Instagram en Laravel

En este proyecto vamos a implementar un Instagram utilizando el framework de Laravel Los pasos para seguir son los siguientes

1. Diseño de la Base Datos y crear proyecto

2. Migraciones, modelos y seeders

3. Relaciones del modelo

4. Login y registro de usuario

5. Configuración de usuarios

6. Imágenes de la aplicación

7. Sistema de comentarios con edición y borrado de imágenes

8. Sistema de likes

9. Perfiles de usuarios

10. Gente y buscador

La **base para el proyecto** será **CRUD libros** de Javi.

0 PREPARACIÓN

# DOCKER

Usando el composer que nos cedió el profesor levantamos los 3 contenedores.

**Docker-compose up**

****

1 CREACIÓN DEL PROYECTO

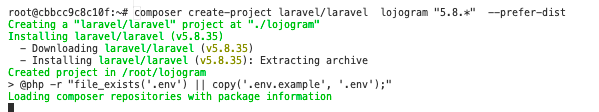
En /**root** se crea el prohecto

Entrar al contenedor del **servidor:**

**docker exec -it lojo-gram\_servidor\_1 ./bin/bash**



**composer create-project laravel/laravel lojogram “5.8.\*” –prefer-dist**

****

Se mueve el proyecto al directorio de **nginx**

**cp lojogram/\* /var/www/html**

**cp lojogram/.\* /var/www/html**

****

**Permisos** al **directorio**

**chown -R www-data:www-data /var/www/html**

****

2 MIGRACIONES, MODELOS, RELACIÓN DEL MODELO

# Conexión con la Base De Datos

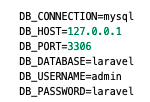
**Config/Database.php** 🡪 base de datos **mysql**

****

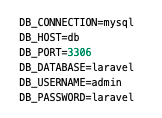
**/.env** 🡪 Colocar datos de **.yml**

Entorno debería estar así pero pasa algo extraño

Para hacer las **migraciones**:



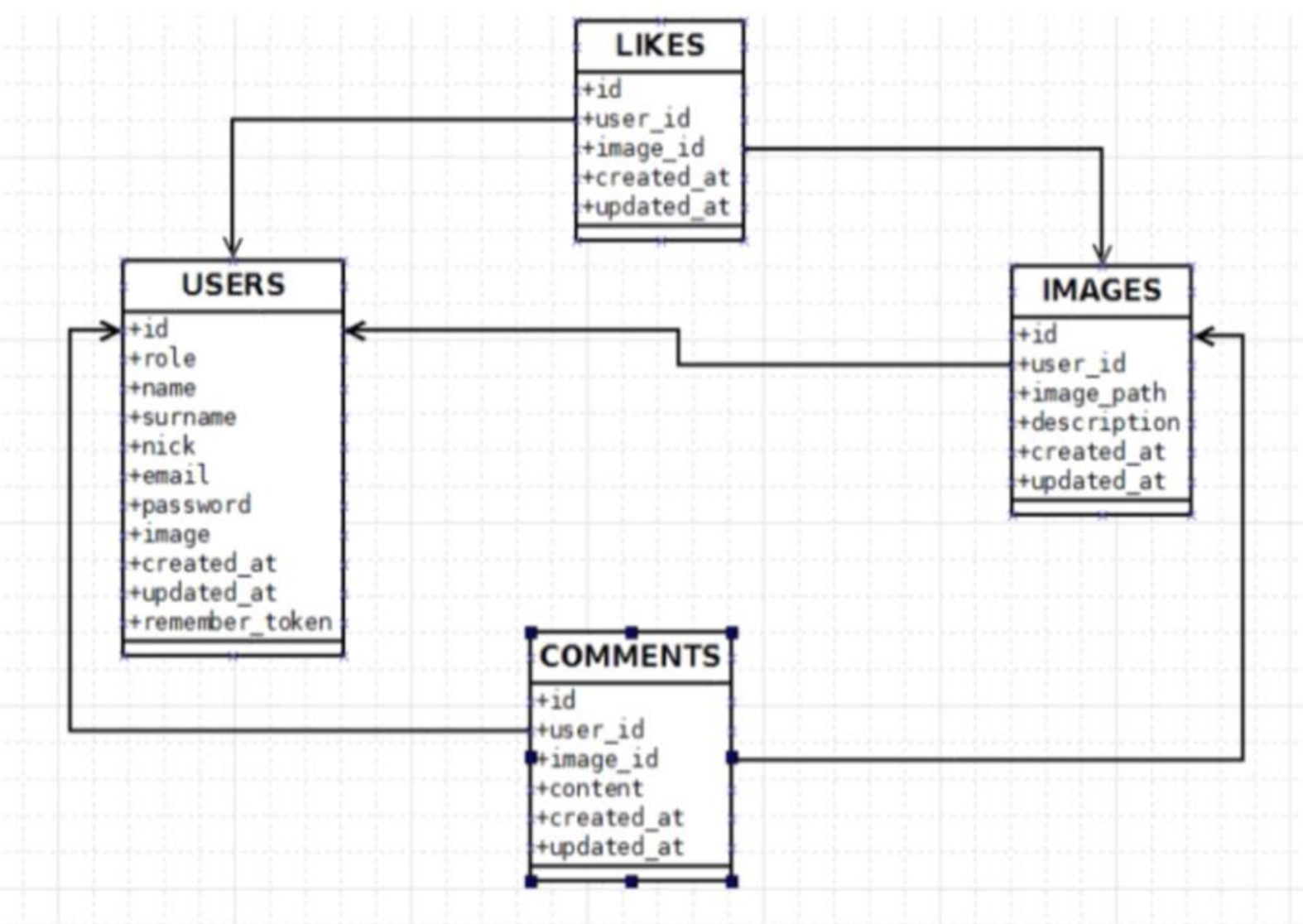
Para **el resto**



# Migraciones y Modelos

Aplicar:

Partimos con el modelo de User medio echo creo la estructura del resto.



