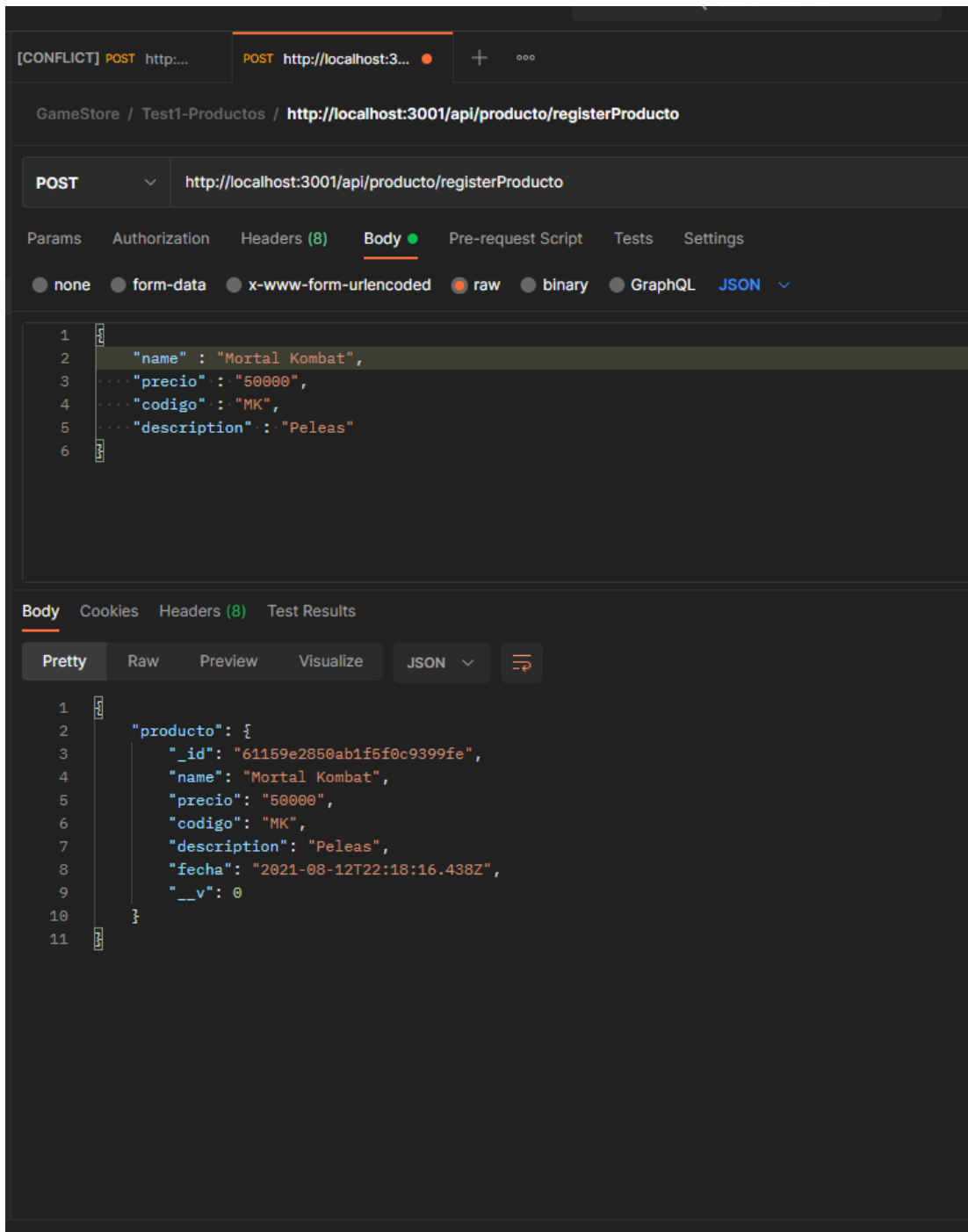


Pantallazos registrados By Juan Diego Sic

función Post:

- **Productos:**

1)



2)

The screenshot displays a REST client interface with a dark theme. At the top, a breadcrumb trail reads "GameStore / Test1-Productos / http://localhost:3001/api/producto/registerProducto". Below this, the request method is set to "POST" and the URL is "http://localhost:3001/api/producto/registerProducto". The "Body" tab is selected, showing a JSON payload with the following fields: "name" (Fifa 18), "precio" (30000), "codigo" (FF), and "description" (Futbol). The response section shows the "Body" tab with a "Pretty" view of the JSON response, which includes an "_id" field and a timestamp "fecha".

