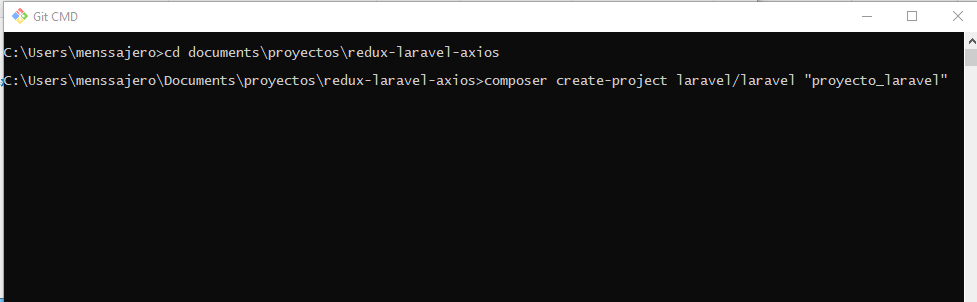
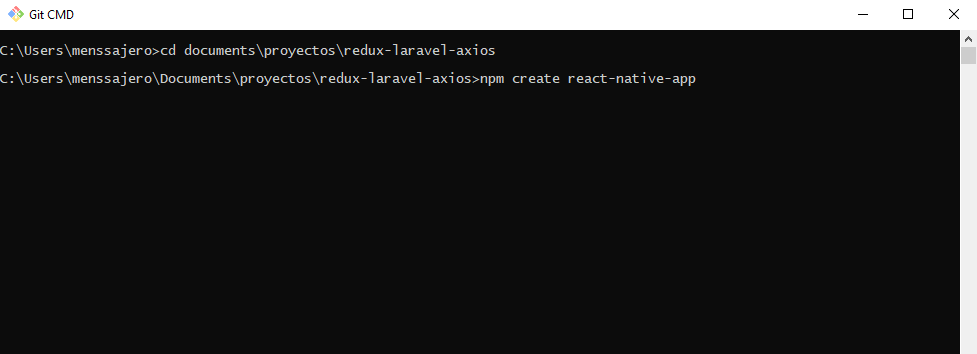
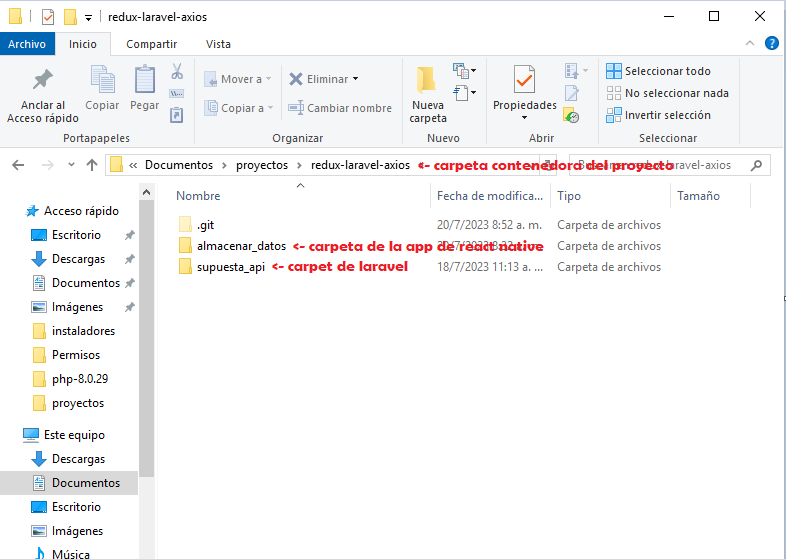
Para crear una API en Laravel que permita guardar los datos de 'nombre', 'tipo', 'descripcion' y 'stock' en una base de datos PostgreSQL y conectarla con una aplicación React Native a través de un crud, conectada con el método post, necesitas seguir los siguientes pasos:

1. Primero debes crear una carpeta en común que tenga los dos proyectos (el de react native y el de laravel).
2. Luego crea con el comando de creación el proyecto de laravel, ej:

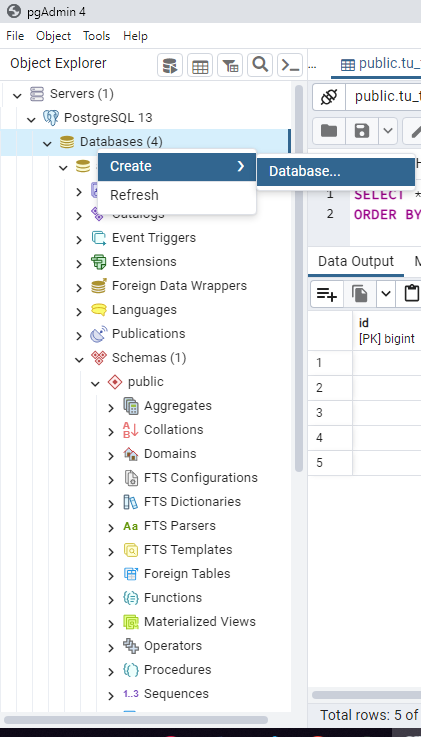


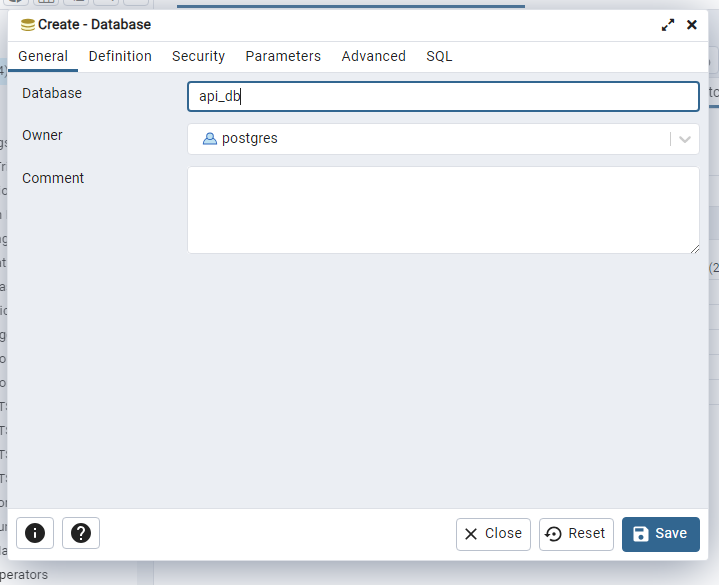
1. Crea el proyecto de react native con el comando correspondiente:



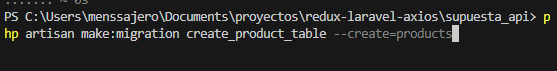


1. Cree la base de datos, en este caso postgreSQL

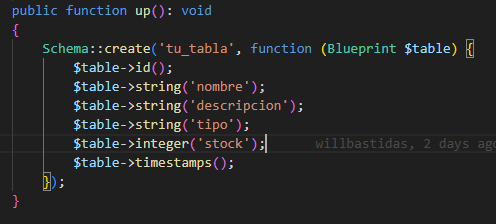


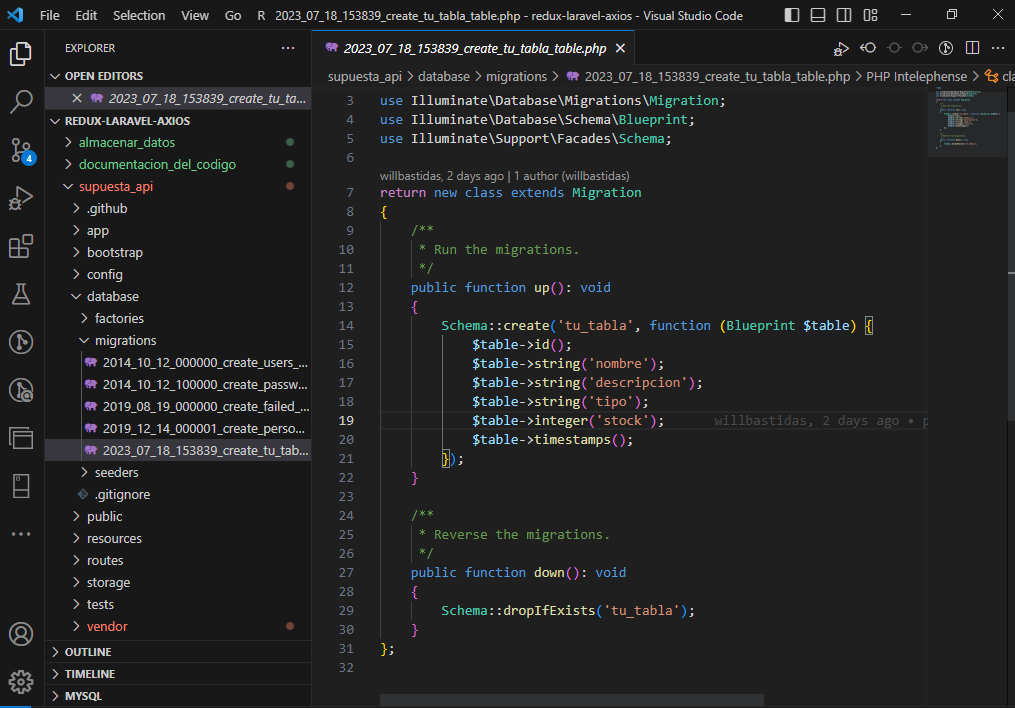


1. Habrá la carpeta contenedora de los proyectos con su entorno de desarrollo en este caso visual code, abra una terminal, diríjase a la carpte del laravel “cd supuesta api” y coloque el siguiente código para crear la migración de los datos deseados: “ php artisan make:migration create\_product\_table --create=products “

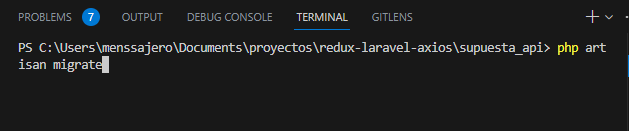


1. Dentro del proyecto se abra creado la migración en supuesta\_api\database\migrations, dentro de la función up cree el esquema deseado, donde dice ‘tu\_tabla’ debe colocar la tabla destino del esquema

