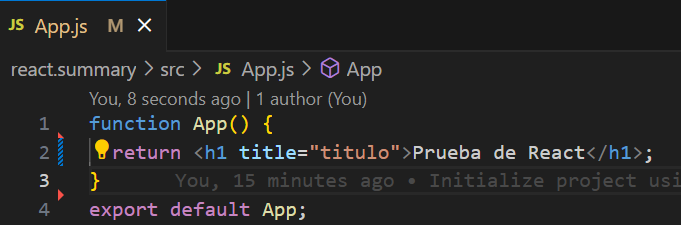
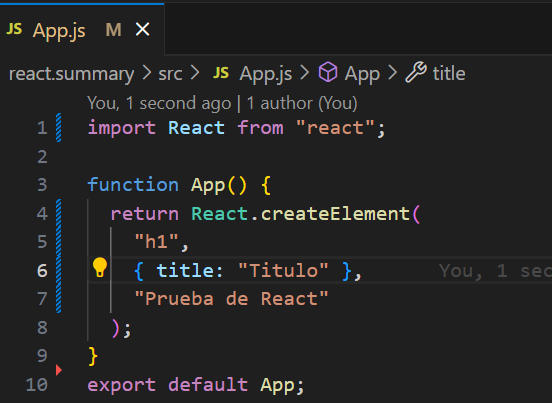
**REACT – SUMMARY**

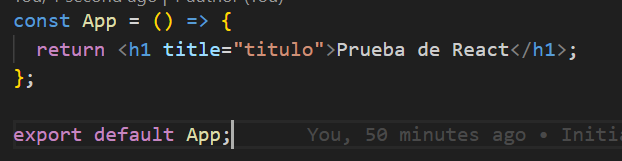
**Diversas formas de trabajar con React:**

* JSX

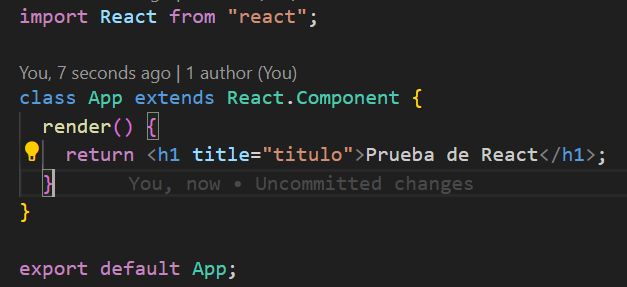


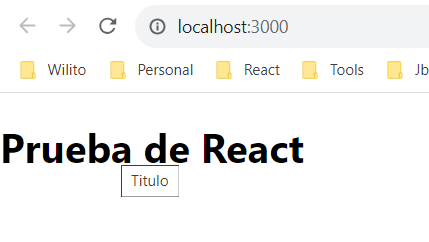
* Componentes funcionales



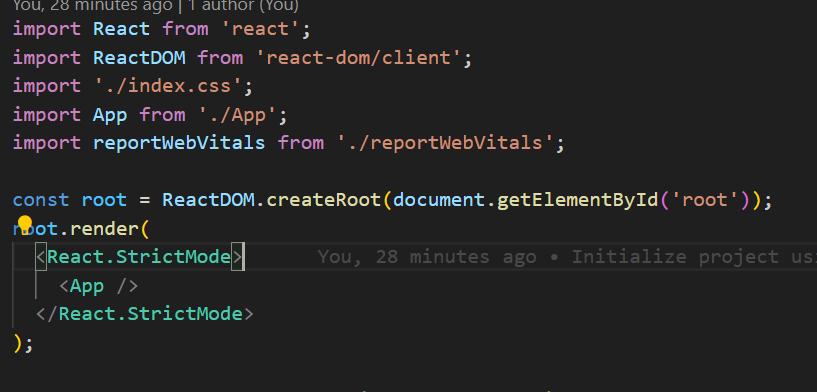


* Clases





* El uso de componentes funcionales debe utilizar mayúsculas para diferenciar un JSX y una aplicación de HTML integrada.



