

Sistemas Transaccionales

Universidad de los Andes



Integrantes:

Yair Andrade Pechene, Camilo Castilla, Santiago
Guevara

Objetivos

- Desarrollar habilidades en el proceso de generación de un modelo de datos en bases de datos documentales.
- Incorporar la validación del esquema del modelo de datos en MongoDB.
- Implementar una aplicación informática de mediana complejidad que involucre bases de datos documentales en MongoDB.

Descripción General

Superandes es una cadena de supermercados con el objetivo de facilitar la adquisición de productos para el consumidor final. La compañía cuenta con varias sucursales estratégicamente distribuidas en ciudades clave, basándose en estudios de oferta y demanda. Cada sucursal es única y tiene diferentes características como nombre, tamaño y ubicación, lo cual le permite adaptarse a las necesidades locales.

Para gestionar sus operaciones, Superandes maneja una compleja estructura de inventario y logística, incluyendo información detallada de los productos en almacenes o bodegas, categorías de productos, proveedores y niveles de reorden para asegurar la disponibilidad constante de mercancía en las sucursales.

Cada bodega almacena productos específicos con un registro de su cantidad, costo promedio y capacidad. Además, las sucursales pueden emitir órdenes de compra cuando el inventario baja de ciertos niveles, lo cual activa un proceso de reposición a través de los proveedores.

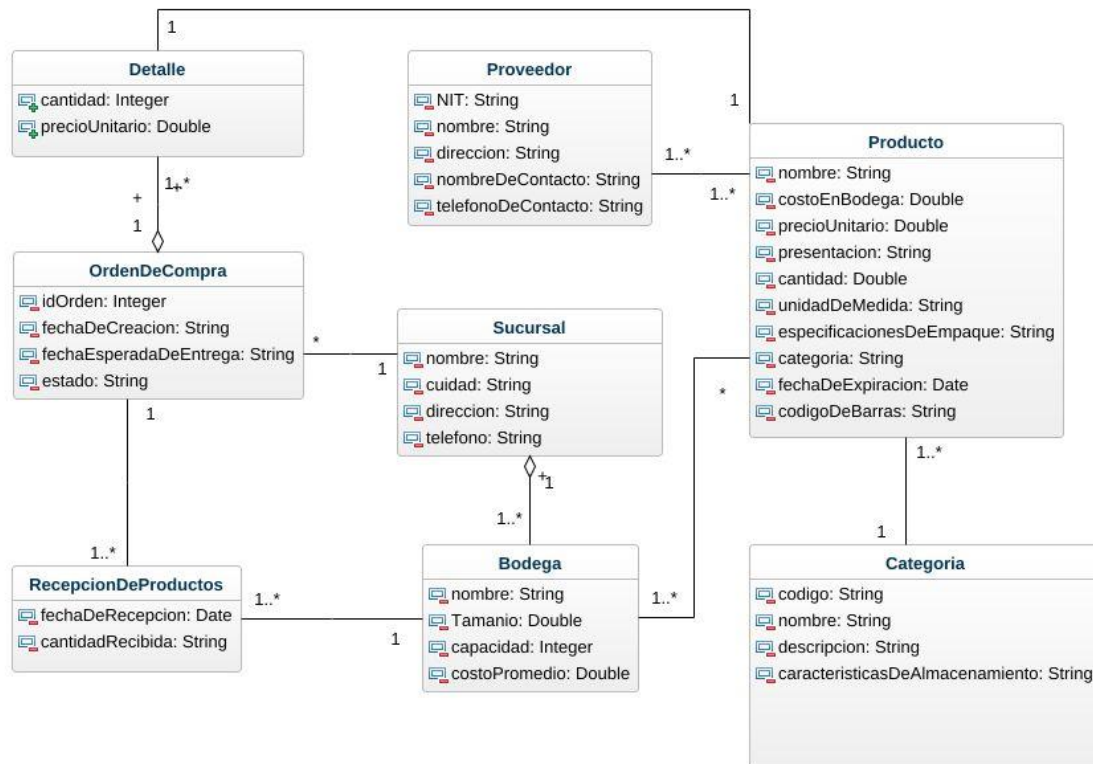
Análisis y Modelo conceptual

Identificación de los elementos fundamentales del negocio

Los elementos identificados según el contexto del problema son:

1. **Sucursal:** Representa las tiendas físicas en distintas ciudades.
2. **Bodega:** Espacios de almacenamiento en cada sucursal.
3. **Producto:** Artículos disponibles para la venta con información como precio, presentación, y categoría.
4. **Categoría:** Clasificación de productos (perecederos, no perecederos, etc.).
5. **Proveedor:** Empresas que suministran productos.
6. **Orden de Compra:** Solicitudes realizadas por sucursales a proveedores.
7. **Recepción de Productos:** Registro de productos entregados por proveedores.

Modelo UML



El diagrama UML presenta la estructura del sistema desarrollado para el caso de estudio de SuperAndes, específicamente para la gestión de inventarios, órdenes de compra, proveedores, y productos en el contexto de una base de datos NoSQL utilizando MongoDB. A continuación, se describe brevemente el diagrama y las principales entidades modeladas:

1. Sucursal

La clase Sucursal representa una sucursal dentro de la empresa. Cada sucursal tiene los atributos básicos como el nombre, dirección, ciudad, y teléfono. Además, cada sucursal puede tener asociadas múltiples bodegas, lo que se refleja en la relación uno a muchos con la clase Bodega.

2. Bodega

La clase Bodega representa los almacenes o bodegas donde se guardan los productos. Una bodega está asociada a una sola sucursal (relación muchos a uno con Sucursal). La bodega tiene atributos como nombre, tamaño y sucursalId, y almacena un conjunto de productos mediante la clase ProductoBodega, que incluye información sobre los productos almacenados, como la cantidad disponible y el costoPromedio.

3. Producto

La clase Producto representa los productos gestionados en el sistema. Cada producto tiene atributos como nombre, precioUnitario, presentacion, categoriaId, codigoBarras, y fechaExpiracion. Los productos están asociados a una categoría mediante el atributo categoriaId, lo que permite la clasificación de los productos en diferentes categorías. Además, un producto puede estar presente en varias bodegas, lo que se refleja en la relación con la clase ProductoBodega.

4. Proveedor

La clase Proveedor almacena la información sobre los proveedores a los cuales la empresa adquiere productos. Los atributos principales incluyen nit, nombre, direccion, nombreContacto, y telefonoContacto. Un proveedor puede estar asociado a múltiples órdenes de compra, lo que se refleja en la relación uno a muchos con la clase OrdenDeCompra.

5. OrdenDeCompra

La clase OrdenDeCompra modela las órdenes de compra realizadas por las sucursales. Cada orden de compra está asociada a una sucursal y a un proveedor, y tiene una lista de productos en el detalle. Los atributos clave de la orden son fechaCreacion, estado, sucursalId, y proveedorId. El detalle de la orden, representado por la clase DetalleOrden, incluye información sobre los productos solicitados, las cantidades y los precios.

6. Categoría

La clase Categoría representa las categorías a las que pertenecen los productos. Cada categoría tiene un codigo y un nombre, y está asociada a múltiples productos a través del atributo categoriaId en la clase Producto.

Diseño de la Base de Datos

1. Identificación de Entidades y Atributos

Primero identificamos las entidades fundamentales que forman parte del negocio:

Entidades:

1. Sucursal

- a. **Atributos:** id, nombre, direccion, ciudad, telefono.
- b. **Operaciones:** CRUD (Crear, Leer, Actualizar, Eliminar).

2. Bodega

- a. **Atributos:** id, nombre, tamaño, sucursalId, productos.
- b. **Operaciones:** CRUD (Crear, Leer, Actualizar, Eliminar).

3. Proveedor

- a. **Atributos:** id, nit, nombre, direccion, nombreContacto, telefonoContacto.
- b. **Operaciones:** Crear, Leer, Actualizar.

4. Categoría de Producto

- a. **Atributos:** id, nombre, descripcion.
- b. **Operaciones:** Crear, Leer.