Creo la estructura de las otras **tres tablas**

**Php artisan make:model -mcr images**

**Php artisan make:model -mcr comments**

**Php artisan make:model -mcr likes**

****

# MIGRACIÓN DE USER



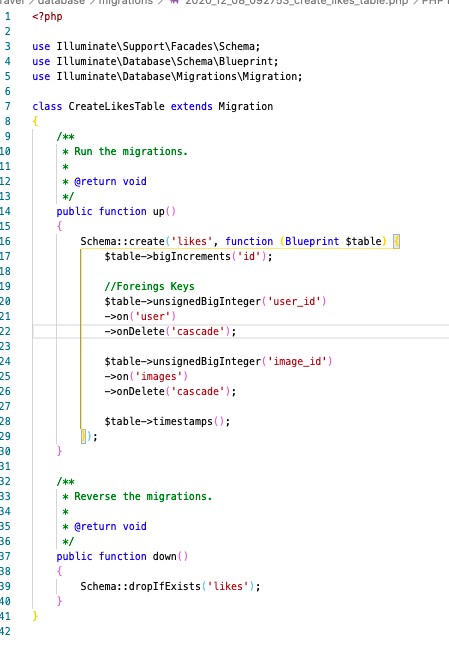
# MIGRACIÓN DE IMAGES



# MIGRACIÓN DE COMMENTS

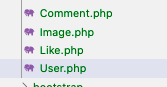


# MIGRACIÓN DE LIKES



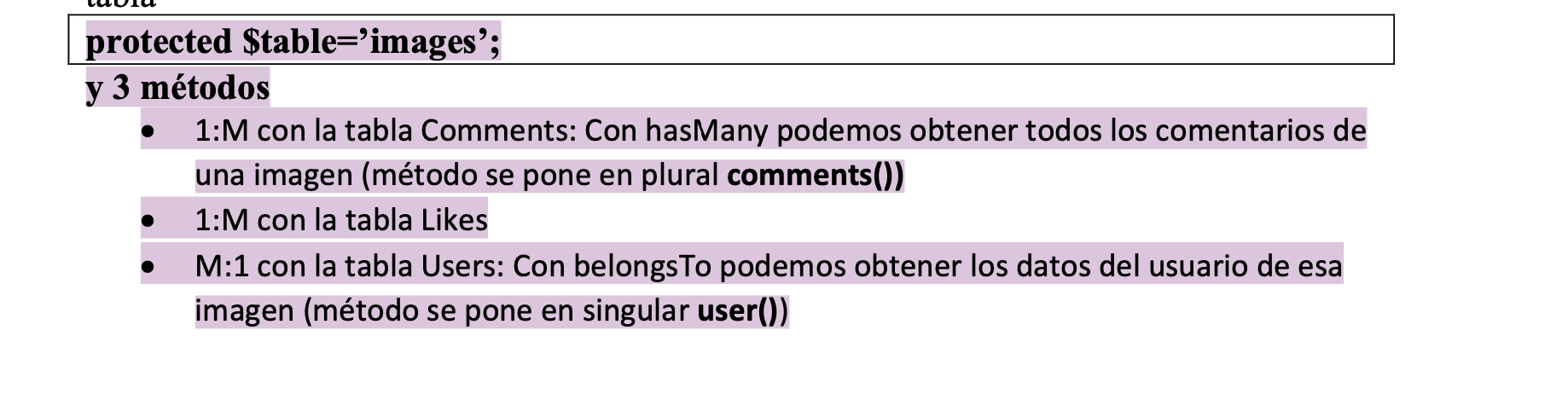
# MODELOS

Cambiar el nombre de los modelos a singular y en mayúsculas



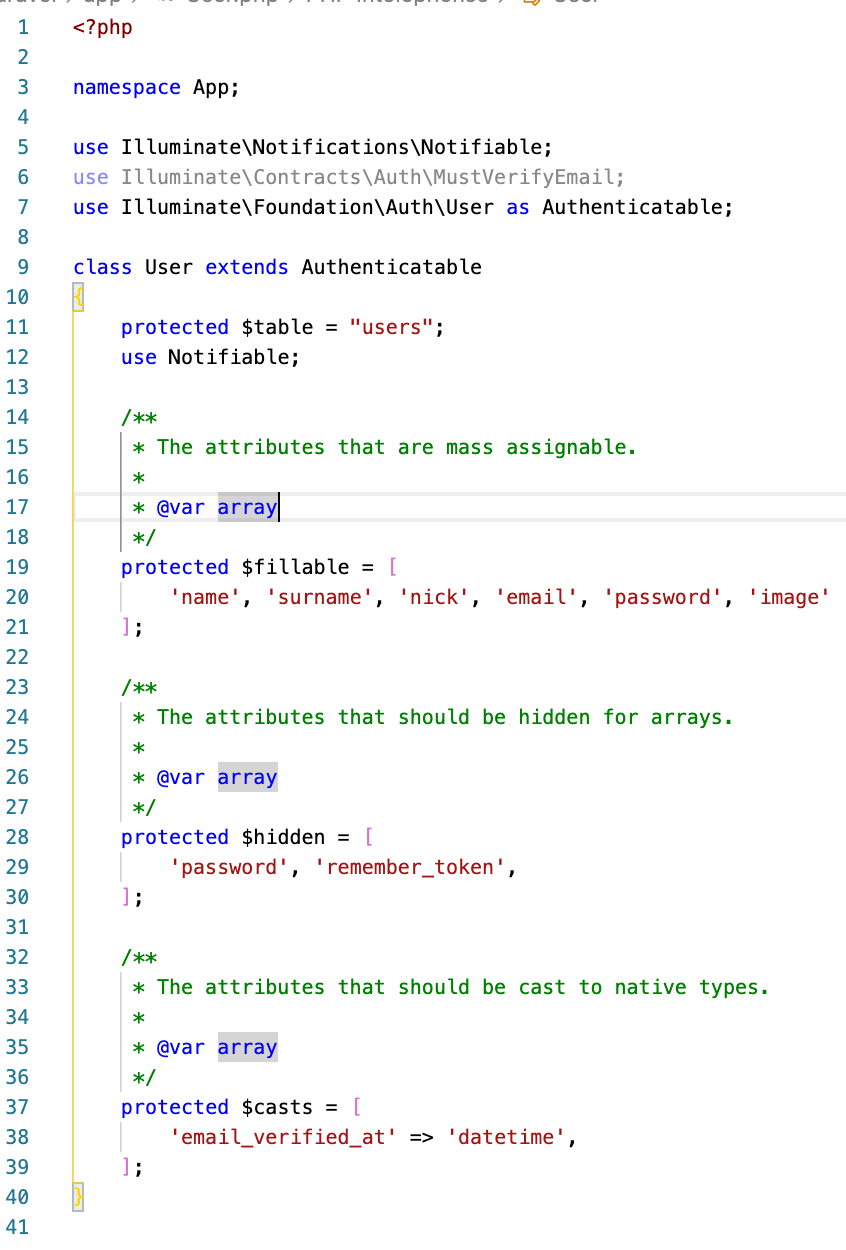
A continuación hay que preparar los modelos con **los campos que se van a poder modificar** y **las relaciones de los modelos**

