

Documentación de API REST - Sistema de E-commerce con MongoDB y PostgreSQL

1. Introducción

Este documento describe el funcionamiento del sistema REST desarrollado con Spring Boot, el cual permite gestionar carritos de compra en MongoDB y realizar transacciones de compra en PostgreSQL. Se detallan los endpoints implementados y se proporcionan ejemplos de peticiones para demostrar su funcionamiento.

2. Tecnologías Utilizadas

- **Spring Boot** para la creación de la API REST.
- **MongoDB** para la gestión de carritos de compra.
- **PostgreSQL** para la gestión de las transacciones de compra.
- **Spring Data JPA y MongoDB** para la capa de persistencia.
- **Swagger/OpenAPI** para documentación interactiva.

3. Repositorio

Es posible encontrar el proyecto alojado en un repositorio de github en el siguiente link.

[Ecommerce-eval-mod5](#)

4. Endpoints y Demostración

4.1 Productos

4.1.1 Crear productos

POST /api/productos

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

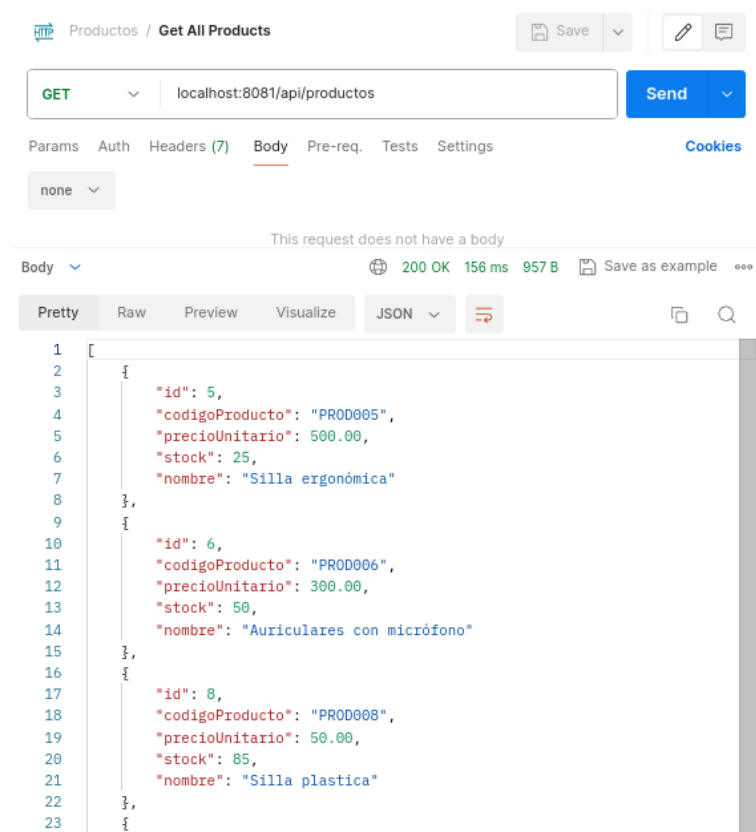
- URL:** localhost:8081/api/productos
- Method:** POST
- Body (raw):**

```
1 {  
2   ... "codigoProducto": "PROD010",  
3   ... "precioUnitario": 50,  
4   ... "stock": 85,  
5   ... "nombre": "Nuevo producto"  
6 }
```
- Response:** 201 Created, 249 ms, 261 B
- Body (pretty):**

```
1 {  
2   "id": 9,  
3   "codigoProducto": "PROD010",  
4   "precioUnitario": 50,  
5   "stock": 85,  
6   "nombre": "Nuevo producto"  
7 }
```

