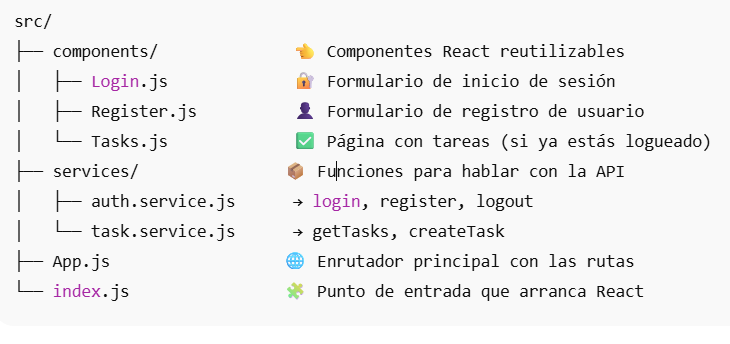
Teniendo node instalado, en la capeta del proyecto, hacemos en consola

npx create-react-app tareas-app

meterme en la carpeta

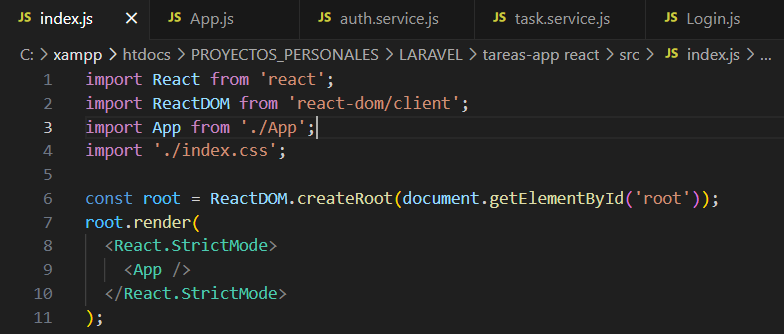
npm install axios react-router-dom

src venia con unos archivos, pero los hemos borrado y hemos dejado esta estructura



Ahora vamos a ir modificando todos estos archivos

**INDEX.JS**

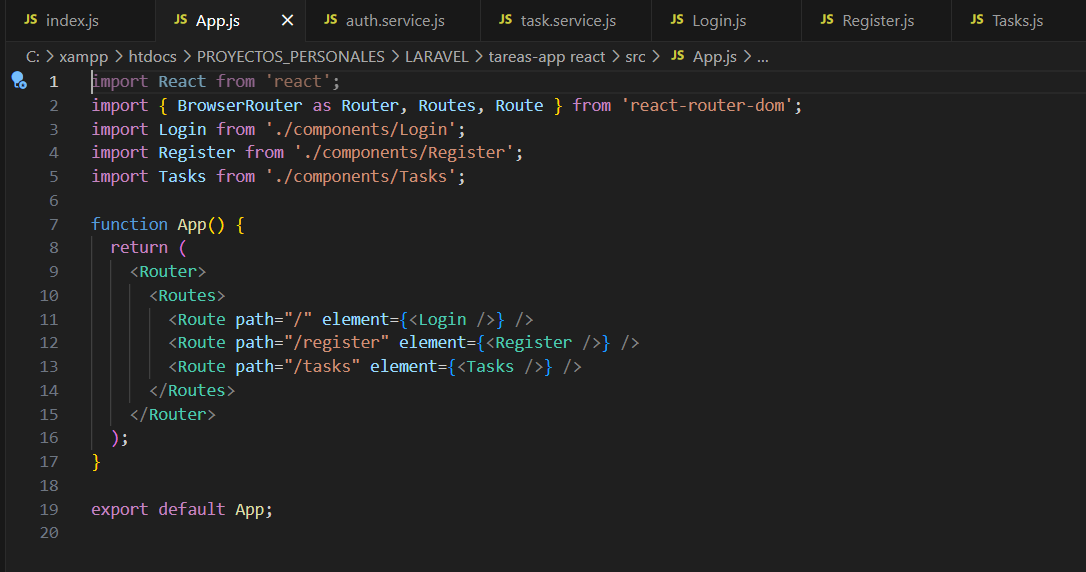
****

Punto de entrada a la aplicación. Importa todo lo necesario para funcionar, crea un root