GameStore / Test1-Productos / http://localhost:3001/api/producto/registerProducto

POST http://localhost:3001/api/producto/registerProducto

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   "name" : "Fifa 18",
3   "precio" : "30000",
4   "codigo" : "FF",
5   "description" : "Futbol"
6 }
```

Body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "producto": {
3     "_id": "61159e4f50ab1f5f0c939a01",
4     "name": "Fifa 18",
5     "precio": "30000",
6     "codigo": "FF",
7     "description": "Futbol",
8     "fecha": "2021-08-12T22:18:55.001Z",
9     "__v": 0
10  }
11 }
```

3)

The screenshot shows a REST client interface with a POST request to `http://localhost:3001/api/producto/registerProducto`. The request body is a JSON object with the following fields:

```
{
  "name": "Dark Souls 3",
  "precio": "70500",
  "codigo": "DS3",
  "description": "RPG"
}
```

The response body is a JSON object with the following fields:

```
{
  "producto": {
    "_id": "61159e7450ab1f5f0c939a04",
    "name": "Dark Souls 3",
    "precio": "70500",
    "codigo": "DS3",
    "description": "RPG",
    "fecha": "2021-08-12T22:19:32.384Z",
    "__v": 0
  }
}
```

The interface includes tabs for Params, Authorization, Headers (8), Body, Pre-request Script, Tests, and Settings. The Body tab is selected, and the response is displayed in the Pretty view.

4)

The image shows the Postman application interface. At the top, there's a search bar labeled "Search Postman". Below it, the request method is "POST" and the URL is "http://localhost:3001/api/producto/registerProducto". The breadcrumb path is "GameStore / Test1-Productos / http://localhost:3001/api/producto/registerProducto".

The "Body" tab is selected, showing a JSON payload:

```
1 {
2   "name": "Batlefield 3",
3   "precio": "63000",
4   "codigo": "B3",
5   "description": "Disparos"
6 }
```

Below the request body, the "Body" tab is selected in the response section, showing the "Pretty" view of the JSON response:

```
1 {
2   "producto": {
3     "_id": "61159eab50ab1f5f0c939a07",
4     "name": "Batlefield 3",
5     "precio": "63000",
6     "codigo": "B3",
7     "description": "Disparos",
8     "fecha": "2021-08-12T22:20:27.471Z",
9     "__v": 0
10  }
11 }
```

5)

The screenshot displays the Postman interface for a POST request. The request is directed to `http://localhost:3001/api/producto/registerProducto`. The body is formatted as JSON and contains the following data:

```
1 {
2   "name": "GTA 5",
3   "precio": "150000",
4   "codigo": "gta",
5   "description": "RPG"
6 }
```

The response is also in JSON format, showing the registered product details:

```
1 {
2   "producto": {
3     "_id": "61159ec950ab1f5f0c939a0a",
4     "name": "GTA 5",
5     "precio": "150000",
6     "codigo": "gta",
7     "description": "RPG",
8     "fecha": "2021-08-12T22:20:57.246Z",
9     "__v": 0
10  }
11 }
```

- Stocks:

1)

The screenshot displays a REST client interface with a dark theme. At the top, a breadcrumb trail shows 'GameStore / Test3-Sock / http://localhost:3001/api/stock/registerStock'. Below this, the request details are shown: a POST method to 'http://localhost:3001/api/stock/registerStock'. The 'Body' tab is selected, showing a JSON payload:

```
{  "cantidad": "2",  "bodega": "Kennedy32"}
```

. The bottom section shows the response in the 'Body' tab, displaying a JSON object:

```
{  "stock": {    "_id": "61159f0ce1a8ae3804c4dae7",    "productoId": "61159e2850ab1f5f0c9399fe",    "cantidad": "2",    "bodega": "Kennedy32",    "fecha": "2021-08-12T22:22:04.006Z",    "__v": 0  }}
```

2)

The screenshot displays a REST client interface with a dark theme. At the top, there are two tabs for POST requests to `http://localhost:3001`. The active tab is titled `GameStore / Test3-Sock / http://localhost:3001/api/stock/registerStock`.

The request configuration shows a **POST** method to `http://localhost:3001/api/stock/registerStock`. The **Body** tab is selected, showing a JSON payload:

```
1 {
2   "cantidad": "3",
3   "bodega": "Suba esquina"
4 }
```

Below the request, the **Body** tab of the response is active, displaying the JSON result in a pretty-printed format:

```
1 {
2   "stock": {
3     "_id": "61159f38e1a8ae3804c4daeb",
4     "productoId": "61159e2850ab1f5f0c9399fe",
5     "cantidad": "3",
6     "bodega": "Suba esquina",
7     "fecha": "2021-08-12T22:22:48.154Z",
8     "__v": 0
9   }
10 }
```

3)

The screenshot shows a REST client interface with a POST request to `http://localhost:3001/api/stock/registerStock`. The request body is a JSON object with the following fields:

```
{
  "cantidad": "1",
  "bodega": "Engativa party"
}
```

The response body is a JSON object with the following fields:

```
{
  "stock": {
    "_id": "61159f63cb284e67dccd3540",
    "productoId": "61159ec950ab1f5f0c939a0a",
    "cantidad": "1",
    "bodega": "Engativa party",
    "fecha": "2021-08-12T22:23:31.977Z",
    "__v": 0
  }
}
```

The interface includes tabs for Params, Authorization, Headers (8), Body, Pre-request Script, Tests, and Settings. The Body tab is selected, and the response is displayed in the Pretty view.