4.1.2 Consultar todos los productos

GET /api/productos



Productos / Get All Products

GET localhost:8081/api/productos

Send

Params Auth Headers (7) Body Pre-req. Tests Settings Cookies

none

This request does not have a body

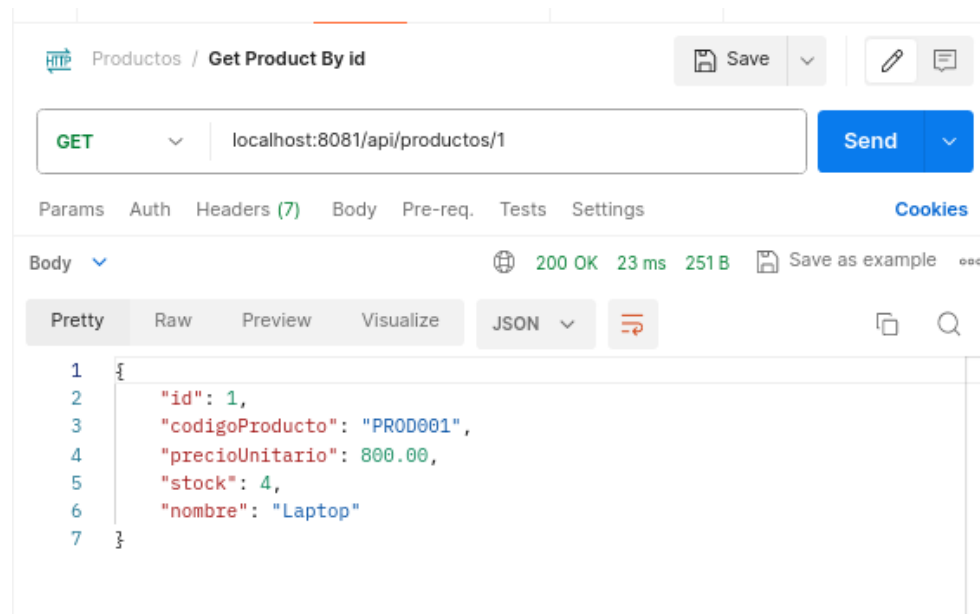
Body 200 OK 156 ms 957 B Save as example

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "id": 5,
4     "codigoProducto": "PROD005",
5     "precioUnitario": 500.00,
6     "stock": 25,
7     "nombre": "Silla ergonómica"
8   },
9   {
10    "id": 6,
11    "codigoProducto": "PROD006",
12    "precioUnitario": 300.00,
13    "stock": 50,
14    "nombre": "Auriculares con micrófono"
15  },
16  {
17    "id": 8,
18    "codigoProducto": "PROD008",
19    "precioUnitario": 50.00,
20    "stock": 85,
21    "nombre": "Silla plastica"
22  },
23 ]
```

4.1.3 Consultar producto por Id

GET /api/productos/{productId}



Productos / Get Product By id

GET localhost:8081/api/productos/1

Send

Params Auth Headers (7) Body Pre-req. Tests Settings Cookies

Body 200 OK 23 ms 251 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 1,
3   "codigoProducto": "PROD001",
4   "precioUnitario": 800.00,
5   "stock": 4,
6   "nombre": "Laptop"
7 }
```

4.2 Carrito

4.2.1 Crear carrito

POST /api/carritos

POST

localhost:8081/api/carritos

Send

Params Auth Headers (10) **Body** Pre-req. Tests Settings

raw

JSON

Beautify

```
1  {
2    "idUsuario": "usuario123",
3    "productos": [
4      {
5        "id": 7,
6        "codigoProducto": "PROD010"
7      },
8      {
9        "id": 8,
10       "codigoProducto": "PROD002"
11     }
12   ]
13 }
14
```

Body



201 Created

102 ms

345 B



Save as example



Pretty

Raw

Preview

Visualize

JSON



```
1  {
2    "id": "67ba52233209a20293079c40",
3    "idUsuario": "usuario123",
4    "productos": [
5      {
6        "id": 7,
7        "codigoProducto": "PROD010",
8        "cantidad": null
9      },
10     {
11       "id": 8,
12       "codigoProducto": "PROD002",
13       "cantidad": null
14     }
15   ]
16 }
```

4.2.2 Consultar todos los carritos

GET /api/carritos

HTTP Carrito / Get Carritos

GET localhost:8081/api/carritos Send

Params Auth Headers (10) Body Pre-req. Tests Settings

Query Params

Key	Value	Desc...	Bulk Edit
-----	-------	---------	-----------

Body 200 OK 36 ms 855 B Save as example

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "id": "679465a41b92ca596c260061",
4     "idUsuario": "usuario123",
5     "productos": [
6       {
7         "id": 1,
8         "codigoProducto": "PROD001",
9         "cantidad": 2
10      },
11      {
12        "id": 2,
13        "codigoProducto": "PROD002",
14        "cantidad": 1
15      }
16    ]
17  },
18  {
19    "id": "67986d6a81b39c5335a0985b",
20    "idUsuario": "usuario456",
21    "productos": [
22      {
23        "id": 3,
24        "codigoProducto": "PROD003",
25        "cantidad": 1
26      },
27    ]
28  }
29 ]
```

4.2.3 Consultar carrito por Id de usuario

GET /api/carritos/{usuarioId}

Carrito / Get Carritos por Id de usuario

GET localhost:8081/api/carritos/usuario789

Send

Params Auth Headers (10) Body Pre-req. Tests Settings

Body 200 OK 12 ms 334 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": "67986d7d81b39c5335a0985c",
3   "idUsuario": "usuario789",
4   "productos": [
5     {
6       "id": 5,
7       "codigoProducto": "PROD005",
8       "cantidad": 1
9     },
10    {
11      "id": 6,
12      "codigoProducto": "PROD006",
13      "cantidad": 1
14    }
15  ]
16 }
```