Y hace root.render de App, que es el enrutador principal de la aplicación

**APP.JS**

(es el archivo que contiene las rutas)

****

Importa todos los archivos de components. Además de react y router. Y pone las rutas principales, que son /login que nos lleva a dentro de la carpeta components a Login.js

/register que nos llevara dentro de la carpeta components a Register

/tasks que nos lleva dentro de la carpeta components a tasks

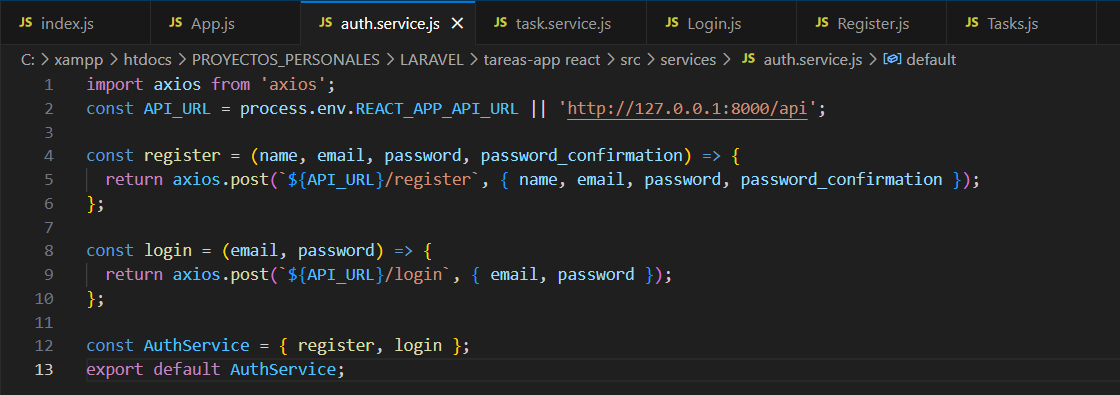
Todo esto lo ha hecho dentro de una funcion App, que exporta

**AUTH.SERVICE.JS Y TASK.SERVICE.JS**

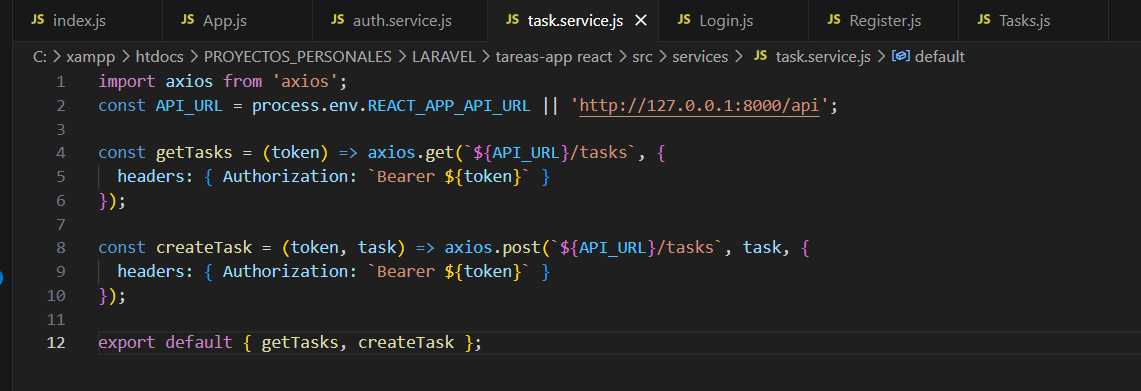
Tienen las funciones para hacer las llamadas a la api. Estas funciones se exportan, y son utilizadas en login.js register.js (las de auth.service) y tasks.js (las de task.service)

