npx create-react-app “tên folder” 🡪 tạo và cài đặt folder project react

cd “tên folder” 🡪 di chuyển vào folder project react

ls -a 🡪 liệt kê các file

npm install 🡪

npm run build 🡪 build project

npm start 🡪 Khởi động máy chủ, mở dự án trong cửa sổ trình duyệt của mình và chuyển tiêu điểm từ thiết bị đầu cuối sang trình duyệt., nếu ko tự mở thì truy cập http://localhost:3000/

1. **Bên trong folder Reacjs project**

* Tên project/
  + REAME.md thư mục chứa mã JavaScript React cho dự án của bạn. Hầu hết công việc bạn làm sẽ nằm trong thư mục đó
  + Node\_modules/ chứa tất cả các thư viện JavaScript bên ngoài được ứng dụng sử dụng.
  + Package.json./

* public/ chứa một số tệp HTML, JSON và hình ảnh cơ sở. Đây là gốc rễ của dự án.
  + index.html #html template
  + favicon.ico
* src/ thư mục chứa mã JavaScript React cho dự án. Hầu hết công việc mình làm sẽ nằm trong thư mục đó
  + App.css
  + App.js
  + App.test.js
  + Index.css
  + Index.js #entry point file chạy đầu tiên
  + logo.svg
* Không thay đổi tên file của public/index.html và src/index.js
* Index.js: File đầu vào của project
* Index.html: File template của project. App sẽ được gắn kết vào node root trong template này
* Package.json: Chứa thông tin các packages mà project sử dụng cũng như một vài config khác
* Node\_modules: folder rất nặng, rất lớn. Nó chứa tất cả các packages mình sử dụng, folder này chỉ có dưới local. Ko push lên Github, ko có trên môi trường production

1. **Push code lên Github repo**

* Tạo project:
* Tạo project:
  + - Bật terminal 🡪 cd tới thư mục cần lưu 🡪 Tạo project theo cú pháp: npx create-react-app “tên folder” 🡪 npm start để khởi động máy chủ
    - Tạo repo trên github 🡪 thực hiện các bước như hướng dẫn trên github

# **JSX**

+ Cú pháp mở rộng cho JavaScript.

+ JSX cho ra những “phần tử”(Element) React.

1. **Nhúng Biểu thức trong JSX**

khai báo một biến tên là name và dùng bên trong JSX bằng cách bao trong cặp dấu ngoặc nhọn:

const name = 'Josh Perez';

const element = <h1>Hello, {name}</h1>;

ReactDOM.render(

element,

document.getElementById('root')

);

Nhúng kết quả của một hàm JavaScript, formatName(user), vào bên trong phần tử <h1>.

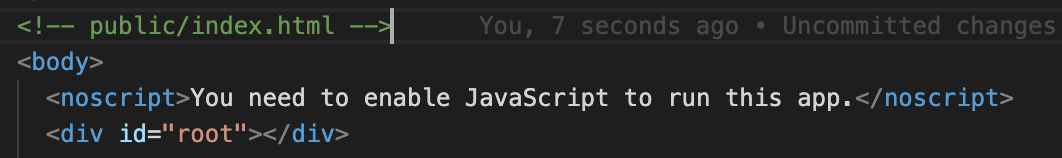


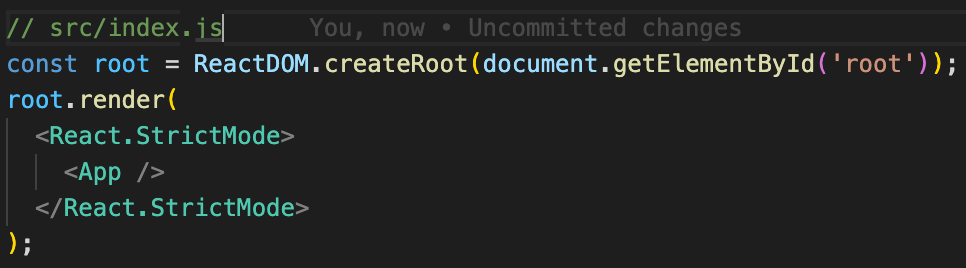
NOTES:

* Trình duyệt không hiểu cú pháp của JSX, nên cần biên dịch JSX sang JS trước khi sử dụng trên trình duyệt.
* Class và for là 2 từ khoá trong js nên ko đc sử dụng trong JSX
* Class 🡪 className
* For 🡪 htmlFor
* Phải import react khi sử dụng JSX

# **ReactDOM.render()**

Render phần app ở chế độ strict Mode (chế độ nghiêm ngặt) vào root 🡪 bỏ vào html của mình có div id= “root”





Theo exp thực tế, hàm ReactDOM.render() chỉ sử dụng đúng 1 lần ở đây

# **Component là gì?**

Một giao diện lớn, phức tạp được chia thành nhiều thành phần UI nhỏ hơn (component) để dễ quản lý và tái sử dụng.

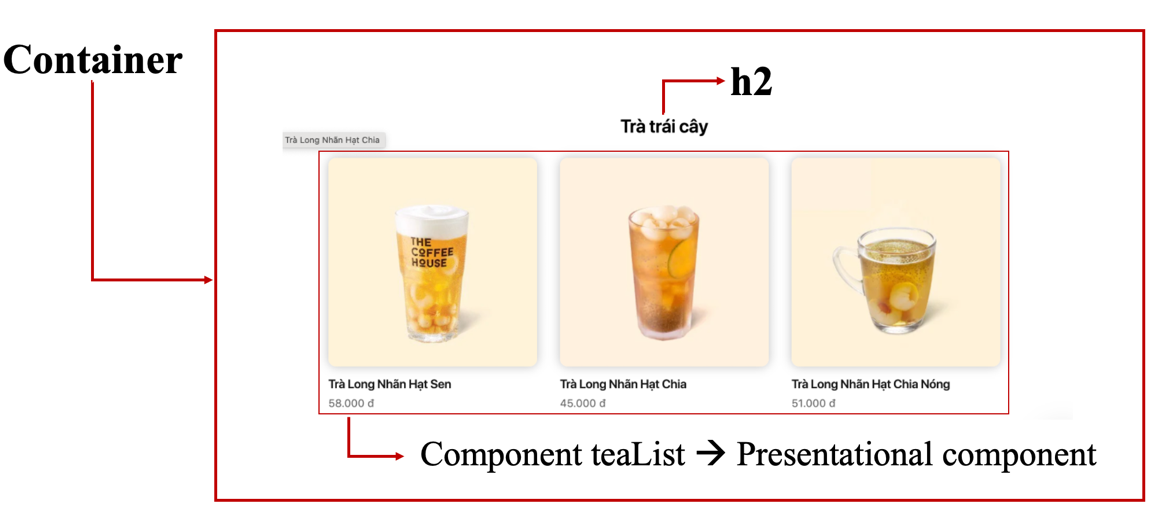
Một website gồm nhiều phần bố cục khác nhau như phần header, phần thân (body), phần footer và bây giờ mình muốn chia các thành phần đó thành 3 component header, body và footer riêng các phần ra để dễ quản lý hoặc tái sử dụng nó