4.2.4 Consultar producto en carrito por ID de usuario y ID de producto

GET /api/carritos/productoById/{usuarioId}?productoId={productoId}

Carrito / Get Producto By id Carrito

GET localhost:8081/api/carritos/productoById/usuario123?productoId=2

Send

Params Auth Headers (8) Body Pre-req. Tests Settings

raw JSON

1

Body 200 OK 39 ms 212 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 2,
3   "codigoProducto": "PROD002",
4   "cantidad": 1
5 }
```

4.3 Orden de compra

4.3.1 Generar orden de compra

POST /api/ordenCompra

Orden Compra / Generar Orden Compra

POST localhost:8081/api/ordenCompra/usuario456

Send

Params Auth Headers (10) Body Pre-req. Tests Settings Cookies

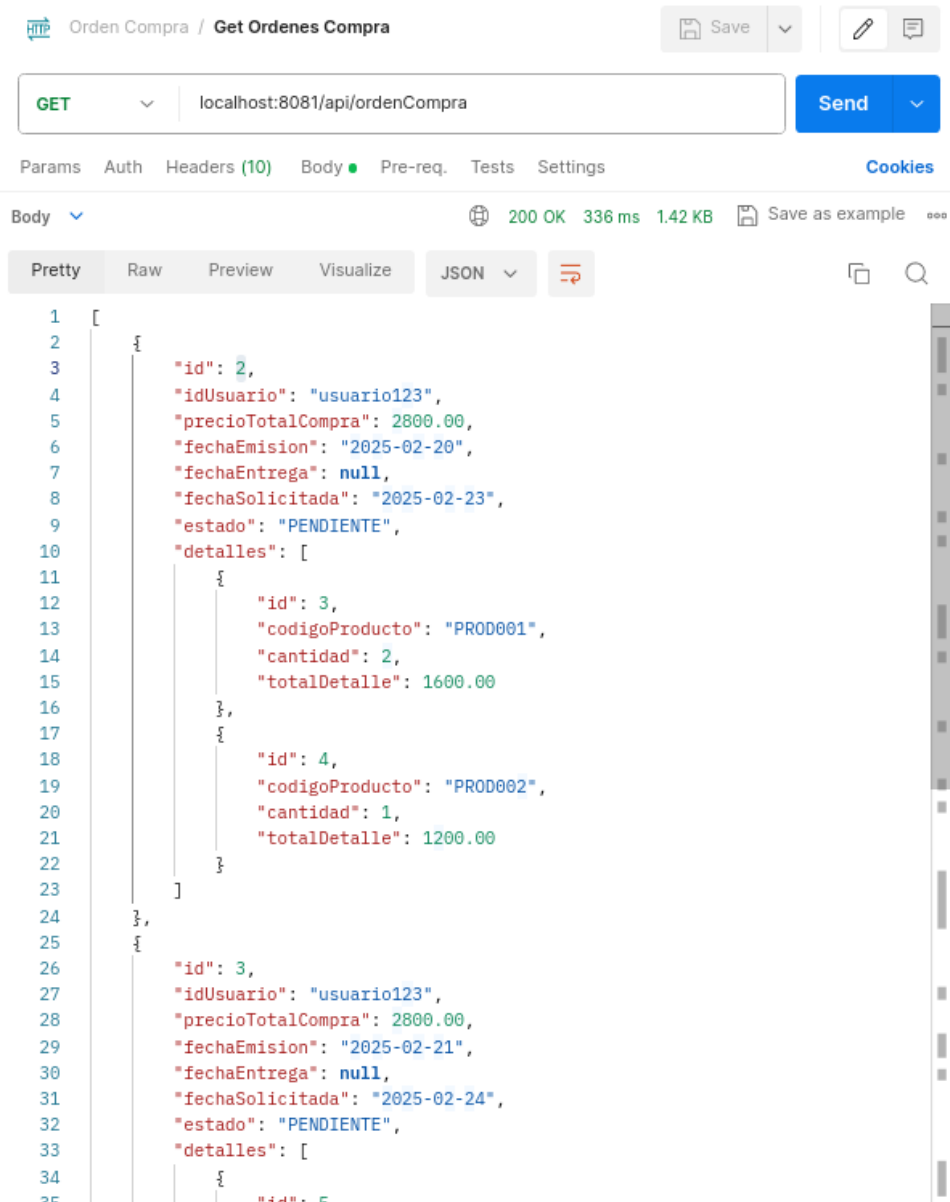
body 201 Created 124 ms 482 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 5,
3   "idUsuario": "usuario456",
4   "precioTotalCompra": 80.00,
5   "fechaEmision": "2025-02-22",
6   "fechaEntrega": null,
7   "fechaSolicitada": "2025-02-25",
8   "estado": "PENDIENTE",
9   "detalles": [
10    {
11      "id": 9,
12      "codigoProducto": "PROD003",
13      "cantidad": 1,
14      "totalDetalle": 50.00
15    },
16    {
17      "id": 10,
18      "codigoProducto": "PROD004",
19      "cantidad": 1,
20      "totalDetalle": 30.00
21    }
22  ]
23 }
```