Simplemente tienen la url de la api, como deben de pasarse los parámetros que reciben la funcion, y exporta la funcion, para que se use desde los archivos register, login y tasks.js, asi estos usan la funcion, pasándole los parámetros correctos de entrada. Y ya procesan la salida de la api, metiéndola en el codigo html de renderizado que tienen

En otras palabras, encapsulan la llamada a la api, contienen una funcion con la llamada

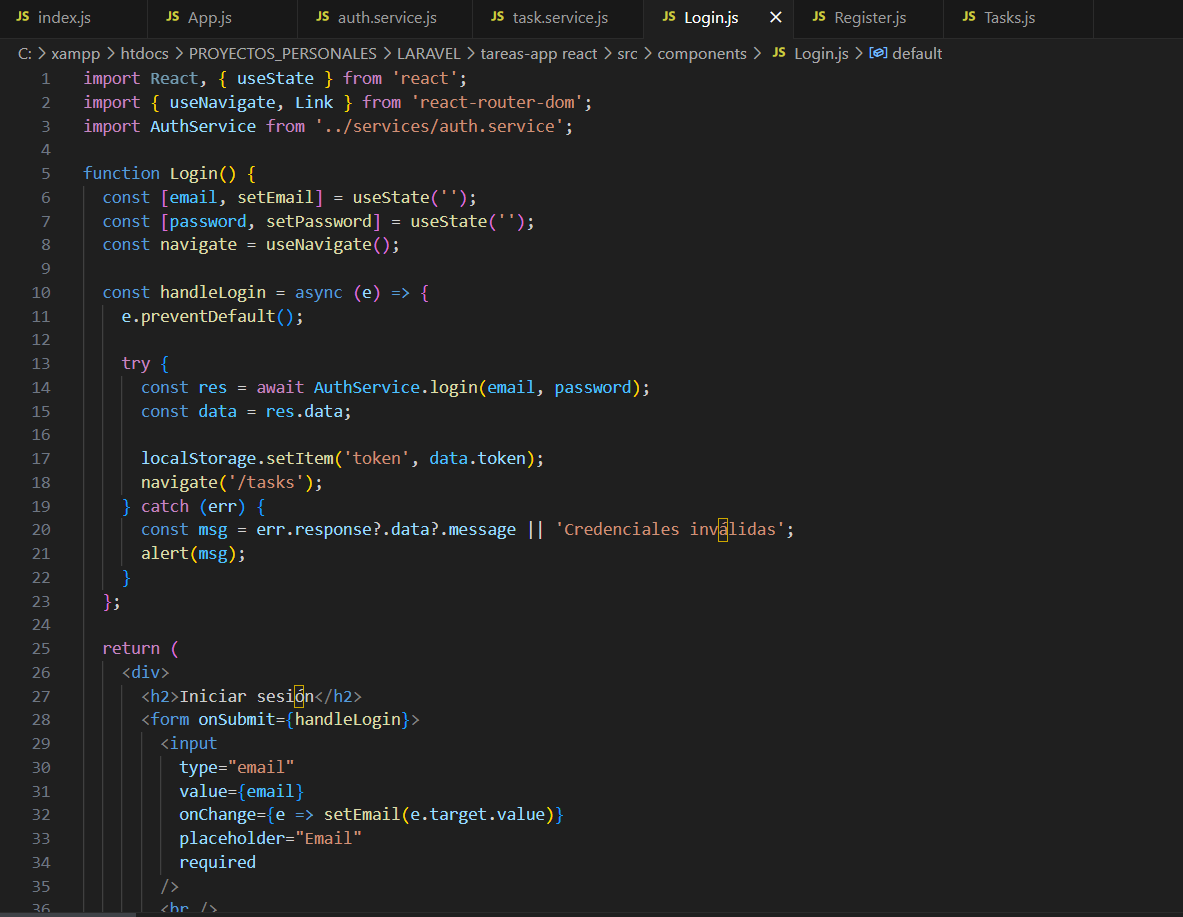


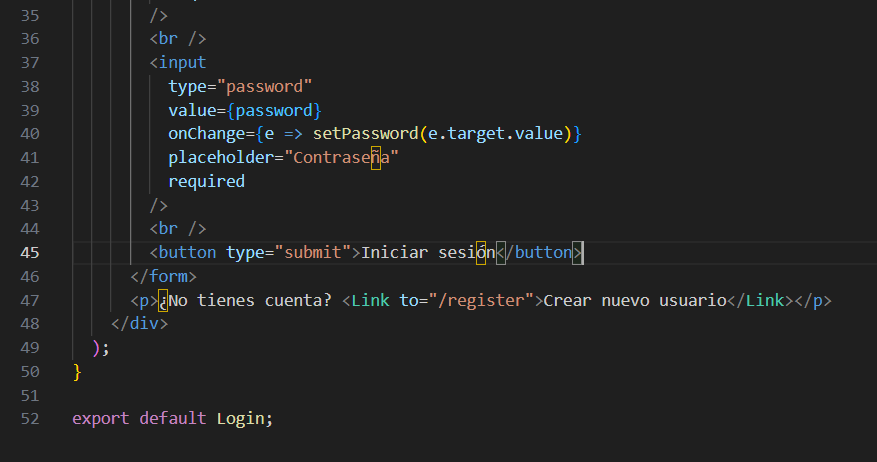
Entonces en login y register con llamar a AuthService.register o login, pasando los parámetros ya estará hecha la llamada a la api



Y aquí, en tasks, con llamar a TaskService.getTasks o create tas, pasando el token, y la task, ya estará hecha la llamada

