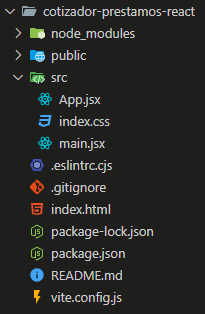
React

React js es una libreria de javascript para creacion de interfaces de usuario

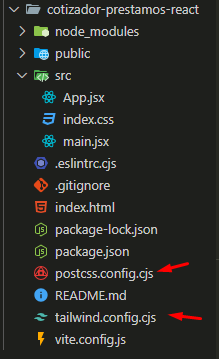
Existen diferentes formas de instalar react, pero actualmente se utiliza vite

Para instalar react con vite puedes hacerlo con npm o yarn

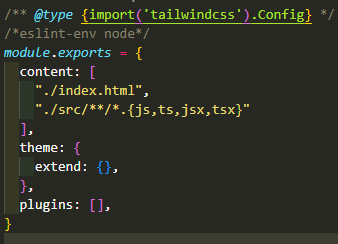
1. Instalar vite npm create vite@latest
2. Nos movemos al Proyecto he instalamos las dependencias npm install
3. Luego hacemos npm run dev y el sistema estará corriendo, nos da una url para acceder al servidor que nos crea
4. Instalar npm i -D autoprefixer@10.4.12 postcss@8.4.18 [tailwindcss@3.1.8](mailto:tailwindcss@3.1.8) para trabajar con tailwin
5. Eliminamos assets css para dejarlo de la siguiente forma



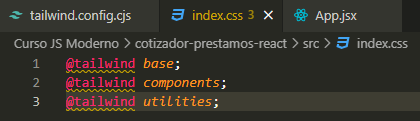
1. Ejecutamos npx tailwindcss init -p para generar la configuración de tailwin



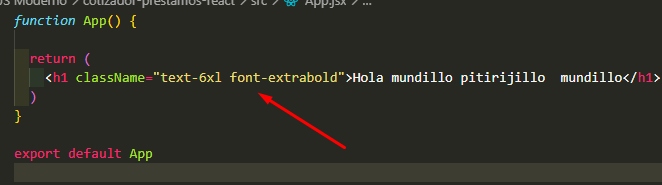
1. Configurando el archivo tailwind, la parte de /\*eslint-env node\*/ no es necesaria se corrige mas delante



1. Configurando el index.css



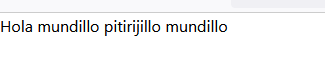
Comprobamos que funciona



Con Tailwind



Sin tailwind

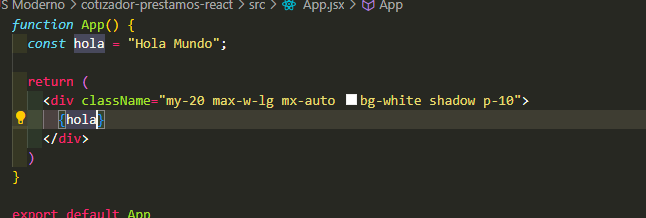


Nota: todas las clase de HTML tiene que tener className

JSX es una sintaxis que nos permite combinar JS y HTML

Los elementos que están entre {} los trata como código de javascript

La lógica de JS no puede ir dentro del return html, esta debe de ir fuera del return

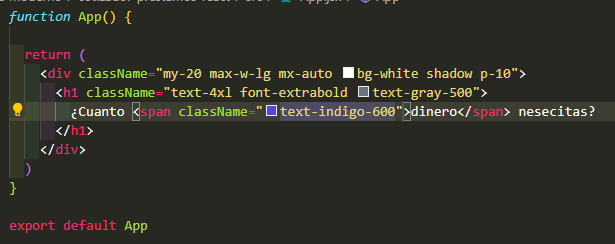


Componentes:

te permiten dividir tu cogido en partes reutilizables.