5. Producto

- a. **Atributos:** id, nombre, precioUnitario, presentacion, categoriaId, codigoBarras, fechaExpiracion.
- b. **Operaciones:** Crear, Leer, Actualizar.

6. Orden de Compra

- a. **Atributos:** id, fechaCreacion, fechaEntregaEsperada, estado, sucursalId, proveedorId, detalles.
- b. **Operaciones:** Crear, Leer.

7. Recepcion de Productos

- a. **Atributos:** id, ordenCompraId, fechaRecepcion, productos.
- b. **Operaciones:** Crear, Leer.

2. Cuantificación de Entidades

A continuación, se cuantifican las entidades y se estima cuántos registros podría tener la base de datos para cada una.

Entidad	Cantidad de Registros	Descripción
Sucursal	5-10	Una cantidad moderada de sucursales en el sistema.
Bodega	30-50	Cada sucursal puede tener múltiples bodegas. Se estiman varias bodegas por sucursal.
Producto	20-50	Una cantidad de proveedores con los que interactúan las sucursales.
Categoria	5-10	El sistema manejará varias categorías de productos.
Proveedor	500-1000	Se estima un número significativo de productos en la base de datos.
Orden de Compra	100-500	Cada sucursal realiza varias órdenes de compra por año.
Recepción	100-500	Cada orden de compra genera una o más recepciones de productos.

3. Análisis de las Operaciones de Lectura y Escritura

En esta sección, detallamos las operaciones que se realizarán con más frecuencia, y qué información se accederá de manera conjunta.

Sucursal

Operación	Descripción	Frecuencia Estimada	Impacto
Crear	Crear una nueva sucursal.	Bajo	Baja, solo inserción de datos.
Leer	Obtener información sobre todas las sucursales o una sucursal por su ID.	Media	Lectura rápida, escalara con la cantidad de sucursales.
Actualizar	Actualizar la información de una sucursal.	Bajo	Bajo, solo actualización de atributos.

Eliminar	Eliminar una sucursal (por ID).	Bajo	Baja, solo eliminación.

Bodega

Operación	Descripción	Frecuencia Estimada	Impacto
Crear	Crear una nueva bodega asociada a una sucursal.	Baja	Baja, solo inserción.
Leer	Obtener información sobre todas las bodegas de una sucursal.	Media	Media, leer varias bodegas.
Actualizar	Actualizar los detalles de una bodega.	Baja	Baja, solo actualización de atributos.
Eliminar	Eliminar una bodega.	Baja	Baja, eliminación de documentos.

Producto

Operación	Descripción	Frecuencia Estimada	Impacto
Crear	Crear un nuevo producto y asociarlo a una categoría.	Alta	Alta, inserción de productos.
Leer	Obtener información sobre un producto por ID o nombre.	Alta	Alta, accesos frecuentes.
Actualizar	Actualizar la información de un producto.	Media	Media, actualizaciones frecuentes.
Eliminar	Eliminar un producto.	Baja	Baja, eliminación de productos.

Orden de Compra

Operación	Descripción	Frecuencia Estimada	Impacto
Crear	Crear una nueva orden de compra asociada a una sucursal y proveedor.	Alta	Alta, inserción de documentos.
Leer	Obtener información sobre una orden de compra.	Media	Media, consulta por ID de orden.
Actualizar	Actualizar el estado de una orden de compra.	Baja	Baja, actualización de estado.

4. Cuantificación de las Operaciones de Lectura y Escritura

Entidad	Operaciones de lectura (diarias)	Operaciones de escritura (diarias)	Total, de Entidades	Notas
Sucursal	1	0.03	4	Consulta semanal
Bodega	1	0.03	4	Consulta semanal
Producto	2000	10000	24	Modificaciones frecuentes
Categoría	24	0	8	Solo consulta ocasional
Proveedor	200	10	5	Consultas frecuentes
Orden de Compra	1000	200	40	Consultas frecuentes
Recepción	40	40	40	Igual que las ordenes

5. Análisis de Selección de Esquema de Asociación (Referenciado vs. Embebido)

En MongoDB, la decisión de usar documentos embebidos o referenciados depende de las consultas más comunes y la frecuencia con que los datos son accedidos o actualizados.

Relaciones en el modelo:

Sucursal - Bodega (1:N): La relación entre sucursal y bodega se representa como referenciada, ya que cada bodega tiene un campo sucursalId y puede haber múltiples bodegas por sucursal. Esta relación no justifica ser embebida, pues las bodegas son entidades relativamente independientes.

Bodega - ProductoBodega (N:1): Embebida. El detalle del producto en cada bodega (como productoId, cantidad, costoPromedio) se guarda embebido porque es más eficiente consultar estos detalles dentro de la bodega misma.

Producto - Categoria (N:1): Referenciada. Los productos hacen referencia a una categoría, pero la categoría no tiene que estar embebida en cada producto. Solo se guarda el categoriaId.

Descripción de las Colecciones y Relaciones entre Entidades

A continuación, se detalla la descripción de las colecciones de datos, las relaciones entre las entidades y su cardinalidad, así como el análisis de la selección de esquema de asociación (referenciado o embebido) para cada relación, con ejemplos en formato JSON.

a. Lista de Entidades con su Descripción

1. Sucursal

Descripción: Representa una sucursal de la empresa. Cada sucursal tiene una ubicación y pertenece a una ciudad. Además, está asociada a varias bodegas.

Atributos:

- id: Identificador único de la sucursal.
- nombre: Nombre de la sucursal.
- direccion: Dirección de la sucursal.
- ciudad: Ciudad en la que está ubicada la sucursal.
- telefono: Número de teléfono de la sucursal.

2. Bodega

Descripción: Cada bodega está asociada a una sucursal y almacena productos. Las bodegas están relacionadas con los productos almacenados, junto con su cantidad y costo promedio.

Atributos:

- id: Identificador único de la bodega.
- nombre: Nombre de la bodega.
- tamaño: Tamaño de la bodega (en metros cuadrados).
- sucursalId: Relación con la sucursal a la que pertenece.
- productos: Lista de productos almacenados, con detalles como cantidad y costo promedio.

3. Proveedor

Descripción: Representa a los proveedores de productos. Los proveedores tienen un NIT, nombre, dirección y un contacto.

Atributos:

- id: Identificador único del proveedor.
- nit: Número de identificación tributaria del proveedor.
- nombre: Nombre del proveedor.
- direccion: Dirección del proveedor.
- nombreContacto: Nombre de la persona de contacto en el proveedor.
- telefonoContacto: Teléfono del contacto del proveedor.

4. Categoría de Producto

Descripción: Las categorías de productos agrupan productos similares.

Atributos:

- id: Identificador único de la categoría.
- nombre: Nombre de la categoría.
- descripcion: Descripción de la categoría.

5. Producto

Descripción: Representa los productos disponibles en la empresa. Cada producto está asociado a una categoría y tiene un precio, presentación, y fecha de expiración.

Atributos:

- id: Identificador único del producto.
- nombre: Nombre del producto.

- precioUnitario: Precio por unidad.
- presentacion: Presentación del producto (por ejemplo, unidad, paquete, caja).
- categoriaId: Relación con la categoría a la que pertenece.
- codigoBarras: Código de barras del producto.
- fechaExpiracion: Fecha de expiración del producto.

6. Orden de Compra

Descripción: Representa una orden de compra realizada por una sucursal a un proveedor.

Atributos:

- id: Identificador único de la orden de compra.
- fechaCreacion: Fecha de creación de la orden.
- fechaEntregaEsperada: Fecha esperada de entrega.
- sucursalId: Relación con la sucursal que realiza la compra.
- proveedorId: Relación con el proveedor al que se le realiza la compra.
- detalles: Detalles de los productos que se compran (cantidad, precio, etc.).

7. Recepcion de Productos

Descripción: Representa la recepción de productos de una orden de compra.

