**Cognos**

composer create-project laravel/laravel aprendiendo-laravel "5.7.\*" --prefer-dist

composer create-project --prefer-dist laravel/laravel laravel-cognos

\*\* al colocar en producción crear otra llave



Subimos al git (tomando en cuenta el .gitignore)

descargamos en la otra compu, vamos a la carpeta descargada en el terminal:

composer update

desbloque el archivo .env (de env.example)

y genero la llave

php artisan key:generate

**Blumbit**

PHP > 7.2

Composer -> administrador de dependencias de PHP

composer create-project --prefer-dist laravel/laravel Producto-app

php artisan serve

**Viernes 10/1/20**

composer.json -> gestiona todo el proyecto

**manejadores de dependencias**

php - composer - composer.json - vendor (carpeta donde se descarga las librerias)

js - npm - package.json - node\_modules

java - maven - pom.xml

composer update (actualiza en base al archivo composer.json)

npm install

php artisan serve --host=0.0.0.0 --port=9000 (port change)

**Routes**

web.php (url)

**Views**

resources/views/welcome.blade.php (blade y php se unen como dos lenguajes de programacion)

**Generar un controlador**

php artisan make:controller CategoriaController

app/Http/Controllers

php artisan make:controller ProveedorController -r (con recursos)

Tarea: Hacer para productos con -r

4 vistas - listar, crear, editar, ver

**Lunes 13/1/20**

Peticiones HTTP

HEAD o GET

POST - guardar

PUT o PATCH - modificar - 200

DELETE - eliminar - 200

**Controlador con -r**

index - listar -get

create - cargar el formulario - get

store - guardar informacion - post

show - mostrar - get

edit - cargar el formulario de edicion - get

update - modificar informacion - put

destroy - delete

get - 200 (respuesta codigo http)

<https://diego.com.es/codigos-de-estado-http>

| Metodo | URL | Controlador - funcion | nombre |
| --- | --- | --- | --- |
| GET | /categoria | CategoriaController@index | categoria.index |
| GET | /categoria/create | CategoriaController@create | categoria.create |
| POST | /categoria | CategoriaController@store | categoria.store |
| GET | /categoria/{id} | CategoriaController@ver | categoria.ver |
| GET | /categoria/{id}/edit | CategoriaController@editar | categoria.editar |
| PUT | /categoria/{id} | CategoriaController@modificar | categoria.modificar |
| DELETE | /categoria/{id} | CategoriaController@eliminar | categoria.destroy |

{{route('guardar\_categoria')}}

lo mismo que /categoria

Tarea: Hacer lo de hoy para proveedor y producto

\*\*\* .env chekar en github

**Martes 14/1/20**

**Plantillas para Laravel**

<https://rimorsoft.com/plantilla-adminlte-en-laravel>

<https://github.com/acacha/adminlte-laravel>

Plantilla con roles y permisos

<https://github.com/spatie/laravel-permission>

Utilizamos este template:

<https://adminlte.io/>

Download/AdminLTE 3.0.1/Source code (zip)

Copiamos las carpetas: build, dist, plugin a Producto-app/public

creamos el archivo admin.blade.php en layouts/ y copiamos el contenido del index.html a utilizar (en este caso del index2.html)

**Hosting Gratuito**

<https://www.heroku.com/>

php artisan migrate

php artisan migrate:fresh (borra todas las tables y vuelve a migrar)

la tarea hacer con la plantilla y en la parte del menu colocar crear, listar, etc.

**Miercoles 15/1/20**

**Migraciones**

categoria tiene productos tiene proovedor

1:N N:M

tabla o migraciones

| categorias | productos | producto\_proveedor | proveedors |
| --- | --- | --- | --- |
| id (int, auto-increment, primary, unsigned) | id | id | id |
| nombre (varchar 50) | nombre (varch 250) | producto\_id | nombre |
| descripcion (text) | cantidad (int 11) | proveedor\_id | nombre\_contacto |
|  | precio (10, 2) |  | telefono |
|  | imagen (string 250) |  | direccion |
|  | detalle |  |  |
|  | estado = 1 (int) |  |  |
|  | categoria\_id |  |  |

\*\*cambiamos el timezone para tener la hora correcta en las migraciones

php artisan migrate

php artisan make:migration create\_categorias\_table

php artisan make:model Categoria (para hacer consultas con eloquent)

php artisan make:model Producto -m (crea modelo y migracion)

php artisan make:model ProductoProveedor -m

php artisan migrate:status

php artisan migrate:rollback (para eliminar la migracion)

php artisan migrate:fresh (para eliminar la migracion y las tablas creadas)

\*\*para que se relacionen las tablas debe estar en InnoDB (con mysam o algo asi, laravel tiene problemas)

php artisan tinker (para verificar las relaciones\*\*)

>>> use App\Categoria;

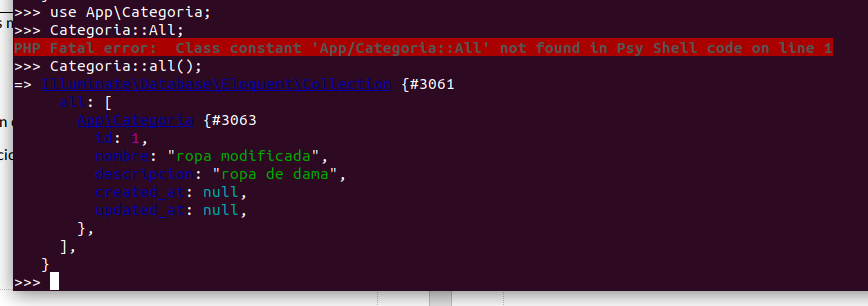
>>> Categoria::All();

cuando tengamos BBDD, nos creamos nuestro modelo

**Consultas**

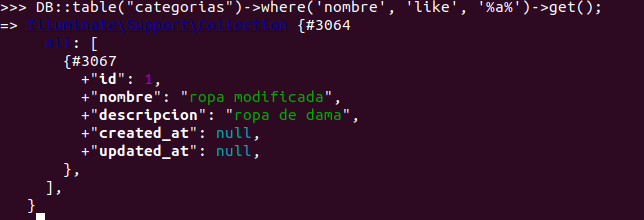
una forma de hacer consultas (no muy recomendada)

php artisan tinker (para realizar consultas\*\*)

