Universidad de las fuerzas armadas espe

Nombre: Ariel llumiquinga

Fecha:06/02/2026

Actividad

La empresa TecnoMega requiere un prototipo de API para compra/venta de productos tecnológicos, usando Redis como base NoSQL clave–valor para almacenar catálogos y transacciones.

Objetivo general

Construir una API REST en Node.js + Express que:

1. Use middleware para parsear JSON (express.json()).
2. Use middleware response-time para medir tiempo de respuesta.
3. Implemente endpoints SET y GET contra Redis.
4. Cargue 10 registros por cada "tabla" desde un archivo JSON (clave–valor) y los almacene en Redis.

Modelo de datos (tablas en Redis)

Se simulan 4 "tablas" (colecciones):

1. clientes
2. productos
3. pedidos
4. detalle_pedido
5. Si el estudiante quiere agregar más colecciones lo puede realizar

Reglas mínimas

- **productos**: deben incluir código, nombre, categoría, precio, stock.
- **clientes**: deben incluir cedula o dni, nombres, email,teléfono,edad,genero
- **pedidos**: deben incluir Código , clienteId, fecha,subtotal,iva, total,estado.
- **detalle_pedido**: debe incluir Código, productoId, cantidad,detalle, precioUnit.

Entregables

1. Repositorio con: https://github.com/arielll669/Leccion-Redis3U-BDA.git
- index.js (API)

- data/tecnomega.json (10 registros por tabla mínimo)

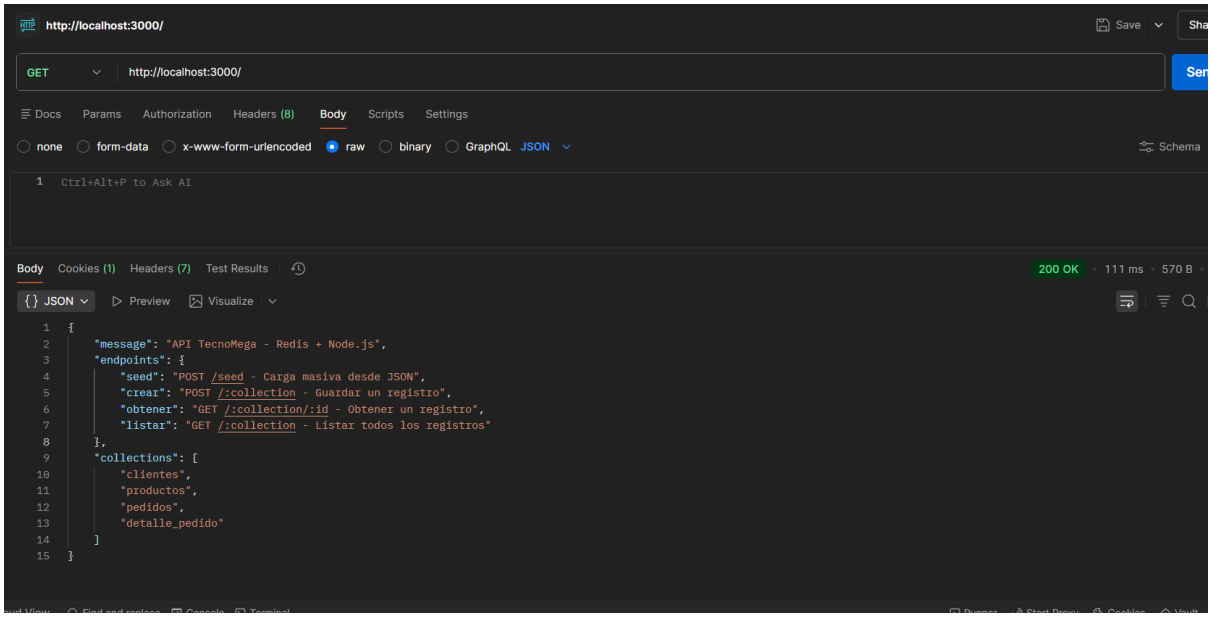
2. Evidencia:

- Capturas de Postman/Thunder Client
- Salida del tiempo de respuesta (Response-Time)