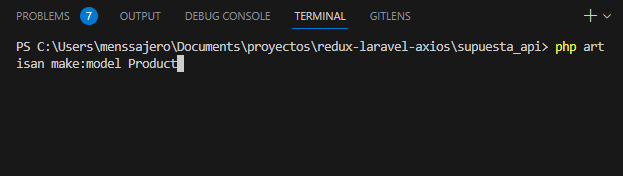




1. Ejecute el siguiente comando en la terminal del proyecto de laravel para ejecutar la migración “php artisan migrate”

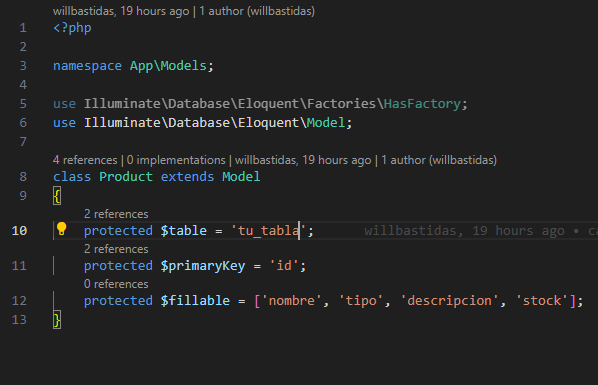


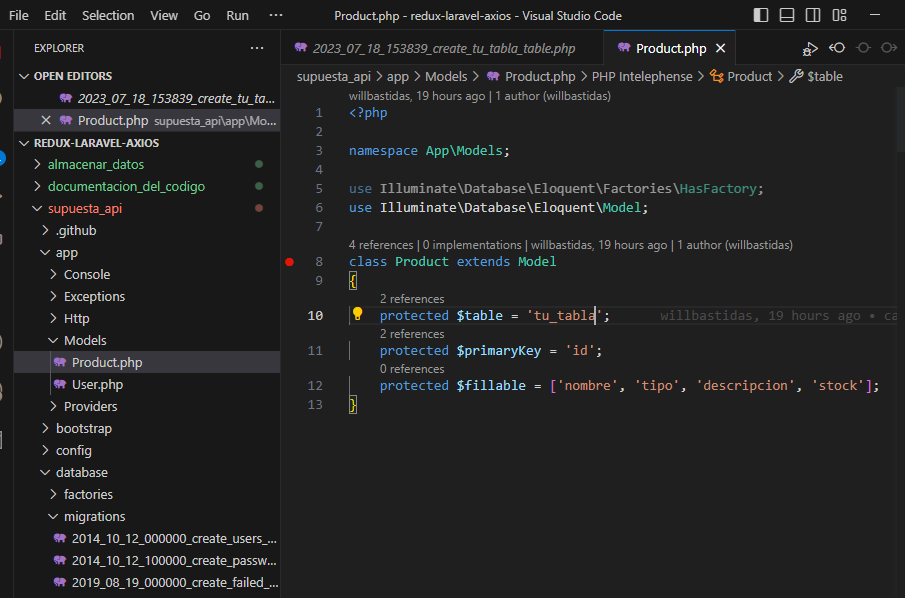
1. Cree el modelo de la tabla deseada con el siguiente comando php artisan make:model Product



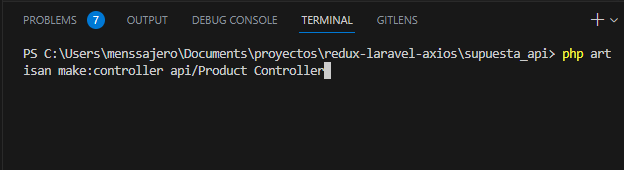
1. El modelo se encuentra en supuesta\_api\app\Models\product , El modelo define los campos de la tabla que se utilizaran.

Protected $tabla = ‘tu\_table’; dirección de la tabla donde van dirigidos los datos.





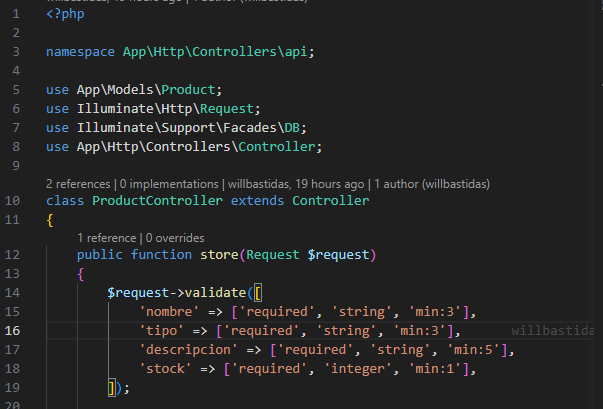
1. Creer el controlador que maneje la lógica de la Api para crear, leer, actualizar, y eliminar los productos con el siguiente comando php artisan make:controller api/Product Controller



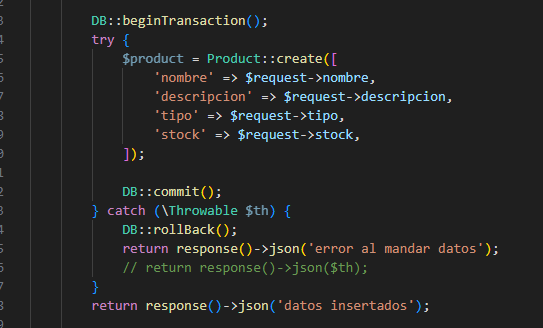
1. El controlador se encuentra en la siguiente ruta supuesta\_api\app\Http\Controller\api.

