Creacion de API Pelicula

Creamos el proyecto con el siguiente comando



Lo arrancamos



Le instalamos las siguientes modulos con Composer



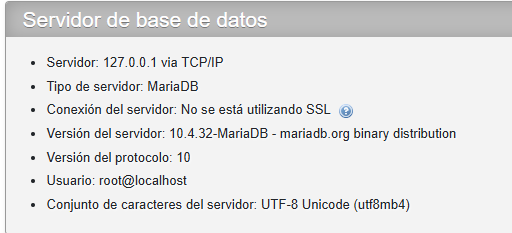




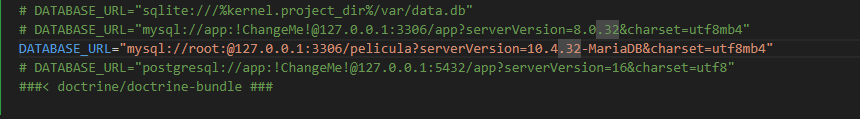
Arrancamos dBeaver para crear la BBDD película



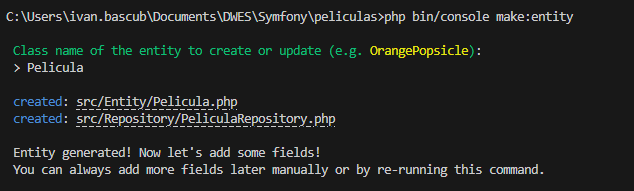
Comprobamos la version del servidor de BBDD con phpmyadmin



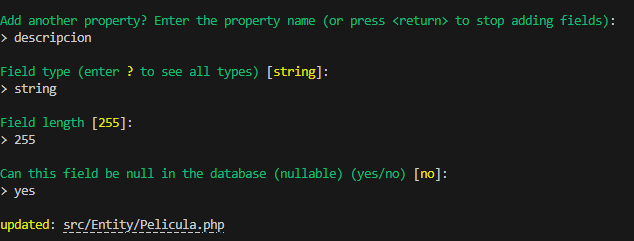
Nos dirigimos al archivo .env y modificamos la variable DATABASE\_URL



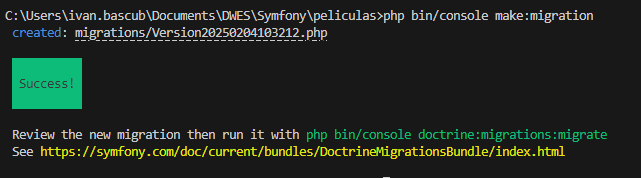
Ahora vamos a crear las entidades que serán las tablas en la BBDD con el siguiente comando



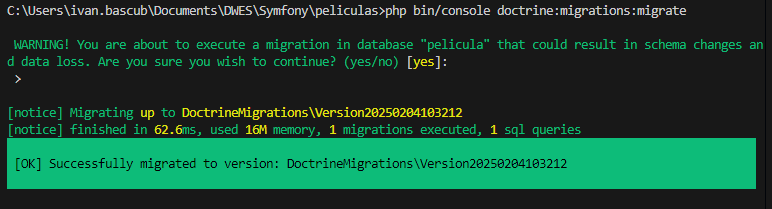
Con este comando también te permite añadir los atributos de la entidad



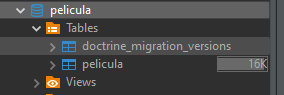
Con el siguiente comando creamos una copia de seguridad para volver en el caso de que nos equivoquemos