**LOGIN.JS**

****

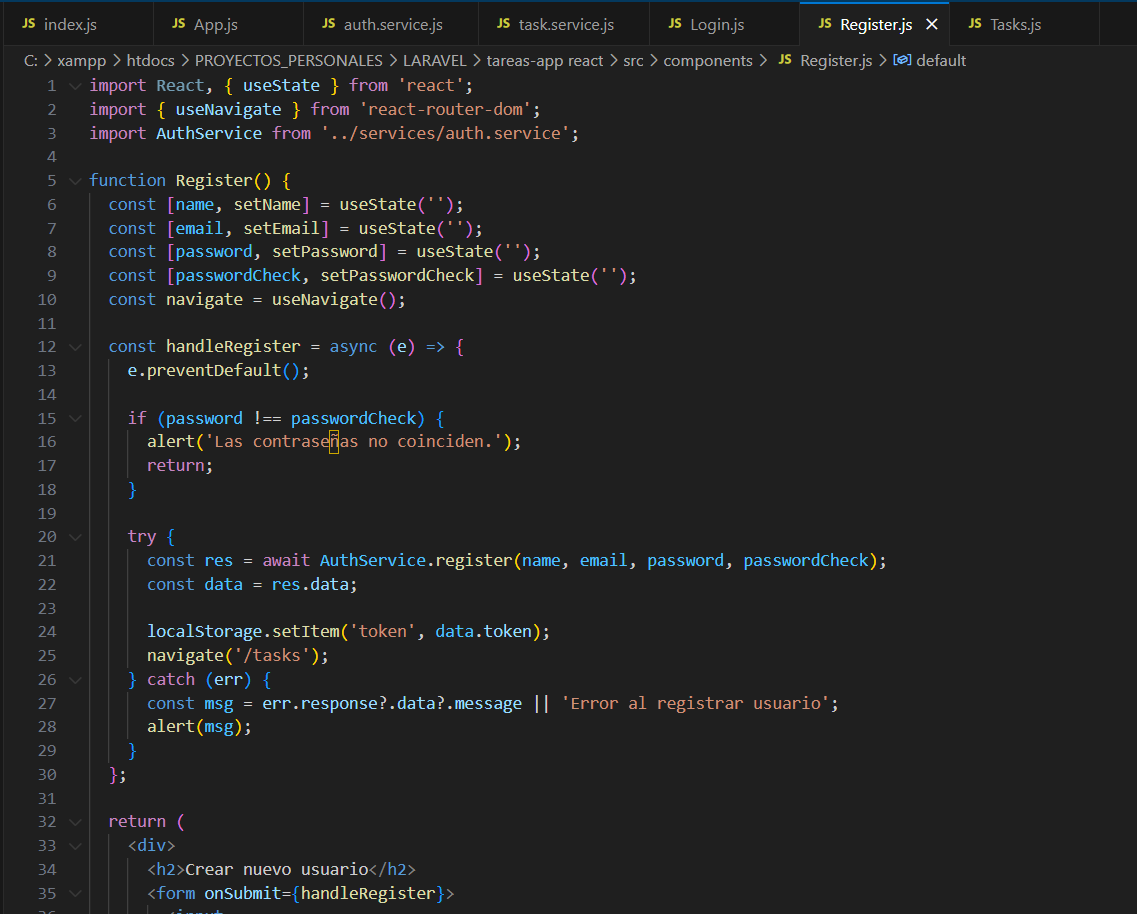
****

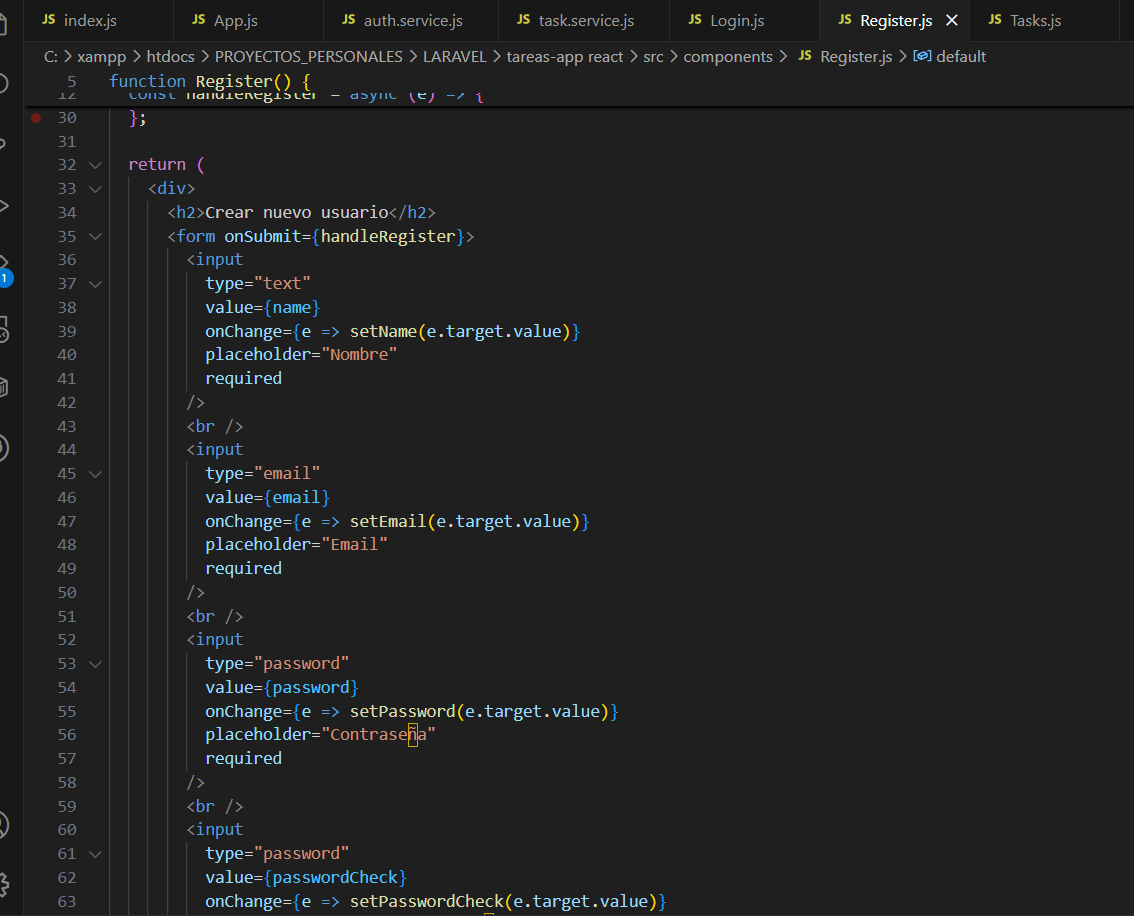
Hay un formulario que establece las variables email y password, cuando hacemos submit llama a la funcion handleLogin que lo que hace es intentar llamar a la funcion que habíamos establecido en Auth.service, para llamar a la api, pasando los parámetros del formulario, y si funciona recibimos data. Guardamos la variable, token en localstorage. Y por la ruta, nos redirige a tasks.js