Atributos:

- id: Identificador único de la recepción.
- ordenCompraId: Relación con la orden de compra asociada.
- fechaRecepcion: Fecha de recepción de los productos.
- productos: Detalles de los productos recibidos (productoId, cantidad, costo promedio).

b. Relaciones entre Entidades y su Cardinalidad

Entidades	Relación	Cardinalidad
Sucursal - Bodega	Una sucursal tiene muchas bodegas.	1: N
Bodega - Producto	Una bodega puede almacenar muchos productos.	N:1
Producto – Categoría	Un producto pertenece a una categoría.	N:1
Orden de Compra - Sucursal	Una orden de compra es realizada por una sucursal.	N:1
Orden de Compra - Proveedor	Una orden de compra es realizada a un proveedor.	N:1
Orden de Compra -Producto	Una orden de compra incluye varios productos.	N:M
Recepcion de Productos – Orden de Compra	Una recepción está asociada a una orden de compra.	N:1

c. Análisis de Selección de Esquema de Asociación

Relación	Tipo de Asociación	Razón
Sucursal - Bodega	Referenciada	Las bodegas son entidades independientes, y cada sucursal puede tener muchas bodegas. Es más eficiente mantenerlas referenciadas.
Bodega - ProductoBodega	Embebida	Los detalles de los productos (como cantidad, costo promedio) son específicos a cada bodega y no necesitan ser consultados por separado.
Producto - Categoria	Referenciada	Los productos necesitan asociarse a una categoría, pero la categoría no necesita estar embebida en cada producto.
Orden de Compra – Producto	Embebida en los detalles	Los detalles de la compra (productos, cantidad, precio) son específicos a cada orden de compra.
Recepcion de Productos – Orden de Compra	Referenciada	Cada recepción se asocia a una orden de compra, pero no es necesario embeber toda la información de la orden de compra en cada recepción.

d. Ejemplo de Relación entre Entidades con JSON

Sucursal con Bodega (Embebido)

```
{
  "_id": "sucursal_001",
  "nombre": "Sucursal Bogotá",
  "direccion": "Calle 123, Bogotá",
  "ciudad": "Bogotá",
  "telefono": "3001234567",
  "bodegas": [
    { "id": "bodega_001", "nombre": "Bodega Central", "tamaño": 500, "sucursalId": "sucursal_001" },
    { "id": "bodega_002", "nombre": "Bodega Norte", "tamaño": 300, "sucursalId": "sucursal_001" }
  ]
}
```

```
}
```

Producto con Categoría (Referencia)

```
{  
  
  "_id": "prod_001",  
  
  "nombre": "Arroz",  
  
  "precioUnitario": 2500,  
  
  "presentacion": "Paquete 1kg",  
  
  "categoriaId": "categoria_001",  
  
  "codigoBarras": "1234567890123",  
  
  "fechaExpiracion": "2024-12-31"  
  
}
```

Bodega con ProductoBodega (Referencia)

```
{  
  
  "_id": "bodega_001",  
  
  "nombre": "Bodega Central",  
  
  "tamaño": 500,  
  
  "sucursalId": "sucursal_001",  
  
  "productos": [  
  
    {  
  
      "productoId": "prod_001",  
  
      "cantidad": 100,  
  
      "costoPromedio": 2500  
  
    },  
  
    {  
  
      "productoId": "prod_002",  
  
      "cantidad": 50,  
  
      "costoPromedio": 3500  
  
    }  
  
  ]  
  
}
```

```
]
}
```

5. Implementación

En esta sección se detallan las decisiones de implementación, el stack tecnológico utilizado, la descripción de los principales componentes del proyecto (como el backend con Spring Boot, la base de datos NoSQL con MongoDB, los endpoints y las pruebas realizadas).

5.1 Stack Tecnológico Utilizado

Backend:

- Spring Boot: Framework para crear aplicaciones Java con enfoque en la facilidad de uso y el desarrollo rápido. Spring Boot fue utilizado para crear el backend de la aplicación, proporcionando endpoints RESTful para gestionar las entidades del sistema (Sucursal, Bodega, Producto, etc.).
- Spring Data MongoDB: Para interactuar con la base de datos NoSQL (MongoDB). Spring Data simplifica el acceso y manipulación de datos en MongoDB.

Base de Datos:

- MongoDB: Base de datos NoSQL utilizada para almacenar las colecciones de datos (Sucursal, Bodega, Producto, etc.). MongoDB fue elegida por su capacidad para manejar datos no estructurados, escalabilidad y flexibilidad en las consultas.

Herramientas:

- Postman: Herramienta utilizada para realizar pruebas y validar las solicitudes HTTP hacia los endpoints del backend. Se utilizaron múltiples colecciones en Postman para validar cada uno de los requerimientos funcionales.

Otras Herramientas:

- MongoDB Compass: Utilizado para visualizar, administrar y consultar los datos almacenados en MongoDB de manera más sencilla.
- Git: Sistema de control de versiones utilizado para gestionar el código fuente y los commits del proyecto.

5.2 Descripción de los Componentes Principales

5.2.1 Estructura de la Base de Datos en MongoDB

El proyecto fue modelado utilizando una base de datos NoSQL (MongoDB), que se ajusta bien al caso de uso debido a la flexibilidad en la representación de los datos. Las colecciones principales y sus relaciones fueron modeladas de la siguiente manera:

Sucursal: Almacena información de las sucursales, como nombre, dirección, ciudad, etc. Cada sucursal puede tener múltiples bodegas asociadas.

Bodega: Representa un almacén dentro de una sucursal. Almacena productos con detalles como cantidad y costo promedio. Las bodegas están asociadas a una sola sucursal, pero pueden almacenar múltiples productos.

Producto: Almacena los productos disponibles, incluyendo detalles como nombre, precio, categoría y fecha de expiración. Los productos están asociados a categorías y pueden pertenecer a una o más bodegas.

Orden de Compra: Representa las órdenes de compra realizadas por las sucursales a los proveedores, con productos, cantidades y precios.

Proveedor: Almacena la información de los proveedores con los que las sucursales realizan compras.

Recepción de Productos: Representa la recepción de los productos que han sido ordenados por una sucursal y recibidos.

5.2.2 Implementación de Endpoints

El backend fue implementado usando Spring Boot y las siguientes funcionalidades fueron cubiertas a través de endpoints RESTful:

Sucursal:

- POST /sucursales/new: Crea una nueva sucursal.
- GET /sucursales: Obtiene todas las sucursales.
- GET /sucursales/{id}: Obtiene una sucursal por su ID.
- PUT /sucursales/{id}: Actualiza una sucursal.
- DELETE /sucursales/{id}: Elimina una sucursal.

Bodega:

- POST /bodegas/new: Crea una nueva bodega asociada a una sucursal.
- DELETE /bodegas/{id}: Elimina una bodega por su ID.
- GET /bodegas/{id}: Obtiene una bodega por su ID.

Producto:

- POST /productos/new: Crea un nuevo producto, asociado a una categoría existente.
- GET /productos/{id}: Obtiene la información de un producto por ID.
- PUT /productos/{id}: Actualiza la información de un producto.
- GET /productos/filtrar: Filtra productos por diferentes características (precio, fecha de vencimiento, categoría, etc.).

Proveedor:

- POST /proveedores/new: Crea un nuevo proveedor.
- GET /proveedores/{id}: Obtiene un proveedor por su ID.
- PUT /proveedores/{id}: Actualiza la información de un proveedor.

Orden de Compra:

- POST /ordenes/new: Crea una nueva orden de compra para una sucursal.
- GET /ordenes/{id}: Obtiene una orden de compra por ID.

Recepción de Productos:

- POST /recepciones/new: Crea una nueva recepción de productos para una orden de compra.
- GET /recepciones/{id}: Obtiene una recepción de productos por ID.

5.2.3 Esquema de Validación de Datos en MongoDB

En MongoDB, utilizamos esquemas de validación para asegurar que los datos cumplan con las restricciones necesarias antes de ser insertados. A continuación, se describe el esquema de validación para algunas colecciones clave:

Sucursal:

Se asegura que los campos id, nombre, direccion, ciudad y telefono sean obligatorios y que el id sea único.

