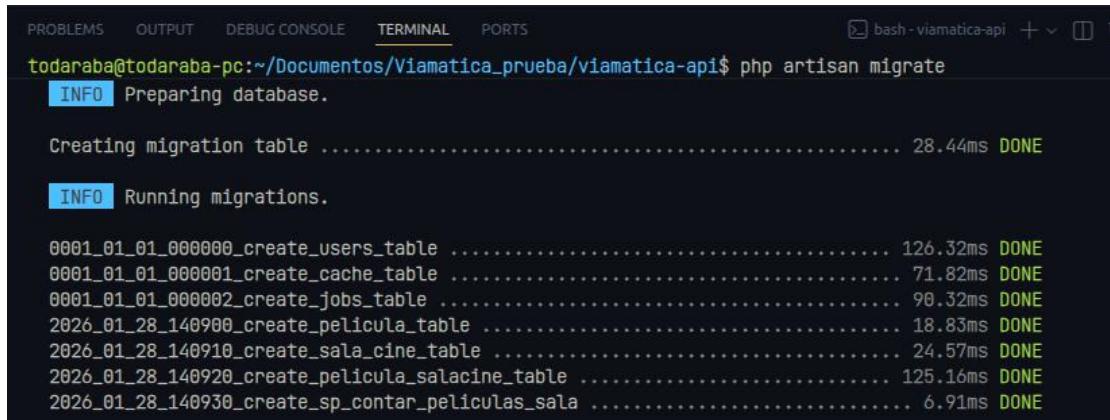


Resultados de crud con postman

1. Crear la base viamatica api
2. Crear las migraciones **php artisan migrate**



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
todaraba@todaraba-pc:~/Documentos/Viamatica_prueba/viamatica-api$ php artisan migrate
INFO Preparing database.

Creating migration table ..... 28.44ms DONE
INFO Running migrations.

0001_01_01_000000_create_users_table ..... 126.32ms DONE
0001_01_01_000001_create_cache_table ..... 71.82ms DONE
0001_01_01_000002_create_jobs_table ..... 90.32ms DONE
2026_01_28_140900_create_pelicula_table ..... 18.83ms DONE
2026_01_28_140910_create_sala_cine_table ..... 24.57ms DONE
2026_01_28_140920_create_pelicula_salacine_table ..... 125.16ms DONE
2026_01_28_140930_create_sp_contar_peliculas_sala ..... 6.91ms DONE
```

3. generar Swagger y verificar /api/documentation/ **php artisan l5-swagger:generate**

4. composer require doctrine/annotations

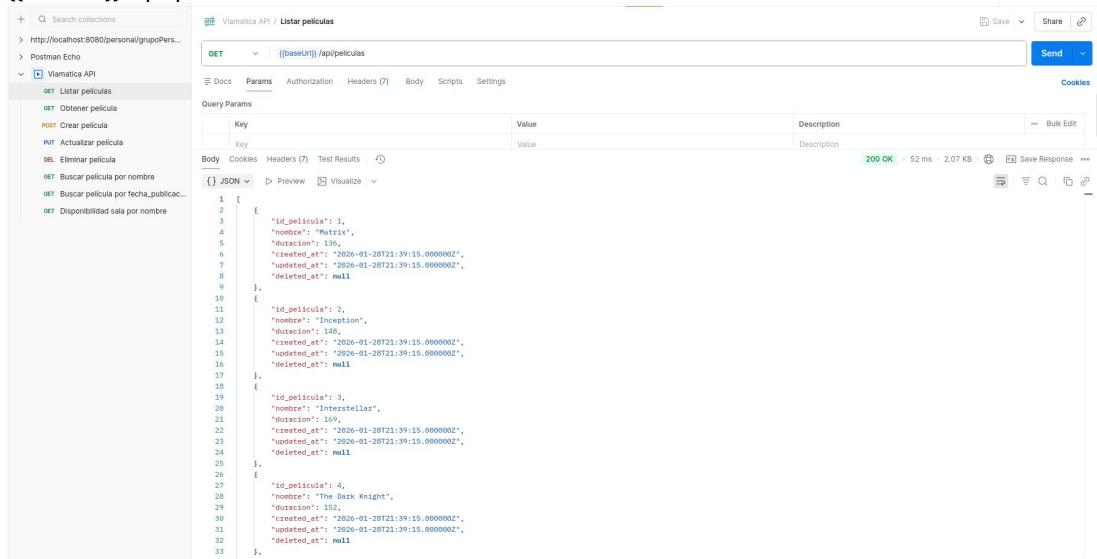
5. php artisan l5-swagger:generate

6. Hice unos seeder para registrar 12 pelicula y 4, salas **php artisan db:seed**

7. Pruebas con Postman

Servicio de peliculas

`{{baseUrl}}/api/peliculas/`



