



UNIVERSIDAD
DON BOSCO

FACULTAD DE INGENIERIA ESCUELA DE COMPUTACIÓN

Desafío 3

Guardado Alvarado Erika Fernanda GA220455

Perez Lue Cristina Aracely PL240092

Portillo Hernandez Yesenia Lisbeth HP240153

Jonathan Baltazar Escalante Guardado

ENLACE: <https://github.com/GA220455/DWFLAB/tree/main>

SOYAPANGO – EL SALVADOR

2025

➤ Pruebas POST:

- 1- POST <http://localhost:8080/auth/sign-up> : Se utiliza para agregar un nuevo usuario dentro de nuestra aplicación.

Ejemplo:

```
{  
  "username": "usuario1",  
  "email": "usuario1@biblioteca.com",  
  "password": "123456"  
}
```

- 2- POST <http://localhost:8080/auth/sign-in> : se utiliza para autenticar a un usuario en nuestra aplicación.

Ejemplo:

```
{  
  "email": "usuario1@biblioteca.com",  
  "password": "123456"  
}
```

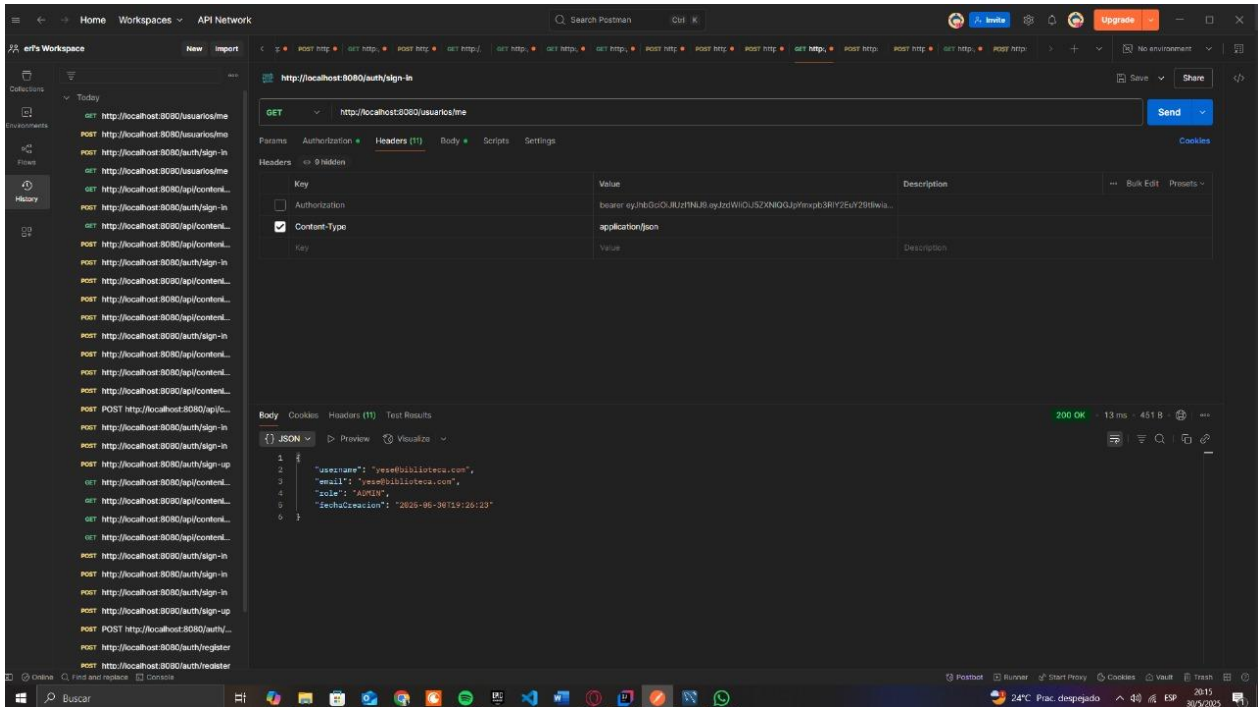
- 3- POST <http://localhost:8080/api/contenido> : En la solicitud POST a <http://localhost:8080/api/contenido>, debes incluir el token generado al momento del endpoint anterior, en los encabezados (headers) de la solicitud. Se usa un encabezado llamado Authorization.