Producto:

Se valida que el campo nombre, precioUnitario, presentacion, categoriaId, y fechaExpiracion estén presentes y cumplan con el formato esperado (por ejemplo, precioUnitario debe ser un número).

Bodega:

Se valida que el campo nombre, sucursalId, y productos (lista de productos) estén presentes.

Los esquemas de verificación se encuentran en el repositorio de GitHub.

5.3 Funcionalidades Implementadas

Las siguientes funcionalidades fueron implementadas y validadas utilizando Postman:

Creación y Gestión de Sucursales:

Los endpoints de Sucursales permiten agregar, actualizar y eliminar sucursales en el sistema. Estas operaciones fueron probadas mediante solicitudes POST, PUT y DELETE en Postman.

Gestión de Bodegas:

La creación de bodegas asociadas a sucursales y su eliminación fueron cubiertas mediante solicitudes POST y DELETE.

Gestión de Productos:

Se permite la creación de productos, así como la actualización de su información y la consulta mediante filtros (por precio, fecha de vencimiento, etc.).

Gestión de Proveedores:

Los proveedores pueden ser añadidos y actualizados en el sistema, con las relaciones adecuadas hacia las órdenes de compra.

Gestión de Órdenes de Compra:

Las órdenes de compra incluyen productos, cantidades y precios, y pueden ser consultadas por su ID.

Recepción de Productos:

Los productos recibidos de las órdenes de compra son registrados junto con su cantidad y costo.

5.4 Conclusión de la Implementación

La implementación de este proyecto ha sido exitosa, ya que hemos cumplido con los requerimientos funcionales y de consulta especificados en el caso de estudio. Los endpoints fueron correctamente implementados, las validaciones de datos en MongoDB se aplicaron con éxito.

6. Pruebas

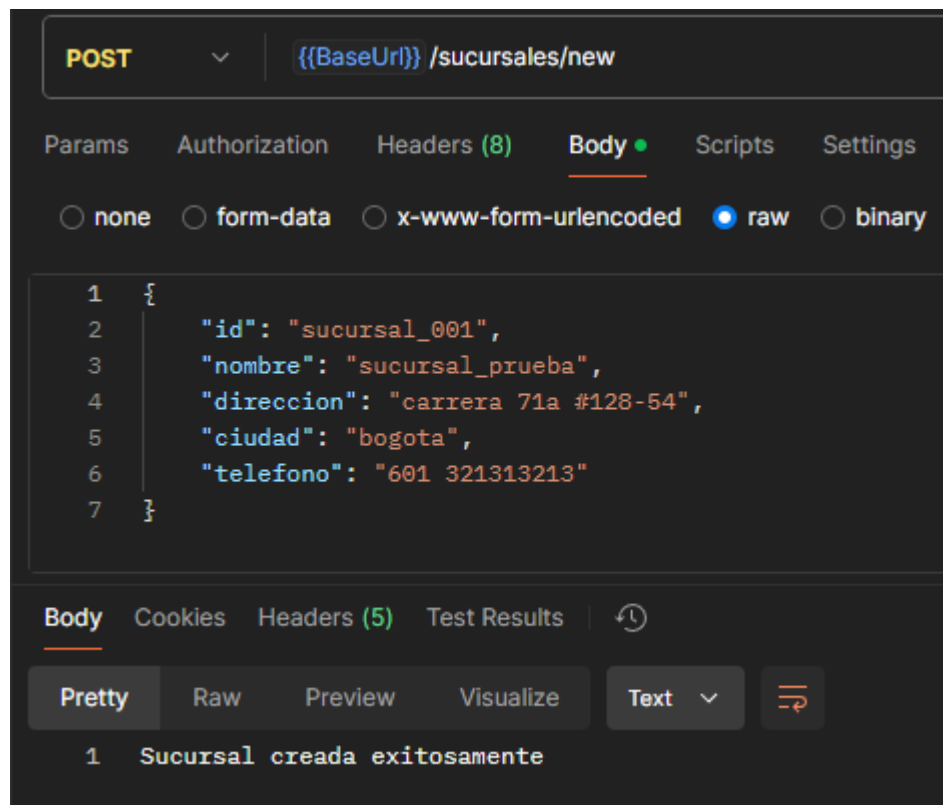
Las pruebas de la implementación fueron realizadas utilizando Postman, una herramienta de desarrollo para pruebas de API que permite enviar solicitudes HTTP a los endpoints del servidor y verificar las respuestas. Cada requerimiento funcional (RF) y de consulta (RFC) fue probado individualmente para asegurarnos de que todas las funcionalidades de la aplicación estén completas y operativas.

6.1 Pruebas de Funcionalidades (RF)

A continuación, se describen las pruebas realizadas para cada uno de los requerimientos funcionales del proyecto (RF1 - RF7):

RF1 - Crear una Sucursal

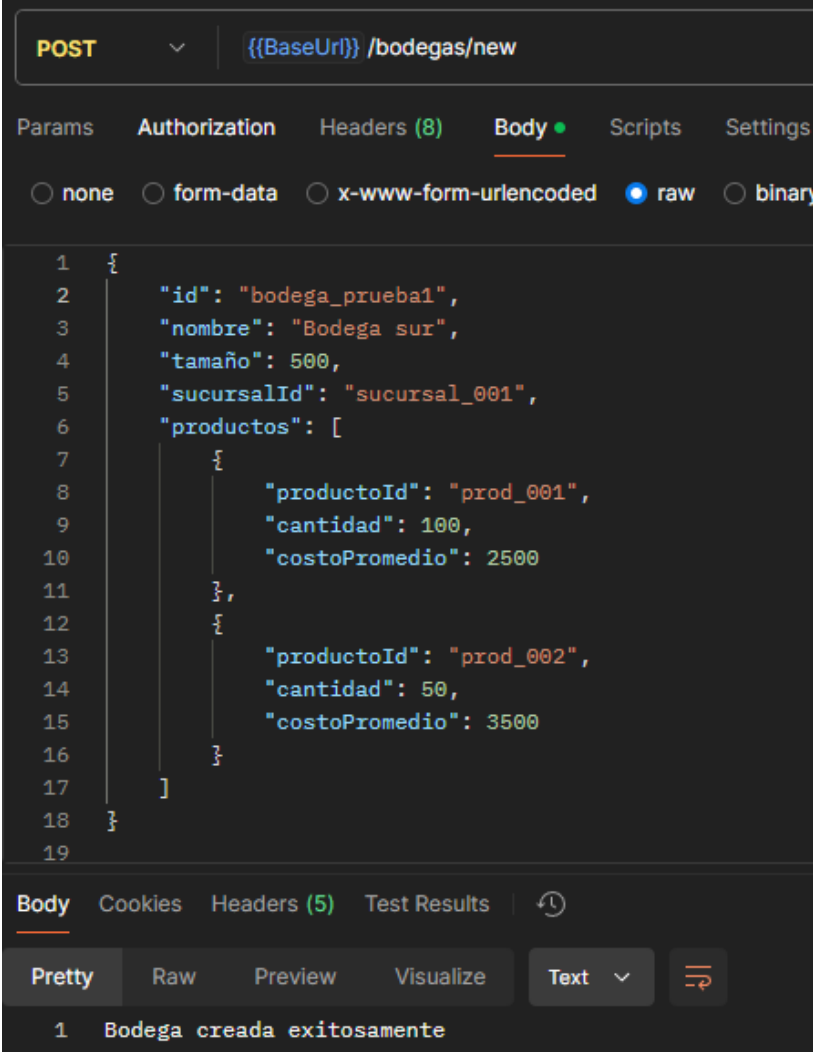
- **Endpoint:** POST /sucursales/new
- **Descripción:** Crear una nueva sucursal.
- **Prueba:**



- **Resultado Esperado:** La sucursal fue creada exitosamente y la respuesta fue HTTP 201 Created.
- **Resultado Obtenido:** HTTP 201 Created con el mensaje "Sucursal creada exitosamente".