DESARROLLO

PRUEBA 1: Verifica que la API está corriendo

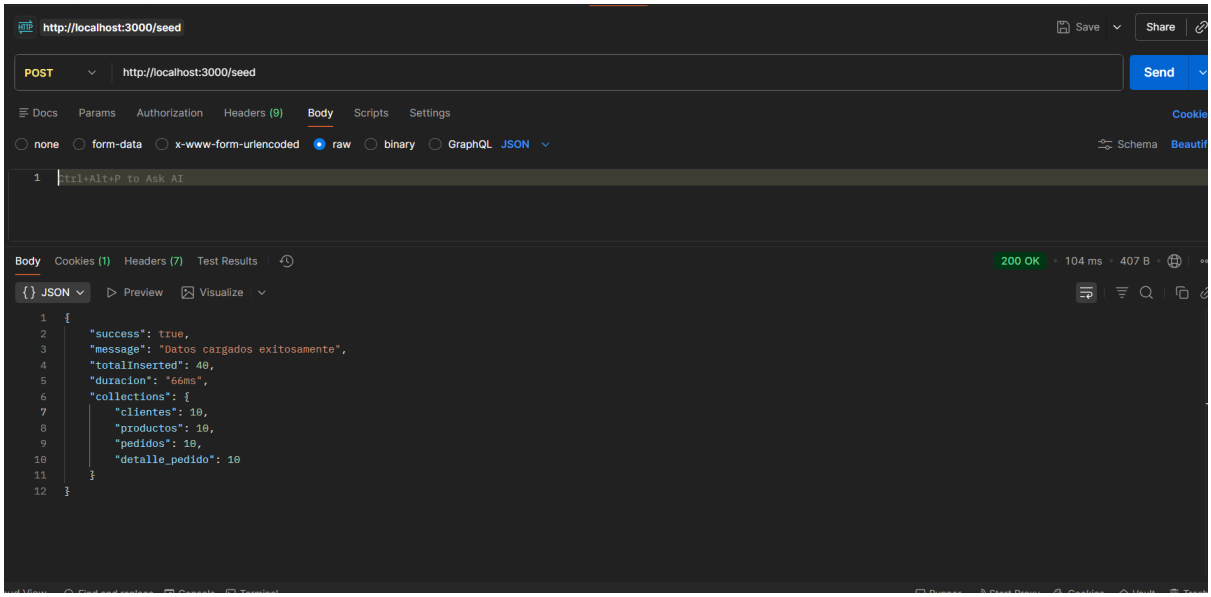
1. Método: **GET**
2. URL: http://localhost:3000/



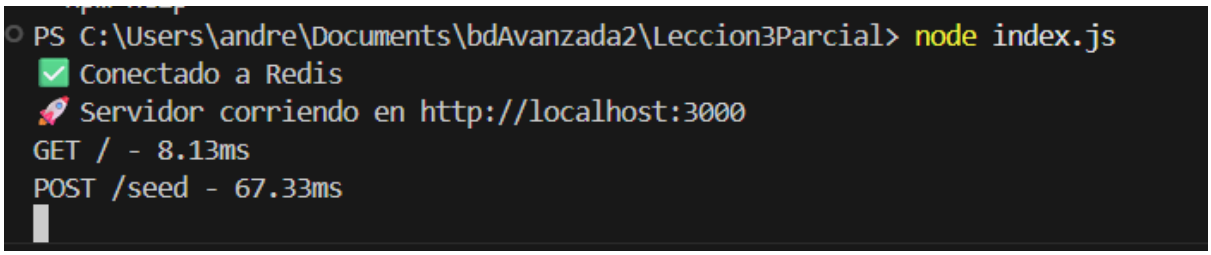
PRUEBA 2: Cargar datos masivos (SEED)

carga todos los datos del JSON a Redis.