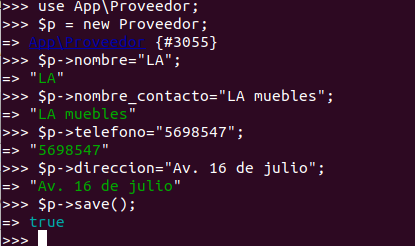


**Jueves 16/1/20**

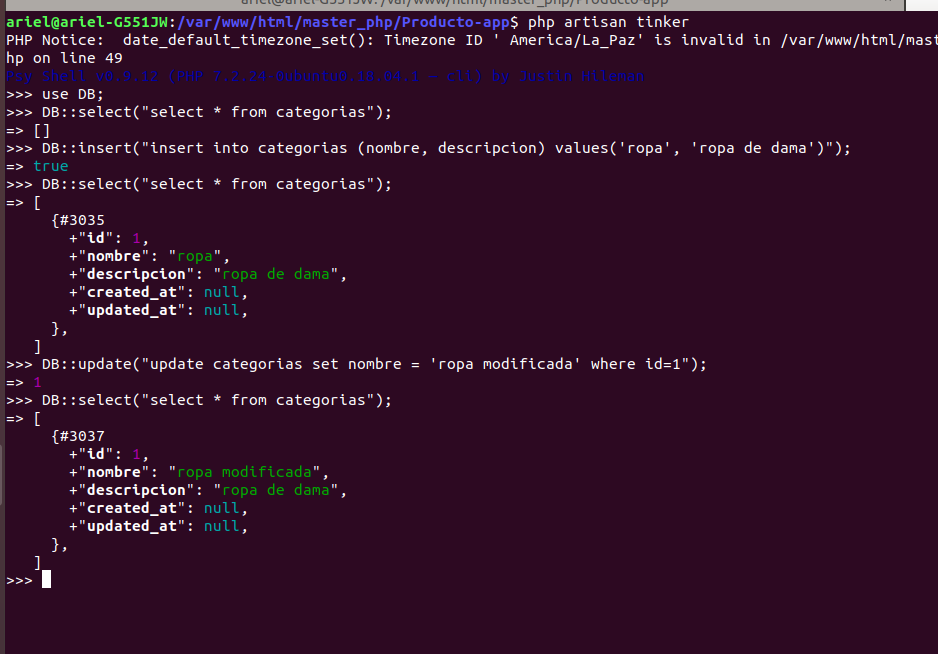




\*\*con eloquent, guardamos un registro

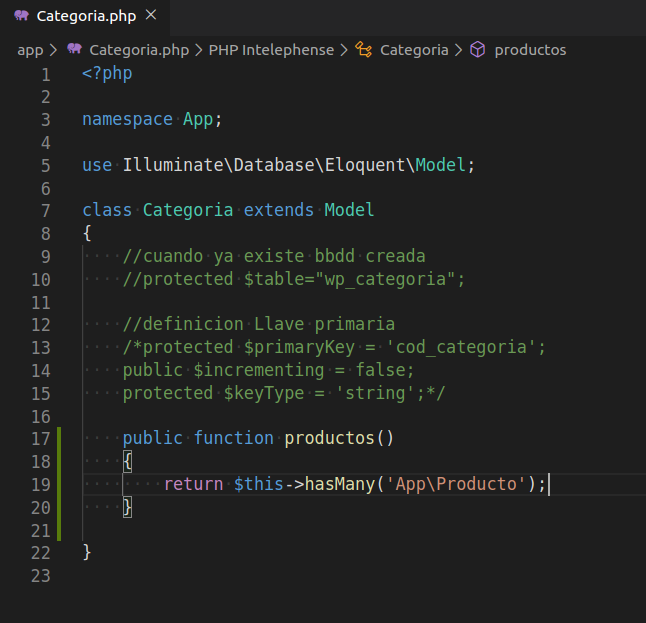




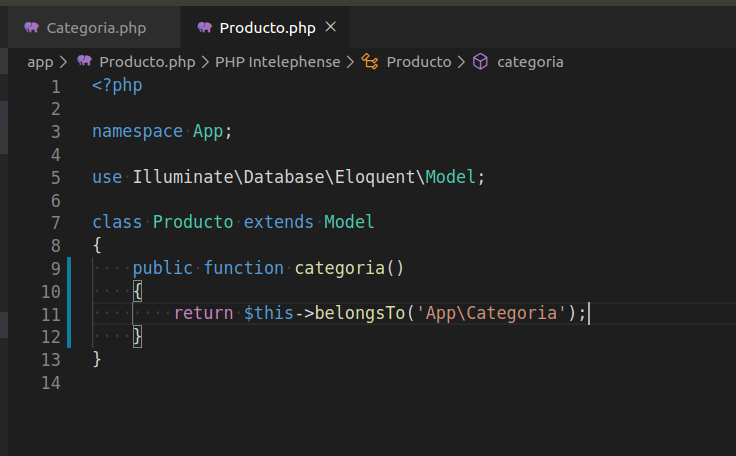


**Relaciones con eloquent**

**Muchos a uno**

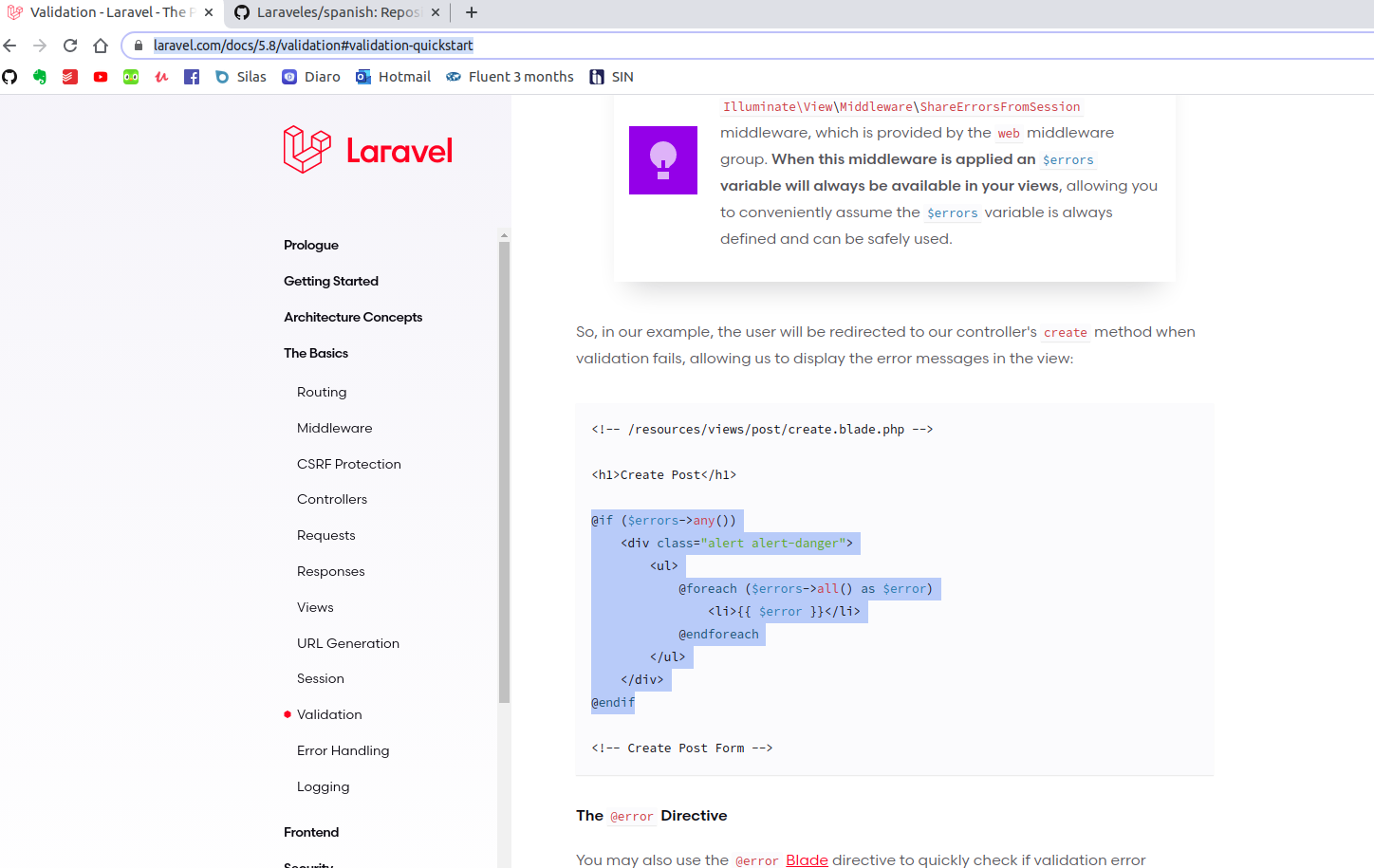


uno a muchos



Validaciones

<https://laravel.com/docs/5.8/validation#validation-quickstart>

[](https://laravel.com/docs/5.8/validation#validation-quickstart)