4)

The screenshot displays a REST client interface with a dark theme. At the top, a breadcrumb trail shows 'GameStore / Test3-Sock / http://localhost:3001/api/stock/registerStock'. Below this, the request configuration is shown: a 'POST' method to 'http://localhost:3001/api/stock/registerStock'. The 'Body' tab is selected, showing a JSON payload:

```
{  "cantidad": "2",  "bodega": "Edificio corcel"}
```

. The bottom section shows the response, also in the 'Body' tab, with a 'Pretty' view of the JSON:

```
{  "stock": {    "_id": "61159f85cb284e67dccd3544",    "productoId": "61159ec950ab1f5f0c939a0a",    "cantidad": "2",    "bodega": "Edificio corcel",    "fecha": "2021-08-12T22:24:05.803Z",    "__v": 0  }}
```

5)

The screenshot shows a REST client interface with a POST request to `http://localhost:3001/api/stock/registerStock`. The request body is a JSON object with the following fields:

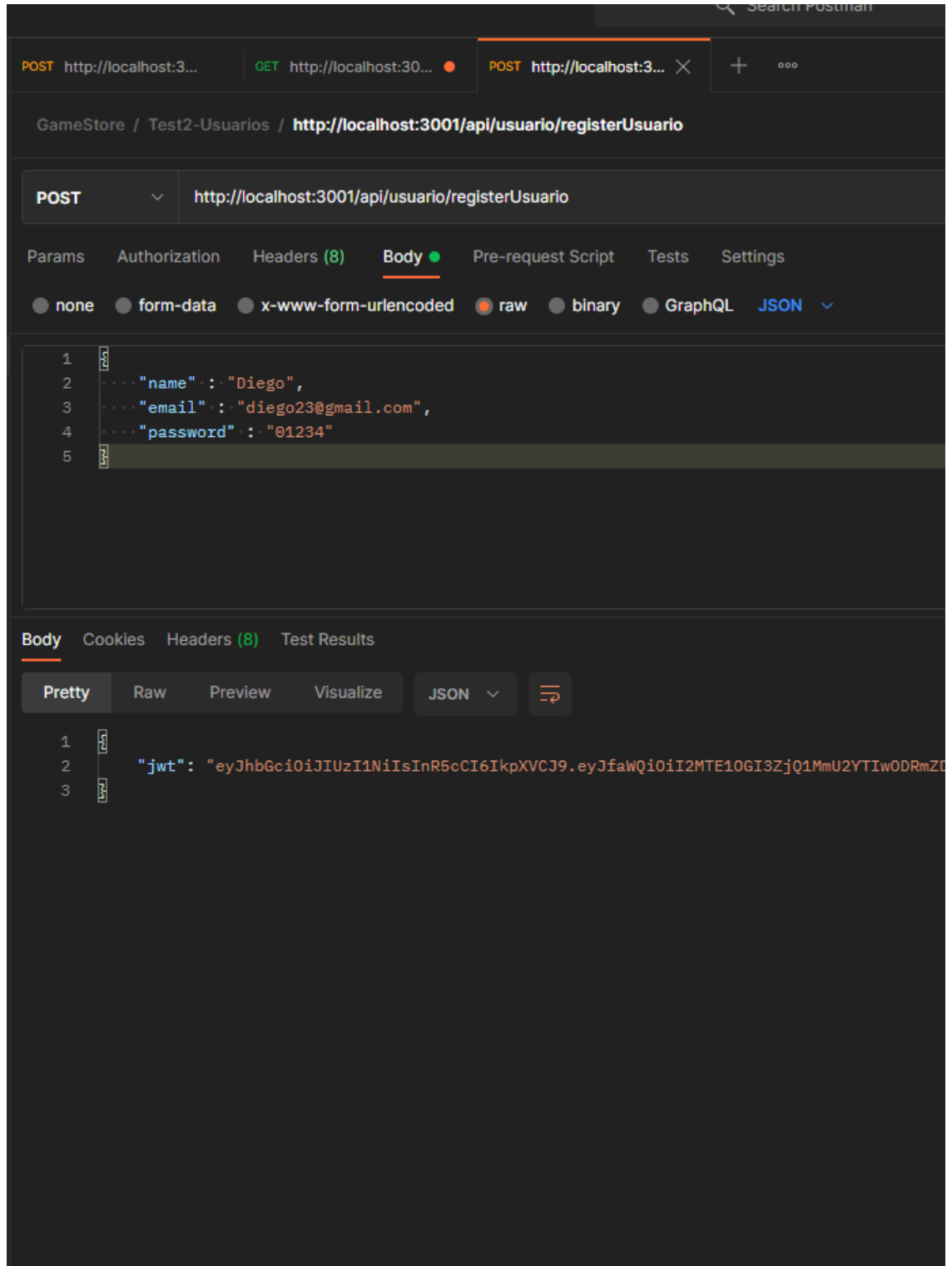
```
{  "cantidad": "14",  "bodega": "Castillo tirano"}
```