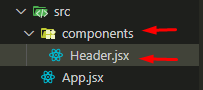
Los componentes utilizan la extension .js o jsx y se importan con un import de JavaScript

Se puede pasar informacion de un componente a otro por Props

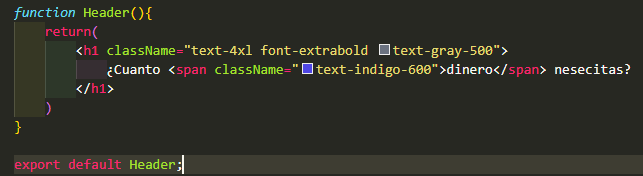


Cuando se crean componentes con vite y react hay dos reglas. 1) tiene que empezar el nombre del archivo en mayusculas y su extension es jsx

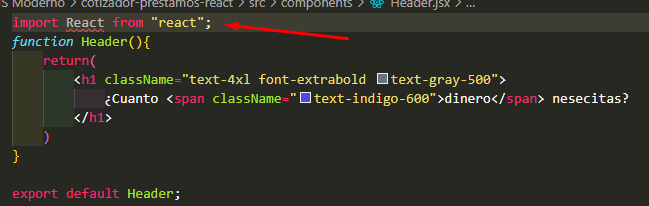
Pasando a crear un componente



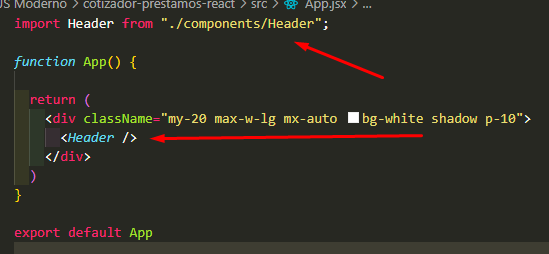
El código del componente seria



El import React en las nuevas versiones ya no es necesario



Ahora hay que importarlo y utilizarlo



//State en React

El state es la fuente de la verdad (source of truth) de tu app

Un listado de clientes, un carrito de compras lleno o vacio, un usuario autenticado es lo que determina cual es el estado o state de tu app, básicamente son variables que guardan los cambios de la aplicación.

Para definir el state en react se importa useState y la funcion al momento de utilizarse nos retorna 2 valores: state y la funcion que modifica el state

Nota: el state estas funciones se les conoce como Hooks ya que hay diferentes tipos de states, los states deben de ir al principio de la función y no pueden ir dentro de una condición.

Administrador de State

useState es suficiente para crear aplicaciones en React

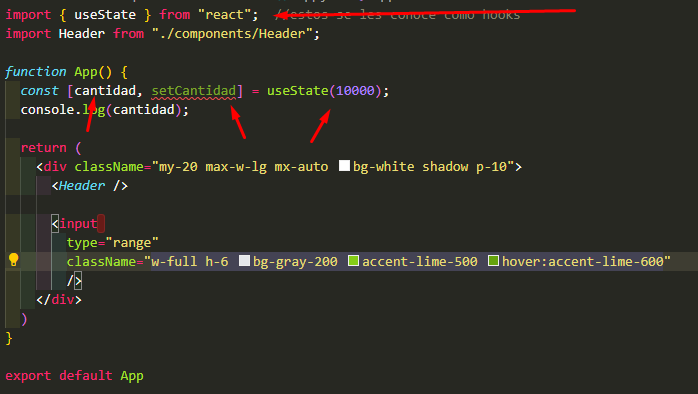
El state no debe modificarse directamente; para eso se debe utilizar la funcion que lo modifica.

En un equipo de trabajo o un proyecto grande se recomienda administrar el state con un state global como Context, Zustand, Recoil o Redux.

En nuestro ejemplo tenemos una barra de tipo range, para obtener cual es el valor de la barra se le podría poner una clase o un id y escucha su evento de change para determinar el rango o valor, pero en react las cosas son diferentes para ello utilizamos el state

