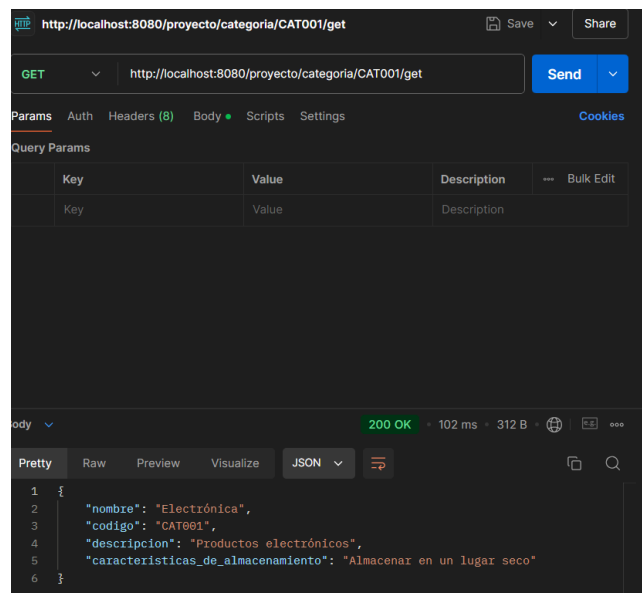


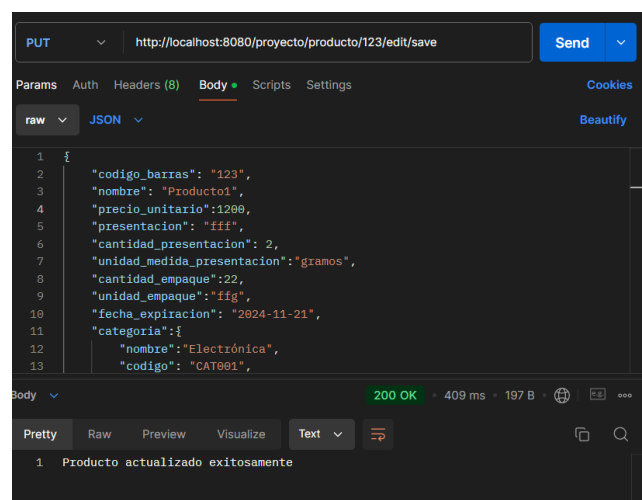
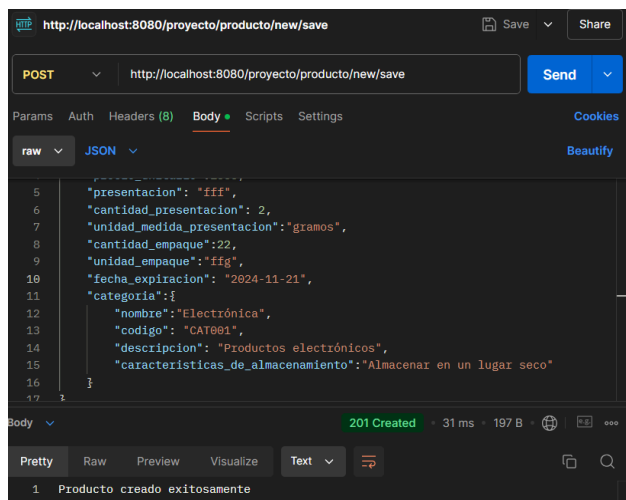
Entrega 2 - Proyecto

PUNTO UNO - REQUERIMIENTOS ANTERIORES

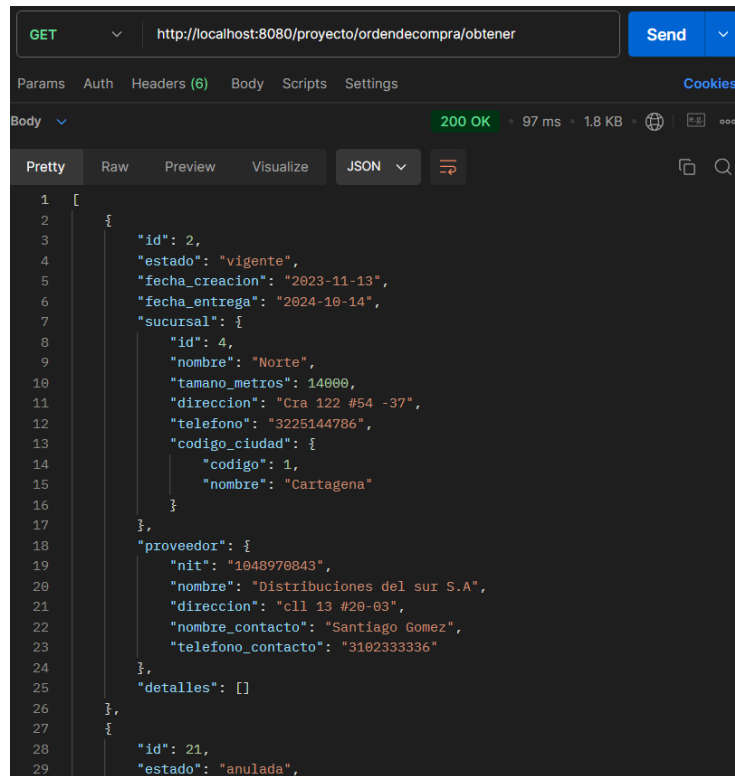
RF5: Faltaba leer una categoría dado un código