Ejemplo:

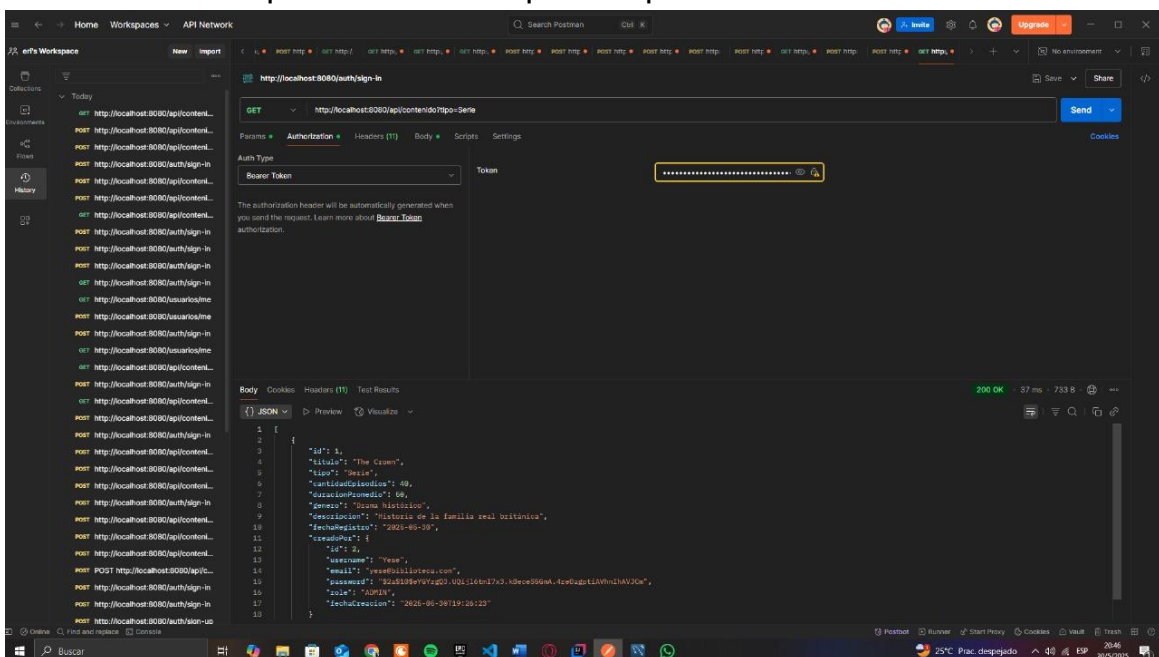
```
{  
  "titulo": "The Mentalist",  
  "tipo": "Serie",  
  "cantidadEpisodios": 151,  
  "duracionPromedio": 50,  
  "genero": "Drama, Accion",  
  "descripcion": "Historia de un detective que es vidente"  
}
```

➤ Pruebas GET:

- 1- GET <http://localhost:8080/api/contenido> : se utiliza para recuperar datos del servidor relacionados con el contenido disponible.
- 2- GET <http://localhost:8080/usuarios/me> : utiliza para recuperar información sobre el usuario que ha iniciado sesión en la aplicación.



- 3- GET <http://localhost:8080/api/contenido?tipo=Serie> : utiliza para recuperar contenido específico filtrado por el tipo "Serie"



➤ Base de datos con MYSQL:

The screenshot displays the MySQL Workbench interface. The top menu bar includes File, Edit, View, Query, Database, Server, Tools, Scripting, and Help. The left sidebar shows the 'Schemas' tree with 'biblioteca_db' selected, containing tables like 'contenido', 'post', 'post_comment', 'post_comment_seq', 'post_seq', 'security_user', and 'user'. The main query editor shows a SQL query: `SELECT * FROM biblioteca_db.security_user;`. The 'Results Grid' displays the query output with columns: id, email, fecha_creacion, password, role, and username. The bottom panel shows the 'Output' tab with a log of query execution actions and their durations.

Table: `post_comment`

Columns:

- `id`: bigint(20)
- `pk`: PK
- `comment_date`: date
- `review`: varchar(255)
- `post_id`: bigint(20)

Table: `security_user`

Columns:

- `id`: int(11)
- `email`: varchar(255)
- `fecha_creacion`: datetime
- `password`: varchar(255)
- `role`: enum('USER', 'ADMIN')
- `username`: varchar(255)

Query Output:

#	Time	Action	Message	Duration / Fetch
4	19:15:07	SELECT * FROM biblioteca_db.security_user LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec
5	19:15:27	SELECT * FROM biblioteca_db.security_user LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec
6	19:30:50	SELECT * FROM biblioteca_db.usuarios LIMIT 0, 1000	0 row(s) returned	0.000 sec / 0.000 sec
7	19:30:53	SELECT * FROM biblioteca_db.security_user LIMIT 0, 1000	2 row(s) returned	0.000 sec / 0.000 sec