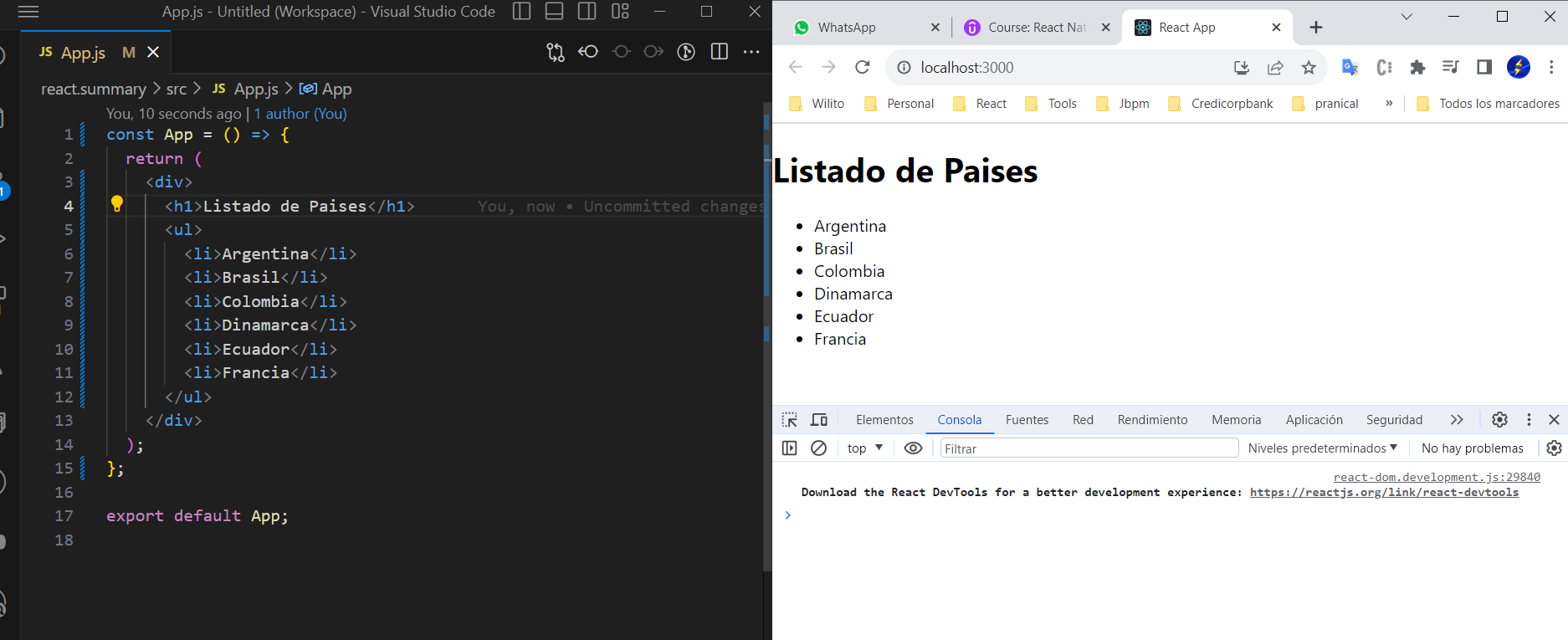
Utilizar <App/> y no <app>

**Primer APP de React**

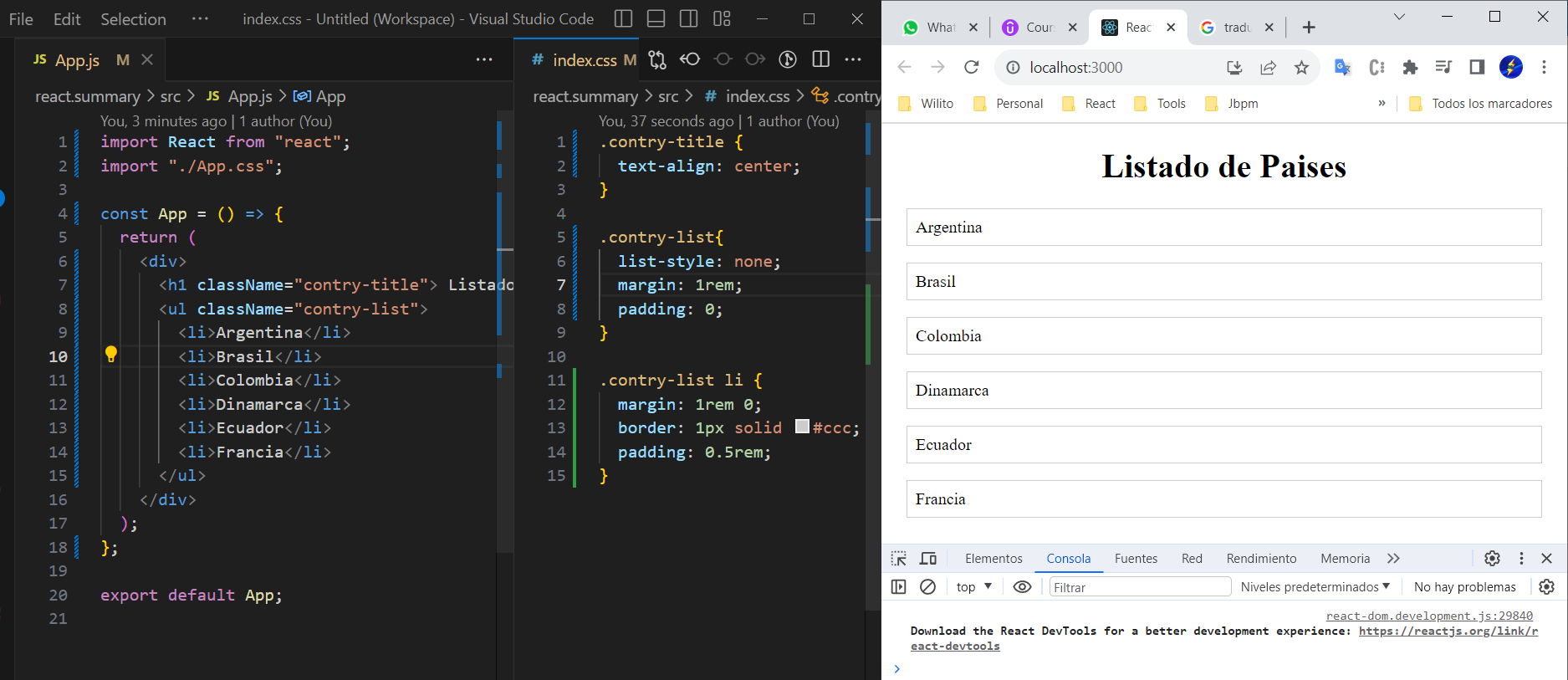
* Crear un nuevo proyecto de nombre react.summary

$> npx create-react-app react.summary

* Crear un componente funcional que muestre una lista de países



* Agregar Estilos



* Generar varios componentes

App.js

import React from "react";

import Countries from "./components/Countries";

import "./App.css";

const App = () => {

  return (

    <div>

      <h1 className="contry-title"> Listado de Paises</h1>

      <Countries />

    </div>

  );

};

export default App;

App.css

.contry-title {

  text-align: center;

}

Country.js

import React from "react";

import "./Countries.css";

const Countries = () => {

  return (

    <ul className="contry-list">

      <li>Argentina</li>

      <li>Brasil</li>

      <li>Colombia</li>

      <li>Dinamarca</li>

      <li>Ecuador</li>

      <li>Francia</li>

    </ul>

  );

};

export default Countries;

Country.css

.contry-list{

    list-style: none;

    margin: 1rem;

    padding: 0;