1. Método: **POST**
2. URL: http://localhost:3000/seed

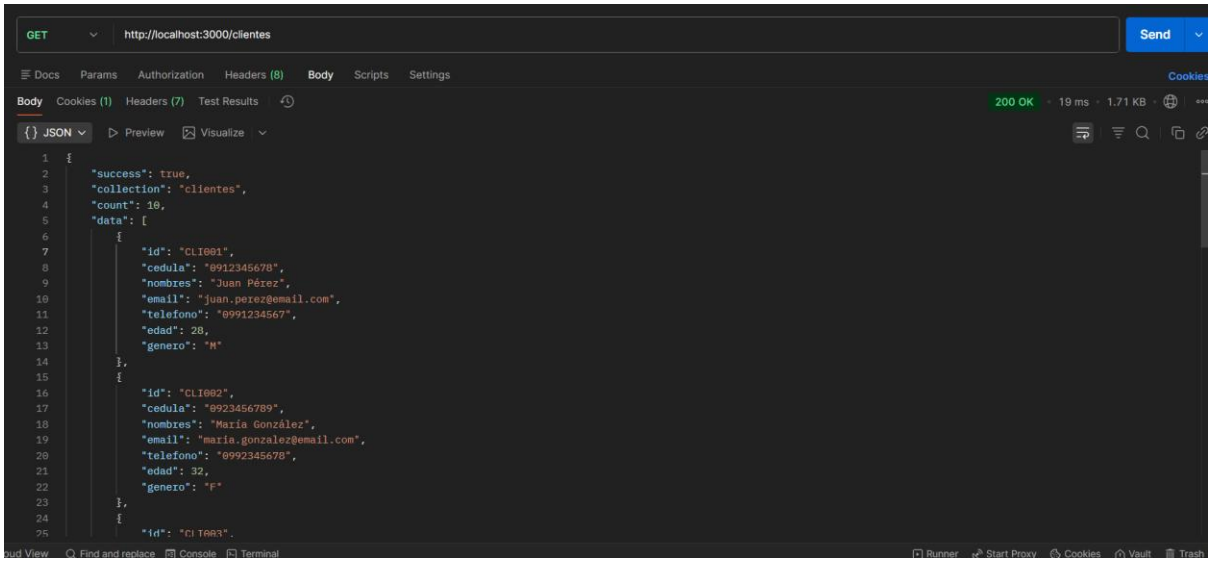


Terminal de vsc:



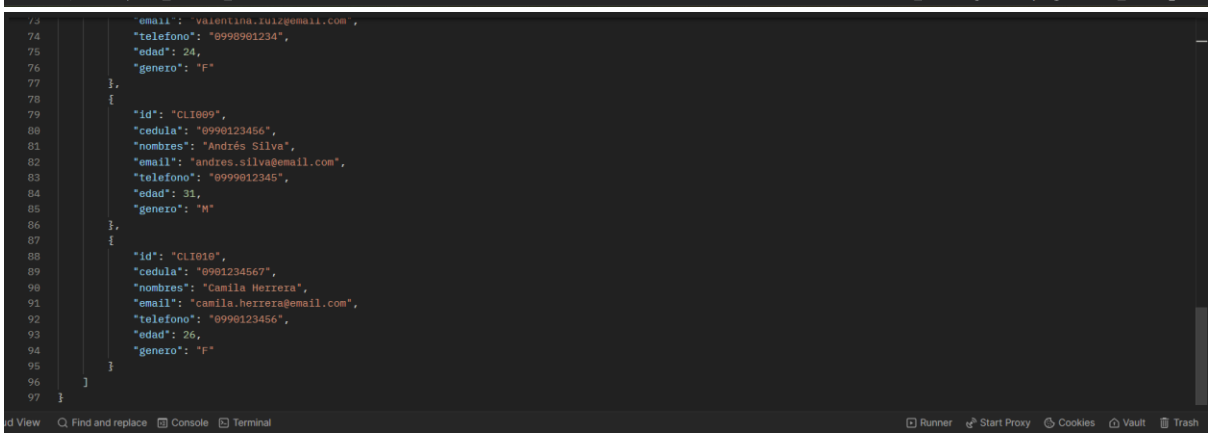
PRUEBA 3: Listar todos los clientes (GET)

1. Método: **GET**
2. URL: http://localhost:3000/clientes


GET
http://localhost:3000/clientes
Send
Docs
Params
Authorization
Headers (8)
Body
Scripts
Settings
Cookies
Body
Cookies (1)
Headers (7)
Test Results
200 OK
19 ms
1.71 KB
{ } JSON
Preview
Visualize
{
"success": true,
"collection": "clientes",
"count": 10,
"data": [
{
"id": "CLI001",
"cedula": "0912345678",
"nombres": "Juan Pérez",
"email": "juan.perez@email.com",
"telefono": "0991234567",
"edad": 28,
"genero": "M"
},
{
"id": "CLI002",
"cedula": "0923456789",
"nombres": "María González",
"email": "maria.gonzalez@email.com",
"telefono": "0992345678",
"edad": 32,
"genero": "F"
},
{
Find and replace
Console
Terminal
Runner
Start Proxy
Cookies
Vault
Trash