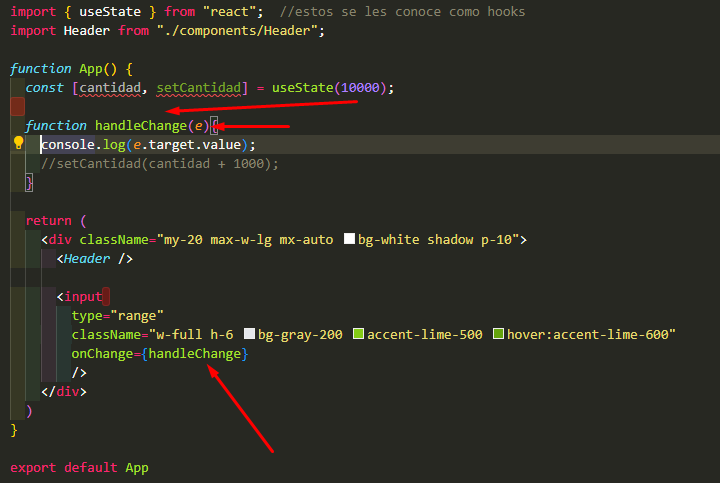


Como llamar a un useState



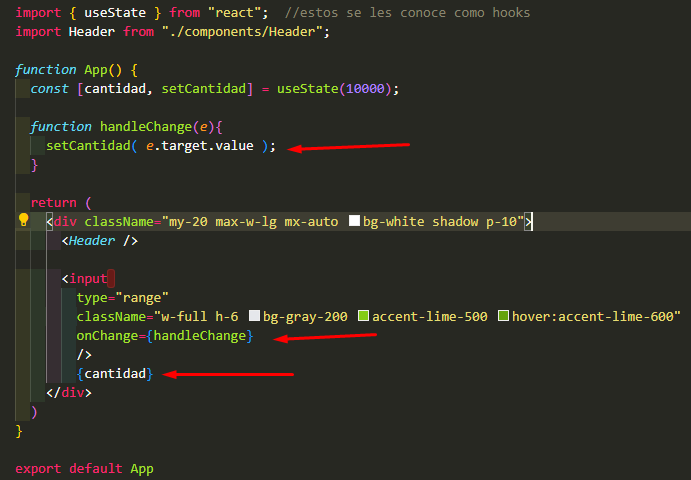
Modificar el valor del state mediante eventos

Como es un evento podemos recuperar e y recuperar el valor



Se nombre como handle mas el nombre del evento en este caso Change por lo que seria handleChange

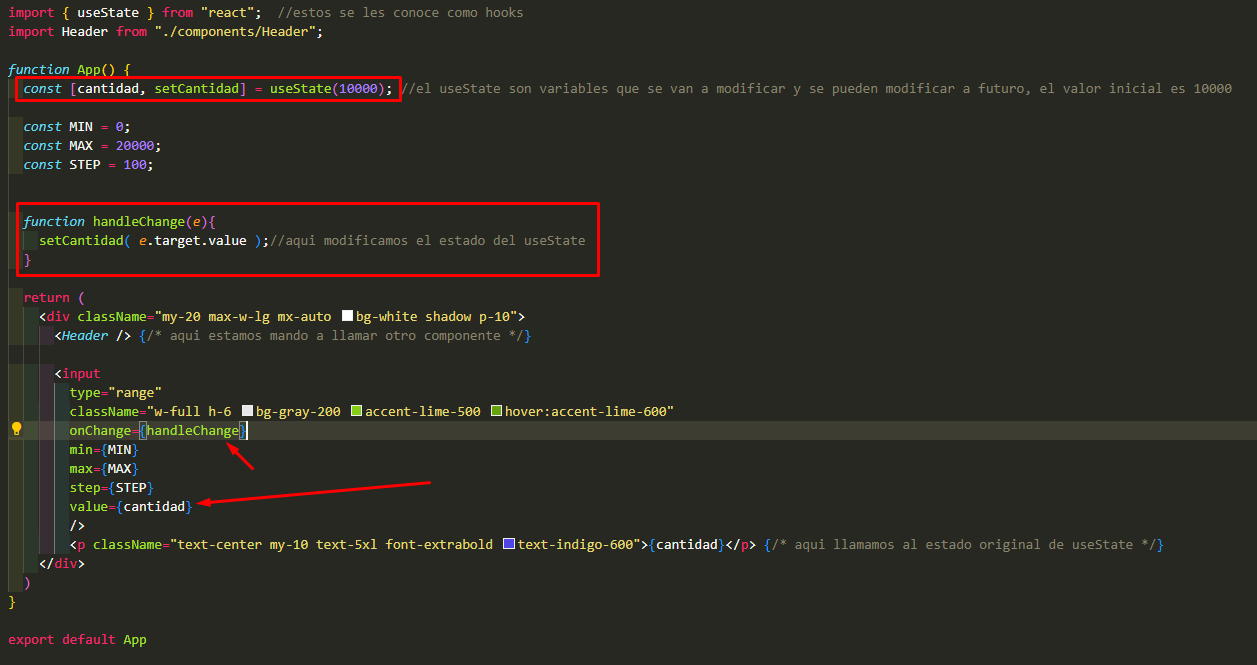
La forma correcta de modificar set cantidad seria de la siguiente forma,