The response body is also shown in a pretty-printed JSON format:

```
{  "stock": {    "_id": "61159fb53a9ad827549f2dd4",    "productoId": "61159e7450ab1f5f0c939a04",    "cantidad": "14",    "bodega": "Castillo tirano",    "fecha": "2021-08-12T22:24:53.382Z",    "__v": 0  }}
```

- Usuarios:

1)



2)

The screenshot displays a REST client interface with a POST request to `http://localhost:3001/api/usuario/registerUsuario`. The request body is a JSON object with the following fields:

```
{  "name": "Felipe",  "email": "feilpepro@gmail.com",  "password": "01234"}
```

The response is also in JSON format, showing a JWT token:

```
{  "jwt": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJfaWQiOiI2MTE1OTVhMDc4MjI4YTFlNDg1MzE2Nj4"}
```

The interface includes tabs for Params, Authorization, Headers (8), Body, Pre-request Script, Tests, and Settings. The Body tab is active, showing the request and response bodies. The response body is displayed in a 'Pretty' JSON view.

3)

The screenshot displays a REST client interface with a POST request to `http://localhost:3001/api/usuario/registerUsuario`. The request body is a JSON object containing user details: `{ "name": "Sebastian", "email": "seb123@gmail.com", "password": "01234" }`. The response is also in JSON format, showing a successful status and a JWT token: `{ "jwt": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJfaWQiOiI2MTE1OTYyNzc4MjI4YTFlNDg1MzE2NmEiL..." }`.

GameStore / Test2-Usuarios / `http://localhost:3001/api/usuario/registerUsuario`

POST `http://localhost:3001/api/usuario/registerUsuario`

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **JSON** ▾

```
1 {
2   "name": "Sebastian",
3   "email": "seb123@gmail.com",
4   "password": "01234"
5 }
```

Body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize JSON ▾

```
1 {
2   "jwt": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJfaWQiOiI2MTE1OTYyNzc4MjI4YTFlNDg1MzE2NmEiL..."
3 }
```

4)

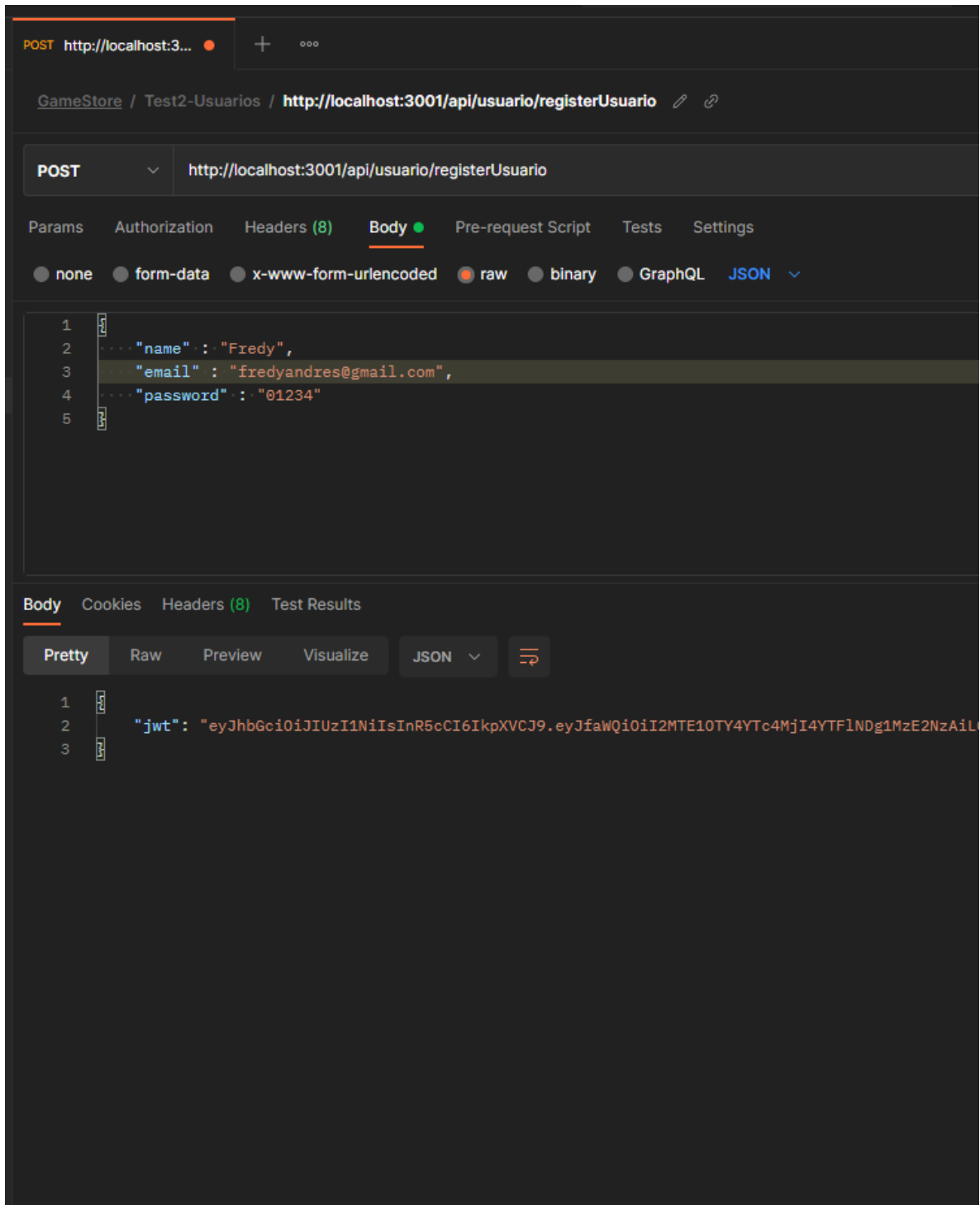
The screenshot displays a REST client interface with the following details:

- Request:**
 - Method: **POST**
 - URL: `http://localhost:3001/api/usuario/registerUsuario`
 - Body Type: **JSON**
 - Body Content:

```
1 {
2   "name": "Antonio",
3   "email": "antonio34@gmail.com",
4   "password": "01234"
5 }
```
- Response:**
 - Tab: **Body**
 - Format: **JSON**
 - Content:

```
1 {
2   "jwt": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJfaWQiOiI2MTE1OTY0MTc4MjI4YTF1NDg1MzE2NmQ"
3 }
```

5)



- Ventas:

1)

The screenshot displays a REST client interface with a POST request to `http://localhost:3001/api/venta/registerVenta`. The request body is a JSON object with the following fields:

```
{  "codigo": "120",  "precio": "50000"}
```

The response is shown in the 'Body' tab, formatted as JSON. It contains a single object with the key 'venta' and a nested object with the following fields:

```
{  "venta": {    "_id": "6115a04724a5b235f4512534",    "codigo": "120",    "productoId": "61159e7450ab1f5f0c939a04",    "usuarioId": "6115962778228a1e4853166a",    "precio": "50000",    "fecha": "2021-08-12T22:27:19.027Z",    "__v": 0  }}
```


2)

The screenshot displays a REST client interface with a POST request to `http://localhost:3001/api/venta/registerVenta`. The request body is a JSON object with `codigo` and `precio` fields. The response body is a JSON object containing a `venta` object with various fields including `_id`, `codigo`, `productoId`, `usuarioId`, `precio`, `fecha`, and `__v`.

Request:

```
POST http://localhost:3001/api/venta/registerVenta
```

Request Body (JSON):

```
{  "codigo": "121",  "precio": "55000"}
```

Response Body (JSON):

```
{  "venta": {    "_id": "6115a068cec76c66f4e25fd5",    "codigo": "121",    "productoId": "61159e7450ab1f5f0c939a04",    "usuarioId": "61158b7f452e6a2084fd5779",    "precio": "55000",    "fecha": "2021-08-12T22:27:52.538Z",    "__v": 0  }}}
```

3)




The screenshot shows a REST client interface with a POST request to `http://localhost:3001/api/venta/registerVenta`. The request body is a JSON object with the following fields:

```
{  "codigo": "122",  "precio": "150000"}
```


The response body is also shown in JSON format:


```
{  "venta": {    "_id": "6115a0927c96d8681003dd90",    "codigo": "122",    "productoId": "61159ec950ab1f5f0c939a0a",    "usuarioId": "61158b7f452e6a2084fd5779",    "precio": "150000",    "fecha": "2021-08-12T22:28:34.698Z",    "__v": 0  }  }
```








4)

POST http://localhost:3...   