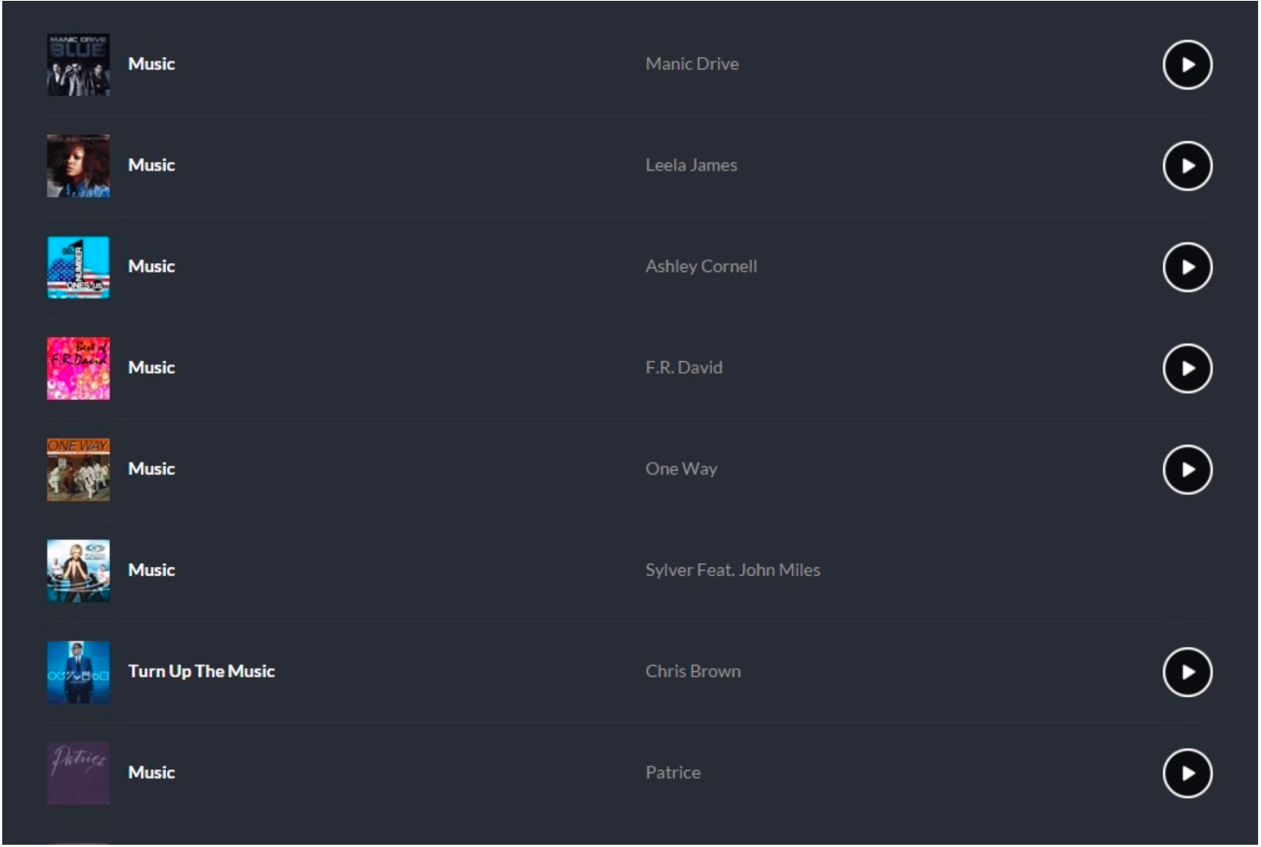
Phân loại component

1. Chia theo cú pháp: Functional và Class component
2. Chia theo chức năng: Container và Presentational component

|  |  |
| --- | --- |
| **Phân biệt container với presentational component** | |
| Container component: quản lý dữ liệu, có dữ liệu nhưng ko bik render như nào | Presentational component: Không biết dữ liệu nhưng biết cách render |
| Quan tâm dữ liệu gì | Quan tâm hiển thị như thế nào |
| Không biết hiển thị gì | Không biết dữ liệu từ đâu ra |
| Truyền dữ liệu xuống presentational component | Nhận dữ liệu từ container để render lên |



Container sẽ lên server lấy dữ liệu, sau đó truyền dữ liệu xuống teaList, teaList sẽ render lên giao diện.