**EJEMPLOS:**

****



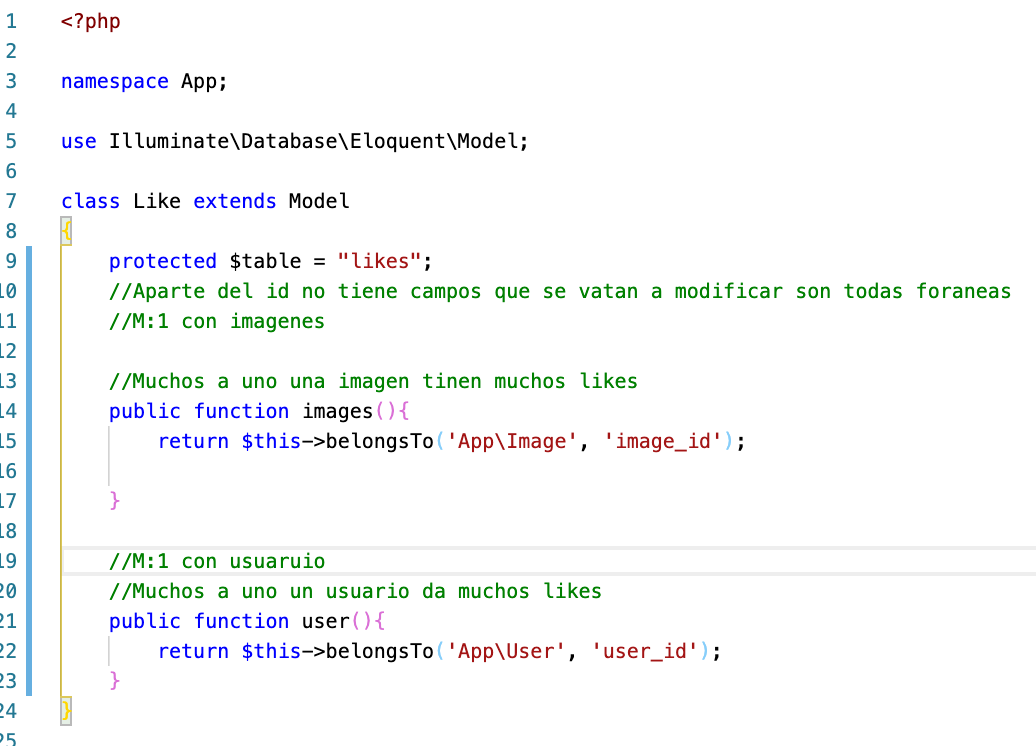
# MODELO USER



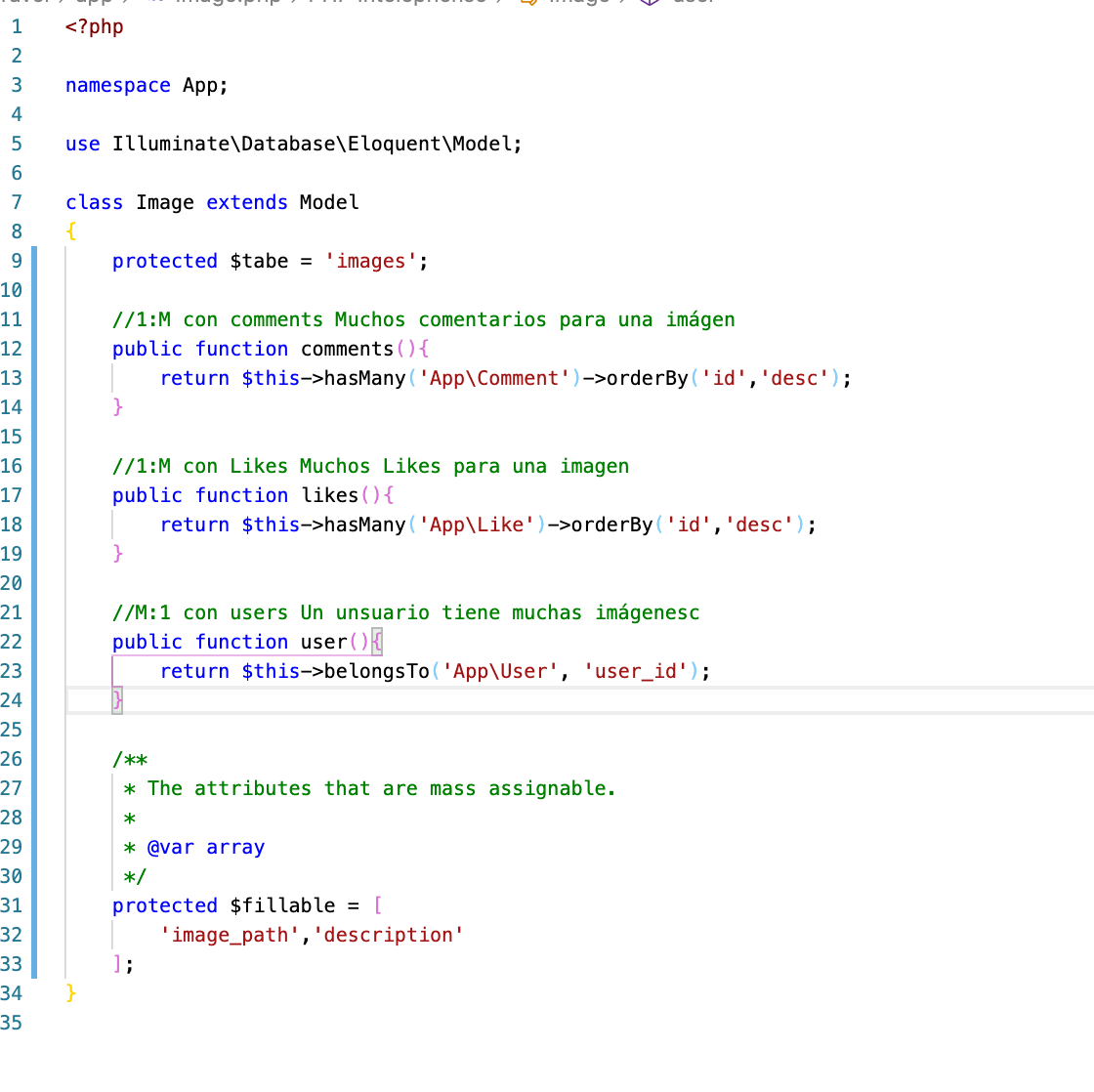
# MODELO COMMENT



# MODELO LIKE

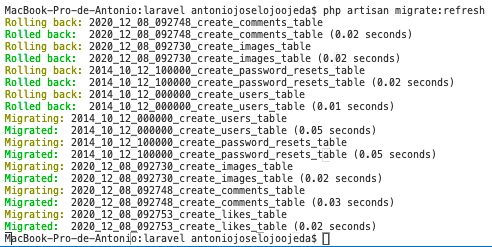


# MODELO IMAGE



# EJECUTAR MIGRACIONES

**Php artisan migrate:refresh** (refresh hace **rollback** de las que están ya creadas)



# EJECUTAR SEEDERS

En **/Database/sedes/DatabaseSeeder.php**

Dentro del método **run()** se colocan los **seeders**

**(de momento solo en users)**

****

Ejecutar Seeder

**php artisan db:seed**



LOGING Y REGISTRO DE USUARIO

LOGING Y REGISTRO

**Php artisan make:auth** 🡪 Configura las ritas y vistas necesarias para el inicio de esesión y restablecer contraseñas.



# Controladores y rutas:

LogingController 🡪 **app/Http/Controllers/Auth**

**GET /loging 🡪** Muestra el formulario de loging

**POST/loging 🡪** Se envía el formulario de loging

**POST/logout 🡪** Cerrar session

RegisterController 🡪 **app/Http/Controllers/Auth**

**GET/register 🡪** Muestra el formulario de registro

**POST/register 🡪** Se envía el formulario de registro

HomeController 🡪 **app/http/Controllers/HomeController.php**

**/Home 🡪** Solo pueden acceder usuarios registrados

# Vistas:

Dentro del directorio normal **resources/views/auth**

**Loging.blade.php**

**Regoster.blade.php**

**resources/views/loyout/**

**app.blade.php**

**resources/views**

[home.blade.php](http://home.blade.php) 🡪 Vista de la ruta /Home

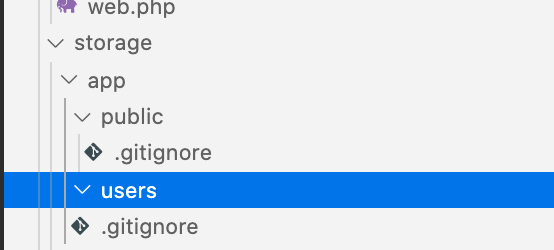
**Imágenes**

Las imágenes se almacenarán en el directorio **app/storage** *(que se* ***linkea internamente*** *con el directorio* ***public****).*

Vamos a crear **dos discos virtuales** que serán dos directorios en el directorio **app/storage**.