GameStore / Test4-Venta / http://localhost:3001/api/venta/registerVenta



POST  http://localhost:3001/api/venta/registerVenta

Params Authorization Headers (8) **Body**  Pre-request Script Tests Settings

 none  form-data  x-www-form-urlencoded  raw  binary  GraphQL **JSON** 

```
1  {
2    "codigo" : "123",
3    "precio" : "80000"
4  }
```

Body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize JSON  

```
1  {
2    "venta": {
3      "_id": "6115a0b99f90e26b70a6a90f",
4      "codigo": "123",
5      "productoId": "61159eab50ab1f5f0c939a07",
6      "usuarioId": "6115964178228a1e4853166d",
7      "precio": "80000",
8      "fecha": "2021-08-12T22:29:13.041Z",
9      "__v": 0
10   }
11 }
```

5)

The screenshot shows a REST client interface with a POST request to `http://localhost:3001/api/venta/registerVenta`. The request body is a JSON object with `codigo` and `precio` fields. The response is a JSON object with a `venta` field containing an object with `_id`, `codigo`, `productoId`, `usuarioId`, `precio`, `fecha`, and `__v` fields.

POST `http://localhost:3...` + ...

GameStore / Test4-Venta / `http://localhost:3001/api/venta/registerVenta`

POST `http://localhost:3001/api/venta/registerVenta`

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL JSON

```
1 {
2   "codigo": "124",
3   "precio": "45000"
4 }
```

Body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "venta": {
3     "_id": "6115a0d89c91d814407e5d9e",
4     "codigo": "124",
5     "productoId": "61159eab50ab1f5f0c939a07",
6     "usuarioId": "6115962778228a1e4853166a",
7     "precio": "45000",
8     "fecha": "2021-08-12T22:29:44.968Z",
9     "__v": 0
10  }
11 }
```

función Get:

- **Productos:**

The screenshot shows a web browser with a REST client interface. The address bar displays the URL `http://localhost:3001/api/producto/listProducto`. The method is set to **GET**. The response body is displayed in JSON format, showing a list of four products:

```
{
  "producto": [
    {
      "_id": "61159e2850ab1f5f0c9399fe",
      "name": "Mortal Kombat",
      "precio": "50000",
      "codigo": "MK",
      "description": "Peleas",
      "fecha": "2021-08-12T22:18:16.438Z",
      "__v": 0
    },
    {
      "_id": "61159e4f50ab1f5f0c939a01",
      "name": "Fifa 18",
      "precio": "30000",
      "codigo": "FF",
      "description": "Futbol",
      "fecha": "2021-08-12T22:18:55.001Z",
      "__v": 0
    },
    {
      "_id": "61159e7450ab1f5f0c939a04",
      "name": "Dark Souls 3",
      "precio": "70500",
      "codigo": "DS3",
      "description": "RPG",
      "fecha": "2021-08-12T22:19:32.384Z",
      "__v": 0
    },
    {
      "_id": "61159eab50ab1f5f0c939a07",
      "name": "Battlefield 3",
      "precio": "63000",
      "codigo": "BF3",
      "description": "FPS",
      "fecha": "2021-08-12T22:19:32.384Z",
      "__v": 0
    }
  ]
}
```

- Stocks:

POST http://localhost:3... GET http://localhost:30... + ...

http://localhost:3001/api/stock/listStock

GET http://localhost:3001/api/stock/listStock

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Query Params

KEY	VALUE
-----	-------

Body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize JSON

```
19 {
20   "_id": "61159f38e1a8ae3804c4daeb",
21   "productoId": {
22     "_id": "61159e2850ab1f5f0c9399fe",
23     "name": "Mortal Kombat",
24     "precio": "50000",
25     "codigo": "MK",
26     "description": "Peleas",
27     "fecha": "2021-08-12T22:18:16.438Z",
28     "__v": 0
29   },
30   "cantidad": "3",
31   "bodega": "Suba esquina",
32   "fecha": "2021-08-12T22:22:48.154Z",
33   "__v": 0
34 },
35 {
36   "_id": "61159f63cb284e67dccc3540",
37   "productoId": {
38     "_id": "61159ec950ab1f5f0c939a0a",
39     "name": "GTA 5",
40     "precio": "150000",
41     "codigo": "gta",
42     "description": "RPG",
43     "fecha": "2021-08-12T22:20:57.246Z",
44     "__v": 0
45   },
46   "cantidad": "1",
47   "bodega": "Engativa party",
48   "fecha": "2021-08-12T22:23:31.977Z",
49   "__v": 0
50 },
51 }
```

- Usuarios:

Search Postman

POST http://localhost:3001/api/usuario/listUsuario/d GET http://localhost:3001/api/usuario/listUsuario/d

GameStore / Test2-Usuarios / http://localhost:3001/api/usuario/listUsuario/d

GET http://localhost:3001/api/usuario/listUsuario/d

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Query Params

KEY	VALUE
Key	Value

Body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize JSON