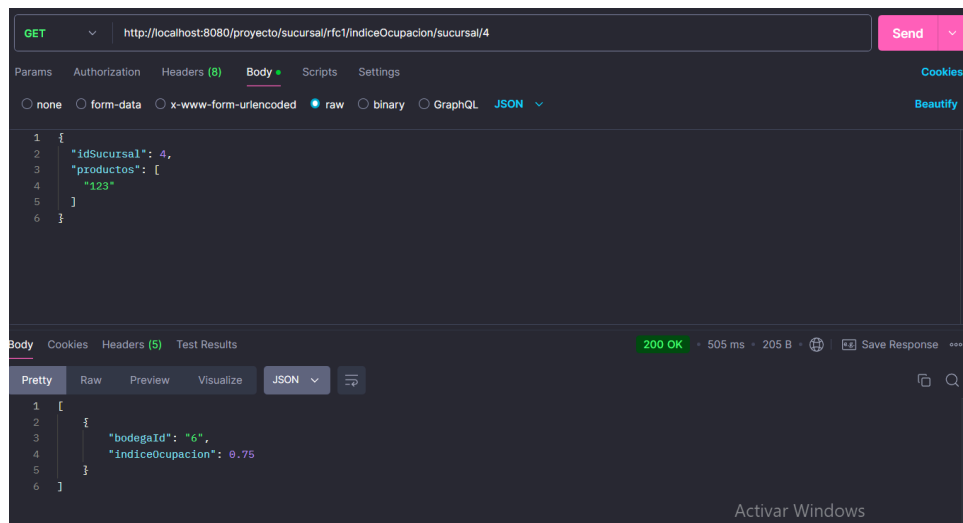
RF6: Faltaba crear y actualizar un producto



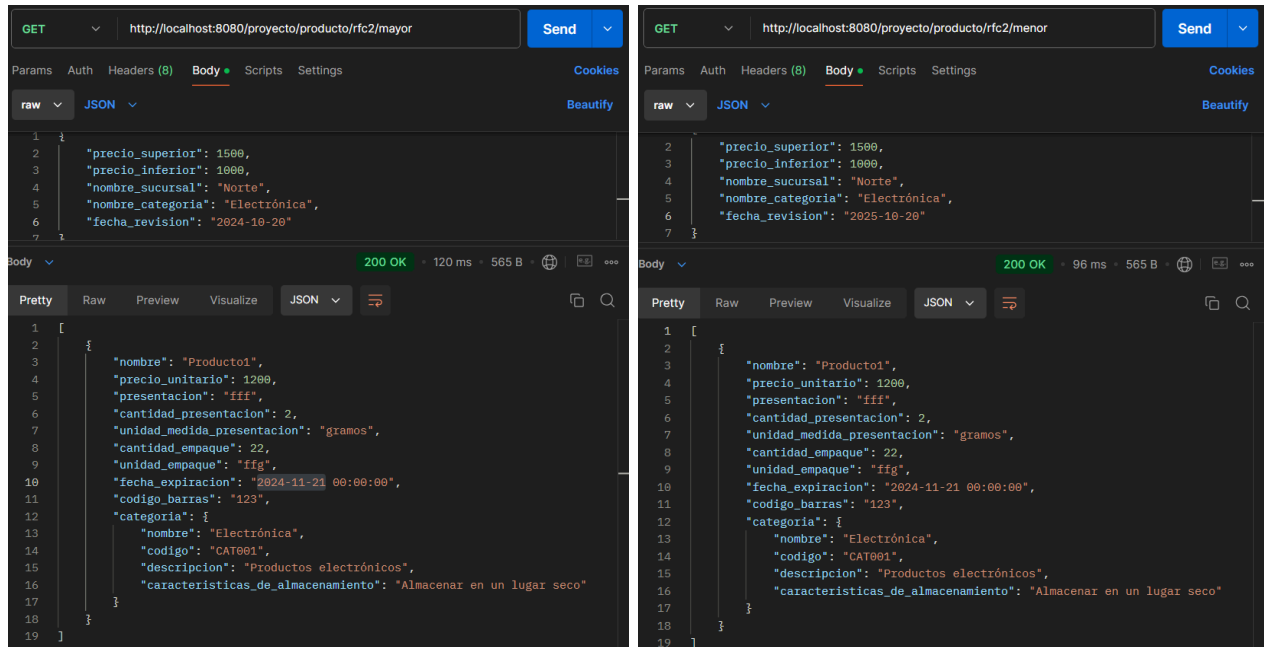
RF9: Previamente salía un error 500 en Postman