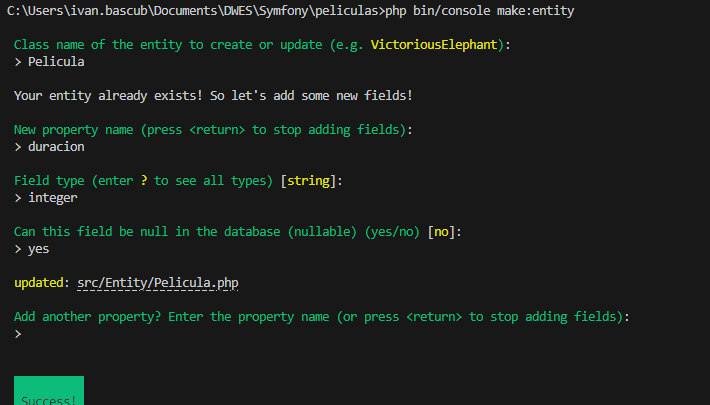
Hacemos la migración a la BBDD



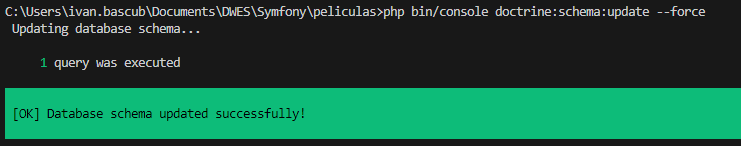
Comprobamos la BBDD si ha habido modificaciones



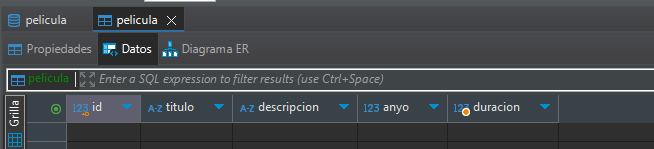
Actualizamos la entidad película añadiéndolo la propiedad duración



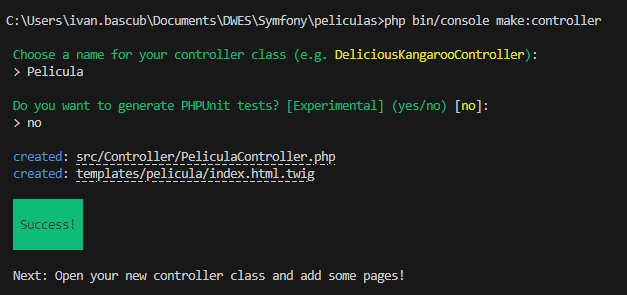
Vamos a forzar a actualizar la BBDD sin migraciones



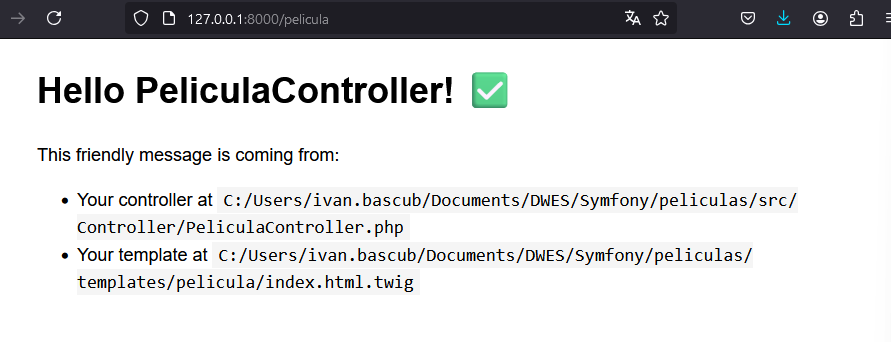
Comprobamos si se ha actualizado



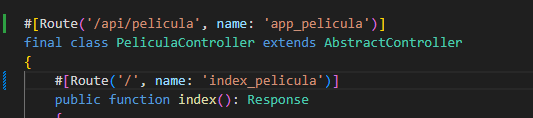
Creamos el controlador de la Entidad Pelicula



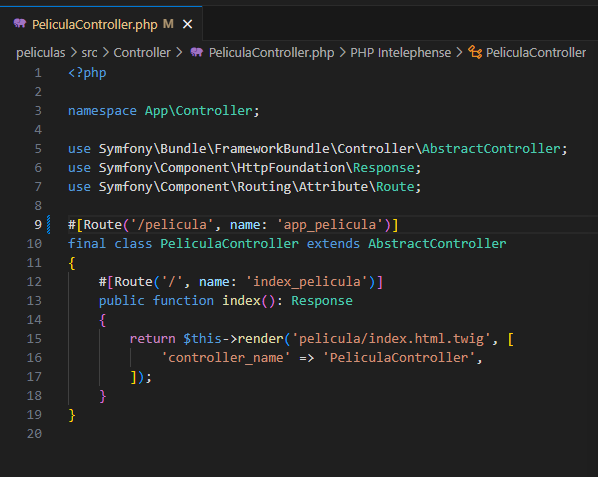
Tras todo eso comprobamos si esta todo bien con el servidor



