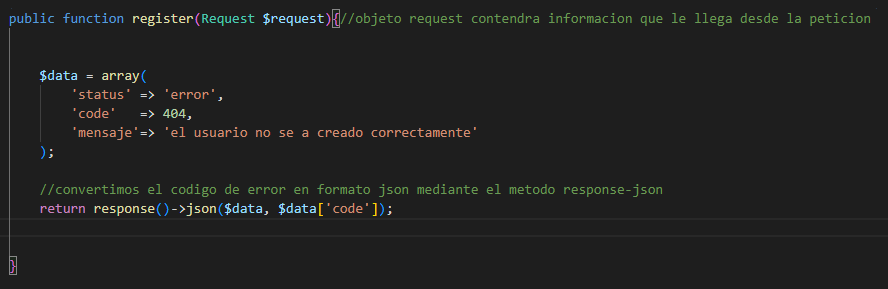
**4 LOGICA PARA REGISTRO**

**LOGICA DE REGISTRO**

Obviamente vamos a modificar el controlador de usuario concretamente el método register en este caso. La funcionalidad de este dado que estamos programando un APIREST, es procesar una serie de datos que se le pasan como entrada y devolver unos datos procesados en formato JSON.

Un ejemplo sencillo, supongamos que vamos a devolver un error, necesitaremos primero crear un array con todos los datos que queramos enviar como respuesta y a continuación mediante el método Response.json, transformar esa respuesta al formato JSON:

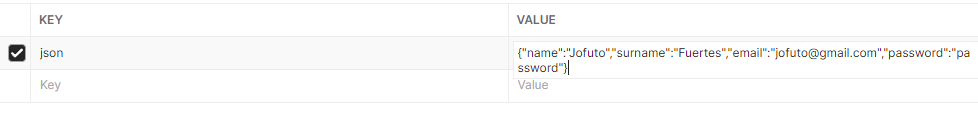


Esto nos servirá mas adelante para devolver en caso de no poder registrar el usuario, en caso de que ya exista etc.

Los pasos para desarrollar la lógica restante serán:

1. Recoger los datos de usuario por post
2. Validar usuario
3. Cifrar contraseña
4. Comprobar que no esté duplicado
5. Crear usuario

Los datos del formulario también los recibiremos en formato JSON, para hacer las pruebas los enviaremos así también:



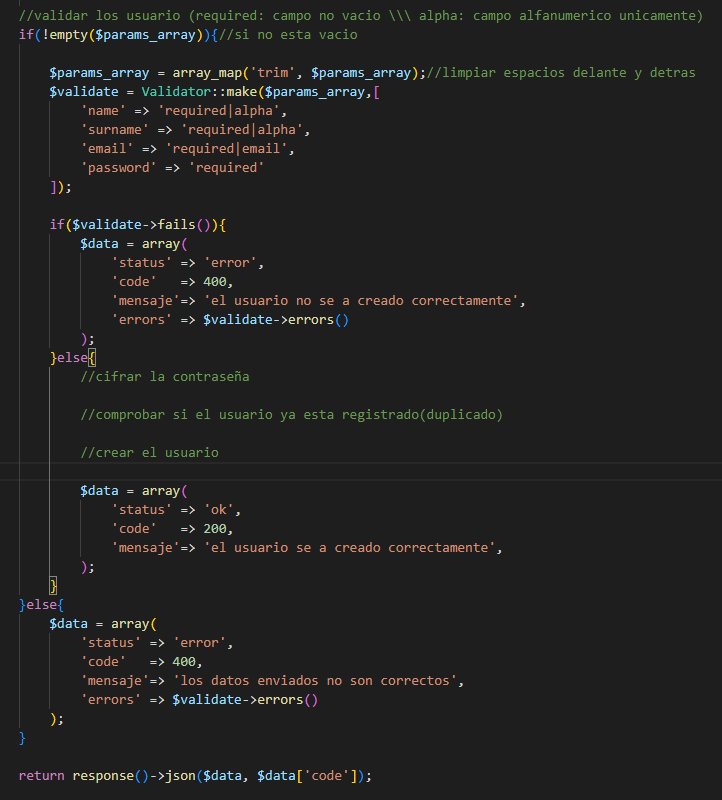
El front que programemos obviamente no nos enviara un parámetro por cada dato nos enviara como en el ejemplo todo junto en un formato JSON y en la lógica nosotros guardaremos cada uno de estos datos en diferentes variables como veremos a continuación.

