

# Ayudantía 2:

# React

Natalia Correal Posada  
Alexandra San Martín

# Contenidos

**01 React**

**03 Componentes**

**02 JSX**

**04 Hooks**

**05 Actividad**

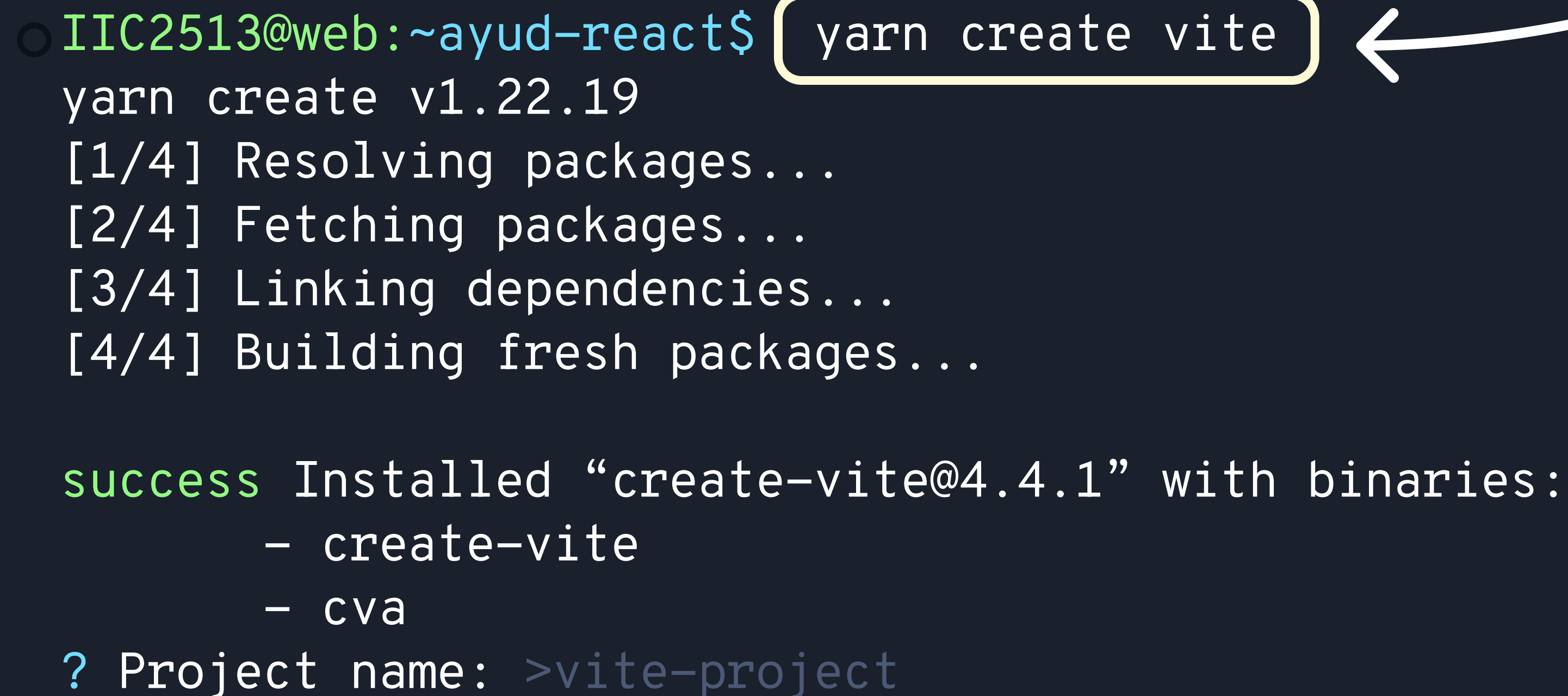
# ¿Qué es React? 🤔

Es una librería de JavaScript para crear UI y facilitar el desarrollo de app dinámicas (no es un framework).