RF2 - Crear y Borrar una Bodega

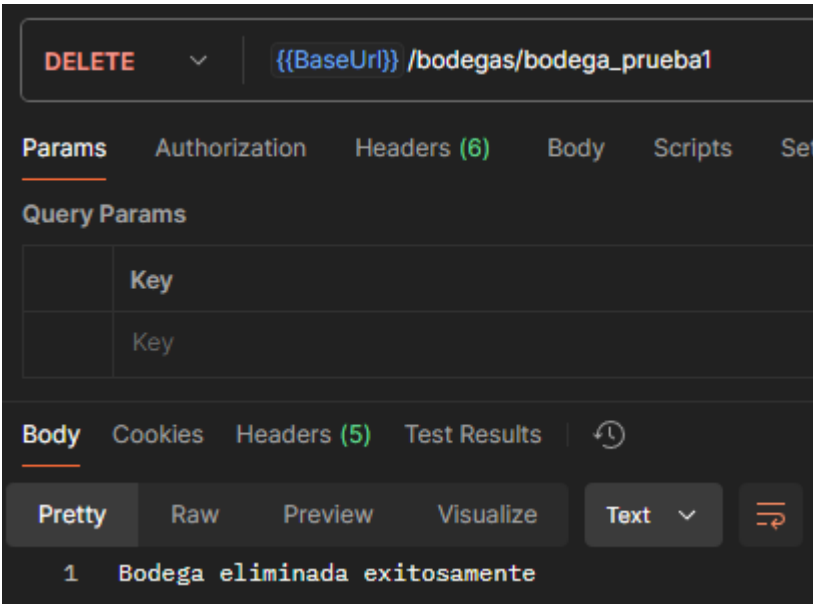
- **Endpoint:**
 - POST /bodegas/new para crear una bodega.
 - DELETE /bodegas/{id} para eliminar una bodega.
- **Prueba:**



The screenshot shows a REST client interface with a POST request to `{{BaseUrl}} /bodegas/new`. The request body is a JSON object with the following structure:

```
1 {
2   "id": "bodega_prueba1",
3   "nombre": "Bodega sur",
4   "tamaño": 500,
5   "sucursalId": "sucursal_001",
6   "productos": [
7     {
8       "productoId": "prod_001",
9       "cantidad": 100,
10      "costoPromedio": 2500
11    },
12    {
13      "productoId": "prod_002",
14      "cantidad": 50,
15      "costoPromedio": 3500
16    }
17  ]
18 }
19
```

The response is displayed in the 'Body' tab, showing the message: `1 Bodega creada exitosamente`.



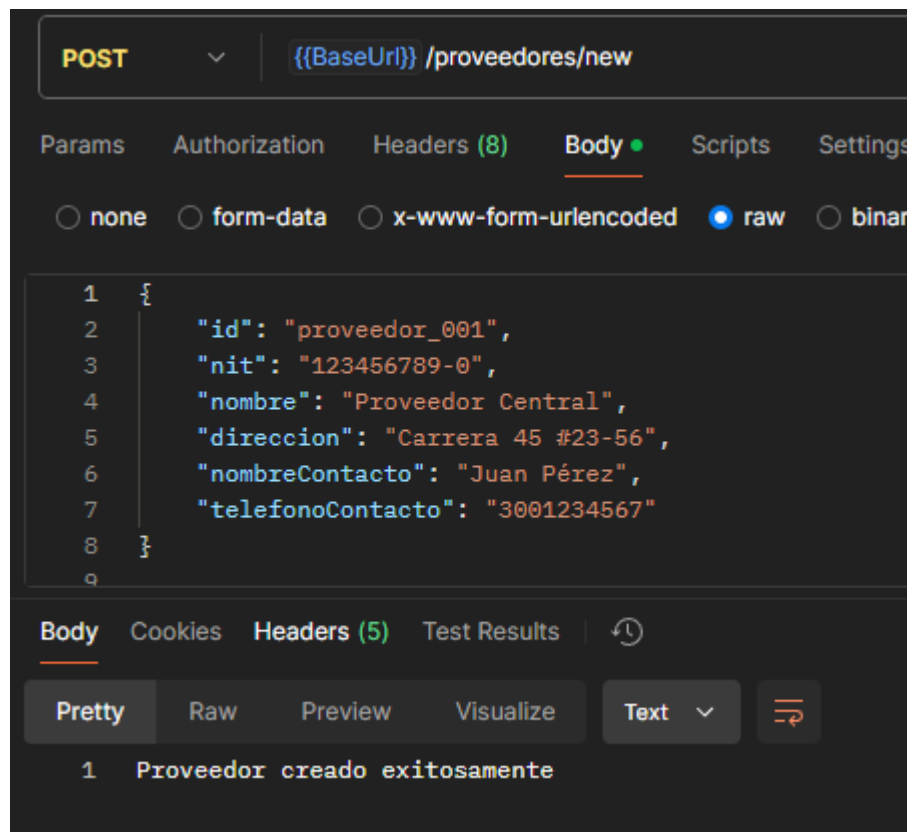
The screenshot shows a REST client interface with a DELETE request to `{{BaseUrl}} /bodegas/bodega_prueba1`. The response is displayed in the 'Body' tab, showing the message: `1 Bodega eliminada exitosamente`.

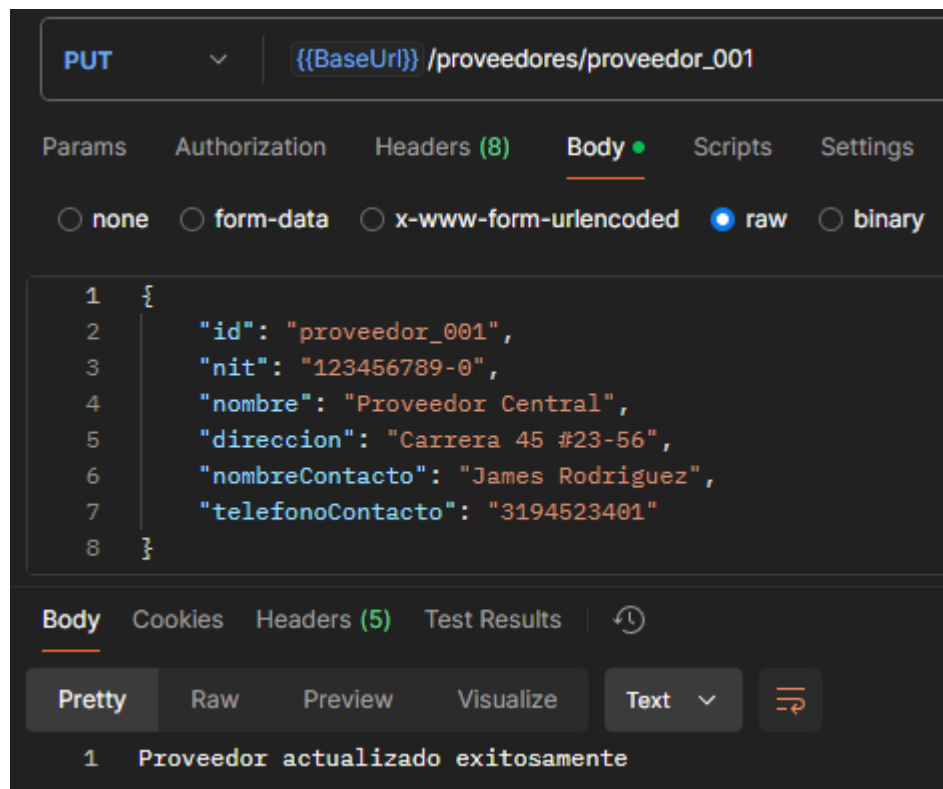
- **Resultado Esperado:** La bodega debe ser creada y luego eliminada correctamente.

- **Resultado Obtenido:** Ambas acciones fueron realizadas exitosamente con respuestas HTTP 200 OK y HTTP 201 Created al crear la bodega.