```
1  {
2    "usuario": [
3      {
4        "_id": "61158b7f452e6a2084fd5779",
5        "name": "Diego",
6        "email": "diego23@gmail.com",
7        "password": "$2b$10$IInoJxqJ8BDUJe38V7n7YN0gIvp3TOMtrE0nd5M2H.TxUujg42/Zwa",
8        "__v": 0
9      },
10     {
11       "_id": "611595a078228a1e48531667",
12       "name": "Felipe",
13       "email": "feilpepro@gmail.com",
14       "password": "$2b$10$oe9B0rpj2YJ5fHfhKn48BejsfSCTarWI20ZIVYRdTYPgx0Dlyc17u0",
15       "__v": 0
16     },
17     {
18       "_id": "6115962778228a1e4853166a",
19       "name": "Sebastian",
20       "email": "seb123@gmail.com",
21       "password": "$2b$10$07B2EoS7ERf6ep14o.jmJuv0iq4.GgVS7nzXfIrFTnbdVM60ZSFWG",
22       "__v": 0
23     },
24     {
25       "_id": "6115964178228a1e4853166d",
26       "name": "Antonio",
27       "email": "antonio34@gmail.com",
28       "password": "$2b$10$ynhr0hZqU5CqJs0stHHLk0sctIq9t7wfHUG9NNQL10fHZqZJF5PyW",
29       "__v": 0
30     }
31   ]
}
```

- Ventas:

GameStore / Test4-Venta / http://localhost:3001/api/venta/listVenta

GET http://localhost:3001/api/venta/listVenta

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Query Params

KEY	VALUE
-----	-------

Body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize JSON ↕

```
2  "venta": [  
3    {  
4      "_id": "6115a04724a5b235f4512534",  
5      "codigo": "120",  
6      "productoId": {  
7        "_id": "61159e7450ab1f5f0c939a04",  
8        "name": "Dark Souls 3",  
9        "precio": "70500",  
10       "codigo": "DS3",  
11       "description": "RPG",  
12       "fecha": "2021-08-12T22:19:32.384Z",  
13       "__v": 0  
14     },  
15     "usuarioId": {  
16       "_id": "6115962778228a1e4853166a",  
17       "name": "Sebastian",  
18       "email": "seb123@gmail.com",  
19       "password": "$2b$10$07B2EoS7ERf6ep14o.jmJuv0iq4.GgVS7nzXfIrFTnbdVW60ZSFWG",  
20       "__v": 0  
21     },  
22     "precio": "50000",  
23     "fecha": "2021-08-12T22:27:19.027Z",  
24     "__v": 0  
25   },  
26   {  
27     "_id": "6115a068cec76c66f4e25fd5",  
28     "codigo": "121",  
29     "productoId": {  
30       "_id": "61159e7450ab1f5f0c939a04",  
31       "name": "Dark Souls 3",  
32       "precio": "70500",  
33       "codigo": "DS3",  
34       "description": "RPG",
```


BD Mongo:

- **Productos:**

gamestore.productos

[Documents](#) [Aggregations](#) [Schema](#) [Explain Plan](#) [Indexes](#) [Validation](#)

FILTER

{ field: 'value' }

ADD DATA

VIEW

```
_id: ObjectId("61159e2850ab1f5f0c9399fe")
name: "Mortal Kombat"
precio: "50000"
codigo: "MK"
description: "Peleas"
fecha: 2021-08-12T22:18:16.438+00:00
__v: 0
```

```
_id: ObjectId("61159e4f50ab1f5f0c939a01")
name: "Fifa 18"
precio: "30000"
codigo: "FF"
description: "Futbol"
fecha: 2021-08-12T22:18:55.001+00:00
__v: 0
```

```
_id: ObjectId("61159e7450ab1f5f0c939a04")
name: "Dark Souls 3"
precio: "70500"
codigo: "DS3"
description: "RPG"
fecha: 2021-08-12T22:19:32.384+00:00
__v: 0
```

```
_id: ObjectId("61159eab50ab1f5f0c939a07")
name: "Battlefield 3"
precio: "63000"
codigo: "B3"
description: "Disparos"
fecha: 2021-08-12T22:20:27.471+00:00
__v: 0
```