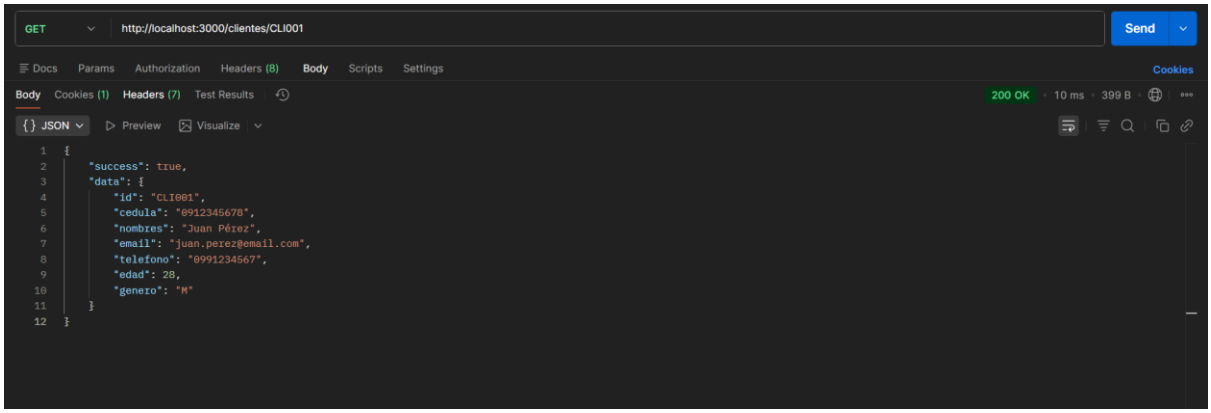

"id": "CLI003",
"cedula": "0934567890",
"nombres": "Carlos Rodríguez",
"email": "carlos.rodriguez@email.com",
"telefono": "0993456789",
"edad": 25,
"genero": "M"
},
{
"id": "CLI004",
"cedula": "0945678901",
"nombres": "Ana Martínez",
"email": "ana.martinez@email.com",
"telefono": "0994567890",
"edad": 30,
"genero": "F"
},
{
"id": "CLI005",
"cedula": "0956789012",
"nombres": "Luis Fernández",
"email": "luis.fernandez@email.com",
"telefono": "0995678901",
"edad": 35,
Find and replace
Console
Terminal
Runner
Start Proxy
Cookies
Vault
Trash


"cedula": "0967890123",
"nombres": "Sofía López",
"email": "sofia.lopez@email.com",
"telefono": "0996789012",
"edad": 27,
"genero": "F"
},
{
"id": "CLI007",
"cedula": "0978901234",
"nombres": "Diego Torres",
"email": "diego.torres@email.com",
"telefono": "0997890123",
"edad": 29,
"genero": "M"
},
{
"id": "CLI008",
"cedula": "0989012345",
"nombres": "Valentina Ruiz",
"email": "valentina.ruiz@email.com",
"telefono": "0998901234",
"edad": 24,
"genero": "F"
Find and replace
Console
Terminal
Runner
Start Proxy
Cookies
Vault
Trash


"email": "valentina.ruiz@email.com",
"telefono": "0998901234",
"edad": 24,
"genero": "F"
},
{
"id": "CLI009",
"cedula": "0990123456",
"nombres": "Andrés Silva",
"email": "andres.silva@email.com",
"telefono": "0999012345",
"edad": 31,
"genero": "M"
},
{
"id": "CLI010",
"cedula": "0901234567",
"nombres": "Camila Herrera",
"email": "camila.herrera@email.com",
"telefono": "0990123456",
"edad": 26,
"genero": "F"
}
]
}
Find and replace
Console
Terminal
Runner
Start Proxy
Cookies
Vault
Trash