Nota: El state basicamente se utiliza para valores que cambian si los valores no cambia se colocan en variables

En la función handleChange le pasamos e que es el evento de change accedemos a su valor y entonces el setCantidad del useState cambia.

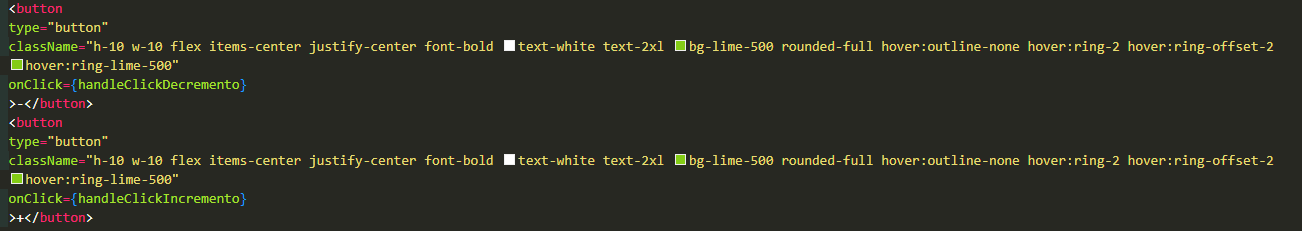
El valor inicial del state es 10000, que se le asigna a cantida, esta cantidad se utiliza para el comienzo del value



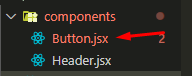




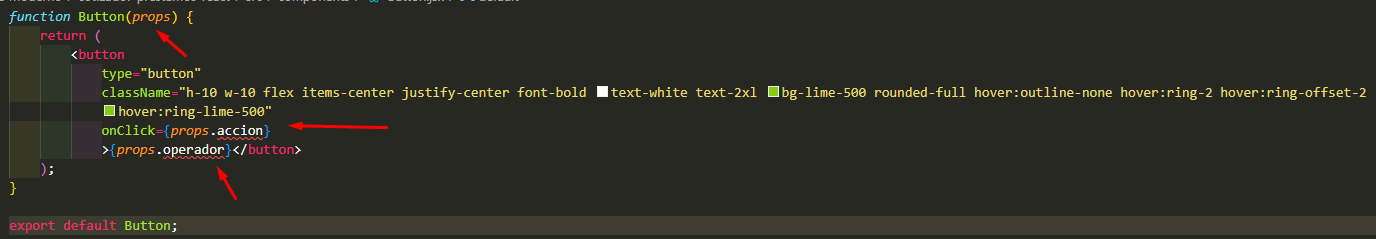
Convertir el siguiente elemento en componente y pasarles props, cuando el código se repite es mejor separarlo y que sea solo uno



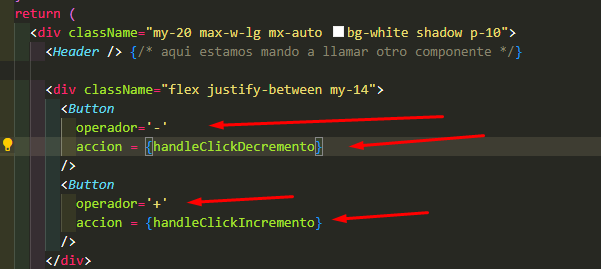
Cramos el componente de botón, se ve en rojo por que eslint se le tenia que cambiar las rules como mas abajo se muestra.