<https://github.com/Laraveles/spanish>

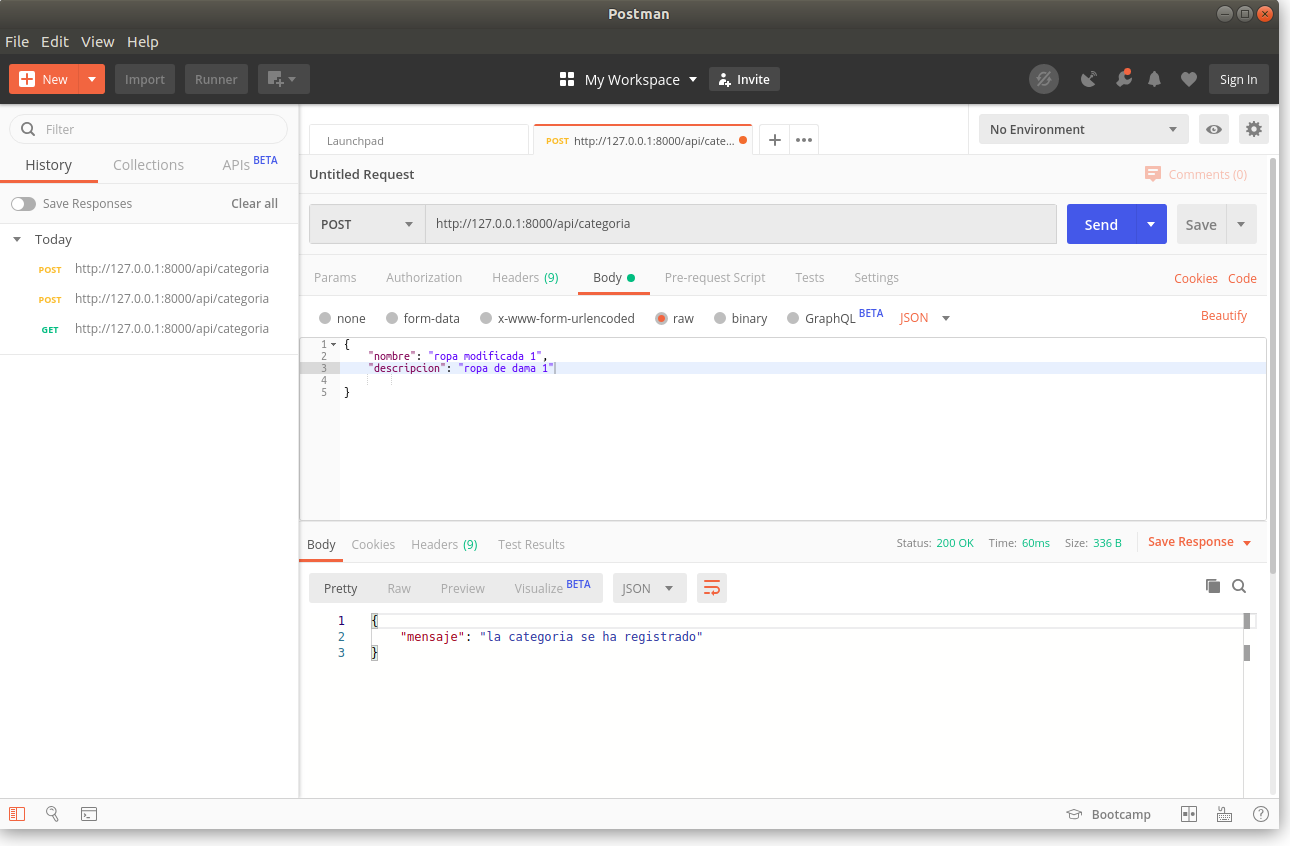
**Viernes 17/1/20**

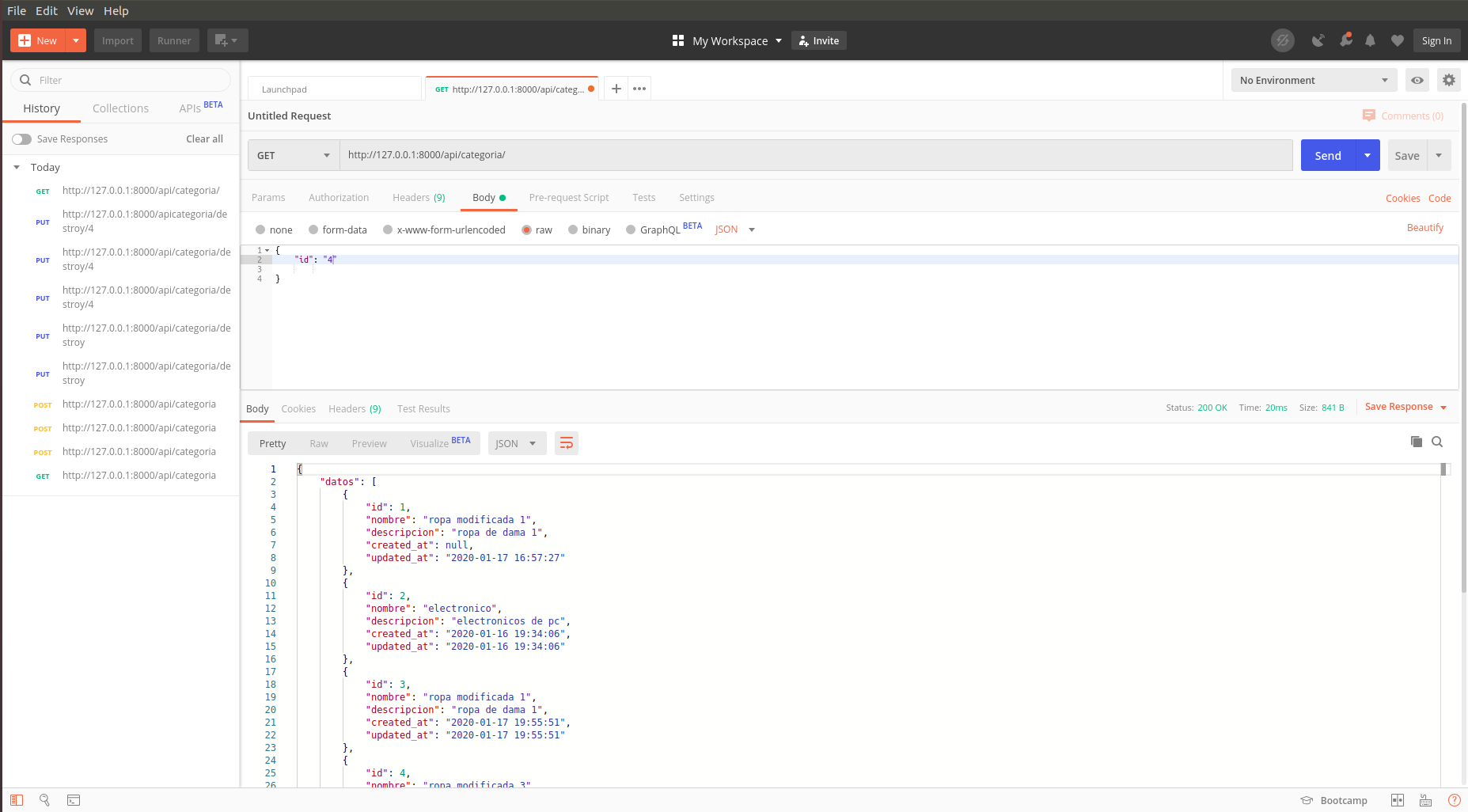
**Servicios Web**

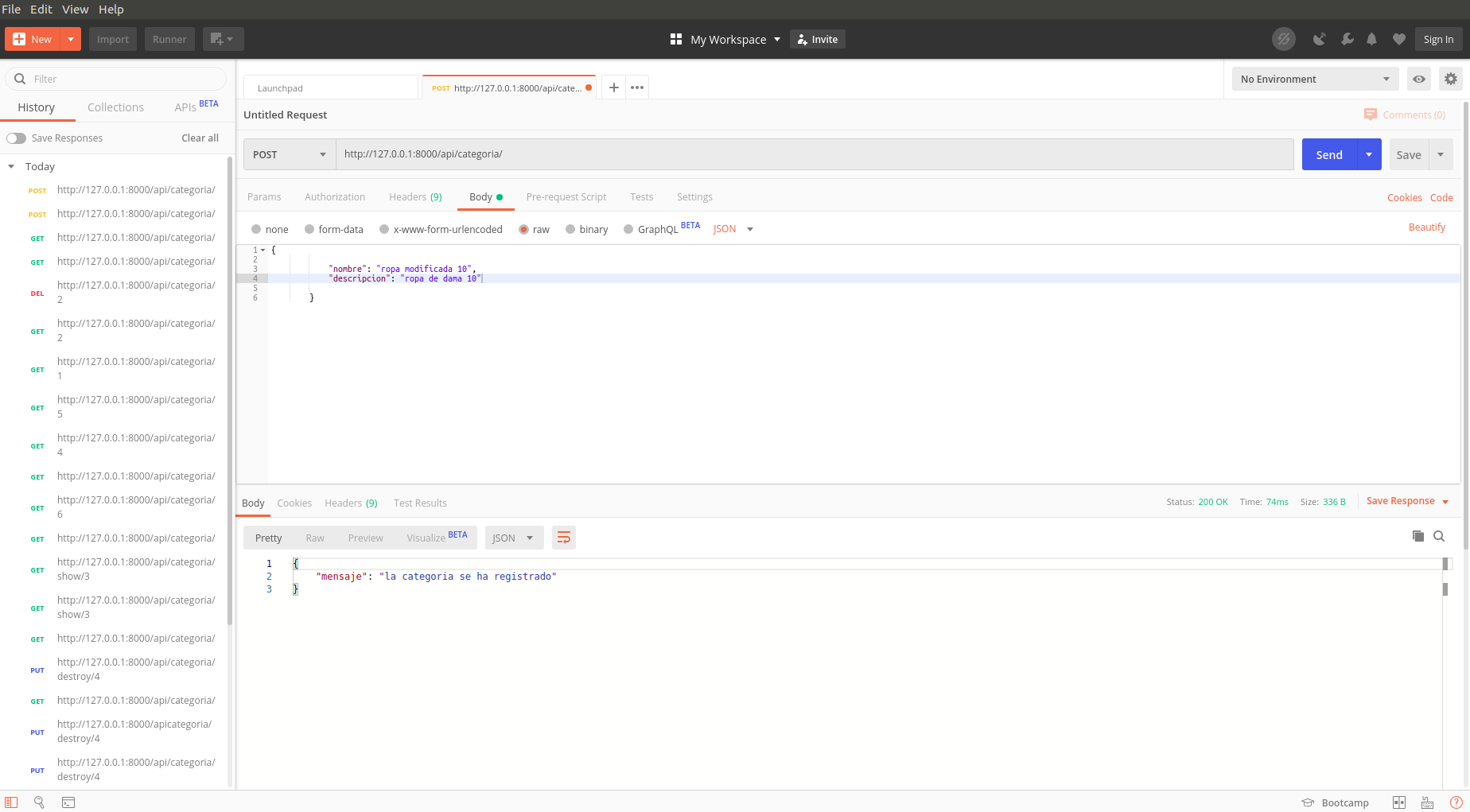
json- rest

xml-soap

postman - cliente http (se reemplazara con angular)



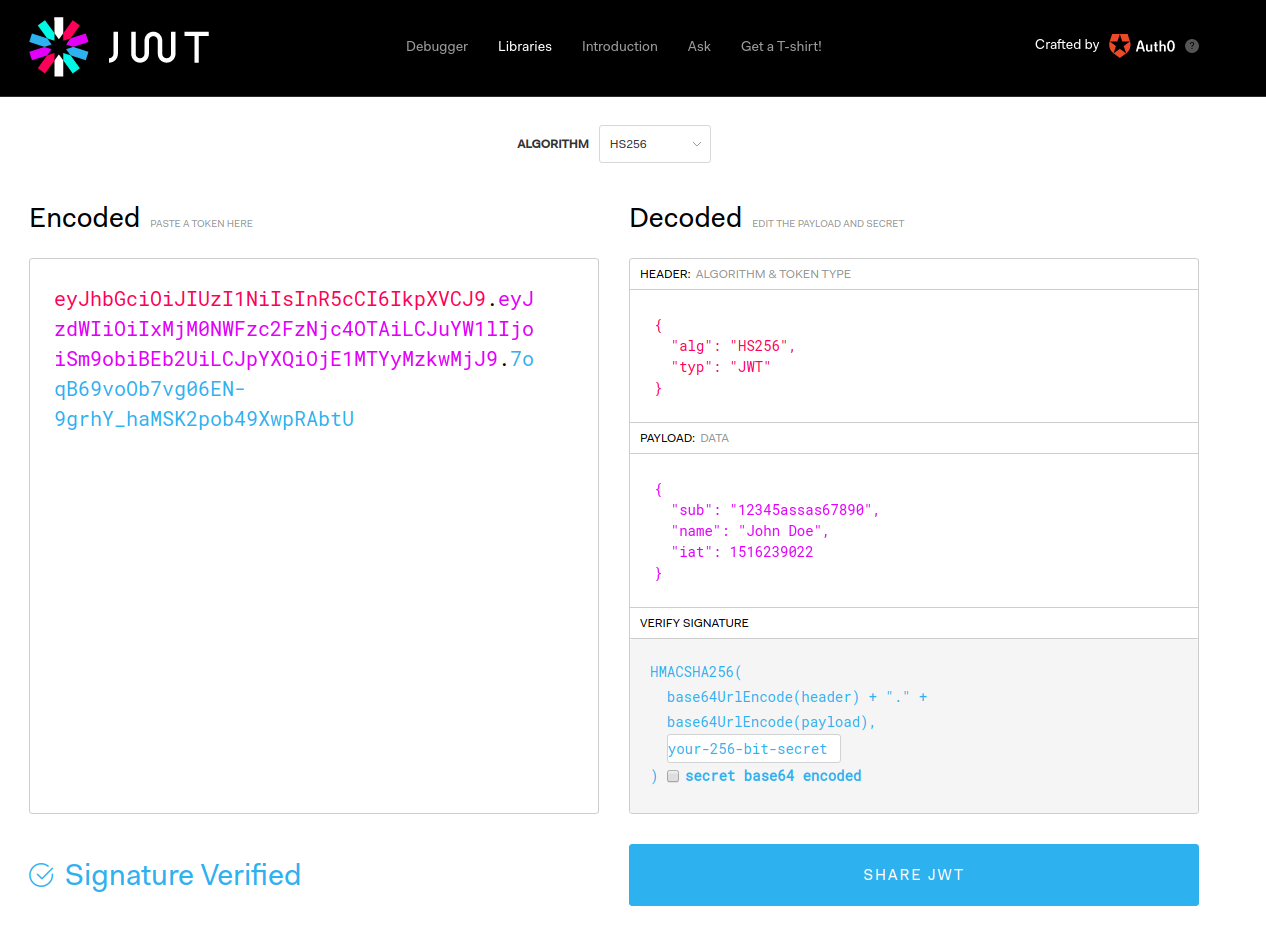




tal como esta, cualquiera puede hacer cambios

ahora haremos **autenticacion basada en tokens**

<https://jwt.io/>

[](https://jwt.io/)

cabecera.tipo.payload,signature

payload es datos

<https://jwt-auth.readthedocs.io/en/develop/>

composer require tymon/jwt-auth "^1.0.0"

si sale error sin comillas probar

<https://laravel-angular.io/>

para verificar si se instala en archivo composer.json

**Instalacion**

<https://jwt-auth.readthedocs.io/en/develop/laravel-installation/#install-via-composer>

composer require tymon/jwt-auth "^1.0.0"

php artisan vendor:publish --provider="Tymon\JWTAuth\Providers\LaravelServiceProvider"