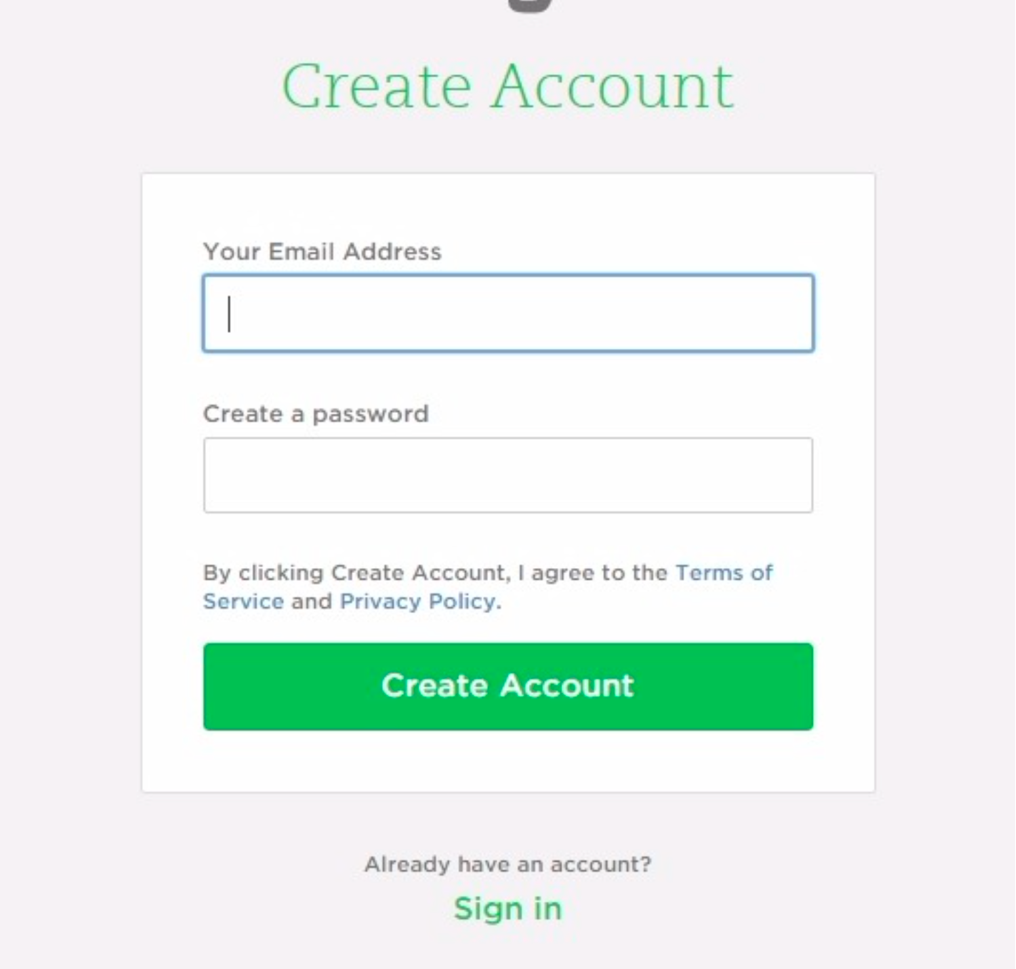


**Container**

**SongList conponent**

**Song conponent**

**Container**

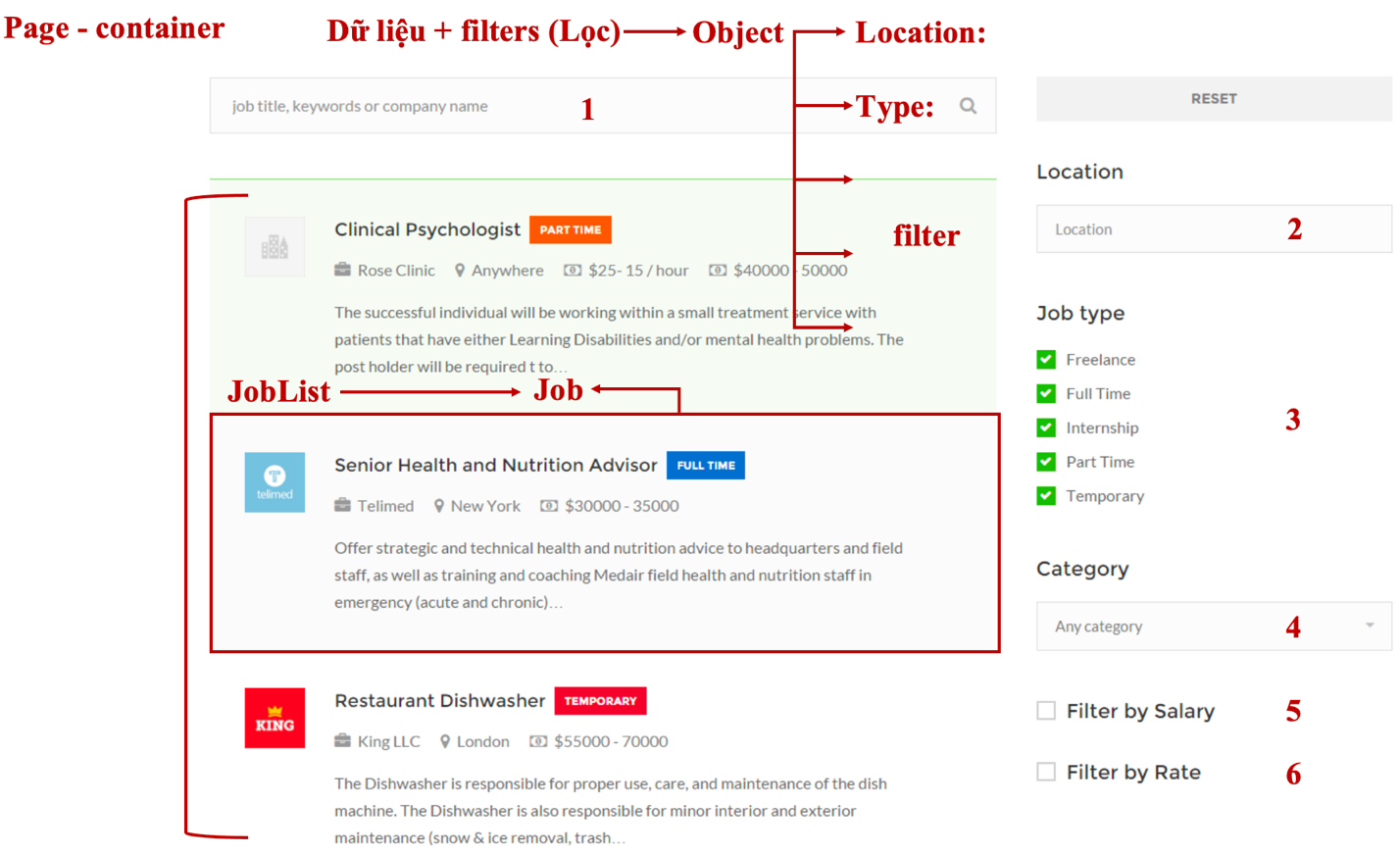


**Submit**

**Sign up Page**

**Sign up Form**

Luôn luôn tạo ra 1 component (vd là component form) có nhiệm vụ là render cái form, quản lý valuedate của form và khi nó submit thì đẩy logic submit lên thằng cha, còn nếu cái form lặp đi lặp lại nhiều nơi trong App mà chỉ làm đúng 1 logic thôi thì handle trong form luôn