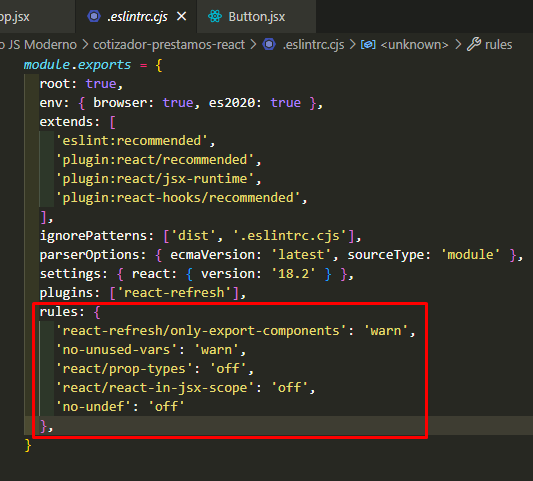
Creamos el código del botón y le pasamos los props, los props se pasan en automatico en todos los componentes de react



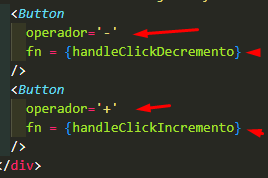
En la parte del botón pasamos los parámetros



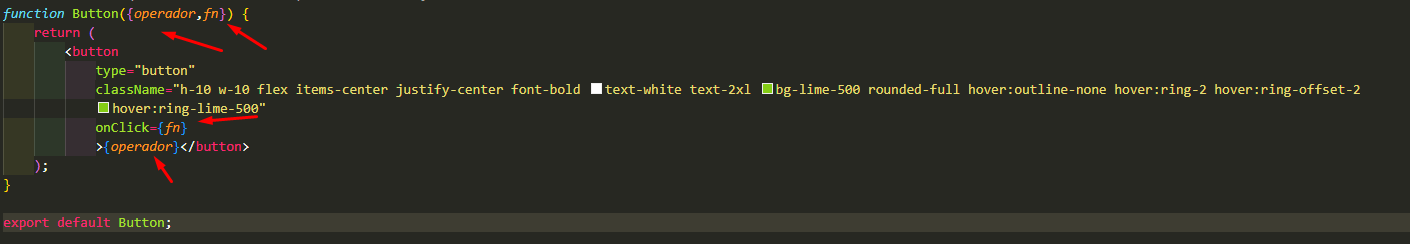
Eslint marcaba errores y se le cambio esto



Se cambio el nombre de los parámetros que se le pasan al componente

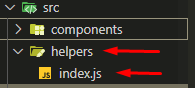


Aplicamos destructuring



Creando un helper para formatear a moneda y para los mensajes

1. Creamos una carpeta llamada helpers sobre src
2. Creamos un archivo en este caso lo llamamos index.js
3. Creamos el código y lo exportamos

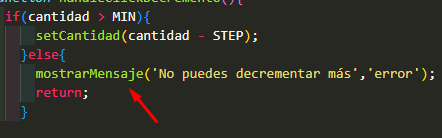




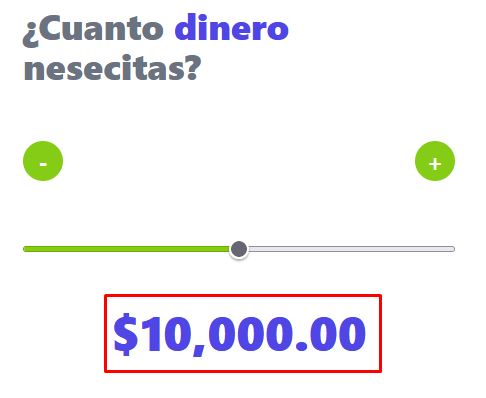
Para utilizarlo solo llamamos a la función

En nuestro caso ahora tenemos dos uno para formatear Dinero y para mostrar mensajes de Swal alert