**Storage/app/users 🡪Imagen de perfil & Imágenes de los usuarios**

**Storage/app/public 🡪 Imágenes Generales de la aplicación**

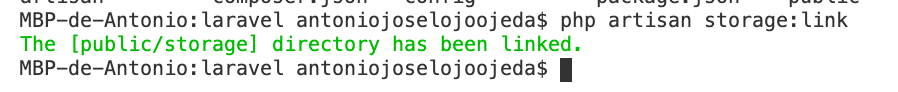


Añadir los **discos virtuales en /config/filesystems.php**

****

Para activar el **link** entre **storage** y **public** Usamos el siguiente comando.

**php artisan storage:link**



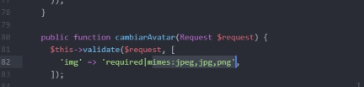
**Cometí un error en la construcción de la tabla de usuarios en el tipo profile\_image**

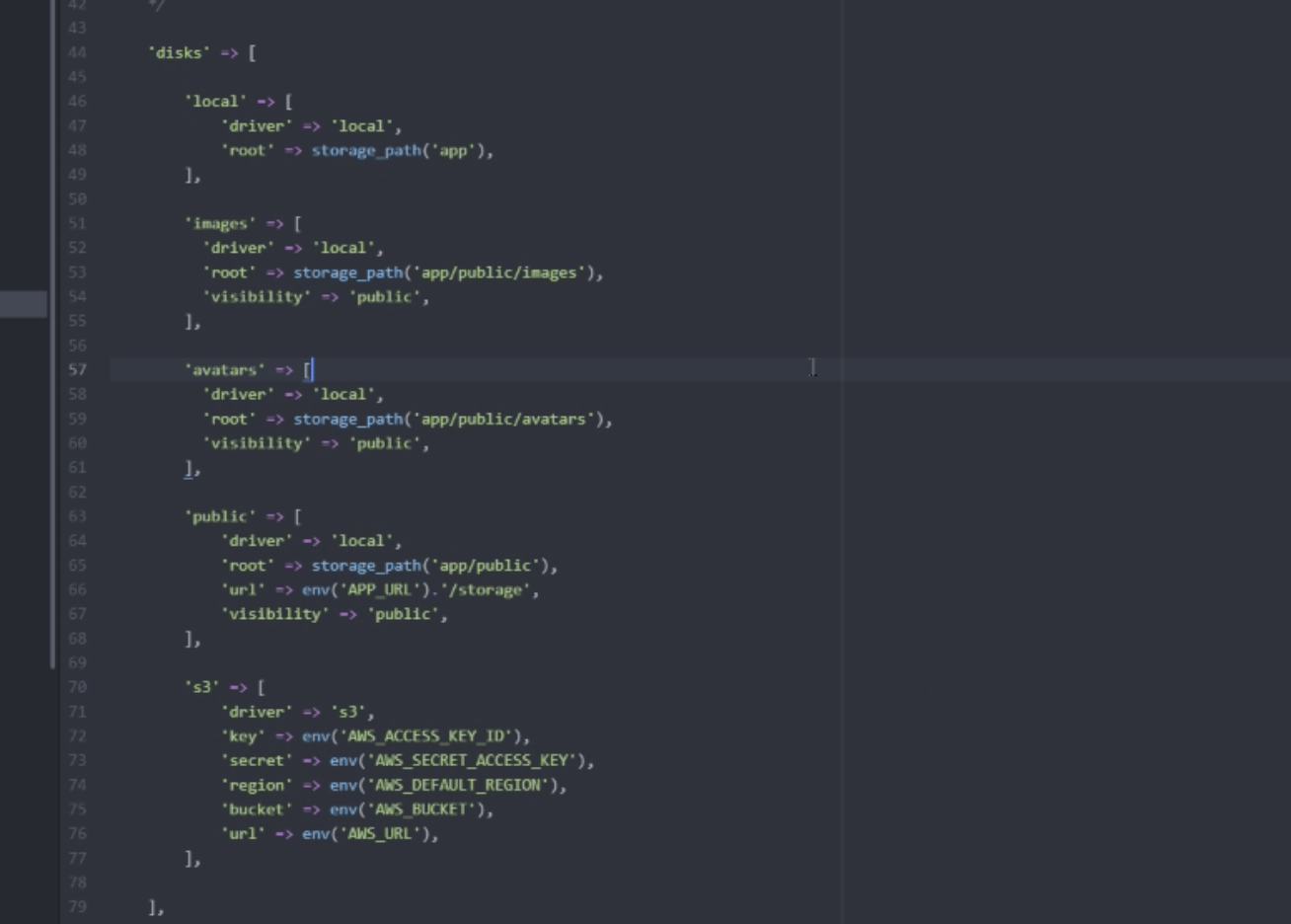
Lo cambiaré a **String** 🡪 Este **String.** Almacenará la ruta de las imágenes

**Nueva migración**

****

A continuación modifico el **controlador del usuario UserController@config** Para incluir la inserción de las imágenes.





TODOS DOCU PERFIL E IMÁGENES SI DIERA TIEMPO

COMPROBAR IMAGEN DE USUARIO EN LA BASE DE DATOS TRAS LA ACTUALIZACIÓN HECHO!!!

CIFRAR CONRTTASEÁ EN EL REGISTROP AL ALMACENAR

Ya tenemos en el directorio dos directorios **db-init** y **laravel**.

Además de tres contenedores funcionando. (bd, phpmyadmin, servidor)



Para crear el proyecto en el directorio del **root**

(el nombre se mira con **docker ps**)

**docker exec -it instagram\_servidor\_1 /bin/bash 🡪** Ejecuta el bash del servidor

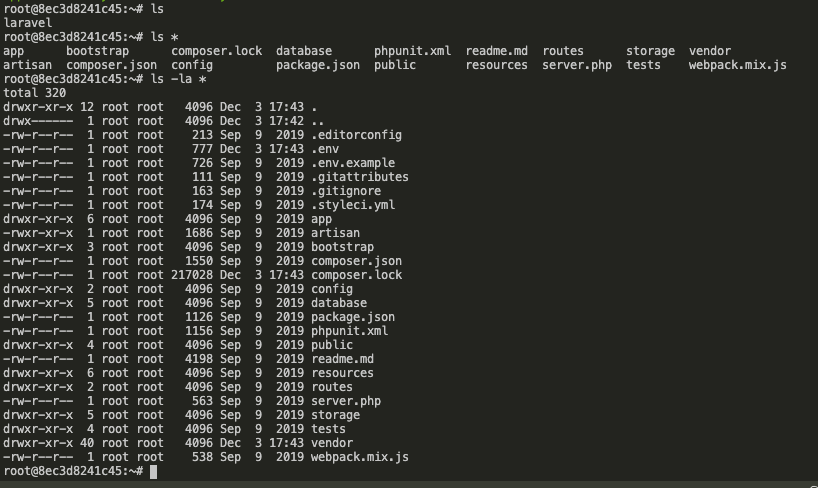
**3º CREAR EL PROYECTO**

Vamos a **root** y desde aquí **ejecuto el comando para crear el proyecto.**

**composer create-project laravel/laravel laravel "5.8.\*" --prefer-dist**

(Crea el Proyecto laravel/laravel en /laravel con la version 5.8.lo que sea)