Cabe destacar que no solo existe el formato JSON, pero es el mas utilizado por su sencillez, pero claramente no es el más seguro.

Texto

Descripción generada automáticamente

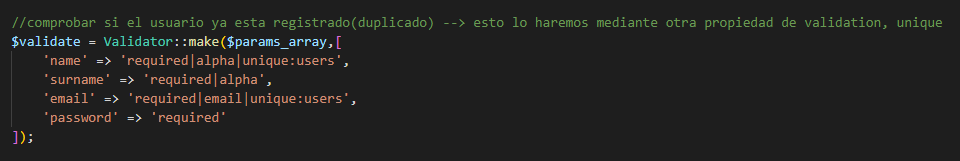
Ahora pasaremos a validar los datos de usuario, para ello en Laravel podremos hacerlo de una manera muy sencilla mediante la librería validator, para mas información: <https://laravel.com/docs/9.x/validation> : AVALIBLE VALIDATION RULES



Como vemos la clase validator nos proporciona mucha ayuda pudiendo seleccionar como queremos que pase el cliente los parámetros, si utilizaremos mayúsculas y minúsculas, que el campo no este vacío, si queremos que únicamente se usen letras(Alpha).

Pasaremos ahora a lo ultimo que nos queda, registrar al cliente en la base de datos para ello una vez validado, cifraremos su contraseña, comprobaremos que no esta duplicado y si todo esta correcto, registraremos el usuario.

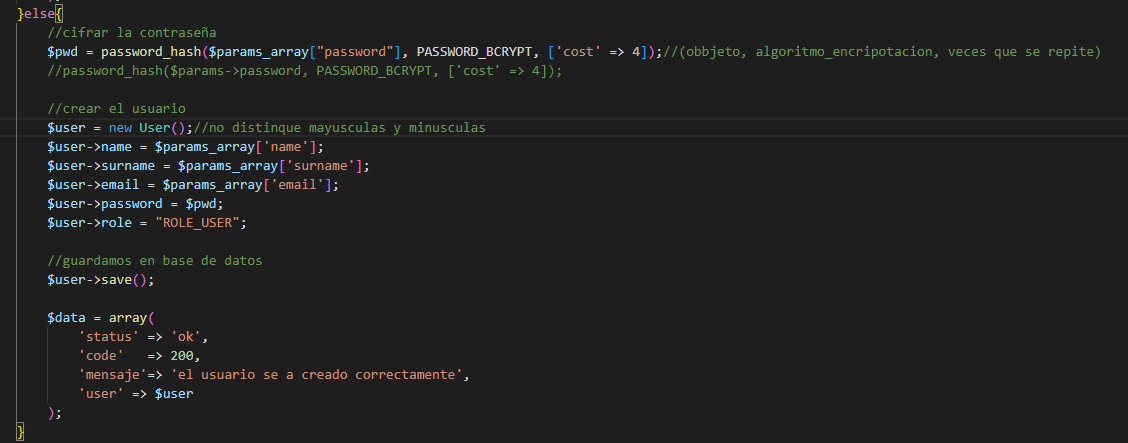
Para la validación utilizaremos otro utilidad de validation, unique:



Tan solo nos faltaría registrar el usuario, para ello necesitaremos importar el modelo, donde tenemos la base de datos :

use App\Models\User;

luego cifraremos la contraseña crearemos el objeto user y por último lo guardaremos en la base de datos usando el método interno llamado sabe();



Por ultimo el return, dado que hay que devolver la respuesta que hemos ido creando en data:

return response()->json($data, $data['code']);