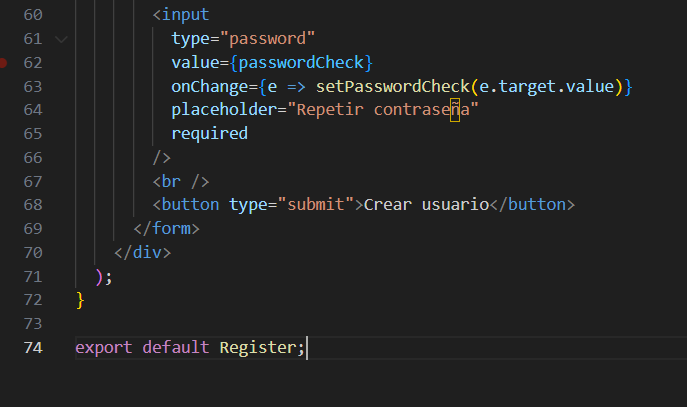
Todo esto al final exporta una funcion llamada login, que es la que se ejecuta cuando accedemos por la ruta que tenia app.js,

También hay un enlace, que nos lleva a register.js por si no tenemos cuenta, a través de la ruta que esta en app.js, y que ejecuta el método register, que exporta register.js

**REGISTER.JS**

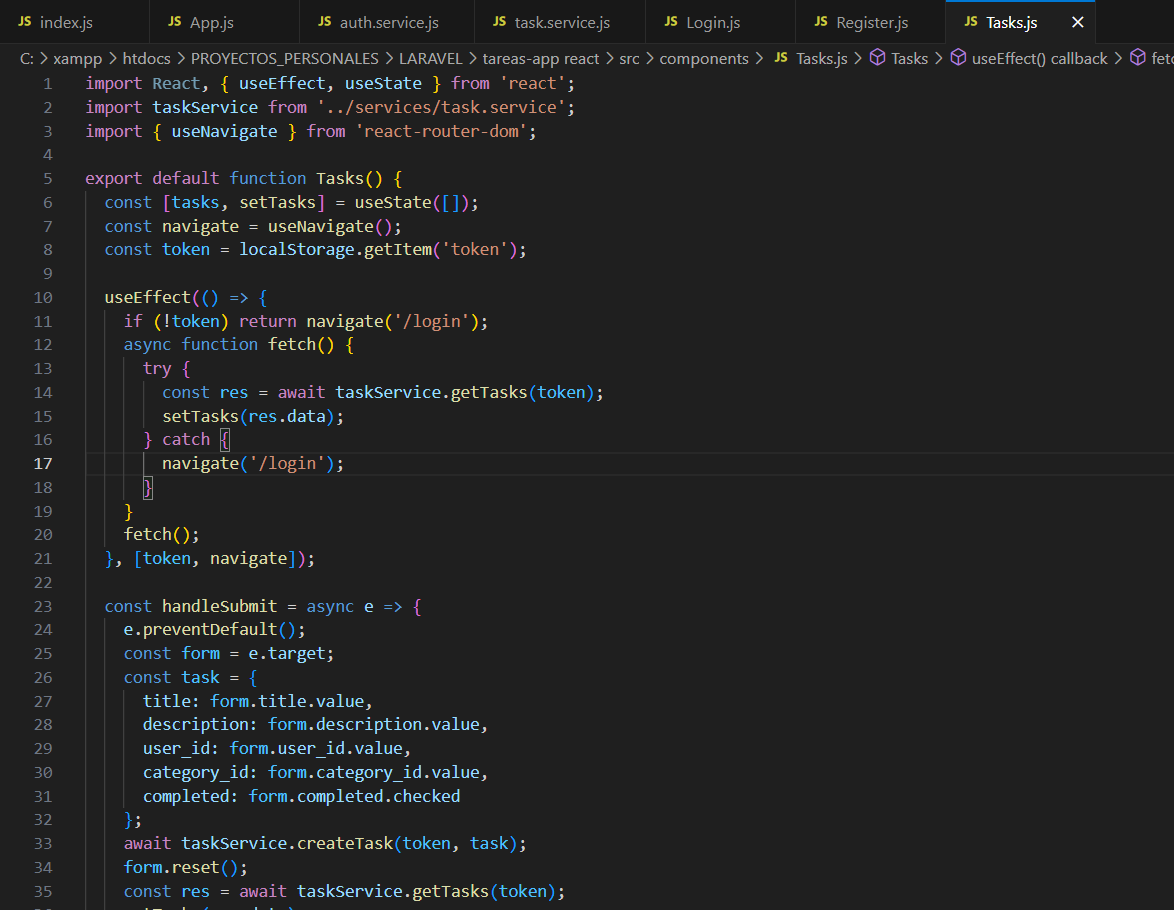
****

****

****

Pues esto lo mismo, pero llama a la funcion password de Auth.register, hace la llamada a la api con todas las variables para crear un nuevo usuario, y si todo funciona y se crea el usuario, guarda data.token en localhost, y nos redirige a tasks.js

**TASKS.JS**





Coge el token de localstorage para guardarlo en una variable,

Si no hay token redirige a login, si lo hay usa el método de task.service.js pasándole el token (el método que muestra todas las tareas)

Y luego llama al método set.tasks, pasándole como parámetro la respuesta data que nos habrá devuelto la api

Luego por otro lado, hay un formulario, que recoge unas variables, para pasarlas a el otro método de task.service.js, que es crear tarea, pasa todos los parámetros que ha recogido, metidos en una variable task, que es como una array de cuando pasamos métodos en una api por post, y el token por otro lado, que lo hemos cogido de la variable que lo recogía de localhost