4.3.2 Consultar todas las ordenes de compra

GET /api/ordenCompra



Orden Compra / Get Ordenes Compra

GET localhost:8081/api/ordenCompra

Send

Params Auth Headers (10) Body ● Pre-req. Tests Settings Cookies

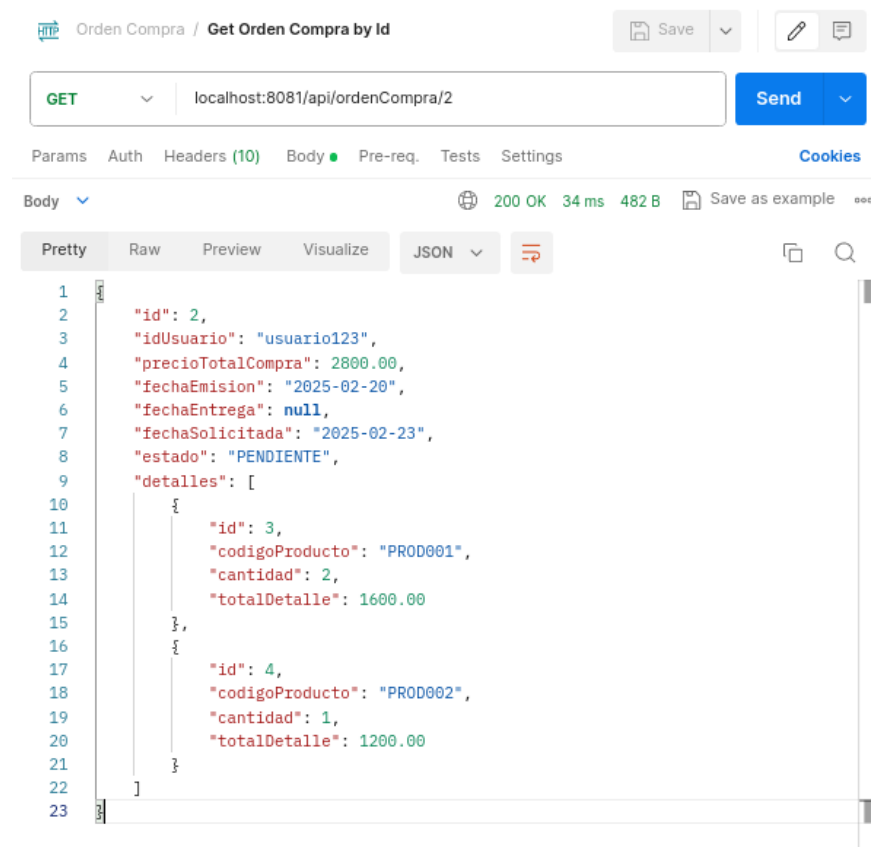
Body 200 OK 336 ms 1.42 KB Save as example

Pretty Raw Preview Visualize JSON

```
1  [
2    {
3      "id": 2,
4      "idUsuario": "usuario123",
5      "precioTotalCompra": 2800.00,
6      "fechaEmision": "2025-02-20",
7      "fechaEntrega": null,
8      "fechaSolicitada": "2025-02-23",
9      "estado": "PENDIENTE",
10     "detalles": [
11       {
12         "id": 3,
13         "codigoProducto": "PROD001",
14         "cantidad": 2,
15         "totalDetalle": 1600.00
16       },
17       {
18         "id": 4,
19         "codigoProducto": "PROD002",
20         "cantidad": 1,
21         "totalDetalle": 1200.00
22       }
23     ]
24   },
25   {
26     "id": 3,
27     "idUsuario": "usuario123",
28     "precioTotalCompra": 2800.00,
29     "fechaEmision": "2025-02-21",
30     "fechaEntrega": null,
31     "fechaSolicitada": "2025-02-24",
32     "estado": "PENDIENTE",
33     "detalles": [
34       {
35         "id": 5,
```