```
1 [
2   {
3     "id_pelicula": 1,
4     "nombre": "Matrix",
5     "duration": 136,
6     "created_at": "2026-01-28T21:39:15.000000Z",
7     "updated_at": "2026-01-28T21:39:15.000000Z",
8     "deleted_at": null
9   },
10  {
11    "id_pelicula": 2,
12    "nombre": "Inception",
13    "duration": 148,
14    "created_at": "2026-01-28T21:39:15.000000Z",
15    "updated_at": "2026-01-28T21:39:15.000000Z",
16    "deleted_at": null
17  },
18  {
19    "id_pelicula": 3,
20    "nombre": "Interstellar",
21    "duration": 169,
22    "created_at": "2026-01-28T21:39:15.000000Z",
23    "updated_at": "2026-01-28T21:39:15.000000Z",
24    "deleted_at": null
25  },
26  {
27    "id_pelicula": 4,
28    "nombre": "The Dark Knight",
29    "duration": 152,
30    "created_at": "2026-01-28T21:39:15.000000Z",
31    "updated_at": "2026-01-28T21:39:15.000000Z",
32    "deleted_at": null
33 ],
```

Obtener película en este caso con el id=5
 {{baseUrl}}/api/pelicula/5

```

1 [
2   "id_pelicula": 5,
3   "nombre": "The Prestige",
4   "duracion": 130,
5   "created_at": "2026-01-28T21:39:15.000000Z",
6   "updated_at": "2026-01-28T21:39:15.000000Z",
7   "deleted_at": null
8 ]
  
```

crear películas

GET: {{baseUrl}}/api/peliculas

```

1 [
2   "nombre": "Avatar",
3   "duracion": 230
4 ]
  
```

POST: Respuesta 201 exitoso al ingresar una nueva película

```

1 [
2   "id_pelicula": 13,
3   "nombre": "Avatar",
4   "duracion": 230,
5   "created_at": "2026-01-28T21:58:02.000000Z",
6   "updated_at": "2026-01-28T21:58:02.000000Z",
7   "deleted_at": null
8 ]
  
```

se actualiza Avatar la duracion de 230 a 250

PUT: {{baseUrl}}/api/peliculas/1?id_pelicula

The screenshot shows the Viamatica API interface. On the left, there's a sidebar with a tree view of collections and endpoints. Under 'Viamatica API', the 'Actualizar pelicula' endpoint is selected. The main area shows a 'PUT' request to {{baseUrl}}/api/peliculas/1?id_pelicula. The 'Body' tab is selected, showing a JSON payload:

```
1 {  
2   "nombre": "Avatar",  
3   "duracion": 250  
4 }
```

Below the body, the response is shown in the 'Body' tab, which displays the updated movie details:

```
1 {  
2   "id_pelicula": 1,  
3   "nombre": "Avatar",  
4   "duracion": 250,  
5   "created_at": "2026-01-28T21:39:15.000000Z",  
6   "updated_at": "2026-01-28T22:03:30.000000Z",  
7   "deleted_at": null  
8 }
```

DELETE: {{baseUrl}}/api/peliculas/13

Respuesta 200 con mensaje "pelicula eliminada"

The screenshot shows the Viamatica API interface. On the left, there's a sidebar with a tree view of collections and endpoints. Under 'Viamatica API', the 'Eliminar pelicula' endpoint is selected. The main area shows a 'DELETE' request to {{baseUrl}}/api/peliculas/13. The 'Body' tab is selected, showing a JSON payload:

```
1 Ctrl+Alt+P to Ask AI
```

Below the body, the response is shown in the 'Body' tab, which displays a message indicating the movie was deleted:

```
1 {  
2   "message": "Pelicula eliminada"  
3 }
```

The status bar at the bottom right indicates a 200 OK response.

Buscar película

en este caso se busca la pelicula el Jocker

GET: {{baseUrl}}/api/peliculas/buscar/nombre?nombre=Joker

The screenshot shows a Postman collection named 'Viamatica API' with various endpoints. A specific GET request is highlighted for '/api/peliculas/buscar/nombre?nombre=Joker'. The response status is 200 OK with a response time of 23 ms. The JSON response body contains one movie record:

```

1 [
2   {
3     "id_pelicula": 12,
4     "nombre": "Joker",
5     "duracion": 122,
6     "created_at": "2026-01-28T21:39:15.000000Z",
7     "updated_at": "2026-01-28T21:39:15.000000Z",
8     "deleted_at": null
9   }
10 ]

```

Buscar película por fecha de publicación

Respuesta 200 exitosa

GET: {{baseUrl}}/api/peliculas/buscar/fecha-publicacion?fecha_publicacion=2026-01-12

The screenshot shows a Postman collection named 'Viamatica API' with various endpoints. A specific GET request is highlighted for '/api/peliculas/buscar/fecha-publicacion?fecha_publicacion=2026-01-12'. The response status is 200 OK. The Query Params section shows 'fecha_publicacion' set to '2026-01-12'. The JSON response body contains two movie records:

```

1 [
2   {
3     "id_pelicula_sala": 3,
4     "id_sala_cine": 2,
5     "fecha_publicacion": "2026-01-12T00:00:00.000000Z",
6     "fecha_fin": "2026-01-22T00:00:00.000000Z",
7     "id_pelicula": 3,
8     "created_at": "2026-01-28T21:39:15.000000Z",
9     "updated_at": "2026-01-28T21:39:15.000000Z",
10    "deleted_at": null,
11    "pelicula": {
12      "id_pelicula": 3,
13      "nombre": "Interstellar",
14      "duracion": 169,
15      "created_at": "2026-01-28T21:39:15.000000Z",
16      "updated_at": "2026-01-28T21:39:15.000000Z",
17      "deleted_at": null
18    }
19  }
20 ]

```

Disponibilidad de sala

Respuesta 200 exitosa

GET: {{baseUrl}}/api/salas/disponibilidad?nombre=Sala%201

The screenshot shows the Postman interface with the following details:

- Collection:** Viamatica API
- Request Type:** GET
- URL:** {{baseUrl}}/api/salas/disponibilidad?nombre=Sala%201
- Headers:** (7 items)
 - Cache-Control: no-cache
 - Postman-Token: <calculated when request is sent>
 - Host: <calculated when request is sent>
 - User-Agent: PostmanRuntime/7.51.0
 - Accept: */*
 - Accept-Encoding: gzip, deflate, br
 - Connection: keep-alive
- Body:** (JSON)

```
[{"idSala": 1, "nombreSala": "Sala 1", "totalPeliculas": 2, "mensaje": "Sala disponible"}]
```
- Response Status:** 200 OK
- Response Time:** 17 ms
- Response Size:** 304 B

Pruebas de documentación con swagger UI

Vista general de Swagger UI mostrando el título y versión

The screenshot shows the Swagger UI homepage with the following details:

- Title:** Viamatica API
- Version:** 1.0.0
- Definition URL:** http://127.0.0.1:8000/api/docs?api-docs.json
- Description:** API REST para gestión de películas y salas de cine
- Filter by tag:** A search bar for filtering API endpoints.
- Tags:**
 - Peliculas
 - Salas

Sección Películas con todos los endpoints actuales

The screenshot shows the Viamatica API documentation generated by Swagger. At the top, there's a header with the Swagger logo and a dropdown menu to "Select a definition". Below the header, the title "Viamatica API" is displayed along with its version "1.0.0" and "OAS 3.0". A URL "http://127.0.0.1:8000/docs?api-docs.json" is also present. A brief description states "API REST para gestión de películas y salas de cine". A search bar labeled "Filter by tag" is at the top left. The main content area is titled "Películas" and contains a list of endpoints:

- GET /api/peliculas** Listar películas
- POST /api/películas** Crear película
- GET /api/películas/{id}** Obtener película por ID
- PUT /api/películas/{id}** Actualizar película
- DELETE /api/películas/{id}** Eliminar película (soft delete)
- GET /api/películas/buscar/nombre** Buscar películas por nombre
- GET /api/películas/buscar/fecha-publicación** Buscar películas por fecha de publicación

This screenshot shows the detailed view for the "Crear película" endpoint (POST /api/películas). It includes sections for "Parameters" (No parameters), "Request body required" (application/json), and "Example Value | Schema" (JSON object with "nombre" and "duracion" fields). The "Responses" section lists two possible outcomes:

Code	Description	Links
201	Creado	No links
422	Validación	No links

GET /api/peliculas/{id} Obtener película por ID

Cancel

Parameters

Name	Description
id * required integer (path)	2

Execute Clear

Responses

Curl

```
curl -X 'GET' \
'http://127.0.0.1:8000/api/peliculas/2' \
-H 'accept: */*' \
-H 'X-CSRF-TOKEN:'
```

Request URL

http://127.0.0.1:8000/api/peliculas/2

Server response

Code Details

200 Response body

```
{ "id_pelicula": 2, "nombre": "Inception", "duracion": 148, "created_at": "2026-01-28T21:39:15.000000Z", "updated_at": "2026-01-28T21:39:15.000000Z", "deleted_at": null }
```

Download

Response headers

```
access-control-allow-origin: *
cache-control: no-cache,private
content-type: application/json
date: Wed, 28 Jan 2026 22:28:25 GMT
host: 127.0.0.1:8000
x-powered-by: PHP/8.2.30
```

Responses

Code	Description	Links
200	OK	No links
404	No encontrado	No links