# Crear proyecto de React

Importante que tienes que haber realizado el Setup de React

Escribir en  
terminal



```
IIC2513@web:~ayud-react$ yarn create vite
yarn create v1.22.19
[1/4] Resolving packages...
[2/4] Fetching packages...
[3/4] Linking dependencies...
[4/4] Building fresh packages...

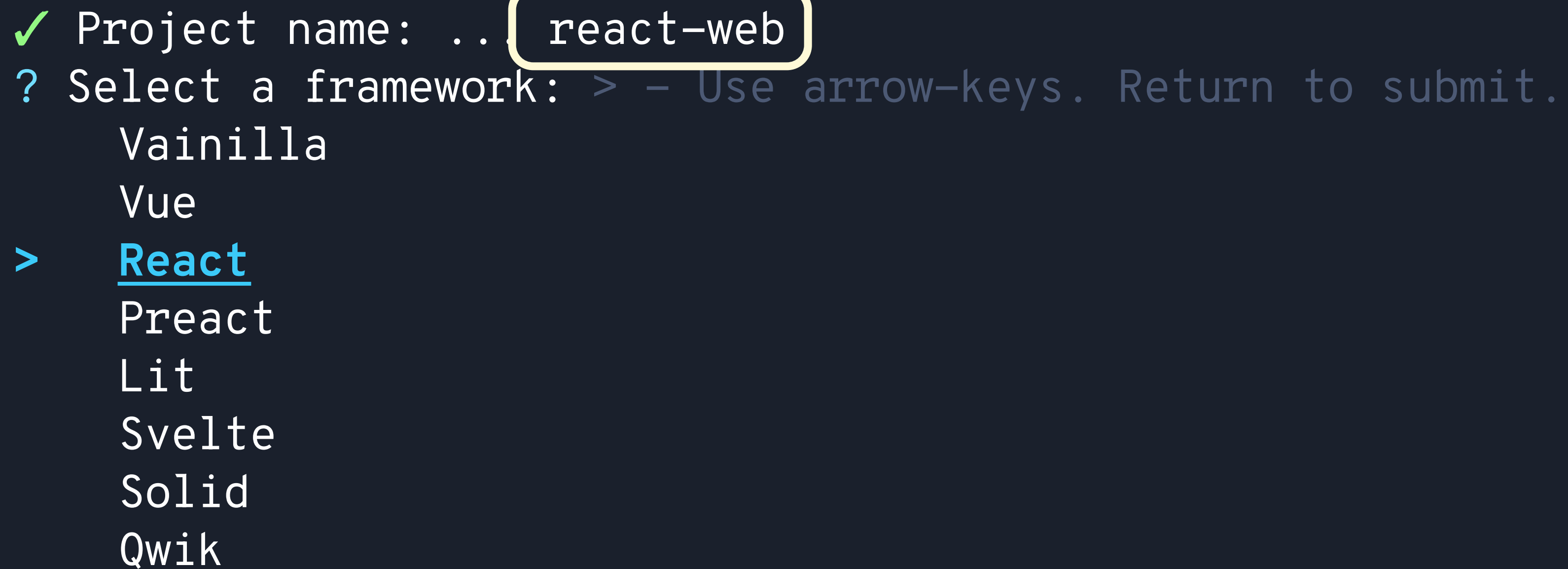
success Installed "create-vite@4.4.1" with binaries:
  - create-vite
  - cva
? Project name: >vite-project
```

A terminal window with a dark background and light-colored text. The prompt is `IIC2513@web:~ayud-react$`. The command `yarn create vite` is entered and highlighted with a yellow box. A white arrow points from the text "Escribir en terminal" to this box. The output shows the progress of installing create-vite, including resolving, fetching, and linking packages. It concludes with "success Installed 'create-vite@4.4.1' with binaries: - create-vite - cva" and prompts for a project name, which is entered as `>vite-project`.



# Crear proyecto de React

Escribir  
nombre del  
proyecto



A terminal window with a dark background and three colored window control buttons (red, yellow, green) in the top-left corner. The text inside the terminal is as follows:

```
✓ Project name: ... react-web
? Select a framework: > - Use arrow-keys. Return to submit.
  Vainilla
  Vue
> React
  Preact
  Lit
  Svelte
  Solid
  Qwik
```

A white arrow points from the text "Escribir nombre del proyecto" to the input field containing "react-web". Another white arrow points from the text "Seleccionar React" to the "React" option in the framework list.