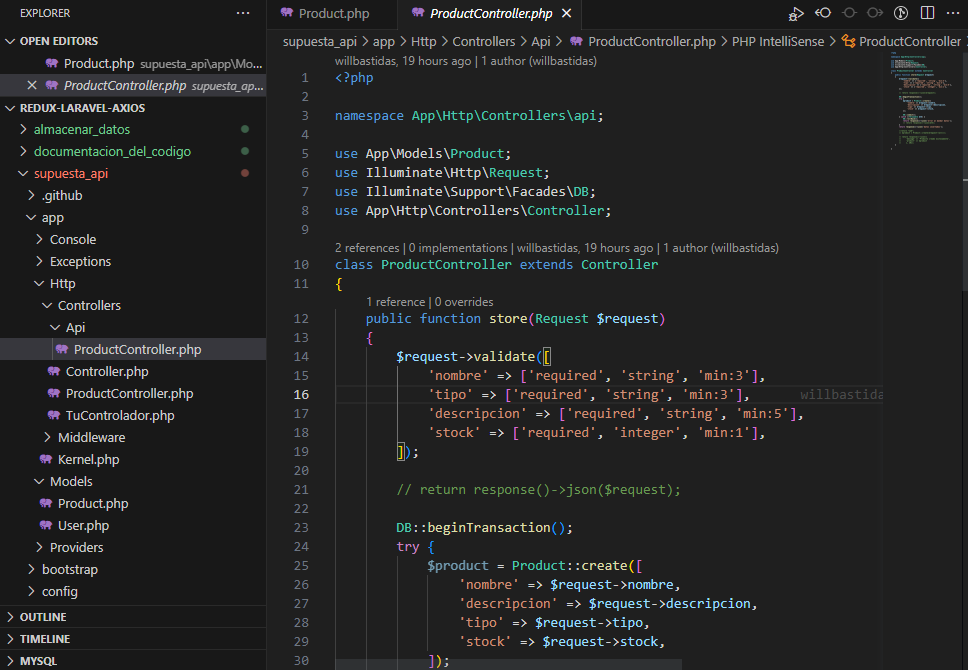
En la primera parte del controlador de la línea 1 a la línea 8 se debe llamar lo que se va a utilizar.

En la línea 12 comienza la función que se va a utilizar el $request->validated funciona para que los datos requeridos se validen o ponerle condiciones en este caso required (campo onligatorio) string (tipo de dato), min:3 (minimo de datos a ingresar).



El try catch sirve para darle ordenes de lo que va hacer y en caso de que no hago lo correspondiente ( catch ) diga un mensaje o haga una función en específico, el rollback es para que se si sale un proceso mal haga que toda la operación falle, por ejemplo si se llega a insertar una letra en stock no se pueda subir a la base de datos, asi se asegura que la información no llegue incompleta.

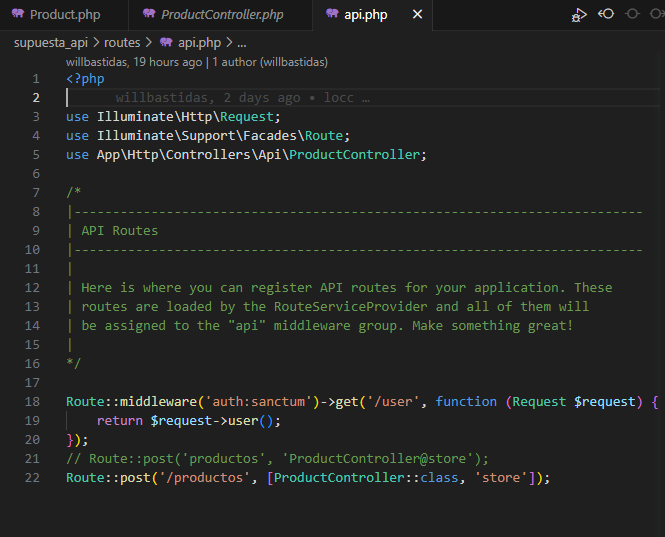




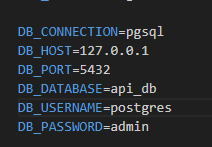
1. Defina las rutas del api en la ubicación supuesta\_api\routes\api.php

En las primeras líneas recuerde convocar lo que se va a utilizar.

En la parte de route:: post, /productos-> nombre del modelo, luego ProductController::class -> nombre del controlador ubicado en la carpeta api, ‘store’-> nombre de la función desea convocar.

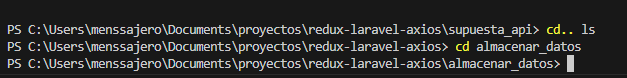


1. Configure la ubicación de la base de datos en el archivo: .env



**Ahora por parte de la app pasos que se deben realizar:**

1. Nos dirigimos a la carpeta de la aplicación de react native, si desea usar la misma terminar puede ejecutar cd.. ls enter y luego cd “nombre de su proyecto” o si desea puede abrir otra terminal y solo coloca cd “nombre de su proyecto”.