****



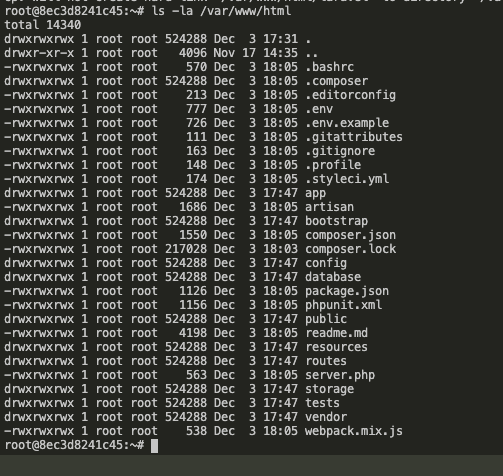
**4º DESPLEGAR EL PROYECTO**

A continuación, copiamos todo al directorio donde publica **nginx** y ya estaría laravel funcionando.

**cp -r laravel/\* /var/www/html**

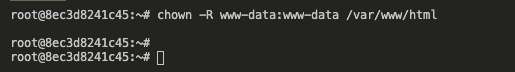
**cp -R laravel/.\* /var/www/html**

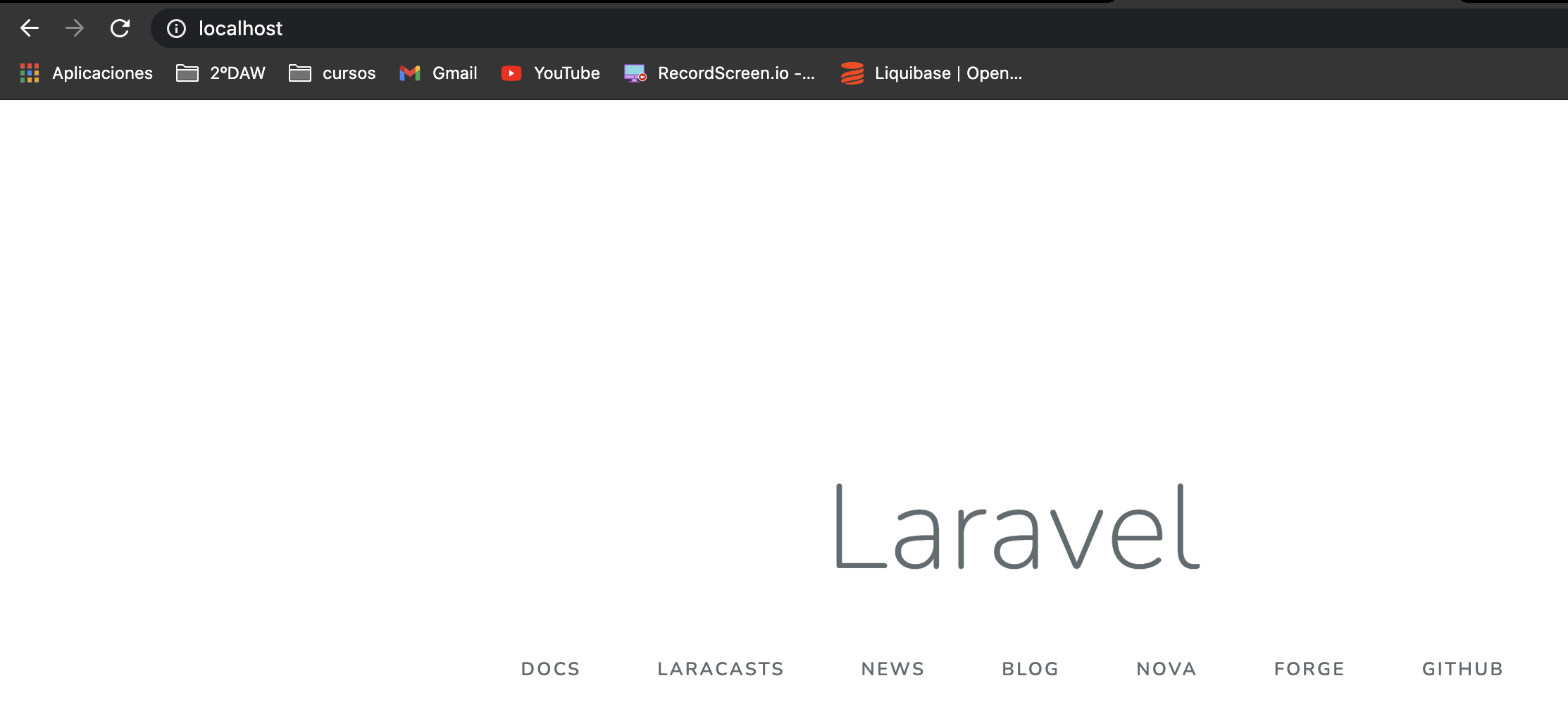
****



A plicamos permisos para poder trabajar con **nginx**.

**chown -R www-data:www-data /var/www/html**

****

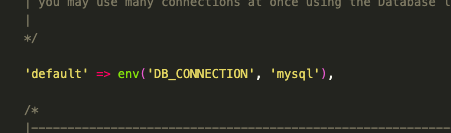
****

1 DISEÑO DE BASE DE DATOS Y CREAR PROYECTO

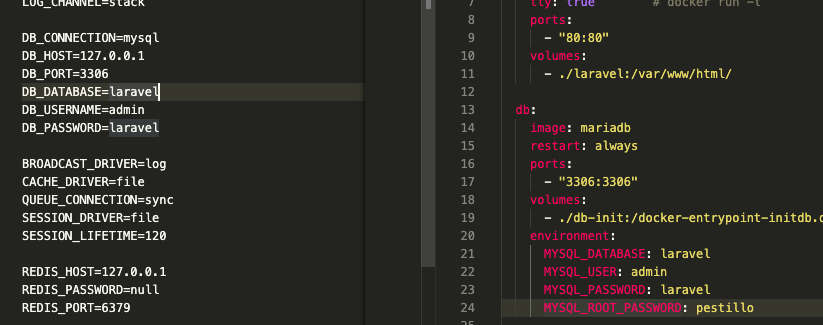
**0 CONEXIÓN CON MYSQL (MIGRATE)**

Ya podemos empezar con VSCode u otro IDE. Comenzamos comprobando que estamos usando **mysql**

**Config/Database.php**

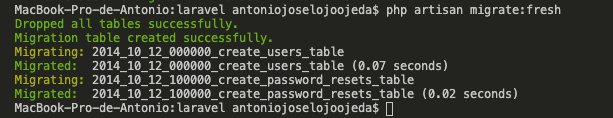
****

Comprobamos que en **.env** usamos la misma configuración que en **docker-compose.yml**