php artisan jwt:secret

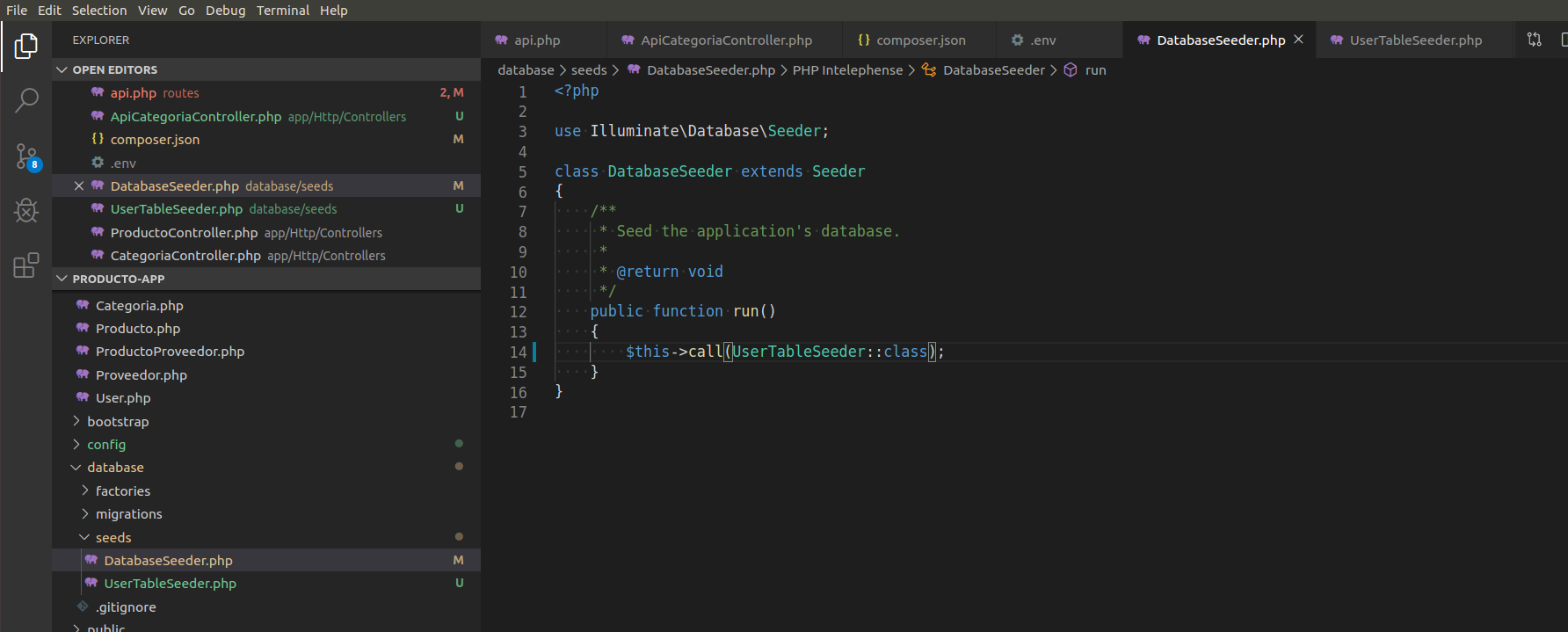
y verificar archivo .env (al ultimo se genera una llave)