1. Instalamos las dependencias a utilizar. Que serían el redux/toolkit y el axios: npm install @reduxjs/toolkit react-redux redux, npm install axios.





1. Ahora creamos la siguiente estructura de carpetas:

./store

* Store.js

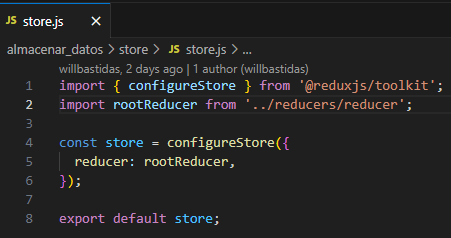
./components

* addItem.js
* Itemlist.js

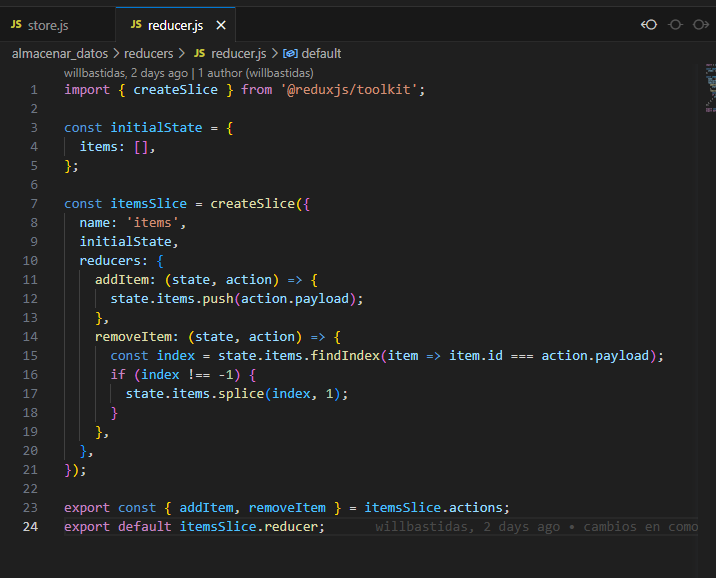
./ reducers

* Reducers.js

1. En store store.js coloque el siguiente código

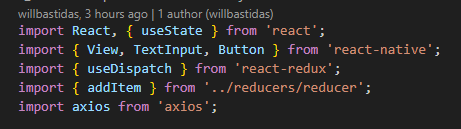


1. Luego, crea un archivo reducers.js donde definirás tus reducers. En este ejemplo, crearemos un reducer para manejar la lista de elementos que se guardarán en nuestra aplicación.

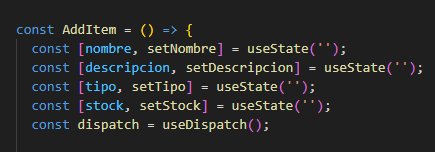


1. Ahora, en AddItem donde el usuario pueda ingresar los datos que quiere guardar y mandarlo a la base de datos por axios. En este ejemplo, el usuario ingresará un nombre y una descripción, tipo y stock

Primero se debe importar lo que se va a utilizar.



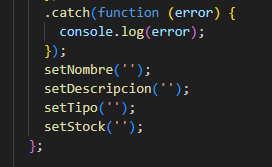
Se crea la función y luego se debe colocar los datos que se desea enviar