Seleccionar  
React

# Crear proyecto de React



```
✓ Project name: ... react-web
✓ Select a framework: > React
? Select a variant: > - Use arrow-keys. Return to submit.
  TypeScript
  TypeScript + SWC
  JavaScript
> JavaScript + SWC
```

Seleccionar  
JS + SWC



# Crear proyecto de React

```
IIC2513@web:~ayud-react$ cd react-web/  
IIC2513@web:~ayud-react/react-web$ yarn  
IIC2513@web:~ayud-react/react-web$ yarn dev
```

→ Local:

<http://localhost:XXXX/>

Para ver el  
proyecto  
de manera  
local

# ¿Qué es JSX?

Es una extensión de la sintaxis de JavaScript.  
Produce “elementos” de React que luego son  
renderizados.

✨HTML + JS✨



# ¿Cómo se ve?



```
const nombre = "Marco"
```

```
//Dentro de los corchetes se pueden agregar expresiones y  
variables de JS
```

```
// Estoy utilizando HTML
```

```
//Este código puede ser renderizado con React
```

```
const saludo = <p>Hola {nombre}!</p>
```

# Componentes en React

Son elementos UI reusables y combinables. Se ordenan y **anidan** para diseñar una página.

Cada componente es como una función en JS, pueden recibir parámetros y retornan elementos de React