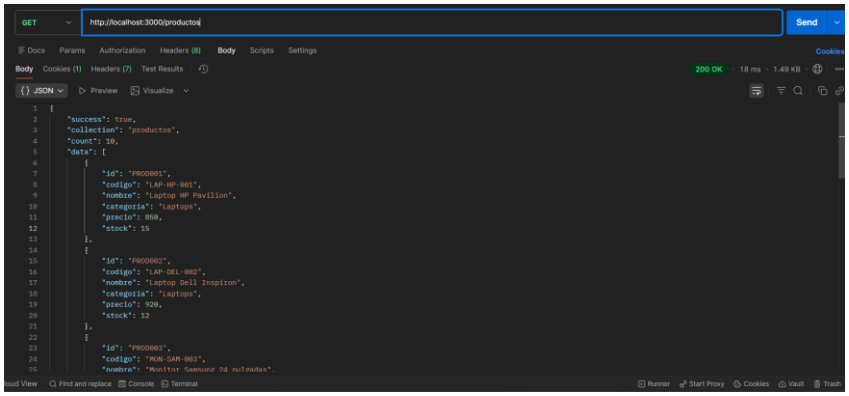
PRUEBA 4: Obtener un cliente específico (GET)

1. Método: **GET**
2. URL: http://localhost:3000/clientes/CLI001



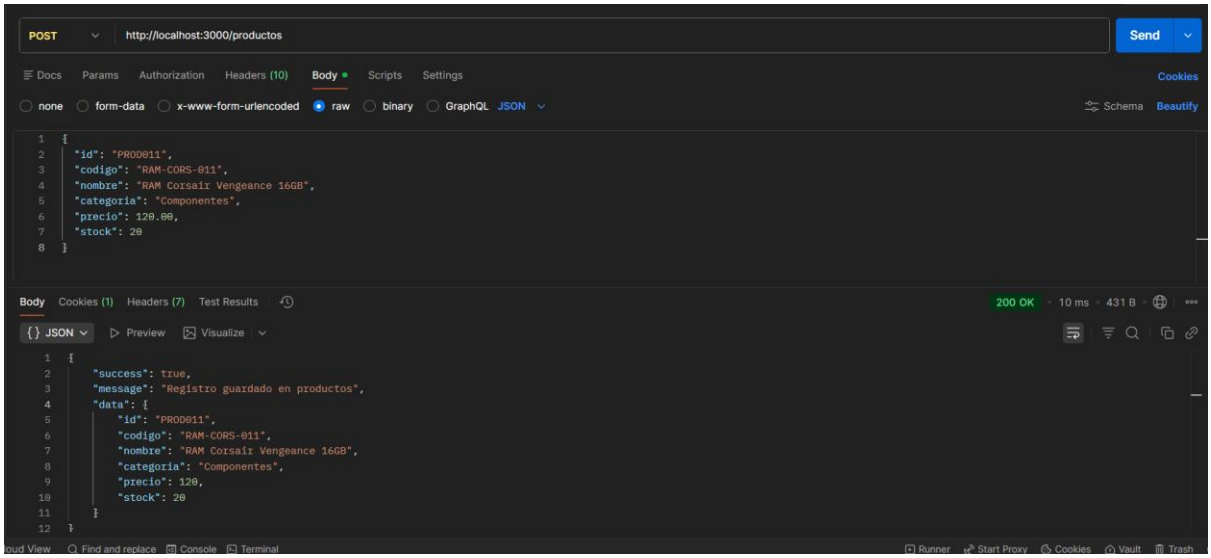
PRUEBA 5: Listar todos los productos (GET)

1. Método: **GET**
2. URL: http://localhost:3000/productos



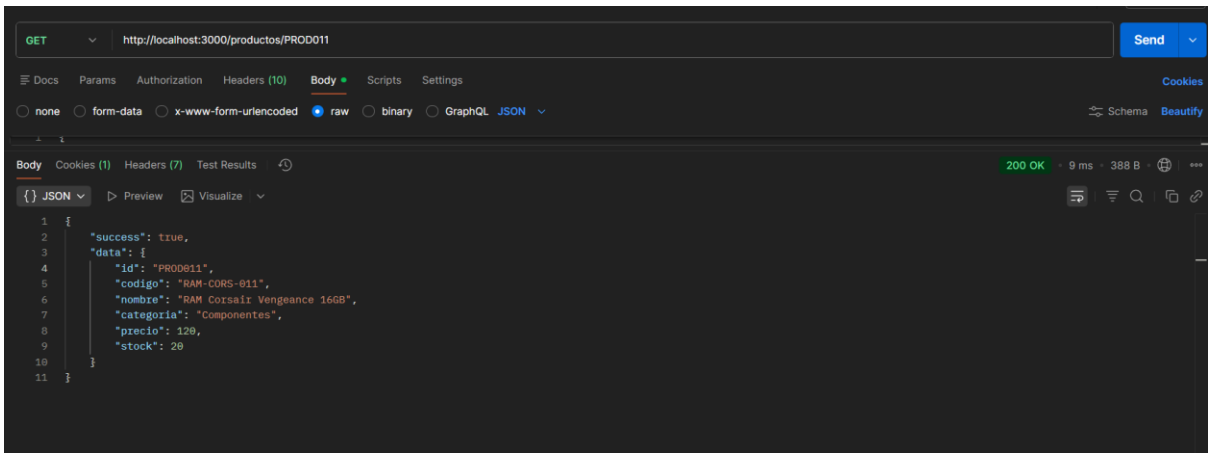
PRUEBA 6: Crear un nuevo producto (POST)

