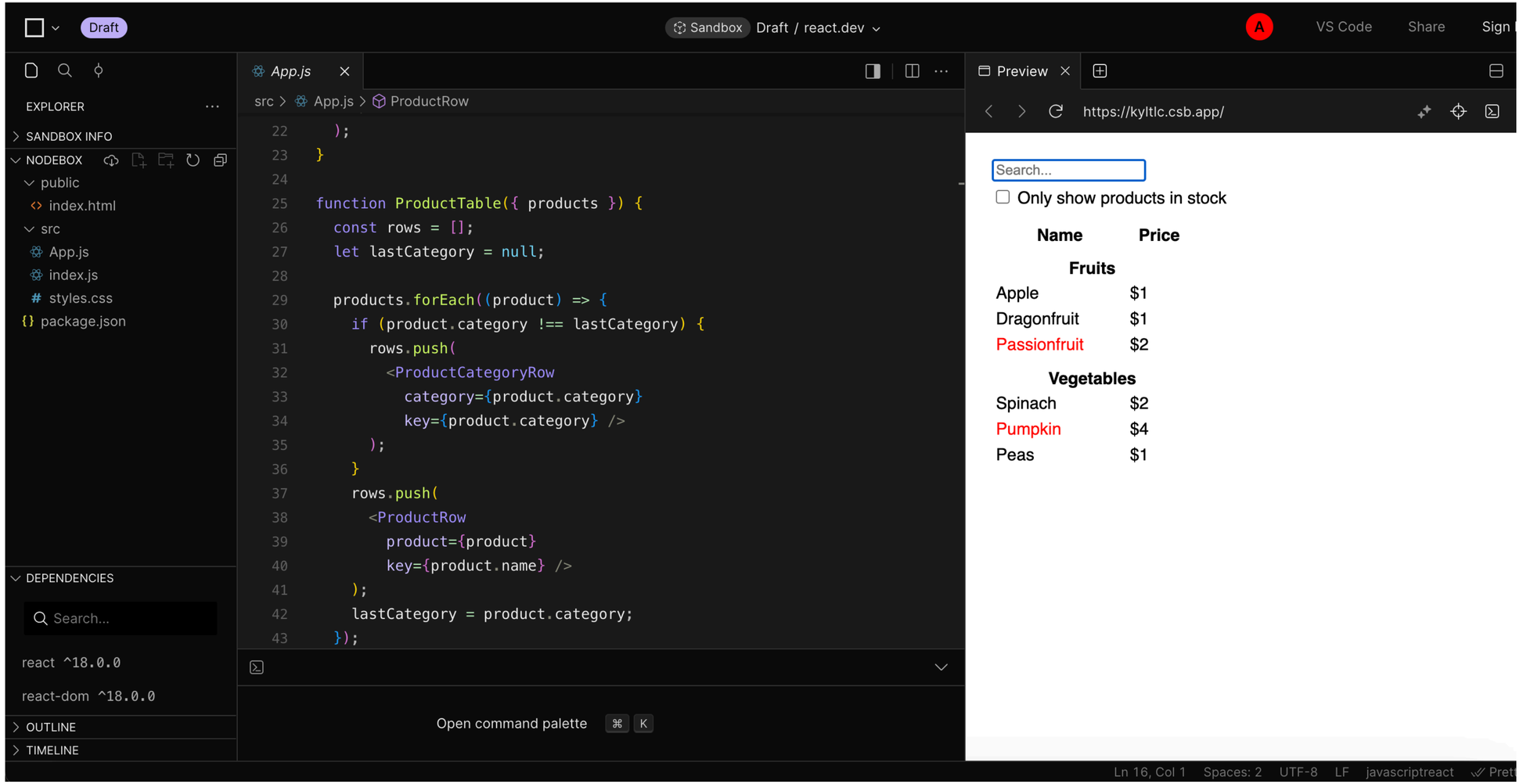
Download Reset Fork

```
1  function ProductCategoryRow({ category }) {
2    return (
3      <tr>
4        <th colSpan="2">
5          {category}
6        </th>
7      </tr>
8    );
9  }
10
11 function ProductRow({ product }) {
12   const name = product.stocked ? product.name :
13     <span style={{ color: 'red' }}>
14       {product.name}
15     </span>;
16
```

Show more

https://react.dev/learn/thinking-in-react

Step 2: Build a static version of React



THANK YOU

- React Introduction
- Fundamentals of React
- Create React Project
- React props/state
- Thinking React