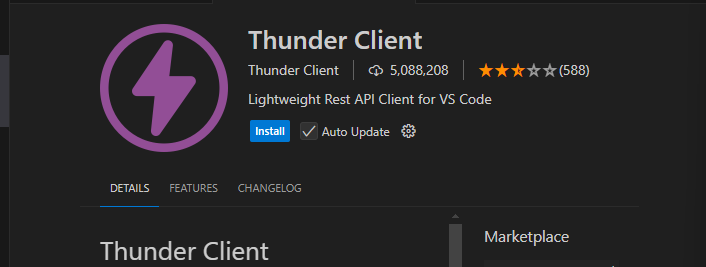
Creamos herencia en el enrutamiento de la web



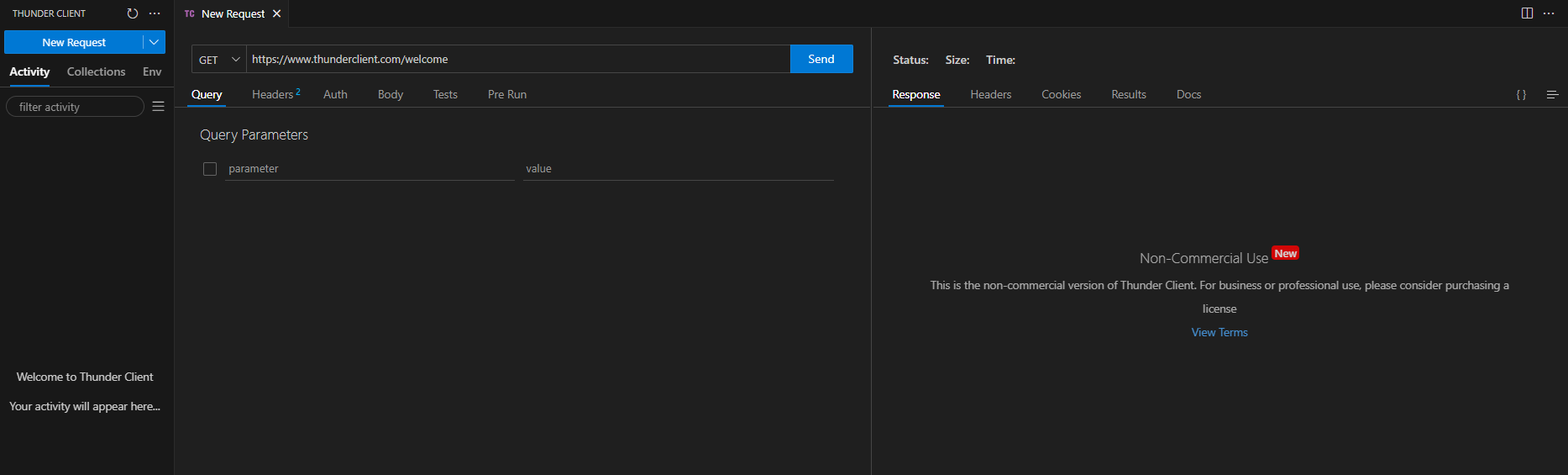
Partimos de esta configuración en el controlador