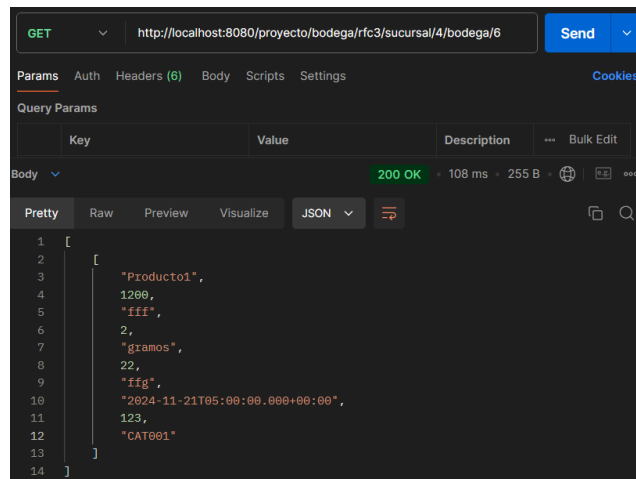
RFC1: Previamente no se presentó



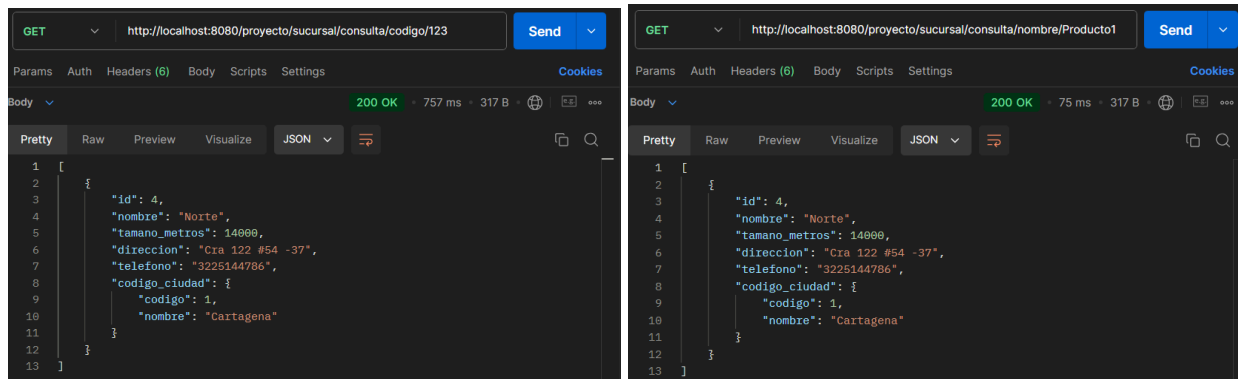
RFC2: Previamente no se presentó



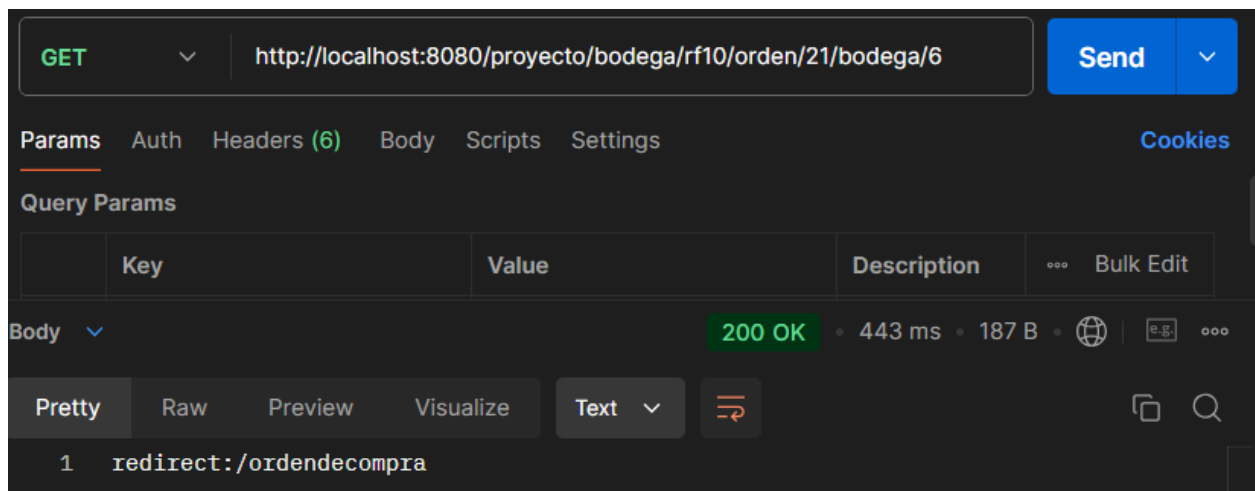
RFC3: Previamente error 400 en Postman



RFC4: Previamente presentaba el mismo error que productos



PUNTO DOS — RF10



Recepción de Producto:

	ID	FECHA_RECEPCION	ID_ORDEN_COMPRA	ID_BODEGA
1	43	29-OCT-24	21	6

Orden de Compra:

	ID	ESTADO	FECHA_CREACION	FECHA_ENTREGA	SUCURSAL	PROVEEDOR
1	2	vigente	13-NOV-23	14-OCT-24	4	1048970843
2	21	entregada	13-SEP-25	13-SEP-25	21	104893
3	22	anulada	10-OCT-27	20-OCT-28	21	104893
4	3	vigente	13-NOV-23	14-OCT-24	4	1048970843

InfoExtraBodega:

Antes de ejecutar:

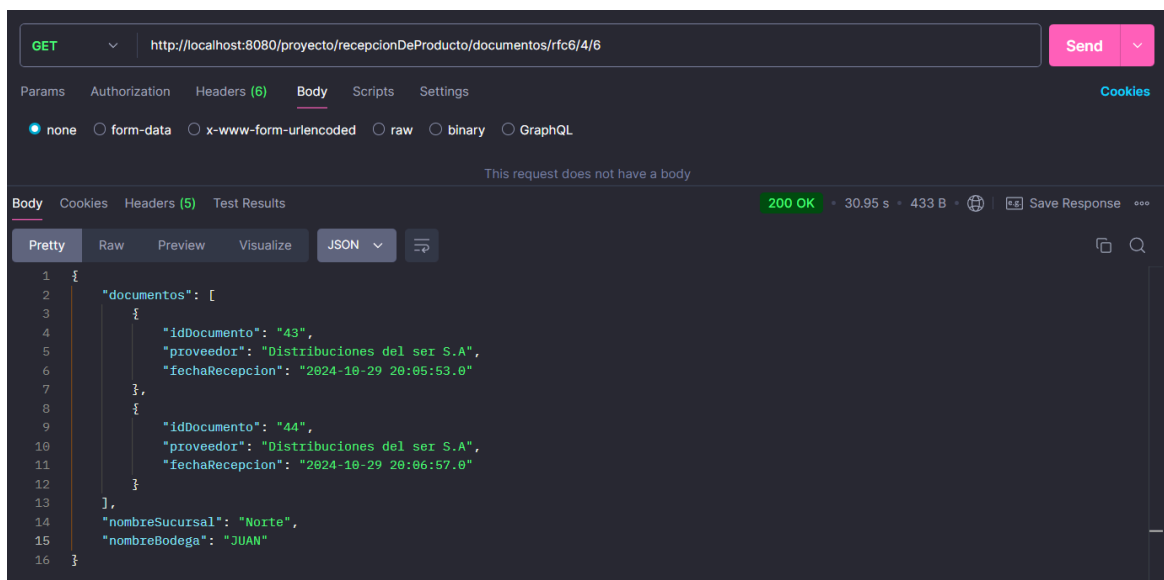
ID_BODEGA	CODIGO_PRODUCTO	NIVEL_MINIMO_REORDEN	CAPACIDAD_ALMACENAMIENTO	COSTO_BODEGA	TOTAL_EXISTENCIAS	COSTO_PROMEDIO
1	6	123	50	1000	50000	900
						137.92

Después de ejecutar:

ID_BODEGA	CODIGO_PRODUCTO	NIVEL_MINIMO_REORDEN	CAPACIDAD_ALMACENAMIENTO	COSTO_BODEGA	TOTAL_EXISTENCIAS	COSTO_PROMEDIO
1	6	123	50	1000	50000	950
						141.19

Para el requerimiento se creó el servicio `RecepcionDeProductoService.java`, en ella se definen las funciones `crearDocumento(ordén, bodega)`, `actualizarAvgCant(ordén, bodega)`, y `cambiarAEntregada(ordén)`. Esto realiza las 3 operaciones solicitadas en el documento de requerimientos. Finalmente, desde el servicio se decide sobre si hacer rollback o commit.