****

**Hacemos la migración.**

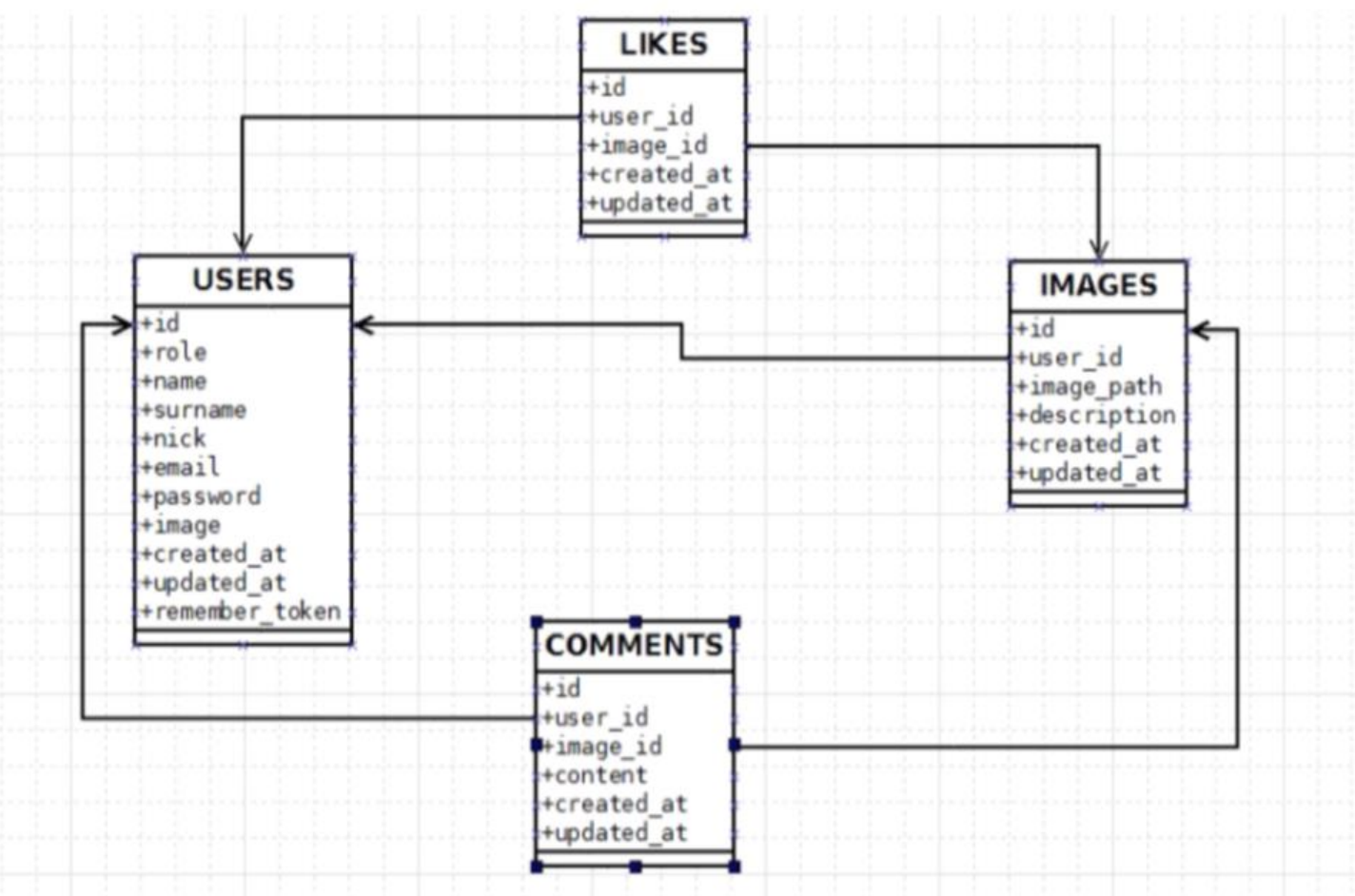
**php artisan migrate:fresh (limpia artisan)**

****

Estos están en **Database/migrations**

****

**Creamos el modelo con ORM**

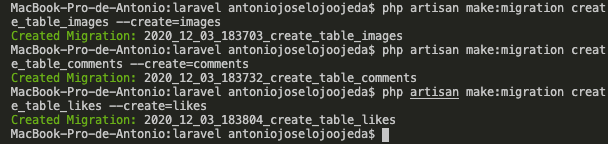
****

En los likes no veo la necesidad de registrar fechas de creación y modificiación. Las obviare.

Usamos el siguiente comando para añadir el resto de modelos.

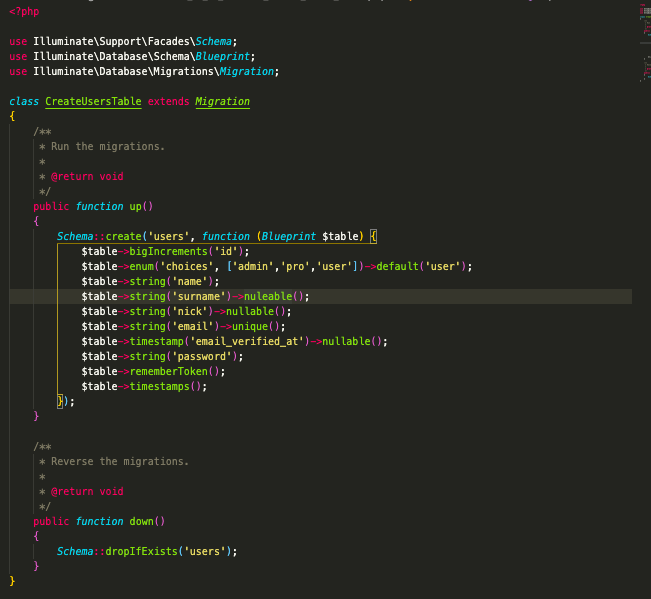
**php artisan make:migration** [crear\_tabla\_nombre] **–create=**[nombre]

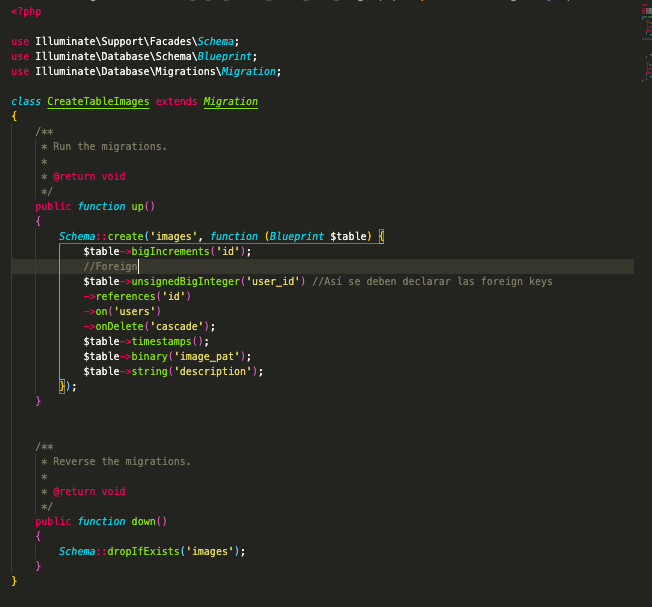
Creo los archivos para los modelos de **images comments y likes**