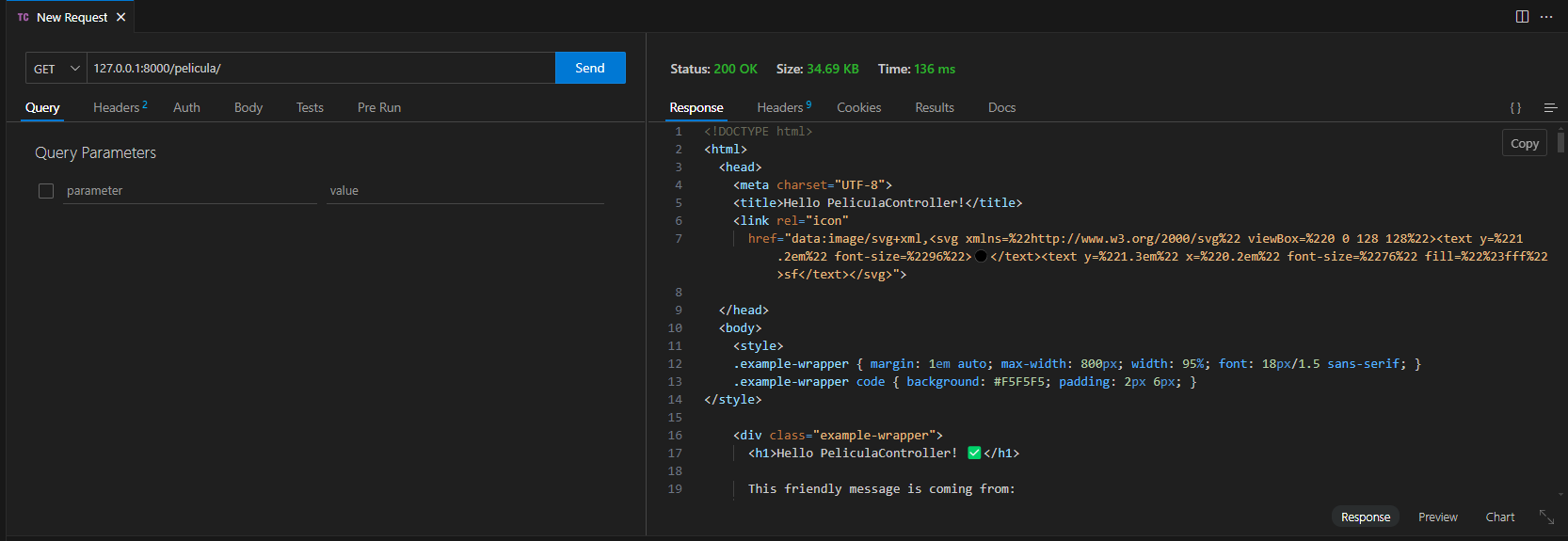
Instalamos la siguiente extensión



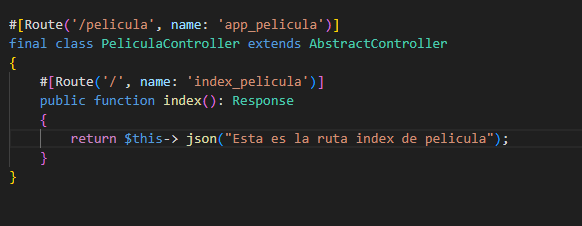
Hacemos una new request en la nueva extensión



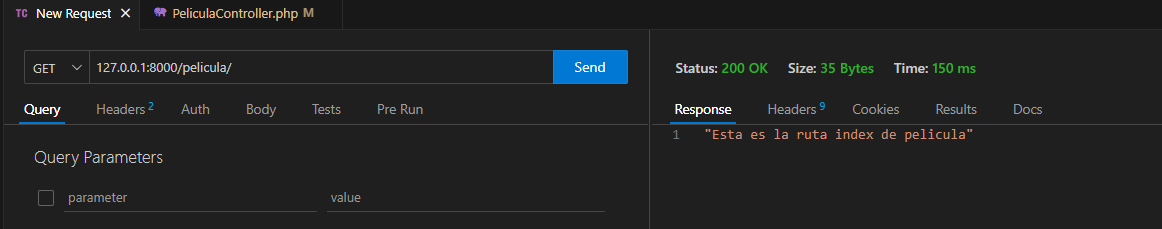
Hacemos una consulta y vemos su resultado



Modificamos el controlador para generar un json

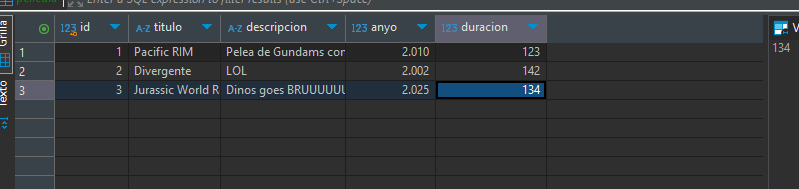


Comprobamos si todo funciona



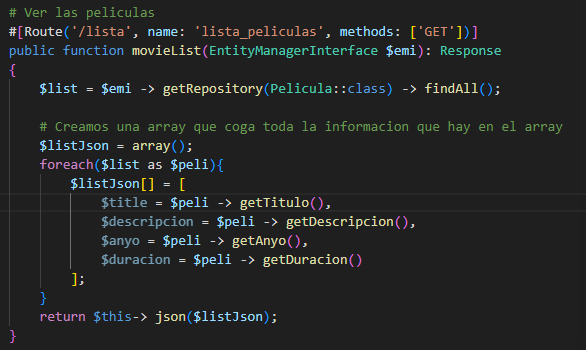
API web (REST= -> CRUD (Pelicula)