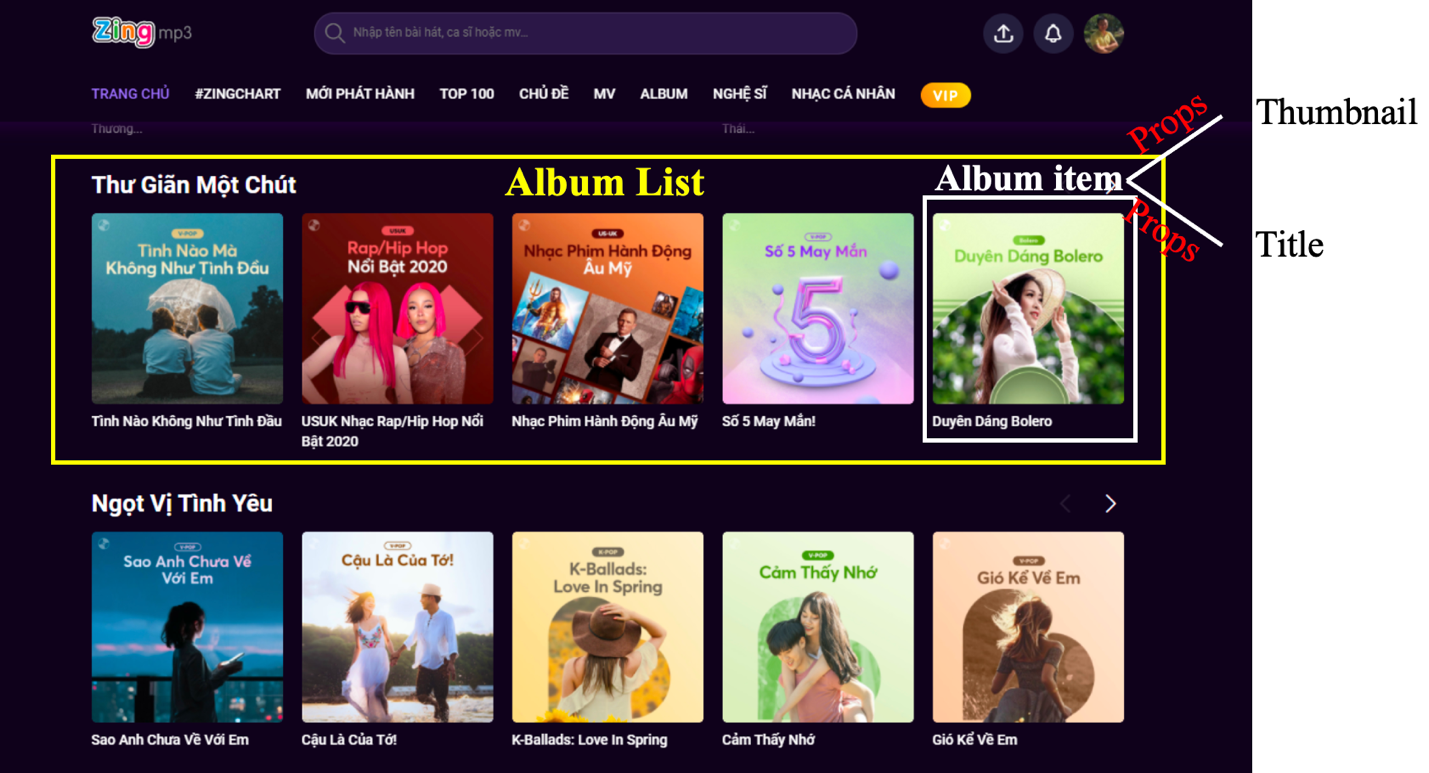
\

Vào API lấy dữ liệu về rồi update vào dữ liệu của container. Dữ liệu này sẽ gọi xuống JobList để JobList render lên List ban đầu. Khi gõ ô search hay đánh dấu tick 🡪 báo lên container sự thay đổi của filter 🡪 gọi lại API để lấy lại dữ liệu 🡪 update dữ liệu mới cho container 🡪 truyền dữ liệu xuống cho JobList để render lại

JobList với filters ko làm việc trực tiếp với nhau mà đều cần phải đi qua Page (container)

# **Props và Compositon**

* Props là dữ liệu được truyền từ component cha xuống component con
* Props là thuộc tính read-only, component con không thay đổi được, muốn đổi thì nhờ component cha.
* Props giúp tạo ra sự đa dạng cho component. Cùng một component với props khác nhau thì render ra khác nhau.



## Khai báo props và kiểm tra loại dữ liệu của props

* Props có thể dùng mà không cần định nghĩa

import React from "react";

function ColorBox(props) {

const { color } = props;

return (

<div className="box" style={

{ backgroundColor: color }}>

</div>

) }

export default ColorBox;

import React from "react";

import ColorBox from "./components/ColorBox";

export default function App() {

return (

<section>

<ColorBox color="red"></ColorBox>

<ColorBox color="green"></ColorBox>

<ColorBox color="blue"></ColorBox>

</section>

); }



## Khai báo props và kiểm tra loại dữ liệu của props

Nên dùng propTypes để kiểm tra đúng loại dữ liệu cho props

import React from "react";

import PropTypes from "prop-types"; // add package prop-types

function ColorBox(props) {

const { color } = props;

return <div className="box" style={{ backgroundColor: color }}></div>;

}

// khai báo tất cả các props được sử dụng trong thành phần này

ColorBox.propTypes = {

color: PropTypes.string.isRequired,

rounded: PropTypes.bool,

};

// đừng quên đặt giá trị mặc định cho các props không bắt buộc

ColorBox.defaultProps = {

rounded: true

};

export default ColorBox;

color: propType.string.isRequired: props bắt buộc phải có hay không, nếu thằng cha lỡ quên truyền props color thì báo lỗi.

Đối với những prop nó ko bắt buộc thì phải truyền cho nó 1 giá trị mặc định