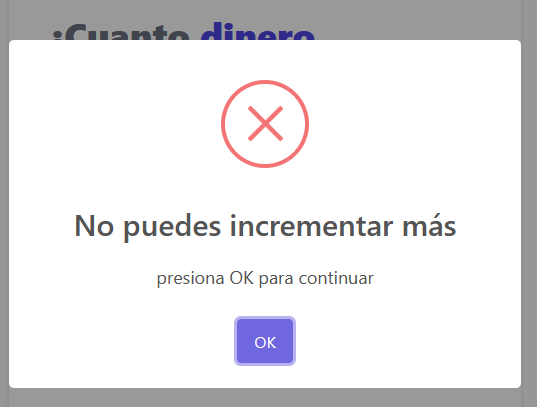




Formateado



Mensaje de error cuando se quiere incrementar de mas o decrementar de mas de los limites permitidos



Otra forma de pasar el state.

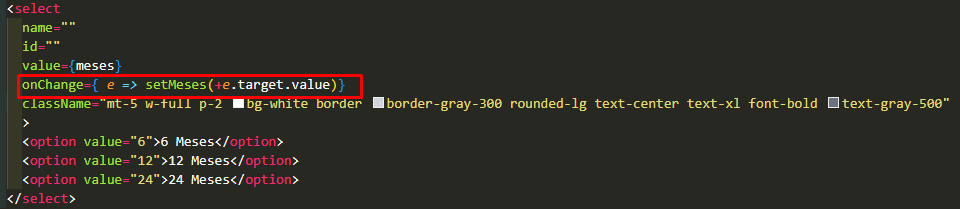
Lo recomendable es crear una función para el cambio del state como se realizo previamente, pero en este caso lo aremos dentro del mismo onChange para que veamos que de las dos formas se puede realizar.

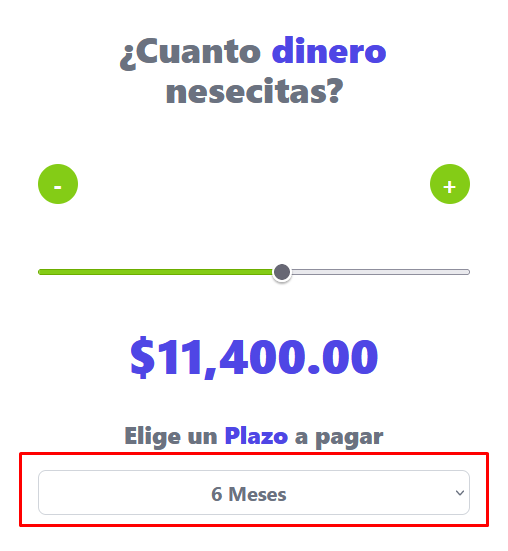
Se creo un useState para un select ya que como sabemos el select tiene opciones y estas pueden ser diferentes,

Estado inicial es meses con un valor de 6 y su estado lo definimos con setMeses



Uso del estado actual en el onChange, el símbolo +e convierte el valor del evento e.target.value a valor y no a string que es el valor por default





useEffect

es un hook que se ejecuta una vez que el componente esta listo, y toma un array de dependencias.

En su listado de dependencias se le puede pasar un state para escuchar los cambios que ocurran en ese state, en caso de que el state se actualice, useEffect se va a ejecutar nuevamente

Nota: el use efecto es como una función que esta a la escucha de los componente que nosotros le pasemos.

Es como un eventlistener que tiene que estar revisando si hay un cambio en un botón, un select, un imput ect. Pero todo esto en conjunto.

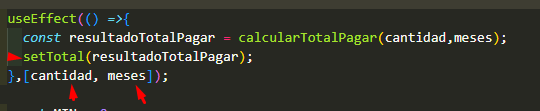
El useffect en nuestro caso esta a la escucha del monto y del plazo a cambiar, para de esta forma cambiar el total.

Como usarlo

Lo llamamos



[cantidad, meses] son los eventos a los cuales esta escuchando cuando cambien, cuando esto ocurra en setTotal se actualiza el calculo.



Otra forma de hacerlo sin usar el useEffect seria, pasar directamente el calculo