RF3 - Crear y Actualizar Proveedores

- **Endpoint:**
 - POST /proveedores/new para crear un proveedor.
 - PUT /proveedores/{id} para actualizar un proveedor.
- **Prueba:**

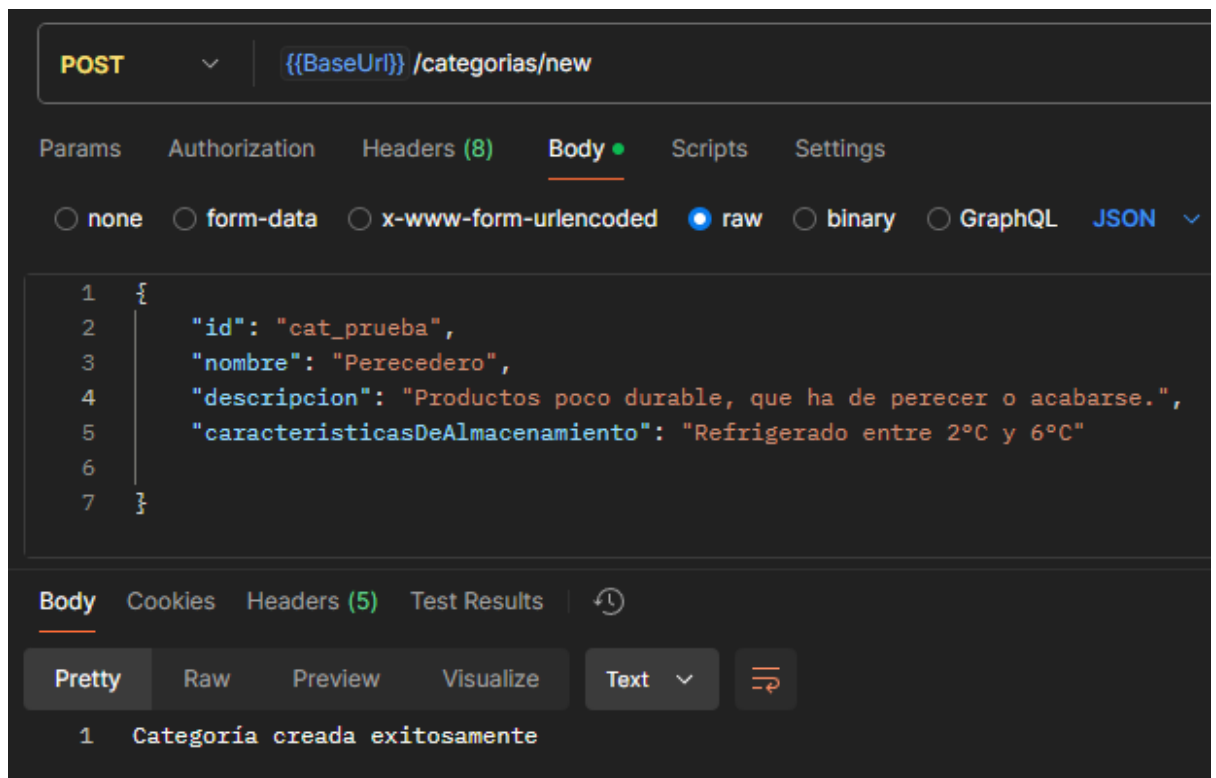




- **Resultado Esperado:** El proveedor fue creado y actualizado correctamente.
- **Resultado Obtenido:** Respuesta HTTP 201 Created y HTTP 200 OK para las operaciones de creación y actualización.

RF4 - Crear una Categoría de Producto

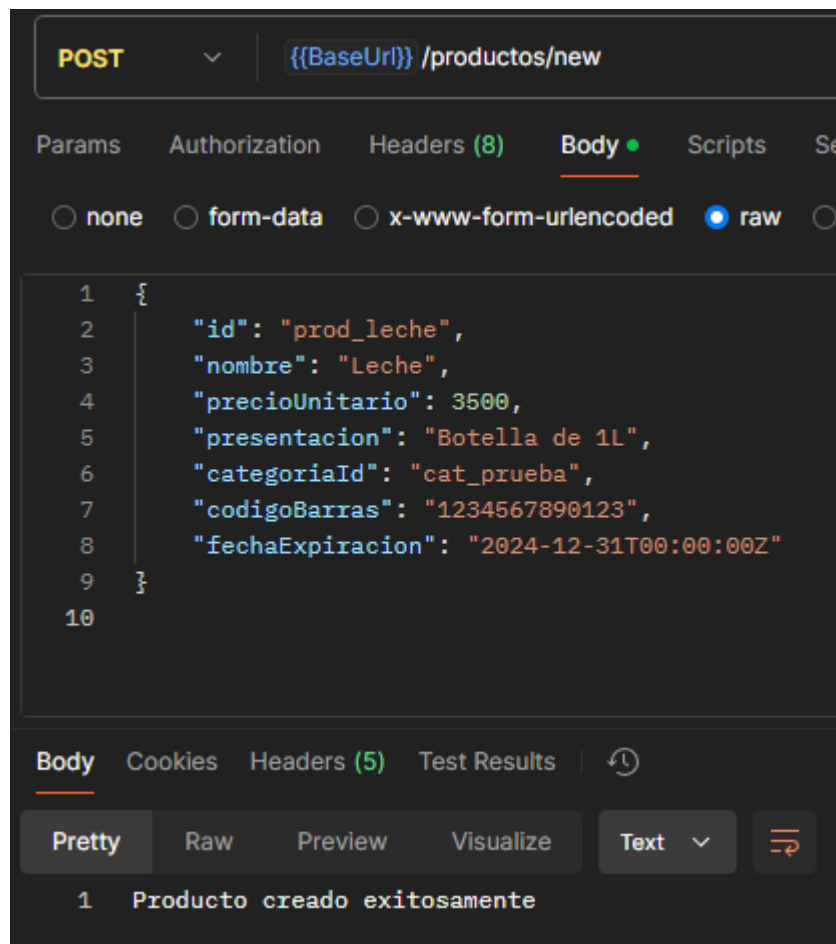
- **Endpoint:** POST /categorias/new
- **Descripción:** Crear una nueva categoría de producto.
- **Prueba:**



- **Resultado Esperado:** La categoría fue creada correctamente.
- **Resultado Obtenido:** Respuesta HTTP 201 Created.