1. Método: **POST**
2. URL: http://localhost:3000/productos



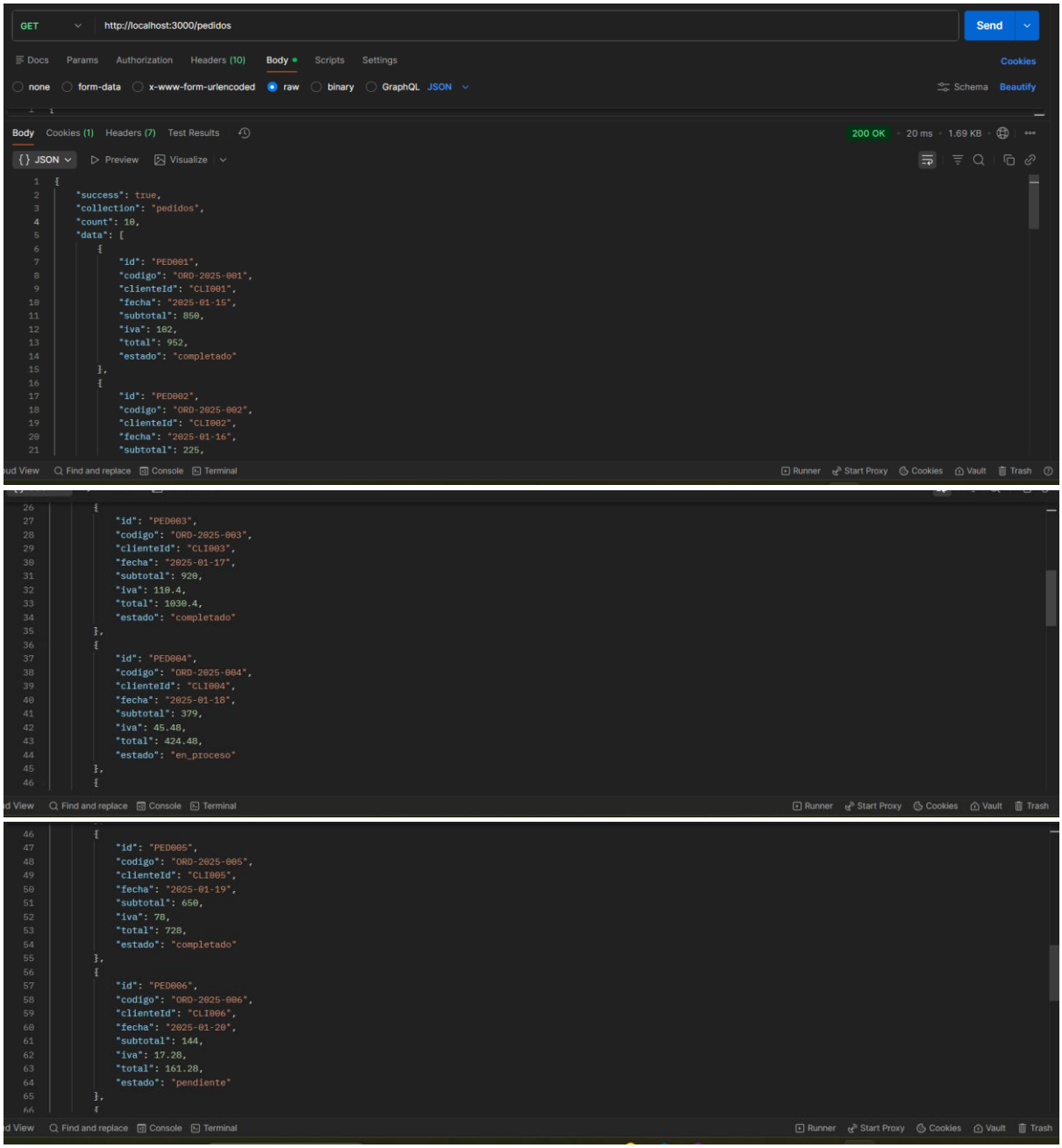
PRUEBA 7: Verificar que el nuevo producto se guardó