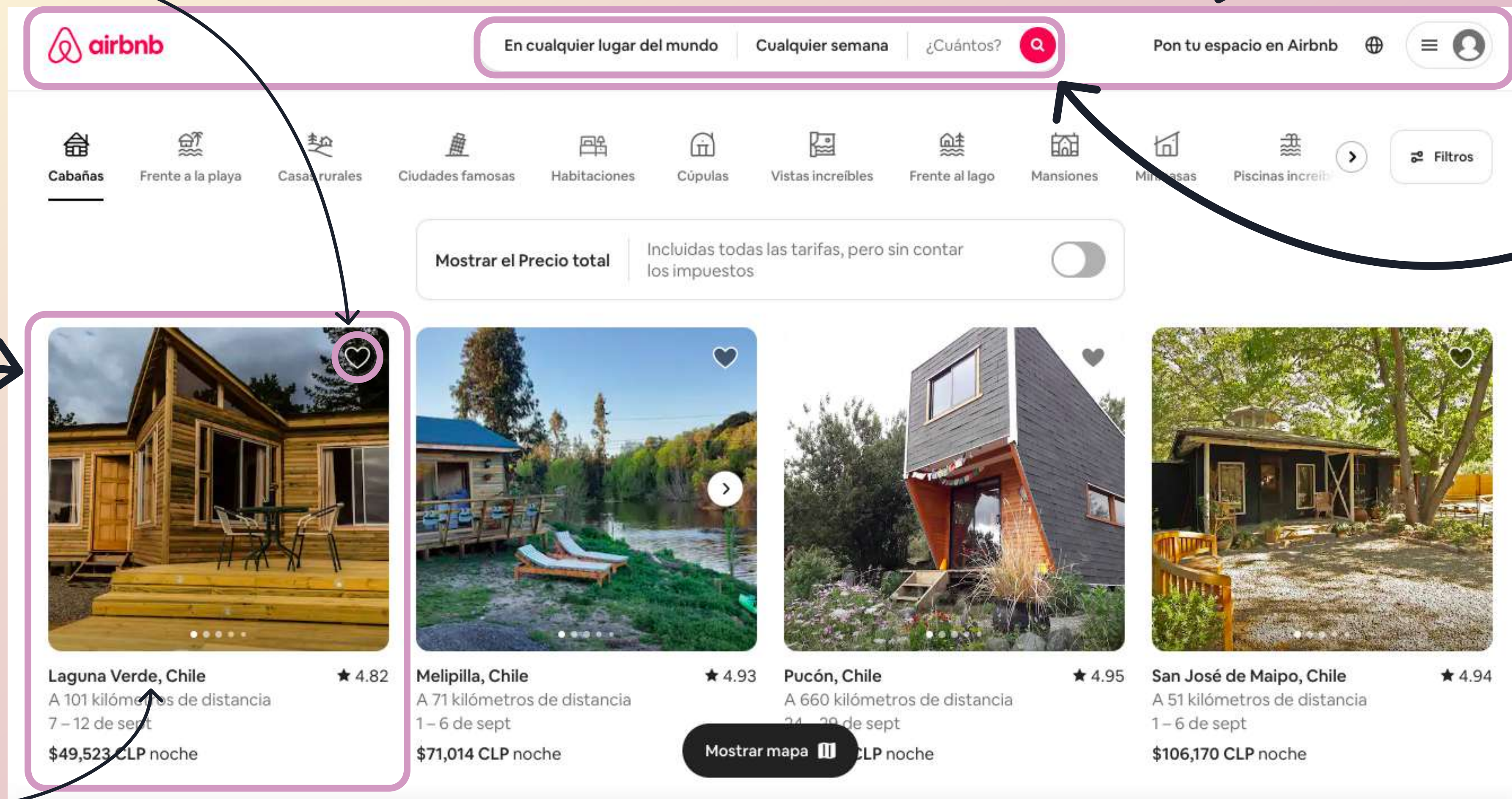


# ¿Cómo se ven?

Componente de *navbar*

Componente de *wishlist*

Componente de buscador (que está dentro de la *navbar*)



Componente de tarjeta

Prop de ubicación



# ¿Y en el editor?

```
src
--- components
    --- Button.css
    --- Button.jsx
    --- Navbar.css
    --- Navbar.jsx
    --- ...
--- App.css
--- App.jsx
--- index.css
--- main.jsx
--- Routing.jsx
--- ...
```

```
App.jsx
import Navbar from './components/Navbar';

function App({nombre_1, nombre_2}) {

    const nombres = nombre_1 + nombre_2

    return (
        <div className="App">
            <Navbar />
            <p>Esta página tendrá una navbar</p>
            <p>Puedo usar variables y texto de JS
            {nombres} dentro de corchetes</p>
        </div>
    );
};
```



# React en *DevTools*



Extensión de Chrome que permite ver y debuggear páginas que han sido realizadas con React.

Se visualiza como se anidan los elementos, los valores de *props*, hooks y otros.

# Hooks

Funciones que permiten usar estados en componentes de React sin tener que definir clases

# useState

Variable que  
contiene el  
valor del  
estado

```
import React, {useState} from 'react';
```

```
function Example() {  
  const [count, setCount] = useState(0);
```

Función que al  
llamarla cambia el  
valor del estado

```
  return (  
    <div>
```

```
      <p>You clicked {count} times</p>
```

```
      <button onClick={() => setCount(count + 1)}>
```

```
        Click me!
```

```
      </button>
```

```
    </div>
```

```
  );
```

```
};
```

Cuando se clickea el  
botón, se usa  
setCount para cambiar  
el valor de count

Importamos  
el Hook

# useEffect

```
import { useState, useEffect } from 'react';
```

```
function Timer() {  
  const [count, setCount] = useState(0);
```

```
    useEffect((useEffect) => {  
      setTimeout(() => {  
        setCount((count) => count + 1);  
      }, 1000);  
    }, [ ]);
```

```
    return <h1>I've rendered {count} times!</h1>  
  };
```

useEffect

Dependencia  
de la variable



# Otros Hooks

Existen muchos Hooks. Para generar IDs únicos, obtener contexto, guardar información en caché y muchos otros...

