

Iniciar proyecto react

Comandos

- `npx create-react-app nombreApp`
- `npm start`
- `npm run build`
- `npm test`
- `npm run eject`

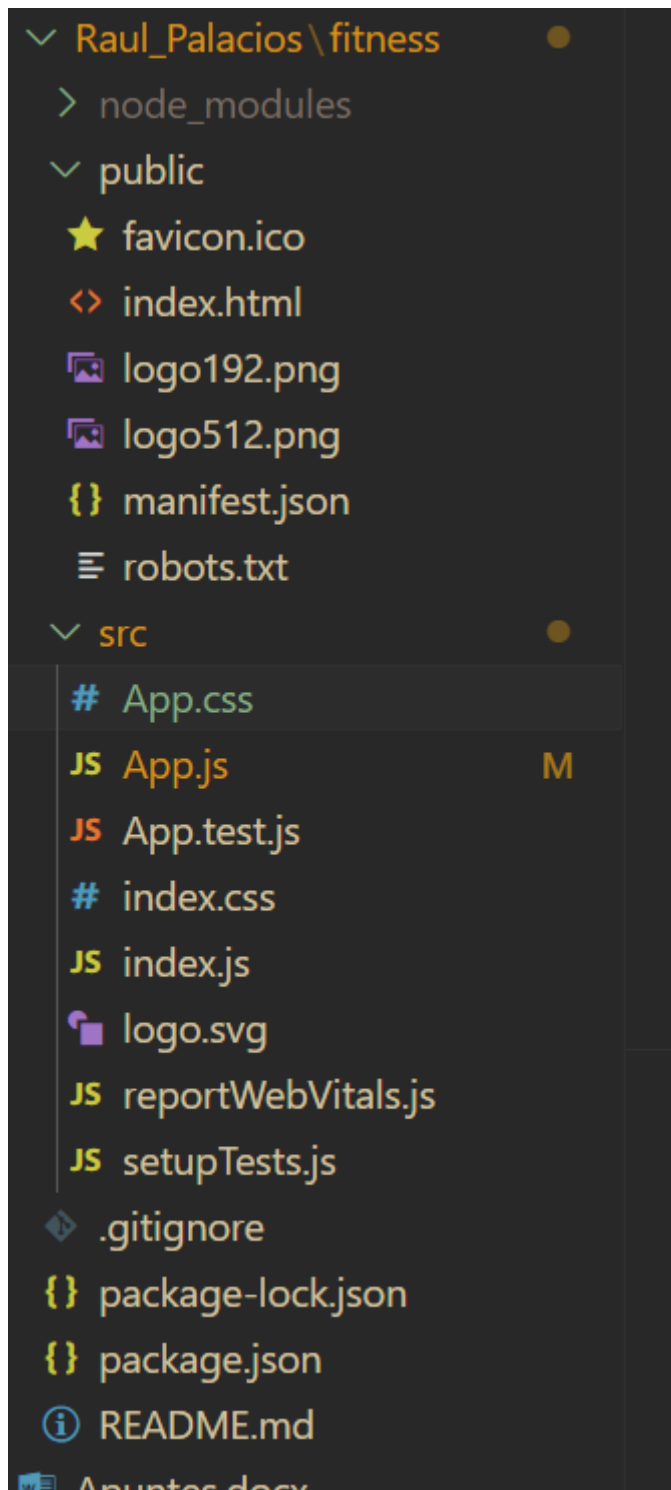
```
npm start
  Starts the development server.

npm run build
  Bundles the app into static files for production.

npm test
  Starts the test runner.

npm run eject
  Removes this tool and copies build dependencies, configuration files
  and scripts into the app directory. If you do this, you can't go back!
```

Eliminando archivos



Se fue:

- App.css
- App.js
- App.test.js
- Index.css
- Logo.svg
- Todo menos el index.js

Primer ejercicio

```
JS index.js M X JS Card.js U
Raul_Palacios > fitness > src > JS index.js > getGreeting
1 //SIN REACT -----
2
3 // const element = document.createElement("h1");
4
5 // element.innerText = "Hello React";
6
7 // const container = document.getElementById("root");
8 // container.appendChild(element);
9
10 //CON REACT -----
11 import React from "react";
12 import ReactDOM from "react-dom";
13
14 const user = {
15   firstName: "Audrie",
16   lastName: "Ochoa",
17   avatar: "https://getwallpapers.com/wallpaper/full/0/7/a/344517.jpg",
18 };
19
20 function getName(user) {
21   return `${user.firstName} ${user.lastName}`;
22 }
23
24 function getGreeting(user) {
25   if (user) {
26     return <h1>Hola! {getName(user)}</h1>;
27   }
28   return <h1>Hola! Extraño</h1>;
29 }
30 const name = "Audrie";
31 const element = (
32   <div>
33     <h1>{getGreeting(user)}</h1>
34     <img src={user.avatar} />
35   </div>
36 );
37 const container = document.getElementById("root");
38 ReactDOM.render(element, container);
```

Ciclo de vida

Montado

constructor(props)

- **Inicializar el estado del componente**
- **Enlazar eventos**
- **Setear variables globales**

componentWillMount()

- **Modificar el estado**
- **No realizar llamados a API's**
- **No realizar suscripción a eventos**

render()

- **Debe ser una funcion pura**
- **No debe modificar el estado**

componentDidMount()

- **El mejor momento para llamar API's**
- **Realizar suscripciones a eventos**
- **Modificar estado**

Actualización

Puede ocurrir múltiples veces o ninguna.

componentWillReceiveProps(nextProps)

- **Realizar cambios en los estados basado en las nuevas propiedades**

**shouldComponentUpdate(nextProps,
nextState)**

- **Validar si queremos renderizar o no el componente**

**componentWillUpdate(nextProps,
nextState)**

- **Realizar cualquier tipo de preparación
Antes de que se actualice la UI**

render()

- **Debe ser una funcion pura**
- **No debe modificar el estado**

ReactJS, Ciclo de vida

componentDidUpdate(prevProps, prevState)

Desmontado

componentWillUnmount()

- **Dejar de escuchar eventos**
- **Desuscribirse de un WebSocket**
- **Cancelar peticiones HTTP**

componentDidCatch()

- **Control de errores**

ReactRouter

Instalación

- Npm install react-router-dom --save

State Up

Usualmente, muchos componentes necesitan reflejar el mismo cambio en los datos. Recomendamos elevar el estado compartido al ancestro común más cercano.