Antes de nada añadimos tres películas

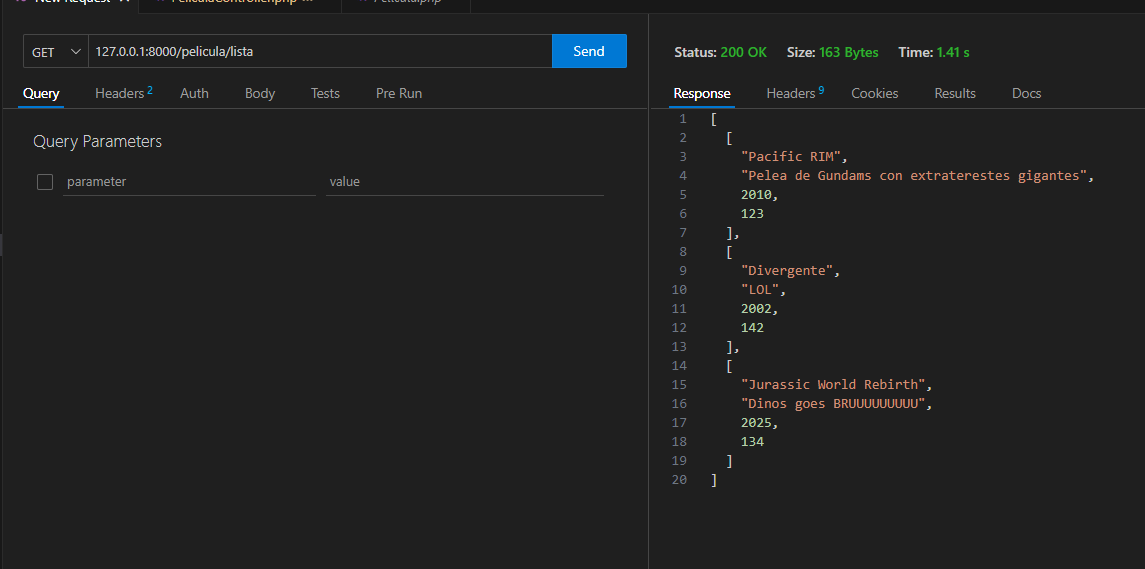


* Ver las películas (Read) (GET)

Creamos el siguiente método con la información de todas las películas

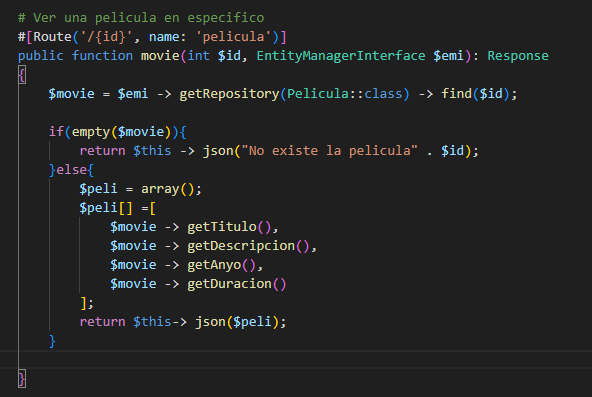


Probamos si funciona

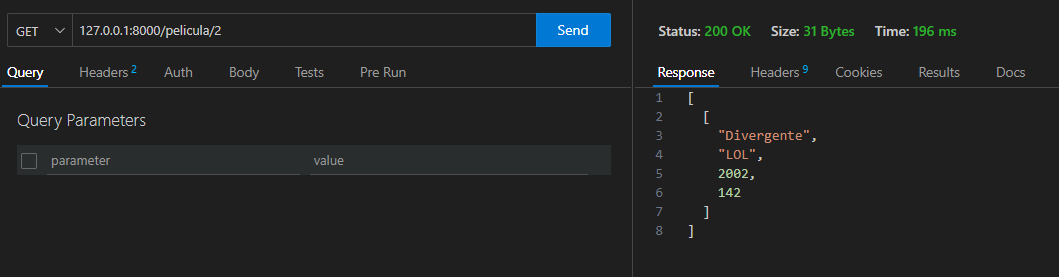


* Ver una película (Read) (GET)