****

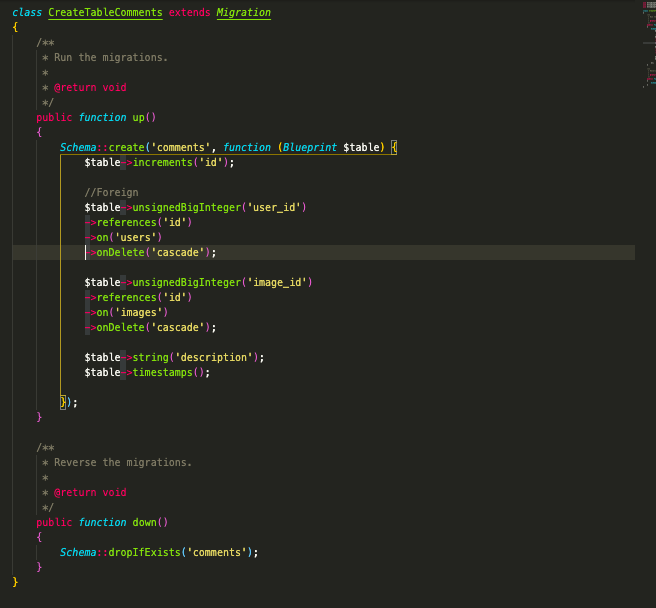
A continuación **definir modelos** [**https://laravel.com/docs/5.0/schema#adding-columns**](https://laravel.com/docs/5.0/schema#adding-columns)

**Users**

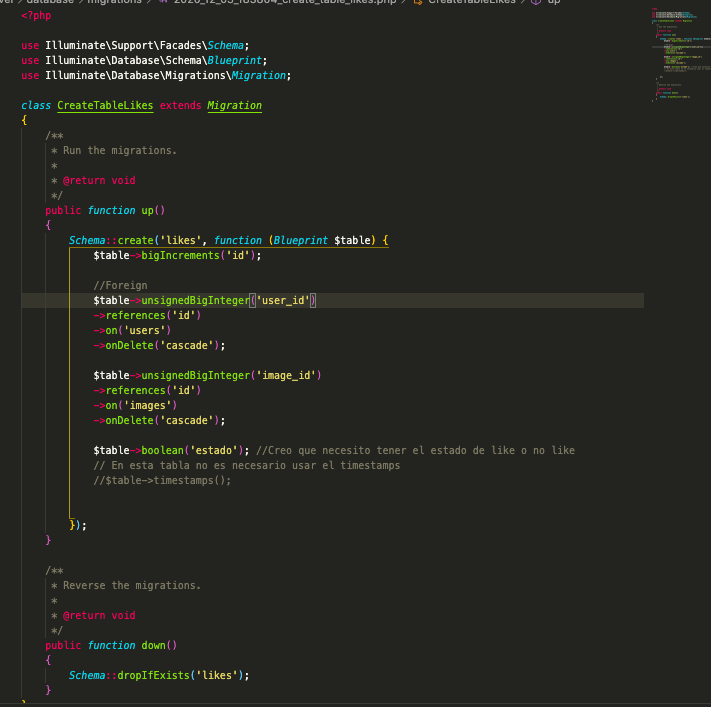


**images**

**comments**

****

**likes**

****

2 MIGRACIONES, MODELOS Y SEEDER

El modelo de User ya está creado solo falta modificarlo para añadirle los campos que falta

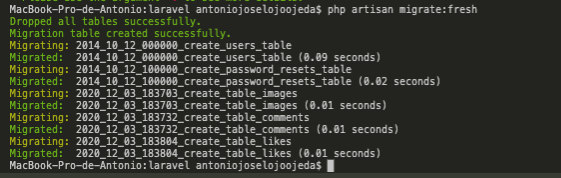
**Asegúrate que todas las claves primarias de las 4 tablas sea increments.**

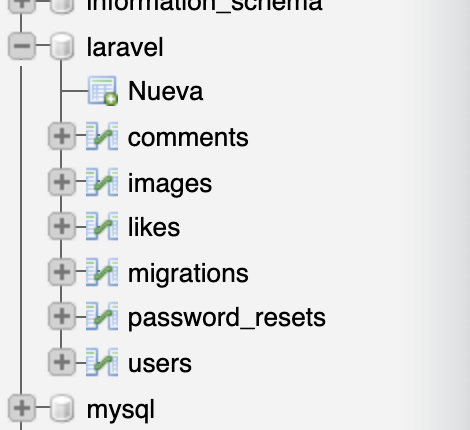
**o Las claves foráneas sean Integer sin tamaño y unsigned(). n.**

**2.1 HACER EL MIGRATE**

Ejecutamos **migrate para aplicar las nuevas tablas**

**Php artisan migrate:fresh**

****

****

Ya tenemos nuestras tablas creadas en la base de datos.

A continuación le toca el punto a los **seeders** o semillas **Estos se encuentran Database/sedes,** y los **modelos**

Para hacer la inserción en nuestra bd ejecutamos **php artisan make:seed instragramSeeders**

Dentro de estos en la función **run** se introducen los datos a insertar

Los modelos se crean ejecutando : **php artisan make:model “**nombre del modelo**”**

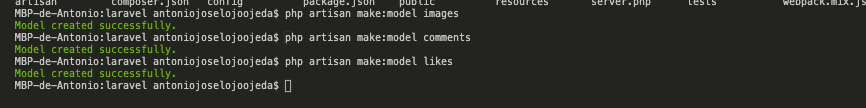
Los modelos son los objetos que vamos ha ir llamando en la aplicación y que se generan a partir de las tablas ejecutadas en las migraciones.

**POR SEGUIR UN POCO LA LÓGICA CREARÉ ANTES LOS MODELOS DE LAS TABLAS Y DESPUÉS SE IRÁN CREANDO LOS SEEDERS (SEMILLAS SEEGÚN SE VAYAN NECESITANDO).**

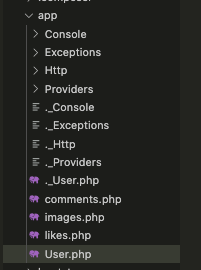
**2.2 CREAR Y MODIFICAR LOS MODELOS**

Se crean solo los modelos de **images comments & likes** ya que el de **users** ya está creado.