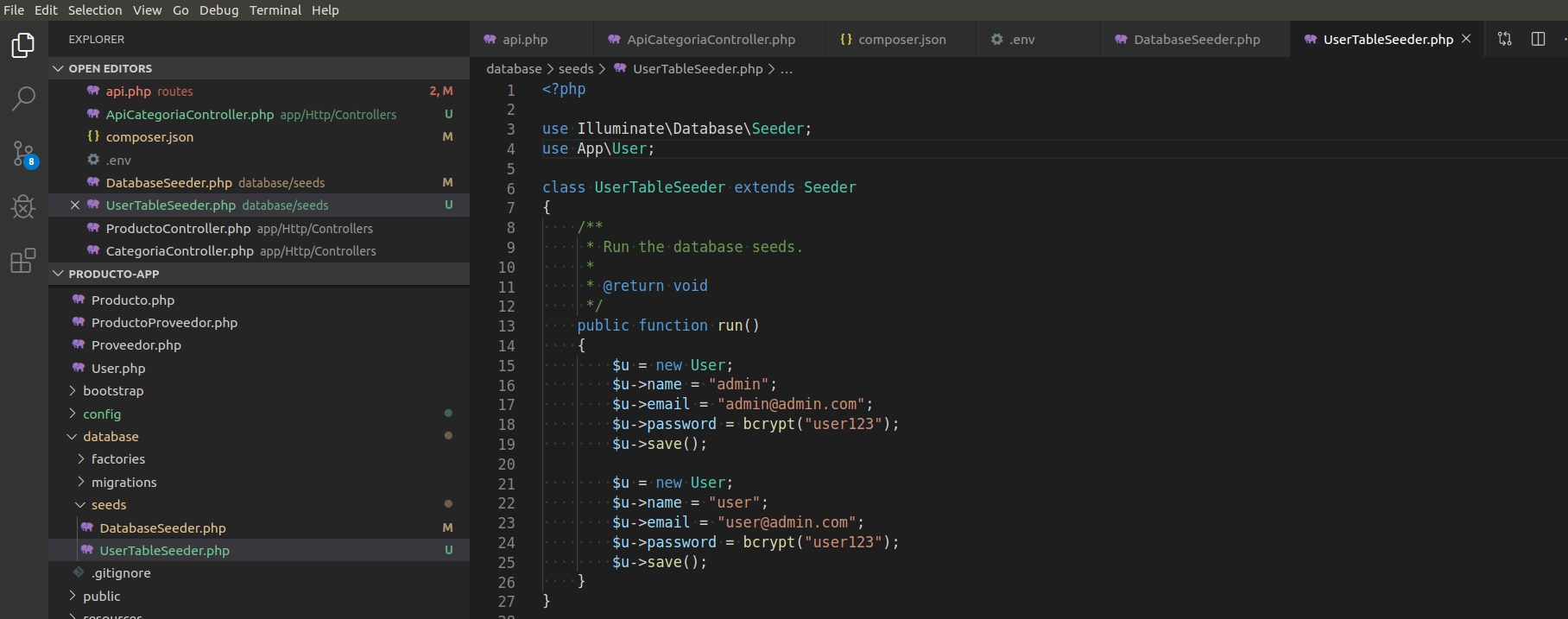
Ahora generamos un seeder

creamos un UsersTableSeeder

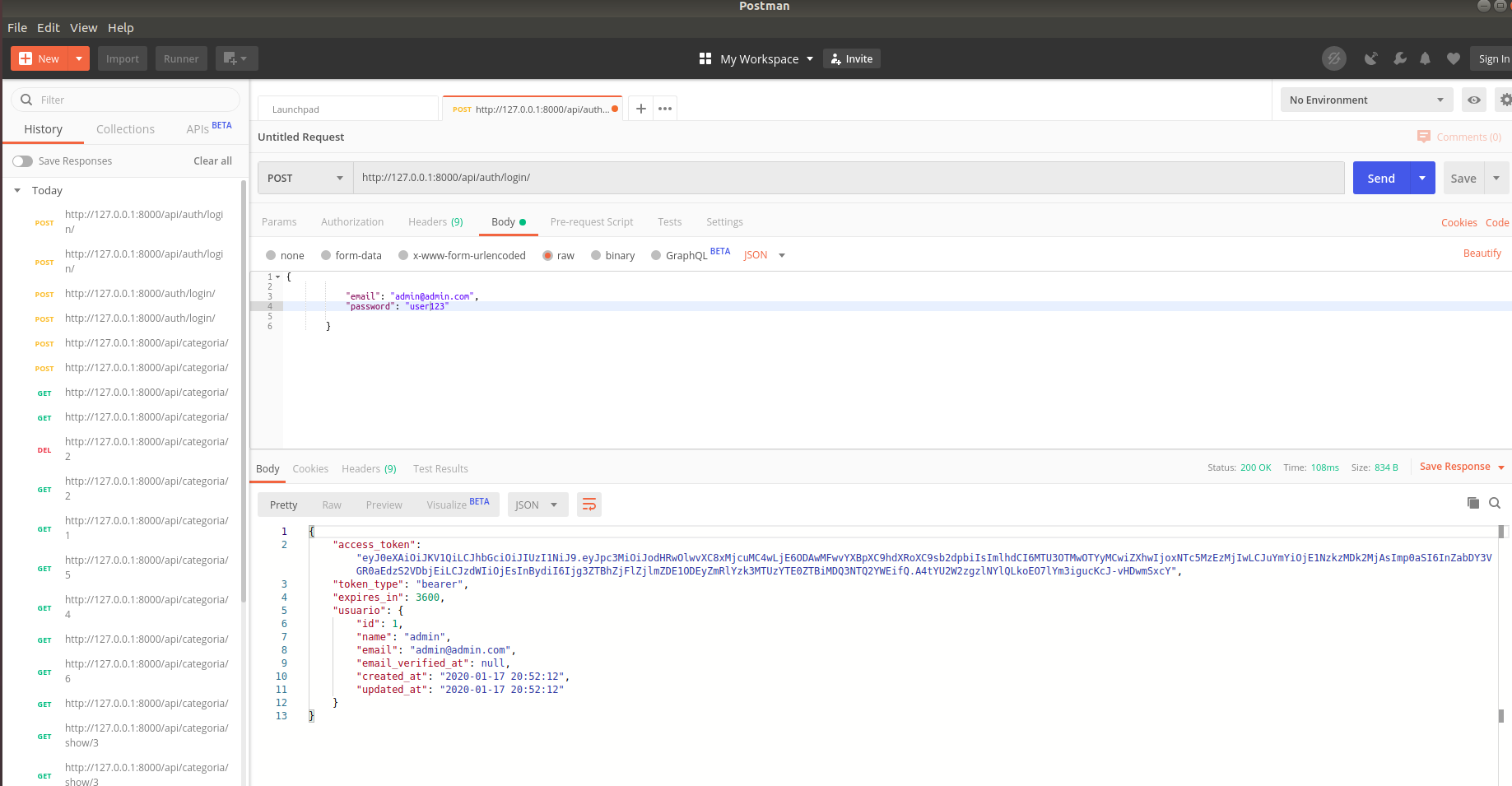
php artisan make:seed UserTableSeeder

primero





php artisan migrate --seed



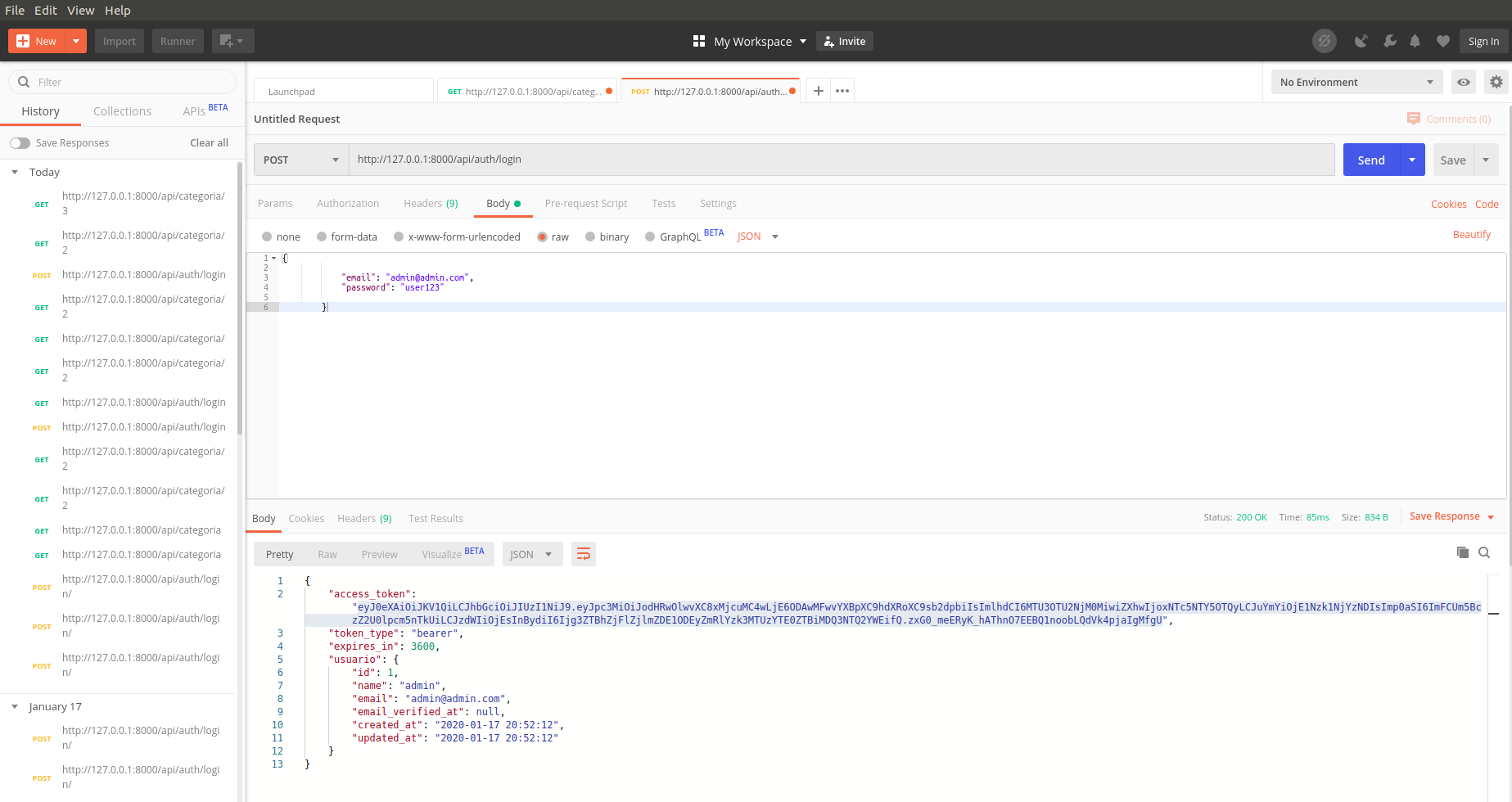
<https://jwt-auth.readthedocs.io/en/develop/quick-start/>