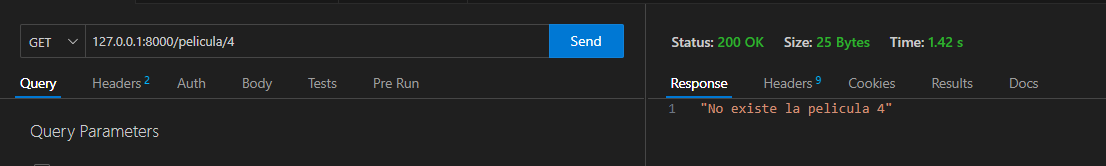
Creamos el siguente método



Comprobamos con una película que existe



Comprobamos con una película que no existe



* Crear la película (Create) (POST)
* Borrar una película (Delete) (DELETE)
* Modificar una película (Update) (PUT)

Web App Mascota Sana

Instalación

Commandos que usamos para instalarlo

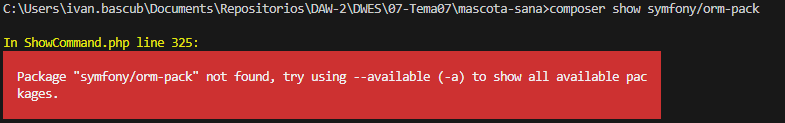
Symphony new my\_porject\_directory –versión=”7.2.x” –webapp

Instalamos los requerimientos dentro de la carpeta del projecto

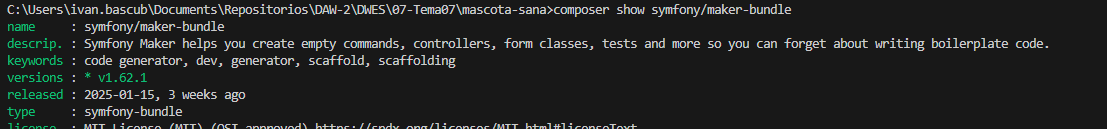
Composer require webapp

Comprobacion de paquetes

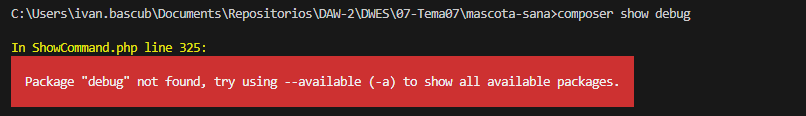
Comprobamos si tenemos instalado el paquete symfony/orm-pack



Comprobamos el paquete Symfony/maker-bundle



Comprobamos si tenemos el paquete debug



Los que no están instalados usamos los siguiente comandos

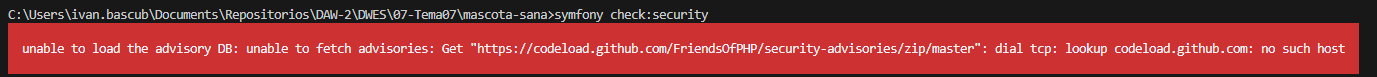
Composer requiere symfony/orm-pack

Composer requiere –dev symfony/maker-bundle

Composer requiere –dev debug

Hacemos un chequeo de seguridad

Symfony check:security



Configuración de la conexión de la BBDD

Configuramos la ruta url de la BBDD en el archivo .env

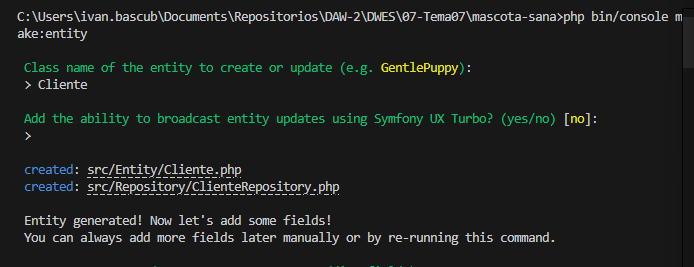


Creamos la BBDD con el siguiente comando php bin/console doctrine:database:create

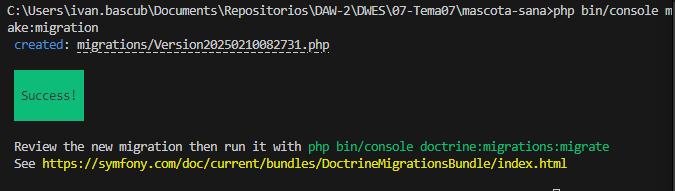


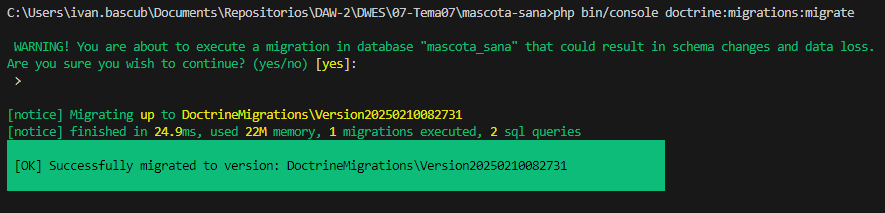
Creacion de entidades

La entidad cliente(ID, nombre, dirección, email, teléfono, fecha\_alta) la creamos con el comando php bin/console make:entity



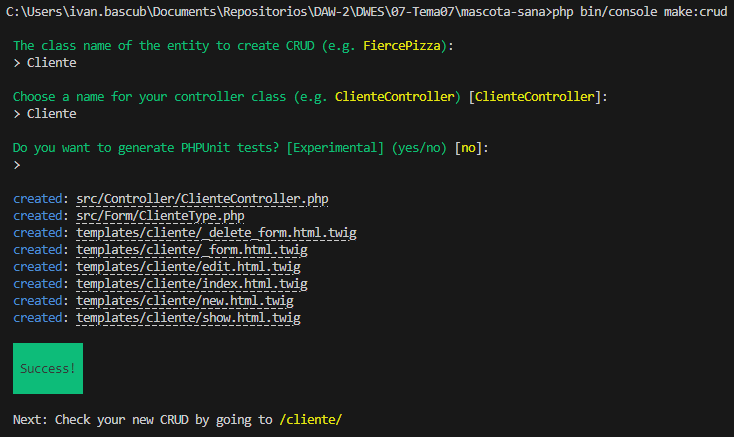
Despues de crearla la migramos a la BBDD



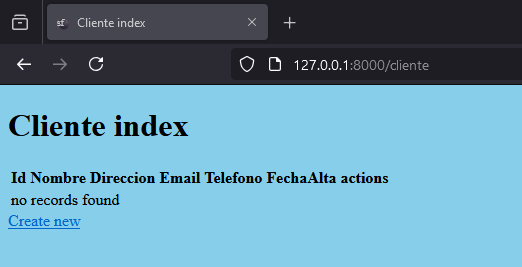


Creación de CRUD con Symfony

Con el siguiente comando creamos el crud php bin/console make:crud



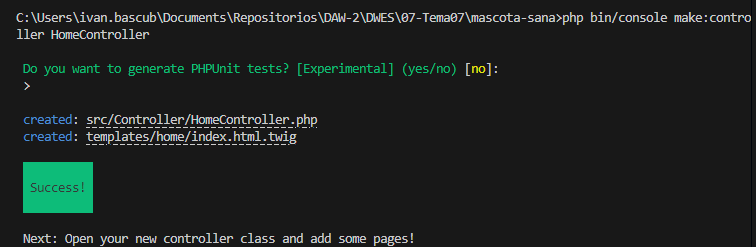
Comprobamos que todo esta bien



Creamos la siguente entidad Mascota(id, nombre, fecha\_nac, cliente), migramos la infomación y creamos el crud

Crear el controlador home

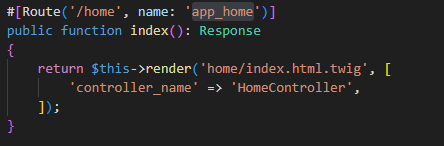
Usamos el comando php bin/console make:controller HomeController