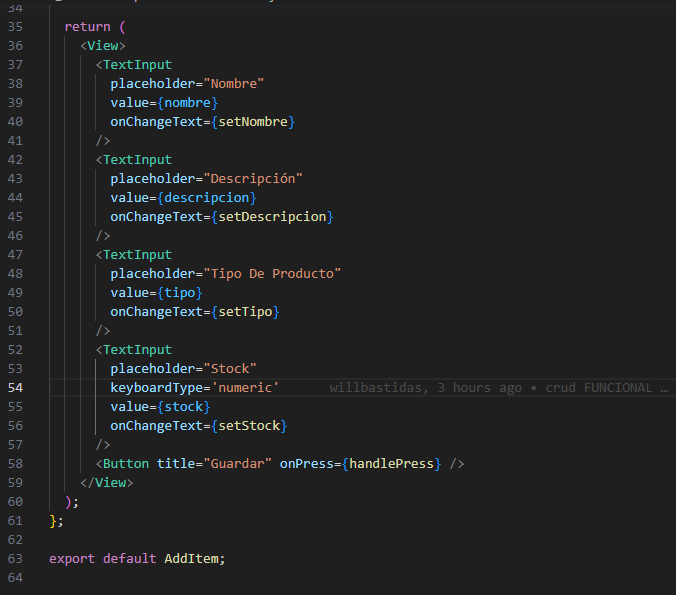
Luego se coloca lo que se va hacer al presionar, dentro de los paréntesis del axios.post se debe colocar la ruta donde se ubica la base de datos en este caso como es local se coloca la ip, luego el puerto, y la dirección de donde se van a almacenar los datos y luego se colocar dentro de una variable los datos que se van a enviar deben coincidir con los datos solicitados en el api para que no cause un error.



Luego se debe colocar el set para que sea enviado el dato.

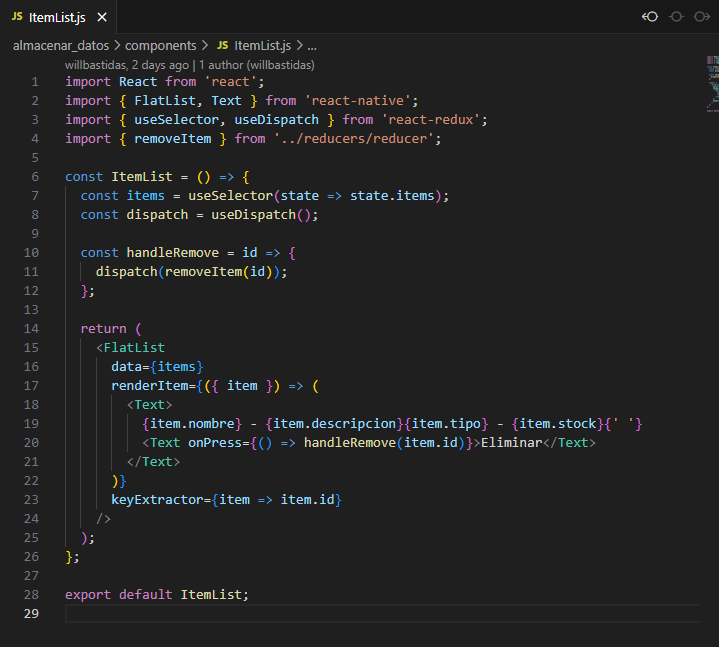


Luego se crea la estructura visual que desea que el usuario vea a través de un return ()

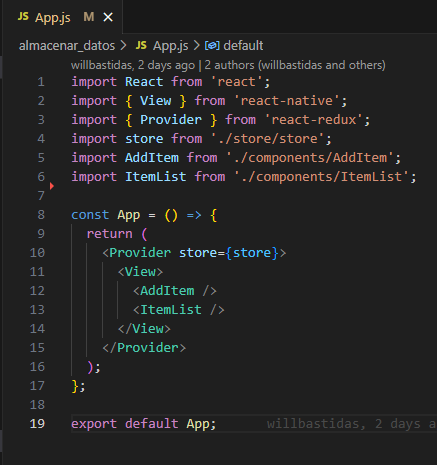


1. Diríjase a el componente ItemList que muestre los elementos guardados. En este ejemplo, simplemente los mostraremos en una lista.

Dentro del return se ingresa el cómo desee se vean los elementos ingresarlos al mostrarlos.



1. Ahora, para utilizar estos componentes en tu aplicación, simplemente importa AddItem y ItemList y colócalos en tu componente principal.



1. Para ejecutar el proyecto y probarlo debe colocar en la ruta del proyecto de laravel con el siguiente comando php artisan serve --host=192.168.85.239 --port=8000



1. Luego debe ejecutar el proyecto de la aplicación con npm run Android.



1. Pruebe su app.