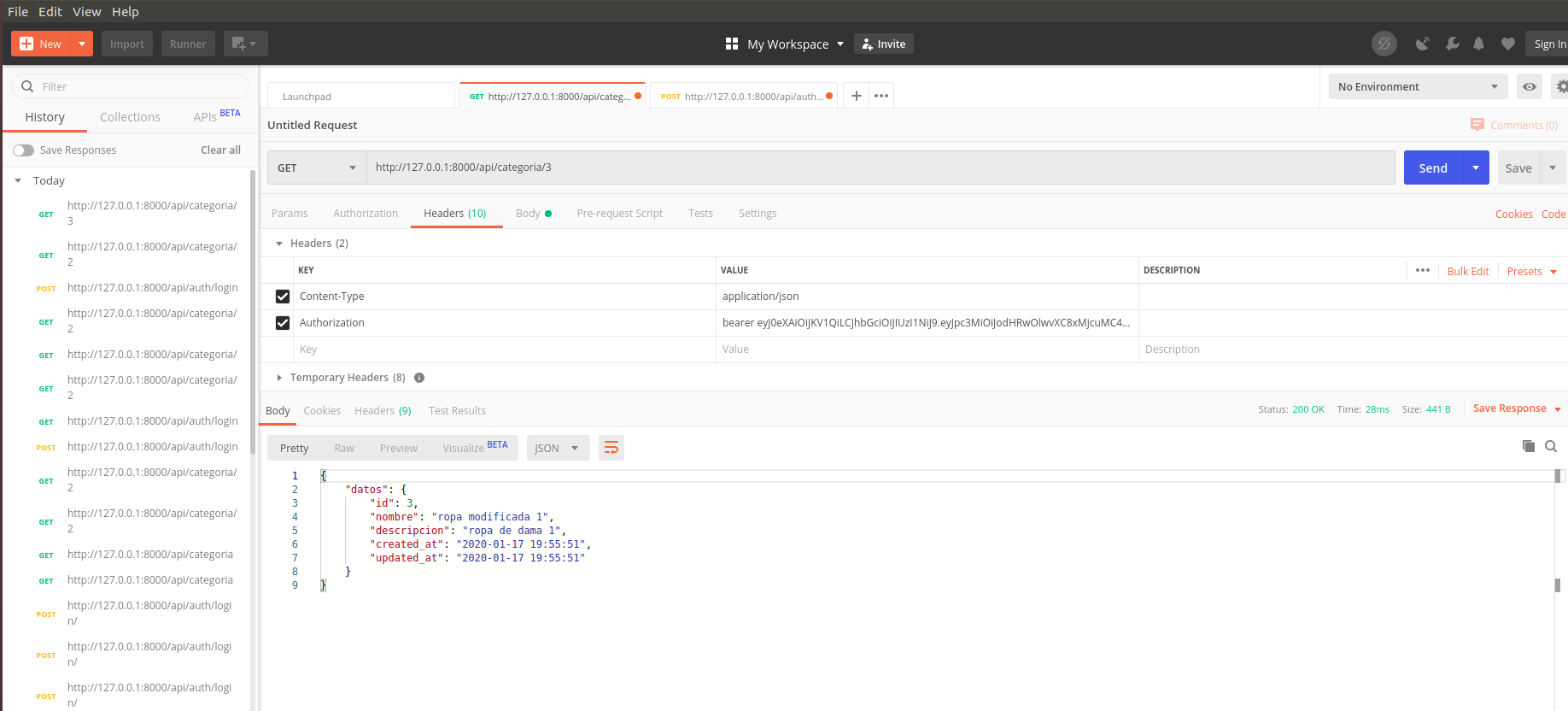
**Lunes 20/1/20**

Vamos a protegefr el ingreso con middleware, se protegera las URL (solo verificara, como condicion if)

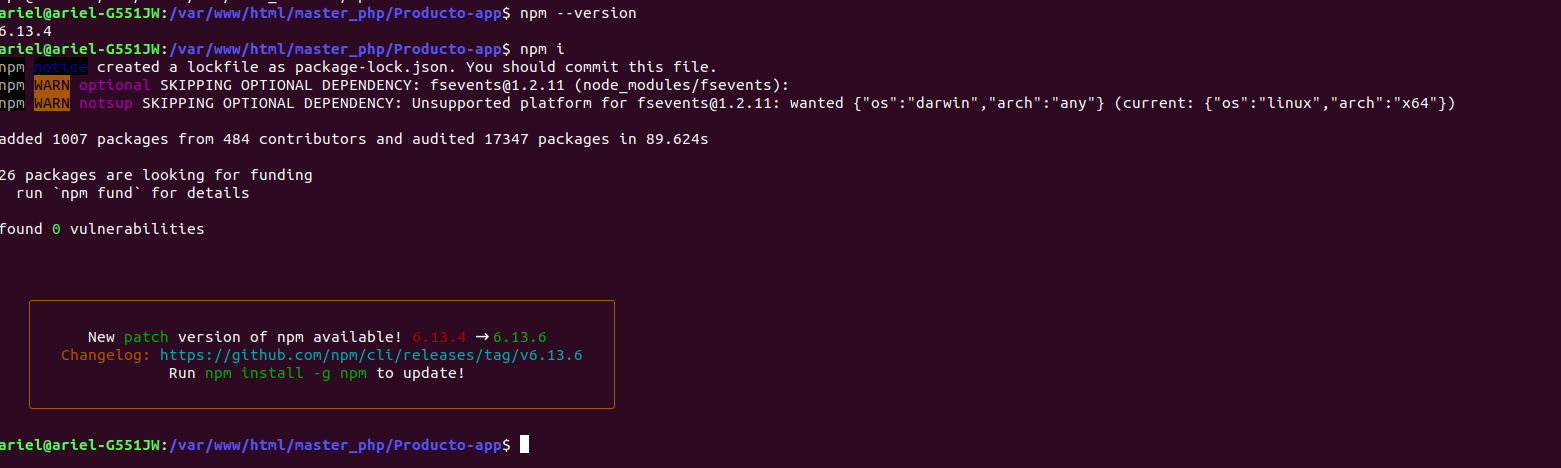
3 niveles de proteccion -> vista, controlador, rutas

a nivel de rutas es mas seguro





Pruebas con Node

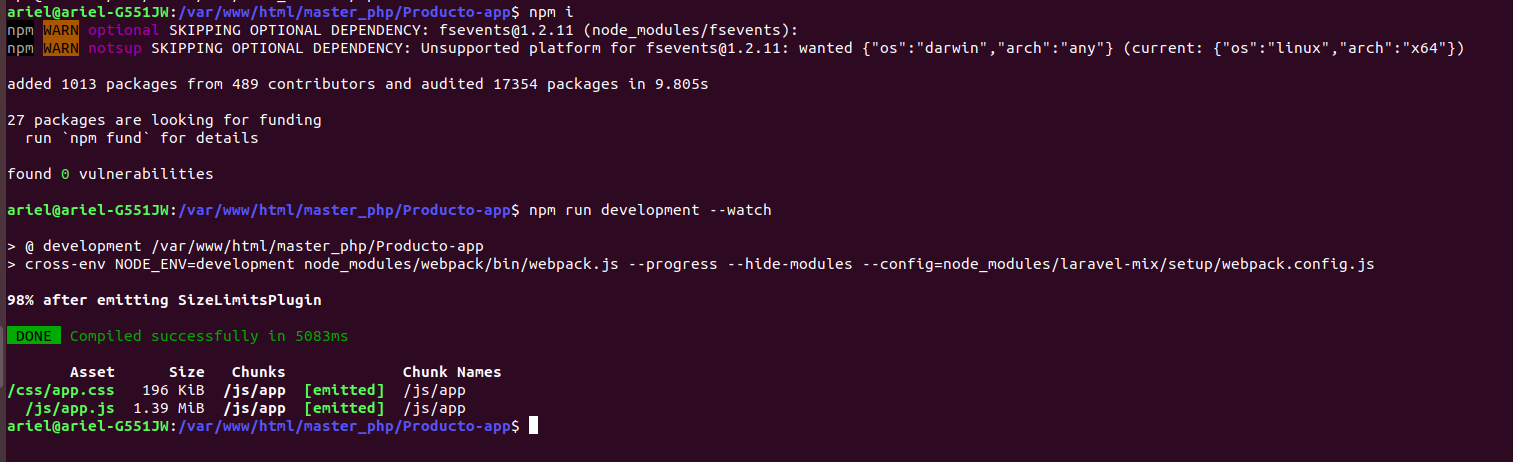


<https://laravel.com/docs/6.x/authentication>

composer require laravel/ui --dev

php artisan ui vue --auth





\*\*\* para probar con API cambiar:

config/auth

linea 17: de web a API

**Martes 21/1/20**

Instalar Angular

<https://cli.angular.io/>

webs para programar y cursos:

<https://stackblitz.com/>

<https://scrimba.com/>

Componente, compuesto por 3 archivos

.css (depende de lo seleccionado - css, stylus, saas, etc)

.html

.typescript - genera un selector <prueba></prueba> -> para llamar

../src/app -> ya es un componente

componenet padre es app y tiene:

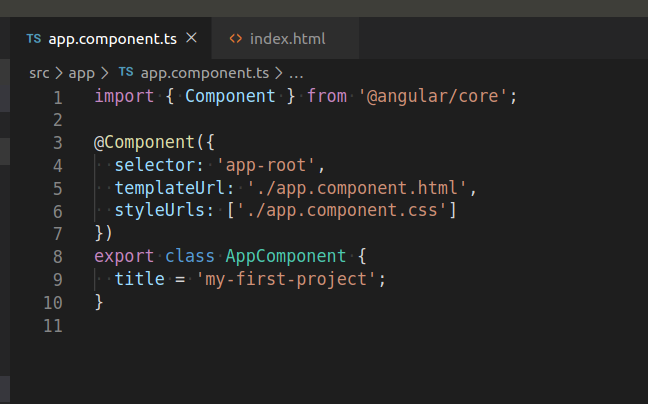
app.component.css

app.component.html

app .component.ts

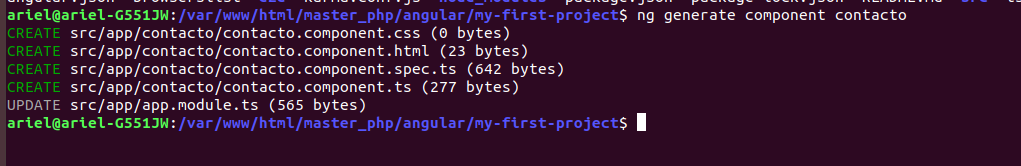
app.module.ts (principal, administrador del proyecto)

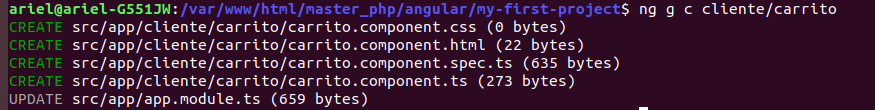
app-routing.module.ts (rutas)



ng serve -o (para iniciar proyecto)

ng generate component contacto





**Jueves 23/1/20**

\*\*probar npm i para esto del github

Lista de Productos

| id | nombre | precio | cantidad | acciones | subtotal | cantidad | acciones |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  | eliminar - add |

Carrito

| id | nombre | cantida | precio | subtotal |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  | total |  |

<https://angular.io/guide/reactive-forms>

**Viernes 24/1/20**

Generar un servicio

ng g s cliente/publicaciones/publicaciones

<https://socket.io/>

Tarea, buscar otro consumo de apis en web y hacer lo mismo

**Lunes 27/1/20**

1. Generamos el modulo admin (crea una carpeta)

ng g m admin

2. Generamos el componente

ng g c admin

3. generamos routes

ng g m admin/admin-routing --flat (--flat es para que no cree una carpeta sino un archivo)

despues habilitar rutas

4. Componentes

ng g c admin/categoria

ng g c admin/producto

ng g c admin/proveedor

ng g c admin/usuario

<https://material.angular.io/>

material es como bootstrap, tambien se puede utilizar bootstrap

ng add @angular/material

<https://material.angular.io/guide/theming#using-a-pre-built-theme>

ng g @angular/material:material-nav --name admin/componentes/navbar

ng g c admin/categoria/categoria-create

ng g c admin/categoria/categoria-index

ng g c admin/categoria/categoria-update

ng g c admin/producto/producto-create

ng g c admin/producto/producto-index

ng g c admin/producto/producto-update

ng g c admin/proveedor/proveedor-create

ng g c admin/proveedor/proveedor-index

ng g c admin/proveedor/proveedor-update

ng g c admin/usuario/usuario-create

ng g c admin/usuario/usuario-index

ng g c admin/usuario/usuario-update

**Martes 28/1/20**

<https://angular.io/guide/reactive-forms>

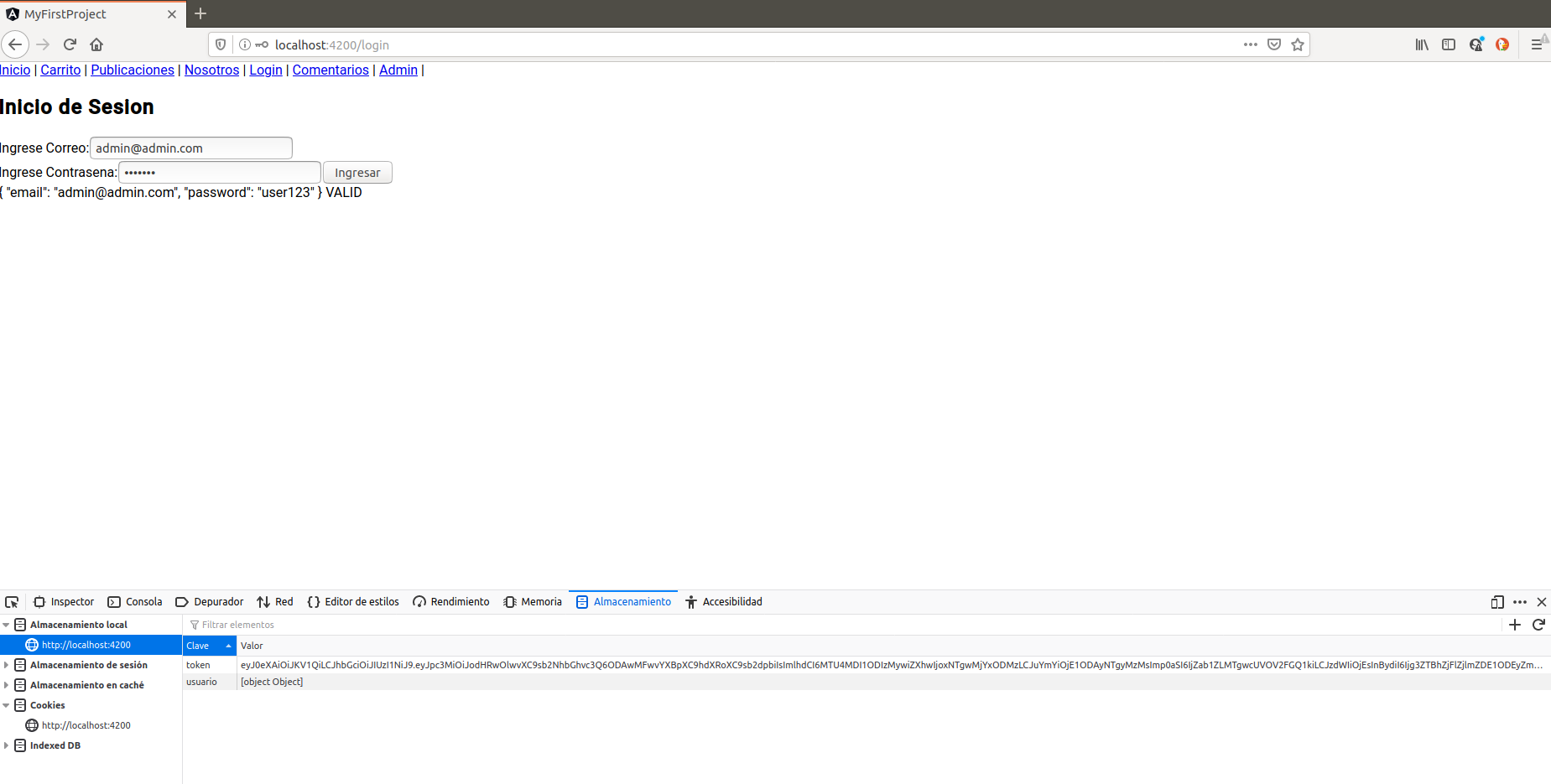
buscar validaciones

creamos servicio

ng g s cliente/login/login

primero el http client

<https://desarrolloweb.com/articulos/solucion-problemas-api-php-cors-post.html>



<https://developer.mozilla.org/es/docs/Web/API/WindowBase64/Base64_codificando_y_decodificando>

**Miercoles 29/1/20**

Creamos servicio

ng g s admin/categoria/categoria

**Jueves 30/1/20**

Guards

ng g guard guards/Auth

ng build --prod

**Viernes 28/8/20**

**Crear un controlador con funciones crud**

php artisan make:controller UsuarioController --resource

**su ruta:**



Crea todas las rutas:

