

Modelo - Vista - Controlador

Primero creamos la ruta en el archivo Web.php

```
Route::get('/', function () {  
    return view('welcome');  
});  
  
Route::get("/actors/add", function () {  
    return view("agregarActor");  
});  
  
Route::post("/actors/add", "ActorController@store");
```

`"/actors/add"` es la ruta que debemos escribir en el navegador para acceder

`return view("agregarActor");` No dirige a la vista `agregarActor.blade.php` en donde se encuentre todo el front end.

`"ActorController@store"` en este caso nos dirige al controlador `ActorController` y específicamente a la función `store`.

Controlador

php artisan make:controller ActorController (lo escribimos en la consola)

```
ActorController.php ×  
ABMLaravel > app > Http > Controllers > ActorController.php > ActorController > store  
1  <?php  
2  
3  namespace App\Http\Controllers;  
4  
5  use Illuminate\Http\Request;  
6  use App\Actor;  
7  class ActorController extends Controller  
8  {  
9      public function store (Request $req) {  
10         $actorNuevo = new Actor();  
11  
12         $actorNuevo->first_name = $req["first_name"];  
13         $actorNuevo->last_name = $req["last_name"];  
14         $actorNuevo->save();  
15         return redirect("/actors/add");  
16     }  
17 }
```

```
use App\Actor;
```

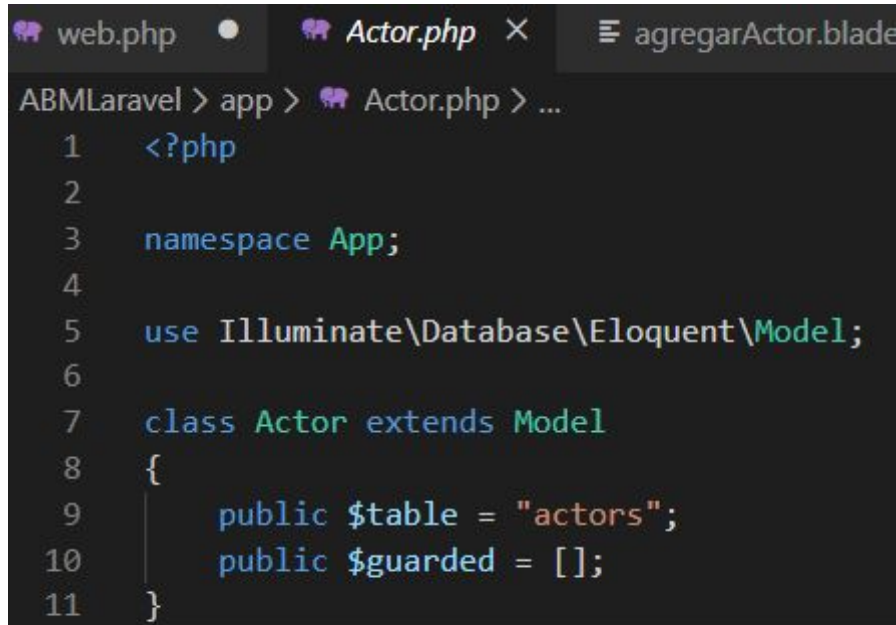
Relacionamos nuestro controlador con el modelo Actor

```
$actorNuevo = new Actor();
```

Creamos un objeto del tipo Actor (nos lo proporciona nuestro modelo)

Modelo

php artisan make:model Actor (lo escribimos en la consola)



```
ABMLaravel > app > Actor.php > ...
1  <?php
2
3  namespace App;
4
5  use Illuminate\Database\Eloquent\Model;
6
7  class Actor extends Model
8  {
9      public $table = "actors";
10     public $guarded = [];
11 }
```

- CREAR MODELOS : los crea dentro de /app

php artisan make:model nombreDelModelo

Se deben aclarar 4 items para que el modelo se relacione con la base de datos:

1) public \$table = "nombreDeLaTabla";

2) public \$primaryKey = "id"; //Si la PK se llama id esta línea puede obviarse.

3) public \$timestamps = ; //Laravel asume que tenemos 2 campos create_at y update_at y los actualiza automáticamente. Si no existen debemos aclarar false. Si existen no hace falta poner este atributo.

4) public \$guarded = []; Laravel no deja que se escriba en base de datos a menos que se explicita mediante este atributo. Si deseamos que alguna columna no se pueda escribir lo especificamos entre las llaves ["sarasa"].

Se puede reemplazar por el atributo \$fillable

5) ForeignKey en caso de que sea necesario.

Podemos agregar métodos y atributos a gusto para utilizar los objetos.

VISTA

Para crear la vista debemos pararnos en `resources/views/` y crear un archivo `nombreDeLaVista.blade.php` y aquí ponemos todo el front end.