RF5 - Crear un Producto

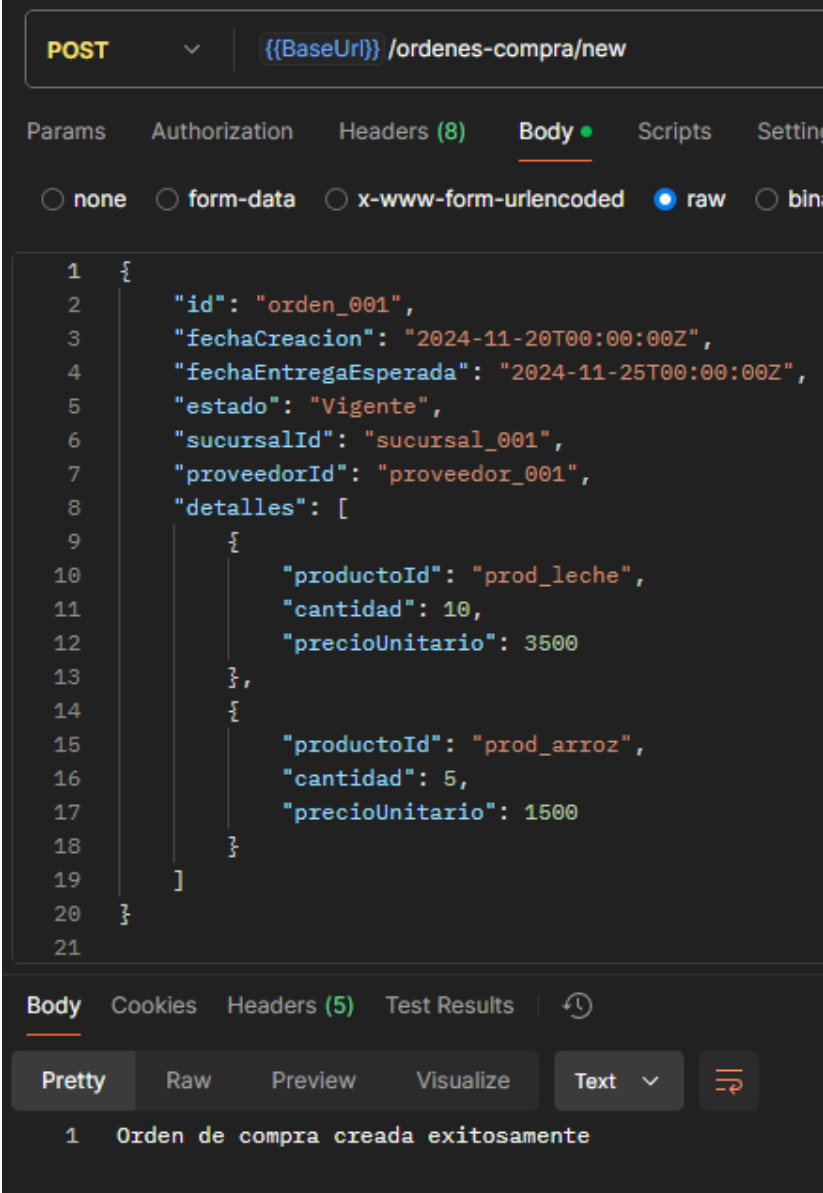
- **Endpoint:** POST /productos/new
- **Descripción:** Crear un producto y asociarlo a una categoría existente.
- **Prueba:**



- **Resultado Esperado:** El producto fue creado y asociado correctamente a la categoría.
- **Resultado Obtenido:** Respuesta HTTP 201 Created.

RF6 - Crear una Orden de Compra

- **Endpoint:** POST /ordenes/new
- **Descripción:** Crear una nueva orden de compra.
- **Prueba:**



The screenshot shows a REST client interface with a POST request to the endpoint `{{BaseUrl}} /ordenes-compra/new`. The request body is a JSON object with the following structure:

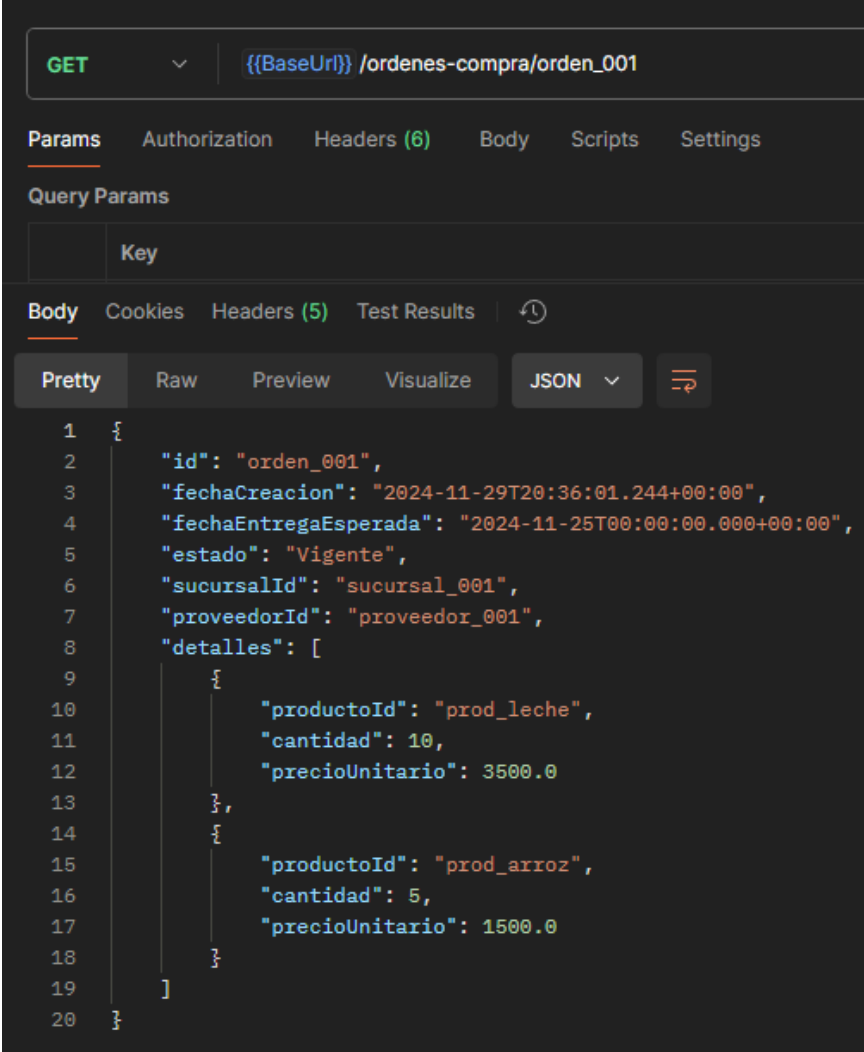
```
1 {
2   "id": "orden_001",
3   "fechaCreacion": "2024-11-20T00:00:00Z",
4   "fechaEntregaEsperada": "2024-11-25T00:00:00Z",
5   "estado": "Vigente",
6   "sucursalId": "sucursal_001",
7   "proveedorId": "proveedor_001",
8   "detalles": [
9     {
10      "productoId": "prod_leche",
11      "cantidad": 10,
12      "precioUnitario": 3500
13     },
14     {
15      "productoId": "prod_arroz",
16      "cantidad": 5,
17      "precioUnitario": 1500
18     }
19   ]
20 }
21
```

The response is displayed in the bottom panel, showing a status of 201 and the message "Orden de compra creada exitosamente".

- **Resultado Esperado:** La orden de compra fue creada exitosamente.
- **Resultado Obtenido:** Respuesta HTTP 201 Created.

RF7 - Leer una Orden de Compra

- **Endpoint:** GET /ordenes/{id}
- **Descripción:** Consultar una orden de compra por su ID.
- **Prueba:**



```
1  {
2    "id": "orden_001",
3    "fechaCreacion": "2024-11-29T20:36:01.244+00:00",
4    "fechaEntregaEsperada": "2024-11-25T00:00:00.000+00:00",
5    "estado": "Vigente",
6    "sucursalId": "sucursal_001",
7    "proveedorId": "proveedor_001",
8    "detalles": [
9      {
10         "productoId": "prod_leche",
11         "cantidad": 10,
12         "precioUnitario": 3500.0
13       },
14       {
15         "productoId": "prod_arroz",
16         "cantidad": 5,
17         "precioUnitario": 1500.0
18       }
19     ]
20  }
```