Los puedes encontrar en

**react.dev**

✨ **Actividad práctica** ✨

# Actividad práctica

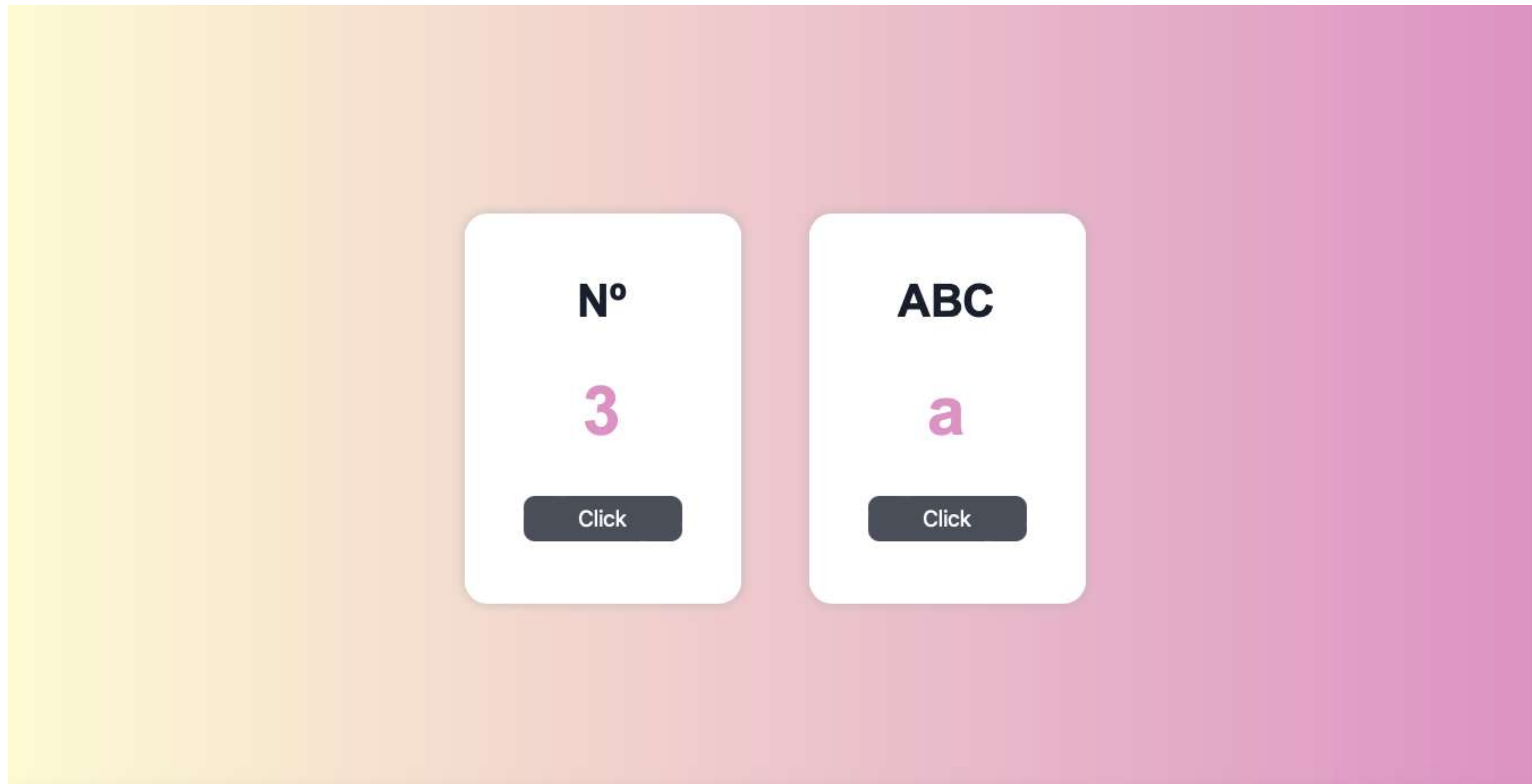
Deberán crear dos tarjetas (iguales en forma, con distinto contenido), cada una de estas llevará un botón que cambiará lo que esta muestra.

- La primera tarjeta comenzará con una “a” y cada vez que presiones el botón de respectiva tarjeta, se mostrará la siguiente letra del abecedario.
- La segunda tarjeta será un contador y cada vez que aprietes el botón, deberás sumarle 1 al valor mostrado.

P.D1: Recuerda que con React puedes recibir componentes, funciones como *props* al crear componentes y así puedes buscar formas de optimizar tu código :)

P.D2: ✨Usar useState ✨

# ¿Cómo se debería ver?





# ¿Cómo se debería ver?

Que las  
tarjetas sean  
un mismo  
componente  
(con distintas  
variables)