```
_id: ObjectId("61159ec950ab1f5f0c939a0a")
name: "GTA 5"
precio: "150000"
codigo: "gta"
description: "RPG"
fecha: 2021-08-12T22:20:57.246+00:00
__v: 0
```

- Stocks:

gamestore.stocks

Documents	Aggregations	Schema	Explain Plan	Indexes	Validation
FILTER { field: 'value' }					
ADD DATA VIEW					
<pre>_id: ObjectId("61159f0ce1a8ae3804c4dae7") productoId: ObjectId("61159e2850ab1f5f0c9399fe") cantidad: "2" bodega: "Kennedy32" fecha: 2021-08-12T22:22:04.006+00:00 __v: 0</pre>					
<pre>_id: ObjectId("61159f38e1a8ae3804c4daeb") productoId: ObjectId("61159e2850ab1f5f0c9399fe") cantidad: "3" bodega: "Suba esquina" fecha: 2021-08-12T22:22:48.154+00:00 __v: 0</pre>					
<pre>_id: ObjectId("61159f63cb284e67dccd3540") productoId: ObjectId("61159ec950ab1f5f0c939a0a") cantidad: "1" bodega: "Engativa party" fecha: 2021-08-12T22:23:31.977+00:00 __v: 0</pre>					
<pre>_id: ObjectId("61159f85cb284e67dccd3544") productoId: ObjectId("61159ec950ab1f5f0c939a0a") cantidad: "2" bodega: "Edificio corcel" fecha: 2021-08-12T22:24:05.803+00:00 __v: 0</pre>					
<pre>_id: ObjectId("61159fb53a9ad827549f2dd4") productoId: ObjectId("61159e7450ab1f5f0c939a04") cantidad: "14" bodega: "Castillo tirano" fecha: 2021-08-12T22:24:53.382+00:00 __v: 0</pre>					

- **Usuarios:**

MongoDB Compass - localhost:27017/gamestore.usuarios

Connect View Collection Help

Local

7 DBS 6 COLLECTIONS

☆ FAVORITE

HOST
localhost:27017

CLUSTER
Standalone

EDITION
MongoDB 5.0.1 Community

Q Filter your data

- > Diegodbscrumboard
- > admin
- > config
- > dbscrumboard
- ▼ gamestore
 - productos
 - stocks
 - usuarios
 - ventas
- > local
- > pruebadb
- > pruebadb2
- > retodbscrumboard

gamestore.usuarios

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER { field: 'value' }

ADD DATA VIEW

```
{
  "_id": ObjectId("61158b7f452e6a2084fd5779"),
  "name": "Diego",
  "email": "diego23@gmail.com",
  "password": "$2b$10$I0xqJ88DUJe38V7n7YNOgIvp3TOMtrEond5M2H.TxUujg42/Zwa",
  "__v": 0
}
```

```
{
  "_id": ObjectId("611595a078228a1e48531667"),
  "name": "Felipe",
  "email": "felipepro@gmail.com",
  "password": "$2b$10$oe9B0rpj2YJ5fHfKn48Bejsf5CTarWI20ZIVYRdTYPgxDlyc17u0",
  "__v": 0
}
```

```
{
  "_id": ObjectId("6115962778228a1e4853166a"),
  "name": "Sebastian",
  "email": "seb123@gmail.com",
  "password": "$2b$10$0782E0s7ERf6ep14o.jm3uvOiq4.GgVS7nzXfIrFTnbdVW60ZSFwG",
  "__v": 0
}
```

```
{
  "_id": ObjectId("6115964178228a1e4853166d"),
  "name": "Antonio",
  "email": "antonio34@gmail.com",
  "password": "$2b$10$ynhrDhZqu5CqJs0stHHLK0sctIq9t7wfHUG9NNQL10fHZqZJF5Pyw",
  "__v": 0
}
```

```
{
  "_id": ObjectId("6115968a78228a1e48531670"),
  "name": "Fredy",
  "email": "fredyandres@gmail.com",
  "password": "$2b$10$Mx753M9LejeRnJUw4JCE30BeHzMFCRn013kOVbGDKa92w1HVVo8TW",
  "__v": 0
}
```

- Ventas

gamestore.ventas

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER { field: 'value' }

ADD DATA



VIEW



```
_id: ObjectId("6115a04724a5b235f4512534")
codigo: "120"
productoId: ObjectId("61159e7450ab1f5f0c939a04")
usuarioId: ObjectId("6115962778228a1e4853166a")
precio: "50000"
fecha: 2021-08-12T22:27:19.027+00:00
__v: 0
```

```
_id: ObjectId("6115a068cec76c66f4e25fd5")
codigo: "121"
productoId: ObjectId("61159e7450ab1f5f0c939a04")
usuarioId: ObjectId("61158b7f452e6a2084fd5779")
precio: "55000"
fecha: 2021-08-12T22:27:52.538+00:00
__v: 0
```

```
_id: ObjectId("6115a0927c96d8681003dd90")
codigo: "122"
productoId: ObjectId("61159ec950ab1f5f0c939a0a")
usuarioId: ObjectId("61158b7f452e6a2084fd5779")
precio: "150000"
fecha: 2021-08-12T22:28:34.698+00:00
__v: 0
```

```
_id: ObjectId("6115a0b99f90e26b70a6a90f")
codigo: "123"
productoId: ObjectId("61159eab50ab1f5f0c939a07")
usuarioId: ObjectId("6115964178228a1e4853166d")
precio: "80000"
fecha: 2021-08-12T22:29:13.041+00:00
__v: 0
```

```
_id: ObjectId("6115a0d89c91d814407e5d9e")
codigo: "124"
productoId: ObjectId("61159eab50ab1f5f0c939a07")
usuarioId: ObjectId("6115962778228a1e4853166a")
precio: "45000"
fecha: 2021-08-12T22:29:44.968+00:00
__v: 0
```