PUNTO TRES -RFC6



Las clases utilizadas para el correcto desarrollo del requerimiento de consulta 6 fueron:

- **Modelo (RecepcionDeProducto):** Representa la entidad de negocio en la base de datos. En este caso, la clase `RecepcionDeProducto` mapea una tabla en la base de datos y

contiene los atributos de un documento de ingreso de productos a una bodega, como la fecha de recepción, el id del documento, el id de la orden de compra a la que está asociada y el id de la bodega en la cuál se reciben los productos.

- **Servicio (RecepcionDeProductoService):** Contiene la lógica de negocio. Actúa como intermediario entre el controlador y el repositorio, garantizando el manejo de las transacciones con aislamiento “serializable”, la espera de 30 segundos, y el rollback en caso de error.

Al momento de definir el método en el service para el aislamiento se utiliza (isolation = Isolation.SERIALIZABLE), para la espera se utiliza (Thread.sleep(30000)) y para el rollback se utiliza (rollbackFor = Exception.class).

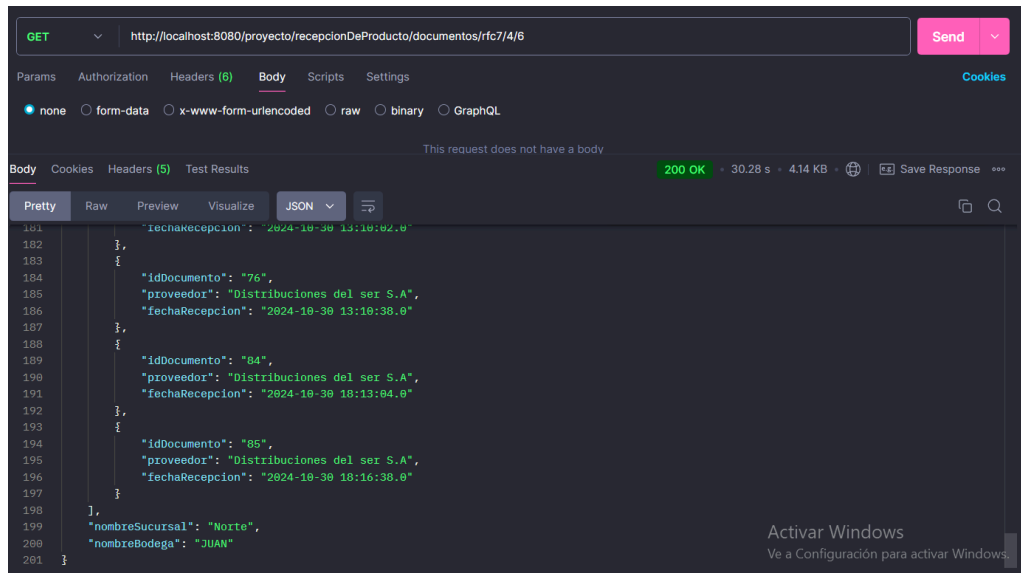
- **Controlador (RecepcionDeProductoController):** Es la capa que maneja las solicitudes HTTP. En este caso, recibe la solicitud del cliente, llama a los métodos del servicio correspondiente (RecepcionDeProductoService) y devuelve la respuesta adecuada al cliente, ya sea el resultado de la consulta o un mensaje de error en caso de problemas.
- **Repositorio (RecepcionDeProductoRepository):** Es la interfaz que interactúa directamente con la base de datos para ejecutar la consulta. En este caso, define la consulta que permite obtener los documentos de ingreso de los últimos 30 días para una sucursal y bodega específicas.