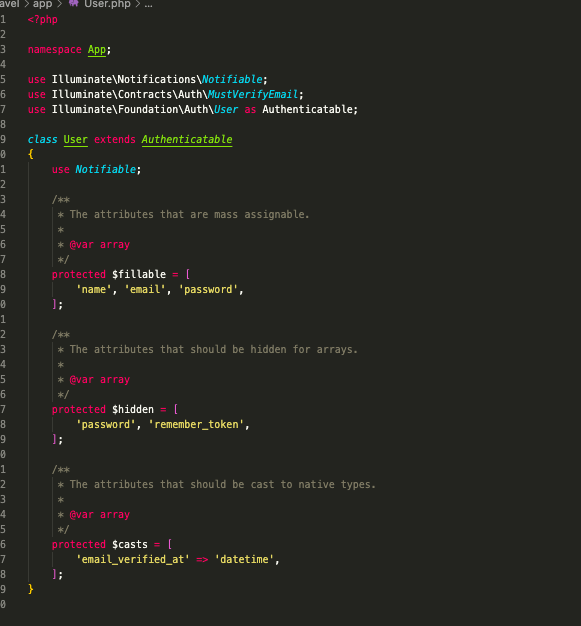
**Php artisan make:model** [nombre del modelo]



Estos se generan en el directorio **/app**

****

Tomando de **ejempo Users** creamos el resto



LOGING Y REGISTRO

**Php artisan make:auth** 🡪 Configura las ritas y vistas necesarias para el inicio de esesión y restablecer contraseñas.

# Controladores y rutas:

LogingController 🡪 **app/Http/Controllers/Auth**

**GET /loging 🡪** Muestra el formulario de loging

**POST/loging 🡪** Se envía el formulario de loging

**POSRT/logout 🡪** Cerrar session

RegisterController 🡪 **app/Http/Controllers/Auth**

**GET/register 🡪** Muestra el formulario de registro

**POST/register 🡪** Se envía el formulario de registro

HomeController 🡪 **app/http/Controllers/HomeController.php**

**/Home 🡪** Solo pueden acceder usuarios registrados

# Vistas:

Dentro del directorio normal **resources/views/auth**

**Loging.blade.php**

**Regoster.blade.php**

**resources/views/loyout/**

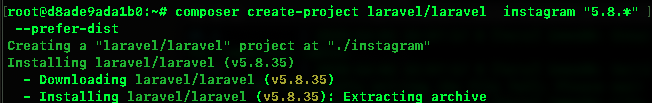
**app.blade.php**

**resources/views**

[home.blade.php](http://home.blade.php) 🡪 Vista de la ruta /Home

**🡪** Crea el Proyecto





A continuación **copiar los archivos a la carpeta donde publica Nginx**

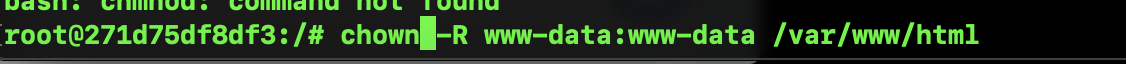
**cp -r instagram/\* /var/www/html/**

**cp -r instagram/.\* /var/www/html/**

****

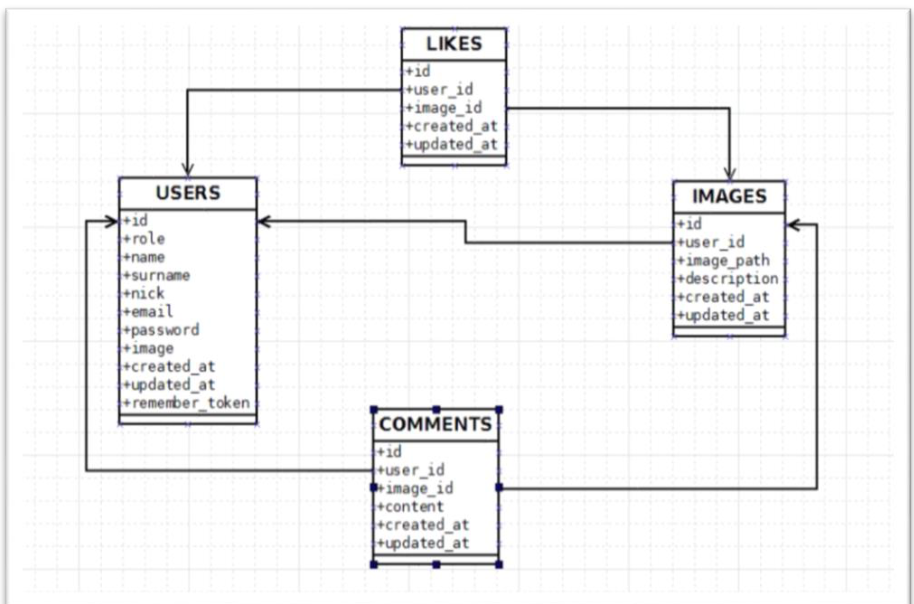
Copiamos los permisos de los archivos de **Nginx para manejarlos**

**chown -R www-data:www-data /var/www/html**

****

Si entramos **a localhost** vemos que está funcionando laravel. **Siempre que se desee cargar docker-compose up**

1 DISEÑO DE LA BASE DE DATOS:

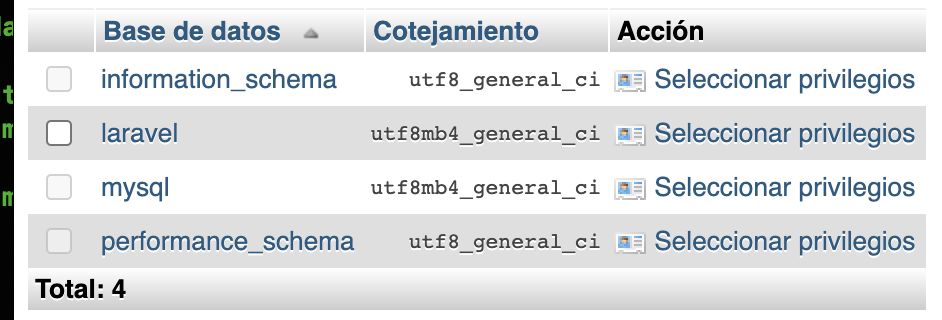
****

**CREAR LA BASE DE DATOS INSTAGRAM:**

**create database Instagram ;**

Desde el navegador entro en <http://localchost:8080>

Database: **db;** User: **root** Password: **pestillo**

****

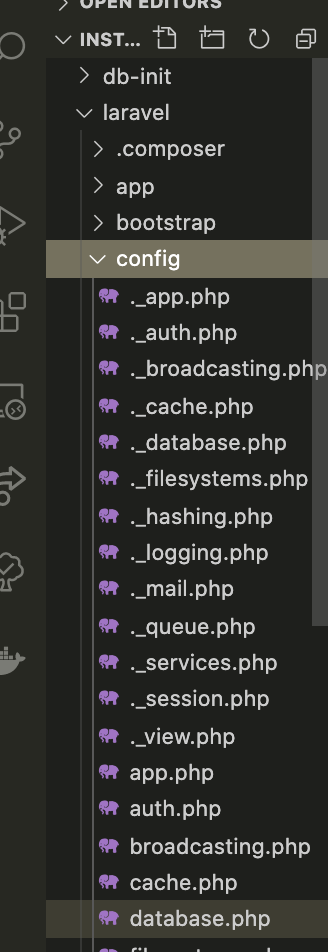
Creo la base de datos:



Realizamos la conexión con la BD .env

**REALIZAR LA CONEXIÓN CON LA BD .env**

Vamos al archivo **database.php** dentro del subdirectorio **config**

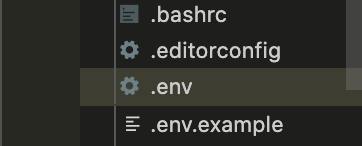
****

En el archivo comprobar la siguiente línea:

**'default' => env('DB\_CONNECTION', 'mysql')**



Una vez verificado que vamos a usar **mysql** vamos al archivo **.env** ubicado en la raíz delo proyecto para configurar **host, bd, nombre, contraseña.**



****

A continuación hacemos la migración de la base de datos **Instagram**

**Php artisan migrate**

2 MIGRACIONES, MODELOS Y SEEDERSC