  }

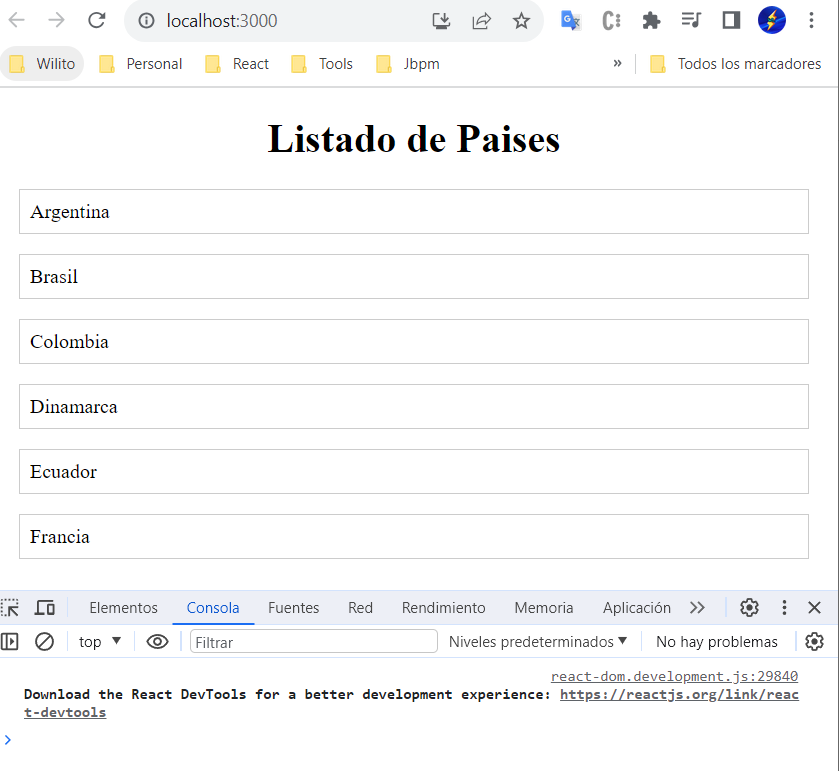
    .contry-list li {

    margin: 1rem 0;

    border: 1px solid #ccc;

    padding: 0.5rem;

  }



**Uso de Props**

* Creo una data Dummy de países

export const countries = [

  { id: "id1", text: "Argentina" },

  { id: "id2", text: "Brasil" },

  { id: "id3", text: "Colombia" },

  { id: "id4", text: "Dinamarca" },

  { id: "id5", text: "Ecuador" },

  { id: "id6", text: "Francia" },

];

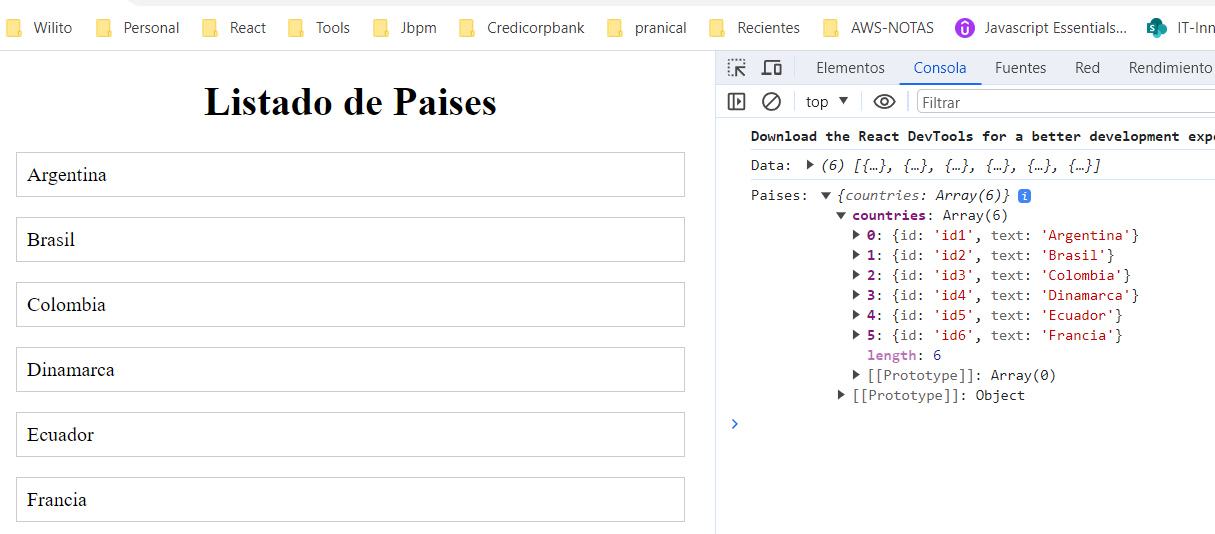
* Se consulta la data desde mi componte principal y se envía a imprimir

import { countries } from "./data/data";

const App = () => {

  console.log("Data:", countries);

  console.log("Paises:", { countries });



  console.log("Data:", countries);

Se muestra los valores del arreglo



  console.log("Paises:", { countries });

Se muestra una expresión JS que contiene un objeto o variable con los valores del arreglo



Este objeto o expresión se envía al objeto que recibe el parámetro (props).



Es diferente enviar [data = “countries”] a [data = {countries}].

Se envía el arreglo de datos y no un texto

* Luego se puede enviar el prop y la clase que lo recibe

import React from "react";

import "./Countries.css";

const Countries = (props) => {

  console.log("PROP", props)

  return (

    <ul className="contry-list">

      {props.data.map((country) => <li key={country.id}>{country.text}</li>)}

    </ul>

  );

};

export default Countries;



* Se puede mejorar el código.

import React from "react";

import "./Countries.css";

const Countries = ({ data }) => {

  console.log("PROP", data);

  return (

    <ul className="contry-list">

      {data.map((country) => (

        <li key={country.id}>{country.text}</li>

      ))}

    </ul>

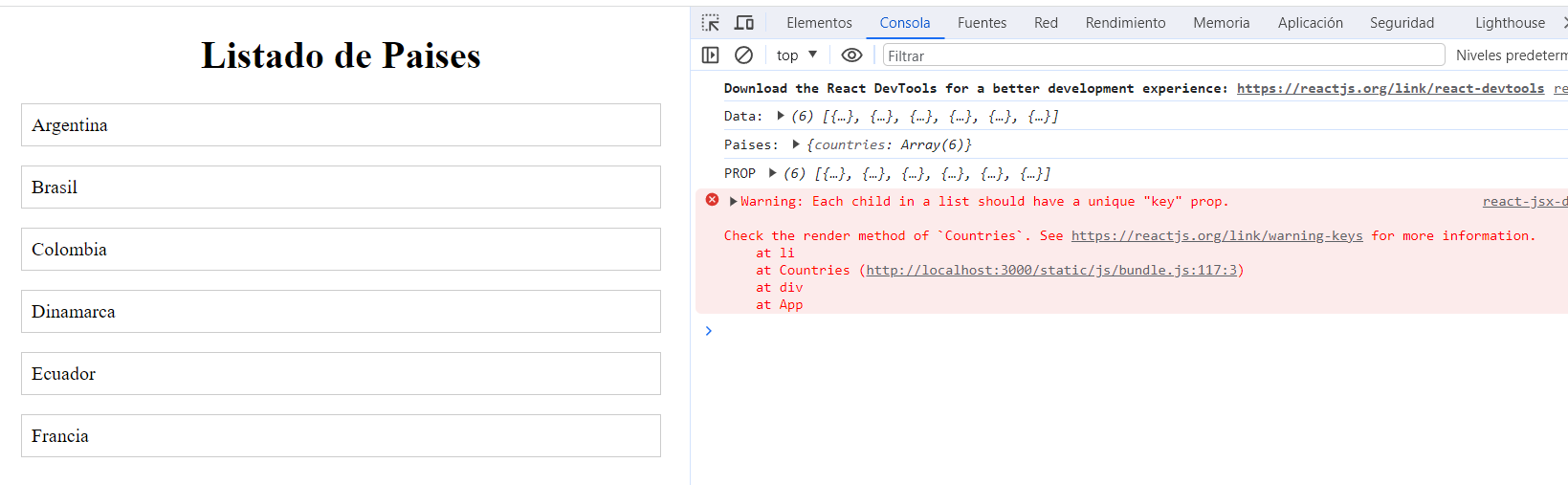
  );

};

export default Countries;



* Si no le paso el key al elemento que se genera en el map se da un warning
* <li>{country.text}</li>



Warning: Each child in a list should have a unique "key" prop.