Esto debería de guardar el método en la base de datos, y luego llamamos a set.tasks, que es el método que devovia todas las tareas

**Pero donde esta definida la funcion set.tasks, si no esta en ningun sitio**

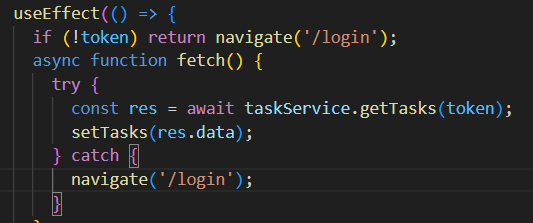
Arriba de tasks tenemos esta línea de código, useState, se ve que lo que hace es crear un **hook de react**, que lo que hace, es cambiar el valor de la variable tasks, cada vez que se llama a setTasks, por lo tanto, cada vez que llamemos a la funcion setTasks, vamos a modificar el valor de esta variable



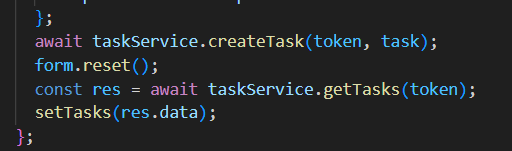
Y la variable tasks, se mapea en esta línea de abajo, poniendo su id, su title, si esta completada



Recordemos que la variable tasks, viene de la respuesta de la api cuando usábamos el método get tasks



O de la respuesta de la api cuando usamos el método create task



La api devuelve toda la lista de tareas, eso lo configuramos en laravel

Una vez puesto todo esto, haciendo

Npm start

React levanta un servidor en <http://localhost:3000>

Donde ya puedo acceder a los formularios, rellenarlos, y obtener la lista de tareas