1. Método: **GET**
2. URL: http://localhost:3000/productos/PROD011



PRUEBA 8: Listar todos los pedidos

1. Método: **GET**
2. URL: http://localhost:3000/pedidos

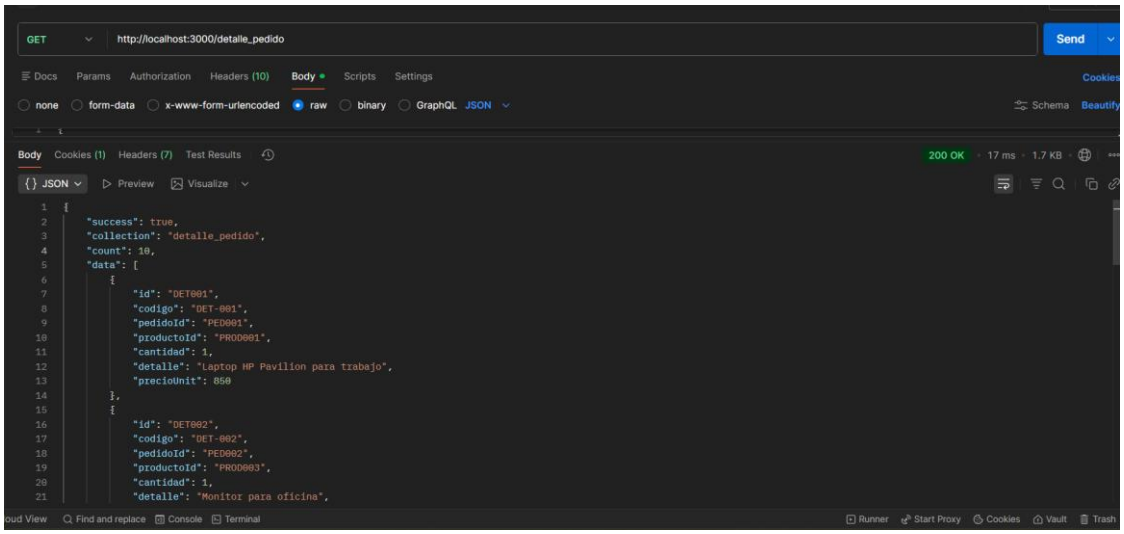

GET
http://localhost:3000/pedidos
Send
Docs
Params
Authorization
Headers (10)
Body
Scripts
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Schema
Beautify
Body
Cookies (1)
Headers (7)
Test Results
200 OK
20 ms
1.69 KB
JSON
Preview
Visualize
{
"success": true,
"collection": "pedidos",
"count": 10,
"data": [
{
"id": "PED001",
"codigo": "ORD-2025-001",
"clienteId": "CLI001",
"fecha": "2025-01-15",
"subtotal": 850,
"iva": 102,
"total": 952,
"estado": "completado"
},
{
"id": "PED002",
"codigo": "ORD-2025-002",
"clienteId": "CLI002",
"fecha": "2025-01-16",
"subtotal": 225,
Find and replace
Console
Terminal
Runner
Start Proxy
Cookies
Vault
Trash

{
"id": "PED003",
"codigo": "ORD-2025-003",
"clienteId": "CLI003",
"fecha": "2025-01-17",
"subtotal": 920,
"iva": 110.4,
"total": 1030.4,
"estado": "completado"
},
{
"id": "PED004",
"codigo": "ORD-2025-004",
"clienteId": "CLI004",
"fecha": "2025-01-18",
"subtotal": 379,
"iva": 45.48,
"total": 424.48,
"estado": "en_proceso"
},
{
Find and replace
Console
Terminal
Runner
Start Proxy
Cookies
Vault
Trash