La sentencia SQL utilizada es la siguiente:

```
"SELECT P.nombre AS proveedor, B.nombre AS nombre_bodega, S.nombre AS nombre_sucursal, " +  
"R.id AS id_Documento, R.fecha_recepcion AS fecha " +  
"FROM RecepcionDeProducto R " +  
"INNER JOIN Bodega B ON B.id = R.id_bodega " +  
"INNER JOIN Sucursal S ON S.id = B.sucursal " +  
"INNER JOIN OrdenDeCompra O ON O.id = R.id_orden_compra " +  
"INNER JOIN Proveedor P ON P.nit = O.proveedor " +  
"WHERE S.id = :idSucursal AND B.id = :idBodega AND R.fecha_recepcion >= CURRENT_DATE - 30",
```

Para llevar a cabo la consulta, la sentencia SQL une todas las tablas requeridas, bajo este contexto, Bodega, Sucursal, OrdenDeCompra, RecepcionDeProducto y Proveedor. Además, a través del WHERE asegura que se cumplan las condiciones presentadas: Los documentos retornados deben ser de una bodega y sucursal específica y deben tener una fecha que se encuentre en el rango de los últimos 30 días. Finalmente, en el select se extraen los elementos que se le desea mostrar al cliente.

PUNTO CUATRO - RFC7



Las clases utilizadas para el correcto desarrollo del requerimiento de consulta 7 fueron las mismas utilizada para el rfc6, la única diferencia es que para este requerimiento el nivel de aislamiento en el service es Read Committed y es representado así: (isolation = Isolation.READ_COMMITTED).

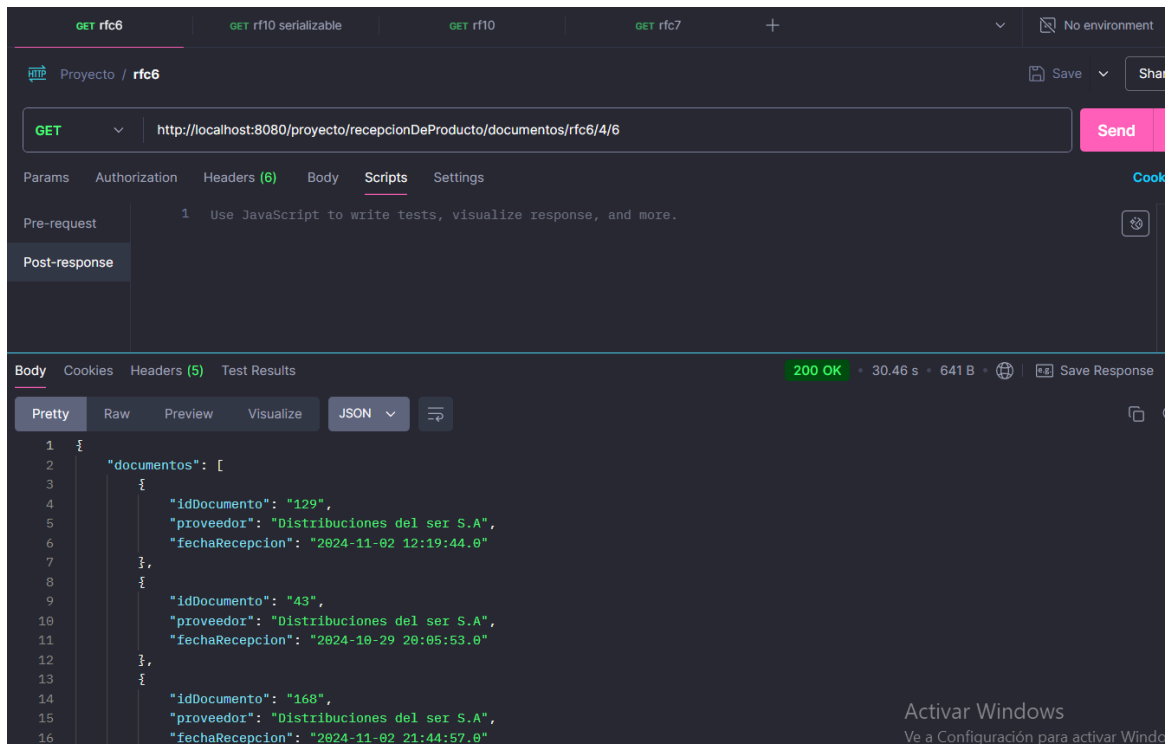
La sentencia SQL utilizada para este requerimiento también es la misma.