Crear rutas a otras partes de la web

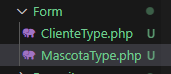


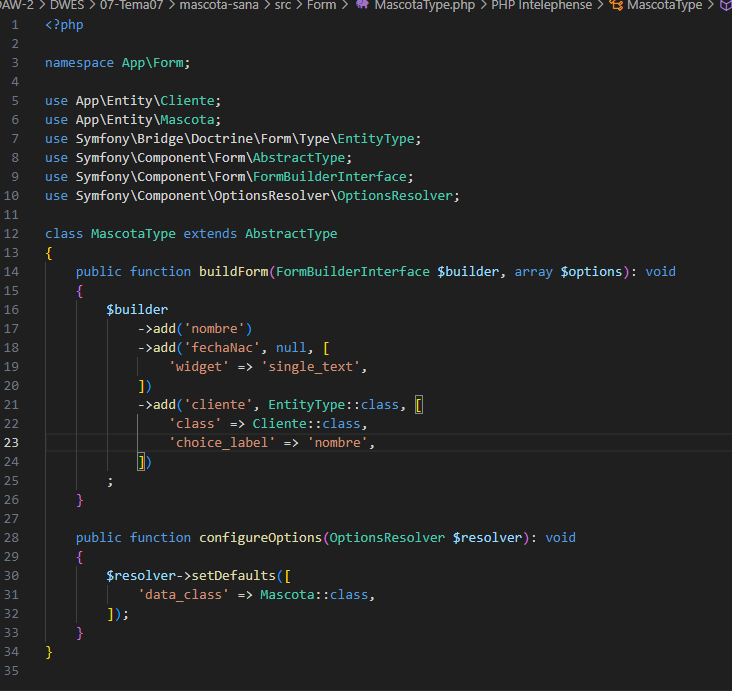
En la parte de href introducimos path y en su interior se indica el name de una función



Modificar los elementos de un formulario

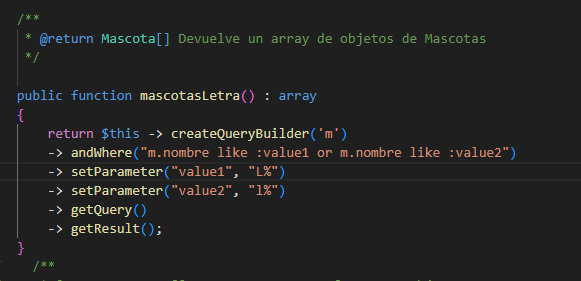
Hay una carpeta form que almacena todos los formularios del proyecto



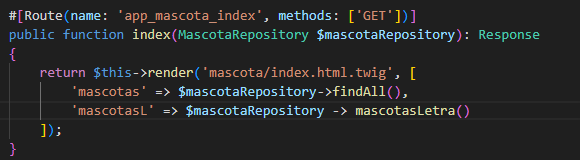


Ejercicio

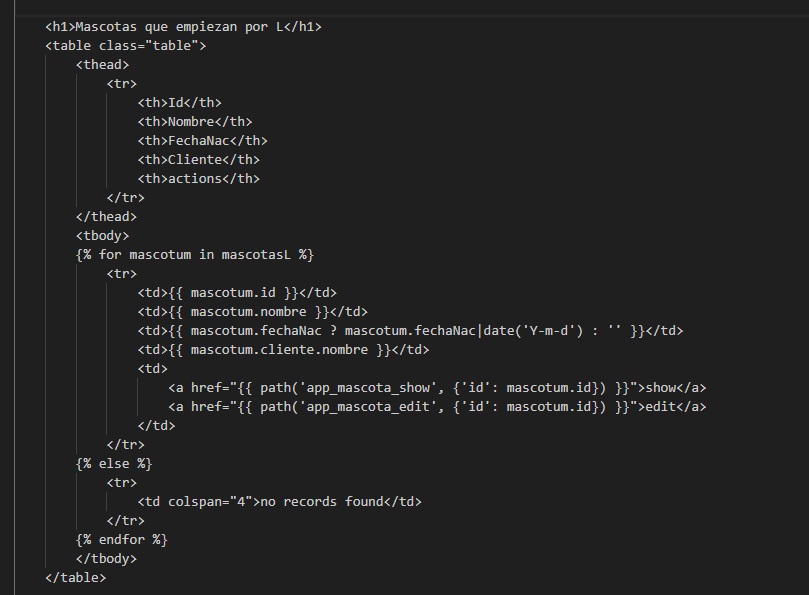
Quiero ver mascota de nombre que comience por L o l. Creamos el siguiente método el el repositorio



Nos vamos al controlador y modificamos el método index



Modificamos el template de index de mascotas



Comprobamos que funciona