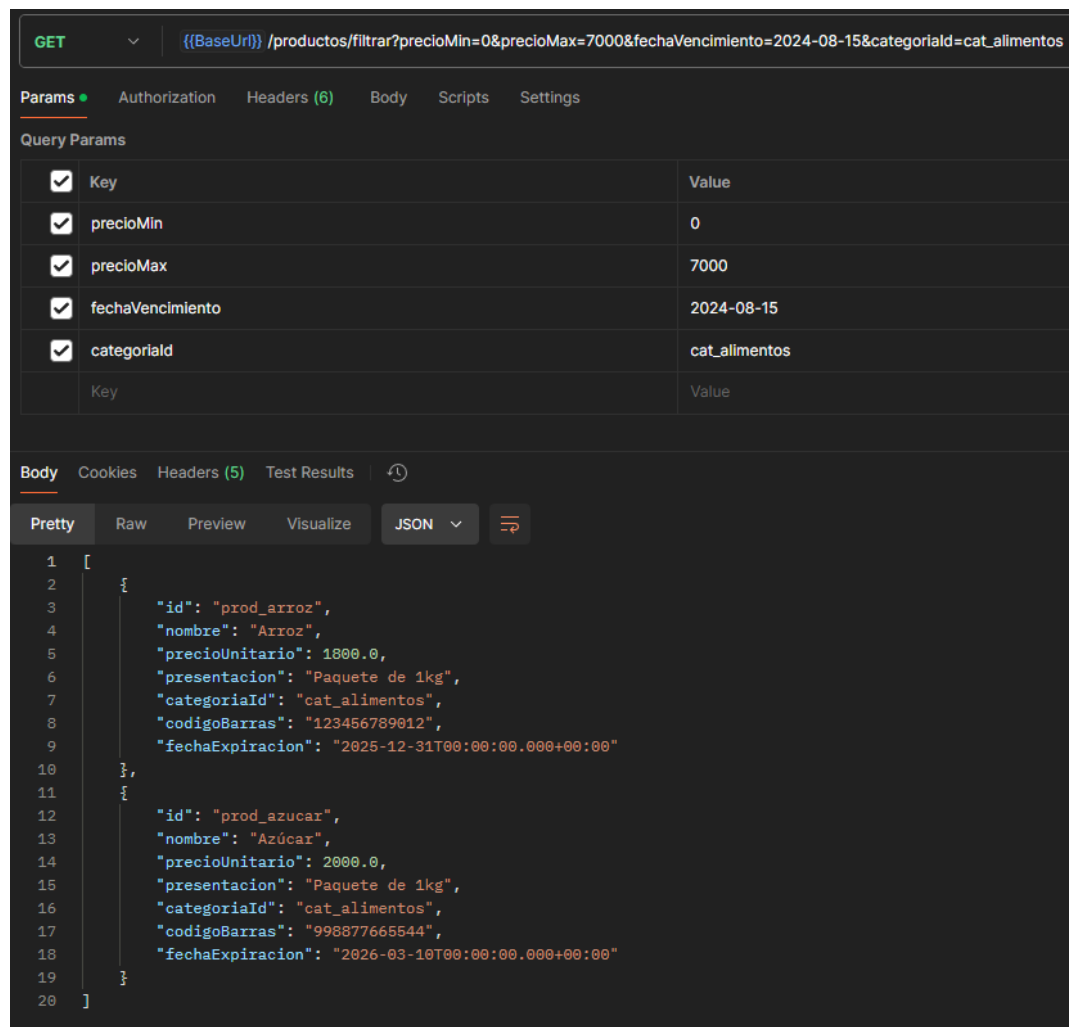
- **Resultado Esperado:** La orden de compra fue obtenida correctamente con todos los detalles.
- **Resultado Obtenido:** Respuesta HTTP 200 OK.

6.2 Pruebas de Consultas (RFC)

Las pruebas de las consultas RFC1 y RFC2 se realizaron de la siguiente manera:

RFC1 - Mostrar los productos que cumplen con cierta característica

- **Endpoint:** GET /productos/filtrar
- **Descripción:** Filtrar productos según precio, fecha de vencimiento, categoría, etc.
- **Prueba:**

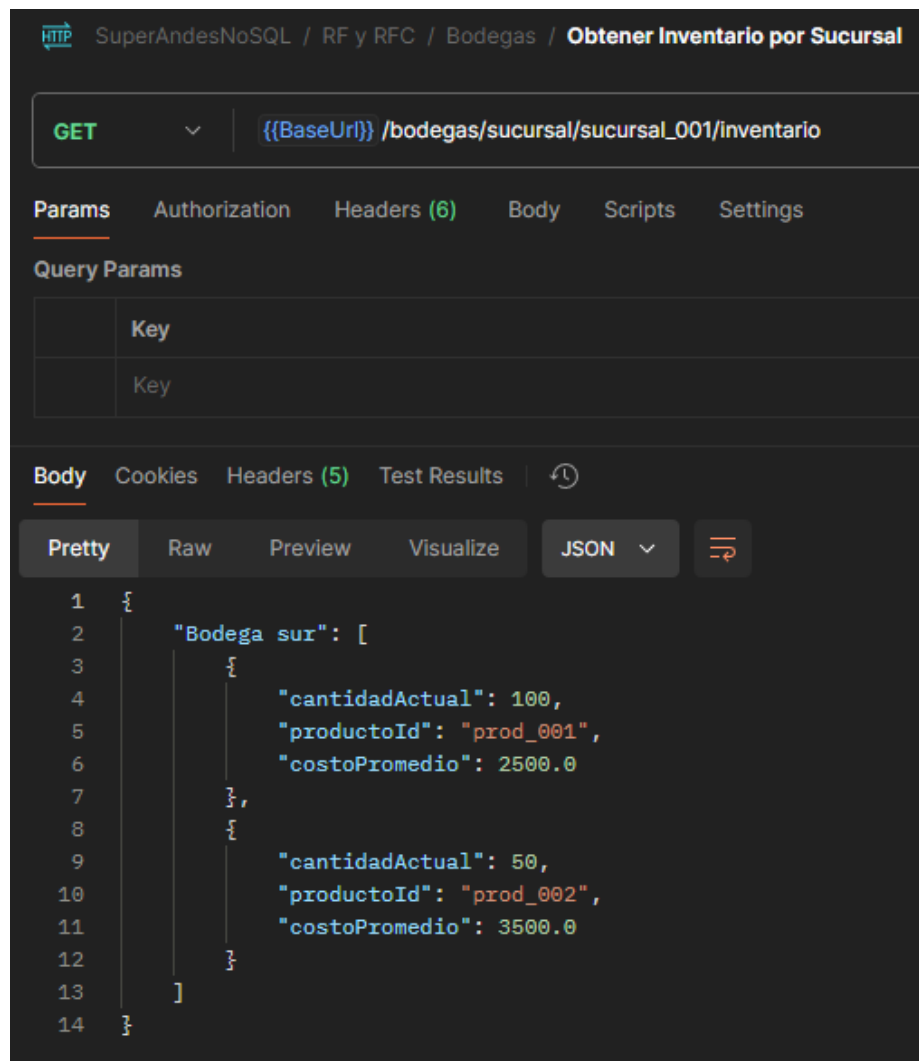


Resultado Esperado: Los productos que cumplen con los filtros especificados fueron devueltos correctamente.

- **Resultado Obtenido:** Respuesta HTTP 200 OK con una lista de productos que cumplieran con los filtros.

RFC2 - Inventario de Productos en una Sucursal

- **Endpoint:** GET /inventarios/sucursal/{sucursalId}
- **Descripción:** Obtener el inventario de productos de una sucursal.
- **Prueba:**



- **Resultado Esperado:** Se obtuvo un reporte con los productos de cada bodega de la sucursal, mostrando la cantidad actual, el costo promedio, etc.
- **Resultado Obtenido:** Respuesta HTTP 200 OK con el inventario de productos detallado.

7. Conclusiones

El proyecto fue implementado exitosamente, cumpliendo con los requerimientos funcionales y de consulta especificados en el enunciado. A través de pruebas exhaustivas con Postman, se validaron todas las operaciones de CRUD para las entidades principales (Sucursal, Bodega, Producto, etc.) y las consultas avanzadas (como los filtros de productos y los inventarios por sucursal).

Aspectos Clave del Proyecto:

- **Escalabilidad y Flexibilidad:** Utilizar MongoDB como base de datos NoSQL permitió una gran flexibilidad en el modelado de datos, especialmente con las relaciones **referenciadas** y **embebidas**.

- **Optimización de Consultas:** La implementación de filtros dinámicos para la consulta de productos (RFC1) y el reporte de inventarios (RFC2) muestra la capacidad del sistema para manejar cargas de trabajo más complejas.
- **Pruebas Exhaustivas:** El uso de Postman para probar cada endpoint aseguró que todas las funcionalidades estuvieran cubiertas y funcionando correctamente. Se validaron tanto las operaciones CRUD básicas como las consultas más complejas.
- **Validación de Datos:** MongoDB fue configurado con esquemas de validación para asegurar la integridad y consistencia de los datos almacenados.