* El *destructuring*de componentes en React es una técnica utilizada en esta librería JavaScript para, como su nombre indica, descomponer estos objetos con el fin de darles mucha más legibilidad. Esta técnica se relaciona especialmente con los componentes, porque son funciones que manejan el objeto *props.*

Componente llama otro componente y envía parámetros (props)

…

<Countries data={} , onPress={()=>{}}, onClose={closehandler}, styles={fileStyles}/>

…

Componente llamado recibe props

…

const Countries = (props) => {

const values = props.data

const onPress = props.onPress

…

Componente llamado recibe props y lo descompone en objetos

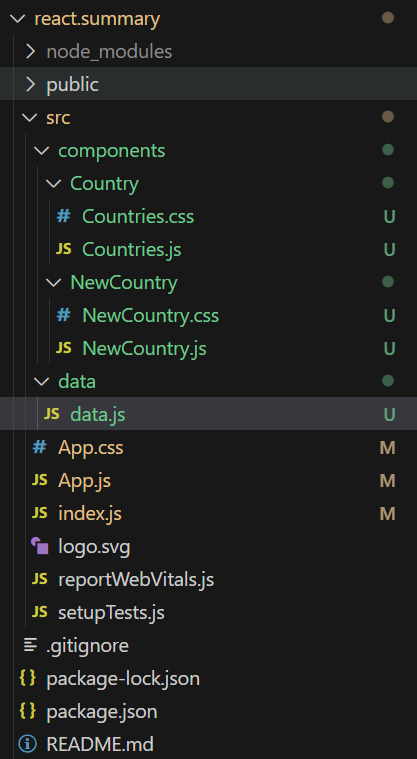
…

const Countries = ({ data, onPress, onClose, styles }) => {

…

**Crear componente para agregar nuevo País. Uso de props y state**

* Se creo carpetas de componentes para ajustar el código y se ajusto la data de entrada



Ajustar App.js

import React, { useState } from "react";

import Countries from "./components/Country/Countries";

import "./App.css";

import { dummy\_countries } from "./data/data";

import NewCountry from "./components/NewCountry/NewCountry";

const App = () => {

  const [countries, setCountries] = useState(dummy\_countries);

  console.log("Data:", countries);

  console.log("Paises:", { countries });

  const addNewCountryHandler = (newCountry) => {

    console.log(111, newCountry)

    setCountries((listCountries) => [...listCountries, newCountry]);

  };

  return (

    <div>

      <h1 className="contry-title"> Listado de Paises</h1>

      <NewCountry onAddCountry={addNewCountryHandler} />

      <Countries data={countries} />

    </div>

  );

};

export default App;

Crear NewCountry.js

import React, { useState } from "react";

import "./NewCountry.css";

const NewCountry = ({ onAddCountry }) => {

  const [name, setName] = useState('');

  const addCountryHandler = (event) => {

    event.preventDefault()

    const newCountry = {

      id: new Date().getTime(),

      text: name

    }

    onAddCountry(newCountry)

    setName('')

  };

  const valueCountryHanlder = ({ target }) => {

    setName(target.value);

  };

  return (

    <form className="new-country" onSubmit={addCountryHandler}>

      <input type="text" value={name} onChange={valueCountryHanlder} />

      <button type="submit">Agregar País</button>

    </form>

  );

};

export default NewCountry;

* Agregar un nuevo País



Al agregarse el nuevo país se limpia el campo input y se refresca la vista



* El uso de userState es importante en el componente newCountry sino al agregar uno nuevo se agrega, pero no se limpia porque no es un state y no refresca la vista en el input

import React from "react";

import "./NewCountry.css";

const NewCountry = ({ onAddCountry }) => {

  let name = ""

  const addCountryHandler = (event) => {

    event.preventDefault()

    const newCountry = {

      id: new Date().getTime(),

      text: name

    }

    onAddCountry(newCountry)

    name =""

  };

  const valueCountryHanlder = ({ target }) => {

    name = target.value

  };

  return (

    <form className="new-country" onSubmit={addCountryHandler}>

      <input type="text" onChange={valueCountryHanlder} />

      <button type="submit">Agregar País</button>

    </form>

  );

};

export default NewCountry;



<https://academind.com/>

<https://www.youtube.com/c/academind>