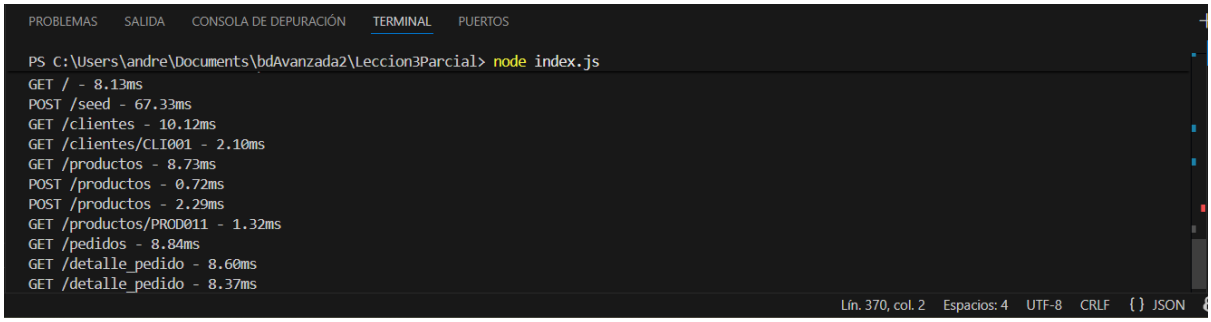
{
"id": "PED005",
"codigo": "ORD-2025-005",
"clienteId": "CLI005",
"fecha": "2025-01-19",
"subtotal": 650,
"iva": 78,
"total": 728,
"estado": "completado"
},
{
"id": "PED006",
"codigo": "ORD-2025-006",
"clienteId": "CLI006",
"fecha": "2025-01-20",
"subtotal": 144,
"iva": 17.28,
"total": 161.28,
"estado": "pendiente"
},
Find and replace
Console
Terminal
Runner
Start Proxy
Cookies
Vault
Trash

PRUEBA 9: Listar todos los detalles de pedido

1. Método: **GET**
2. URL: http://localhost:3000/detalle_pedido



Captura de la Terminal:



Evidencias Ubuntu

Ver todas las claves almacenadas:

```
andre@BEYKER:~$ redis-cli
127.0.0.1:6379> ping
PONG
127.0.0.1:6379> KEYS *
 1) "clientes:CLI005"
 2) "pedidos:PED010"
 3) "productos:PROD011"
 4) "estudiante:"
 5) "estudiante:1724966849"
 6) "pedidos:PED003"
 7) "detalle_pedido:DET004"
 8) "clientes:CLI003"
 9) "estudiante:1850598010"
10) "productos:PROD003"
11) "detalle_pedido:DET005"
12) "estudiante:1725600819"
13) "clientes:CLI001"
14) "pedidos:PED001"
15) "detalle_pedido:DET002"
16) "clientes:index"
17) "productos:PROD004"
18) "detalle_pedido:index"
19) "empleado:1"
20) "estudiante:1754459160"
21) "detalle_pedido:DET001"
22) "clientes:CLI002"
23) "estudiante:1722651567"
24) "clientes:CLI007"
25) "clientes:CLI009"
26) "estudiante:1723167514"
27) "key"
28) "detalle_pedido:DET003"
29) "test"
30) "productos:PROD005"
31) "estudiante:1850677764"
32) "clientes:CLI008"
33) "user:1002"
34) "productos:PROD009"
35) "pedidos:PED007"
36) "pedidos:PED006"
37) "estudiante:1727641407"
```

Ver un registro específico (JSON)

JSON almacenado del cliente

```
127.0.0.1:6379> GET clientes:CLI001
"{\"id\":\"CLI001\",\"cedula\":\"0912345678\",\"nombres\":\"Jua
n P\xc3\xa9rez\",\"email\":\"juan.perez@email.com\",\"telefono\
":\"0991234567\",\"edad\":28,\"genero\":\"M\"}"
```

Ver el contenido de un índice (Set)

todos los IDs de clientes

```
127.0.0.1:6379> SMEMBERS clientes:index
 1) "CLI001"
 2) "CLI002"
 3) "CLI003"
 4) "CLI004"
 5) "CLI005"
 6) "CLI006"
 7) "CLI007"
 8) "CLI008"
 9) "CLI009"
10) "CLI010"
```

Contar cuántas claves hay

total de claves

```
127.0.0.1:6379> DBSIZE
(integer) 72
```