PUNTO CINCO - ESCENARIO DE PRUEBA DE CONCURRENCIA 1

- **Pasos de ejecución:** Se inicia la ejecución de la consulta RFC6 con nivel de aislamiento SERIALIZABLE, que busca documentos de ingreso de productos. Antes de que finalice el temporizador de 30 segundos en RFC6, se inicia RF10, la transacción de registro de ingreso de productos, la cual intenta:
 - Insertar el nuevo documento de ingreso en la base de datos.
 - Actualizar el inventario y el costo promedio de los productos en la bodega.
 - Cambiar el estado de la Orden de Compra a “entregada”.
- **Descripción de lo sucedido:** Inicialmente al ejecutar la consulta RFC6, esta se queda en espera, aguardando a que el temporizador termine. Al ejecutar el RF10 en medio de este temporizador se queda cargando de igual manera, debido a que el RFC6 tiene un nivel de aislamiento SERIALIZABLE. Lo que impide la interferencia en la consulta y las lecturas fantasma. Debido al nivel de aislamiento de RFC6, RF10 se ve obligado a esperar hasta

que RFC6 libere los recursos bloqueados. Cuando estos recursos son desbloqueados RF10 puede ejecutarse.

- **Resultado presentado por RFC6:** El requerimiento termina su consulta después del tiempo de espera y obtiene los datos correspondientes a los documentos de ingreso de productos en los últimos 30 días. Como el RF10 debe esperar la finalización de RFC6, el documento de ingreso de productos creado por RF10 no se muestra en el resultado de RFC6, ya que al momento de ejecutar la consulta la transacción de registro aún no había sido completada.

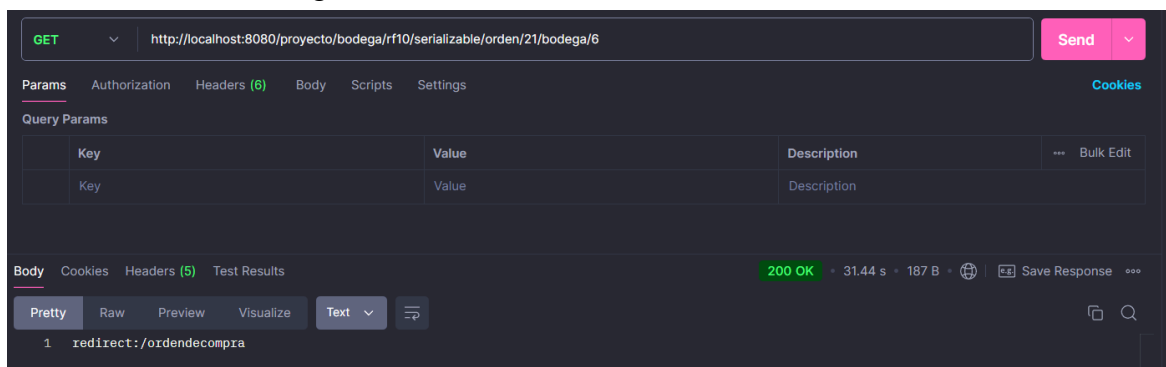


The screenshot shows a REST client interface with the following details:

- Method:** GET
- URL:** `http://localhost:8080/proyecto/recepcionDeProducto/documentos/rfc6/4/6`
- Status:** 200 OK
- Body (JSON):**

```
{  "documentos": [    {      "idDocumento": "129",      "proveedor": "Distribuciones del ser S.A",      "fechaRecepcion": "2024-11-02 12:19:44.0"    },    {      "idDocumento": "43",      "proveedor": "Distribuciones del ser S.A",      "fechaRecepcion": "2024-10-29 20:05:53.0"    },    {      "idDocumento": "168",      "proveedor": "Distribuciones del ser S.A",      "fechaRecepcion": "2024-11-02 21:44:57.0"    }  ]}
```

- **Resultado presentado por RF10:** Una vez que RFC6 terminó y liberó los recursos, RF10 continuó con el registro exitosamente.