4.3.3 Consultar orden de compra por Id

GET /api/ordenCompra/{ordenCompraId}



Orden Compra / Get Orden Compra by Id

GET localhost:8081/api/ordenCompra/2

Send

Params Auth Headers (10) Body Pre-req. Tests Settings Cookies

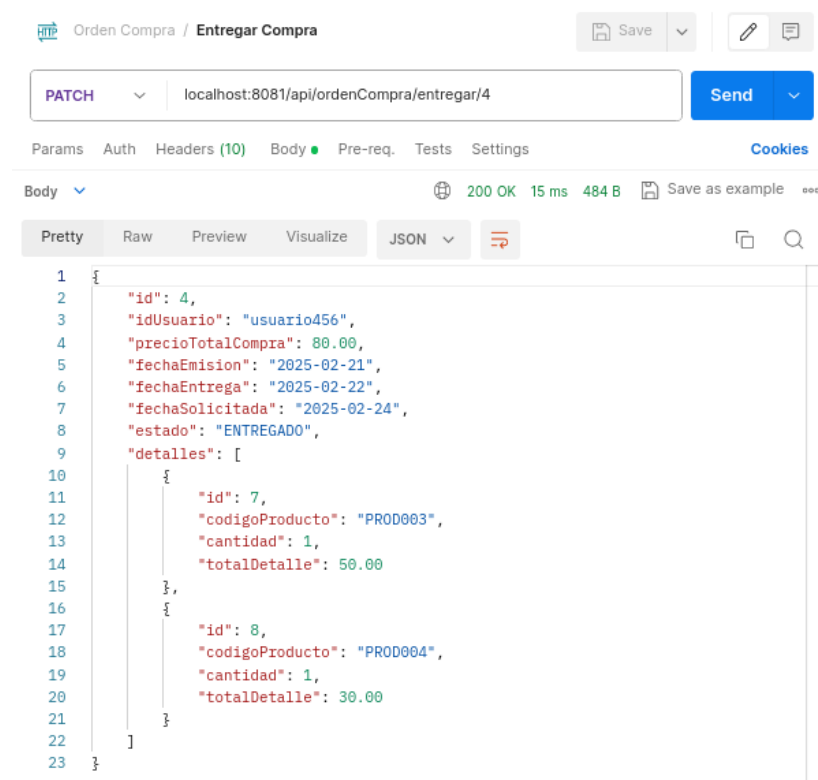
Body 200 OK 34 ms 482 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 2,
3   "idUsuario": "usuario123",
4   "precioTotalCompra": 2800.00,
5   "fechaEmision": "2025-02-20",
6   "fechaEntrega": null,
7   "fechaSolicitada": "2025-02-23",
8   "estado": "PENDIENTE",
9   "detalles": [
10    {
11      "id": 3,
12      "codigoProducto": "PROD001",
13      "cantidad": 2,
14      "totalDetalle": 1600.00
15    },
16    {
17      "id": 4,
18      "codigoProducto": "PROD002",
19      "cantidad": 1,
20      "totalDetalle": 1200.00
21    }
22  ]
23 }
```

3.3.4 Marcar estado de orden de compra como entregada

PATCH /api/productos/entregar/{ordenCompraId}



Orden Compra / Entregar Compra

PATCH localhost:8081/api/ordenCompra/entregar/4

Send

Params Auth Headers (10) Body Pre-req. Tests Settings Cookies

Body 200 OK 15 ms 484 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 4,
3   "idUsuario": "usuario456",
4   "precioTotalCompra": 80.00,
5   "fechaEmision": "2025-02-21",
6   "fechaEntrega": "2025-02-22",
7   "fechaSolicitada": "2025-02-24",
8   "estado": "ENTREGADO",
9   "detalles": [
10    {
11      "id": 7,
12      "codigoProducto": "PROD003",
13      "cantidad": 1,
14      "totalDetalle": 50.00
15    },
16    {
17      "id": 8,
18      "codigoProducto": "PROD004",
19      "cantidad": 1,
20      "totalDetalle": 30.00
21    }
22  ]
23 }
```

5. Conclusión

Se ha desarrollado una API REST funcional que permite manejar carritos de compra en MongoDB y realizar transacciones en PostgreSQL. La documentación presentada demuestra mediante Postman, el correcto funcionamiento de los endpoints principales del sistema.