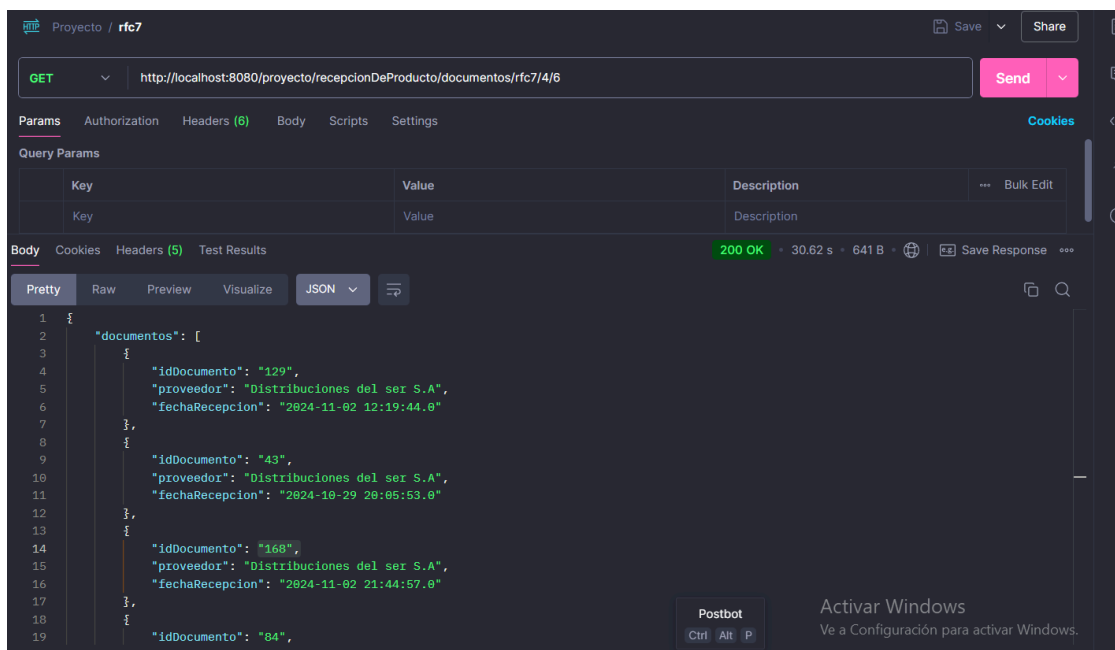
The screenshot shows a REST client interface with the following details:

- Method:** GET
- URL:** `http://localhost:8080/proyecto/bodega/rf10/serializable/orden/21/bodega/6`
- Status:** 200 OK
- Body (Text):**

```
1  redirect:/ordendecompra
```


PUNTO SEIS - ESCENARIO DE PRUEBA DE CONCURRENCIA 2

- **Pasos de ejecución:** Se inicia la ejecución de la consulta RFC7 con nivel de aislamiento *READ-COMMITTED*, que busca documentos de ingreso de productos. Antes de que finalice el temporizador de 30 segundos en RFC7, se inicia RF10, la transacción de registro de ingreso de productos, la cual intenta:
 - Insertar el nuevo documento de ingreso en la base de datos.
 - Actualizar el inventario y el costo promedio de los productos en la bodega.
 - Cambiar el estado de la Orden de Compra a “entregada”.
- **Descripción de lo sucedido:** Al ejecutar RFC7, esta consulta queda en espera, esperando a que el temporizador de 30 segundos termine. Al iniciarse RF10 durante este periodo, puede modificar los datos ya que el nivel de aislamiento *READ COMMITTED* de RFC7, puede permitir visualizar cambios en los datos que otras transacciones están efectuando simultáneamente.
- **Resultado presentado por RFC7:** La consulta RFC7 finaliza después del tiempo de espera, obteniendo los datos de los documentos de ingreso de productos en los últimos 30 días. Y debido al nivel de aislamiento *READ COMMITTED*, permite la lectura del RF10.



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

- Method:** GET
- URL:** `http://localhost:8080/proyecto/recepcionDeProducto/documentos/rfc7/4/6`
- Status:** 200 OK
- Time:** 30.62 s
- Size:** 641 B
- Body:** A JSON array of documents.

```
1 {
2   "documentos": [
3     {
4       "idDocumento": "129",
5       "proveedor": "Distribuciones del ser S.A",
6       "fechaRecepcion": "2024-11-02 12:19:44.0"
7     },
8     {
9       "idDocumento": "43",
10      "proveedor": "Distribuciones del ser S.A",
11      "fechaRecepcion": "2024-10-29 20:05:53.0"
12     },
13     {
14       "idDocumento": "160",
15       "proveedor": "Distribuciones del ser S.A",
16       "fechaRecepcion": "2024-11-02 21:44:57.0"
17     },
18     {
19       "idDocumento": "84",
```

- **Resultado presentado por RF10:** Antes que el RFC7 termine, el RF10 género el registro exitosamente

GET

▼

http://localhost:8080/proyecto/bodega/rf10/orden/21/bodega/6

Send

▼

Params

Authorization

Headers (6)

Body

Scripts

Settings

Cookies

Query Params



	Key	Value	Description	...	Bulk Edit
	Key	Value	Description		

Body

Cookies

Headers (5)

Test Results

200 OK • 1506 ms • 187 B •   Save Response ...


Pretty


Raw


Preview

Visualize

Text ▼







1 redirect:/ordendecompra