Plano de Testes Funcionais

Este plano de testes funcionais tem como objetivo validar os principais fluxos da API REST e do microsserviço de consumo de pedidos, garantindo a persistência e a consulta correta das informações.

1. Teste do Microsserviço: Escuta e armazenamento da fila de pedidos

## Cenário de Teste:

Verificar se o microsserviço (MS) de escuta e armazenamento da fila de pedidos consegue receber e armazenar corretamente os dados de um novo pedido no banco de dados.

## Pré-condições:

* RabbitMQ em execução e fila de pedidos criada.
* MS deve estar em execução aguardando mensagens na fila de pedidos.
* RabbitMQ e banco de dados MongoDB acessíveis pelo microsserviço.

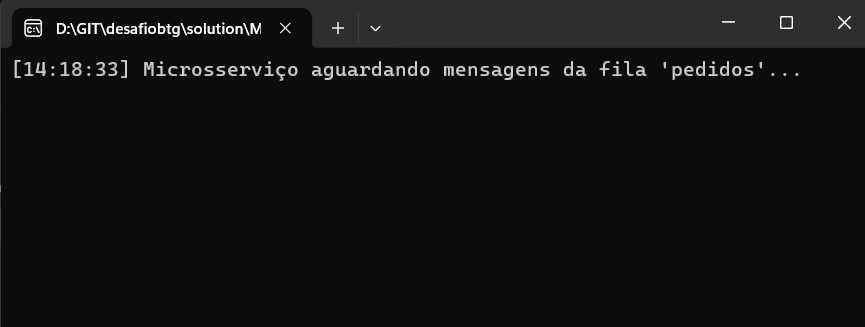
## Passos de Teste:

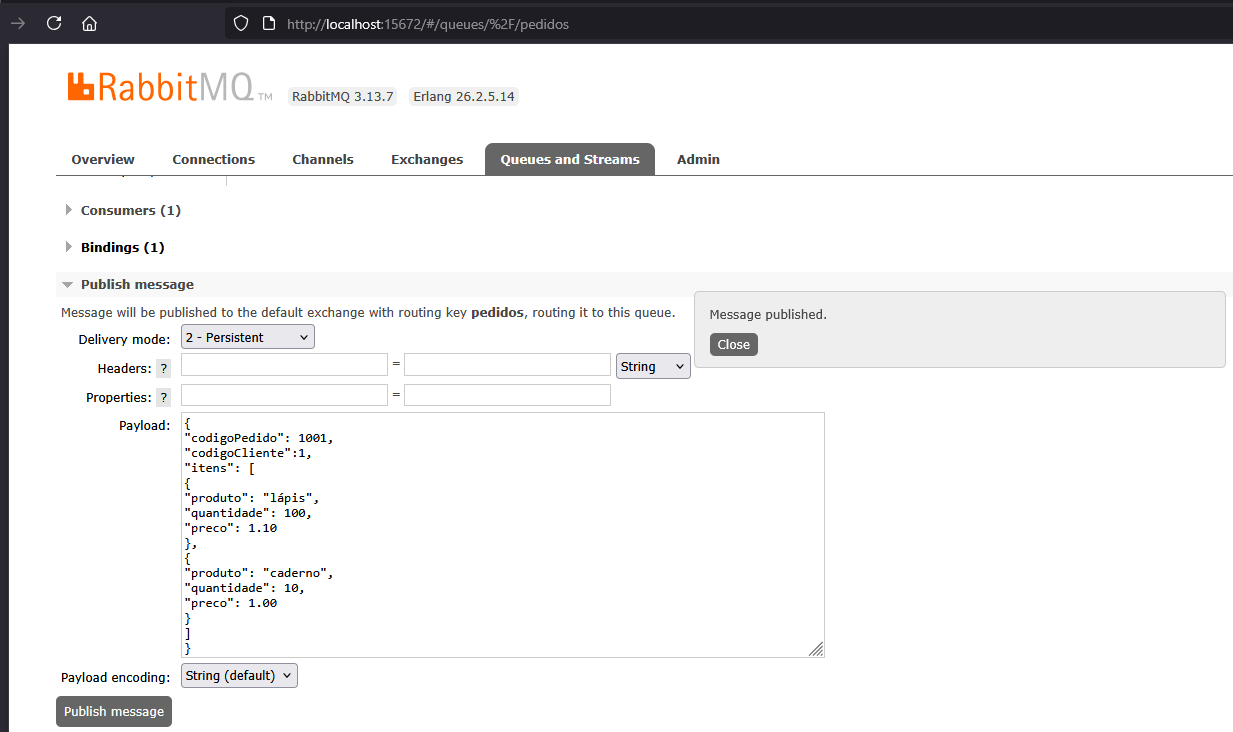
1. Enviar manualmente um novo pedido na fila pedidos do RabbitMQ.
2. Verificar a mensagem de resposta (esperado: 200 OK ou 201 Created).
3. Consultar o banco de dados para confirmar se os dados foram salvos corretamente.

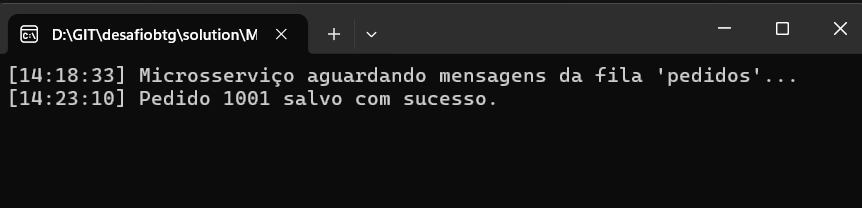
## Resultado Esperado:

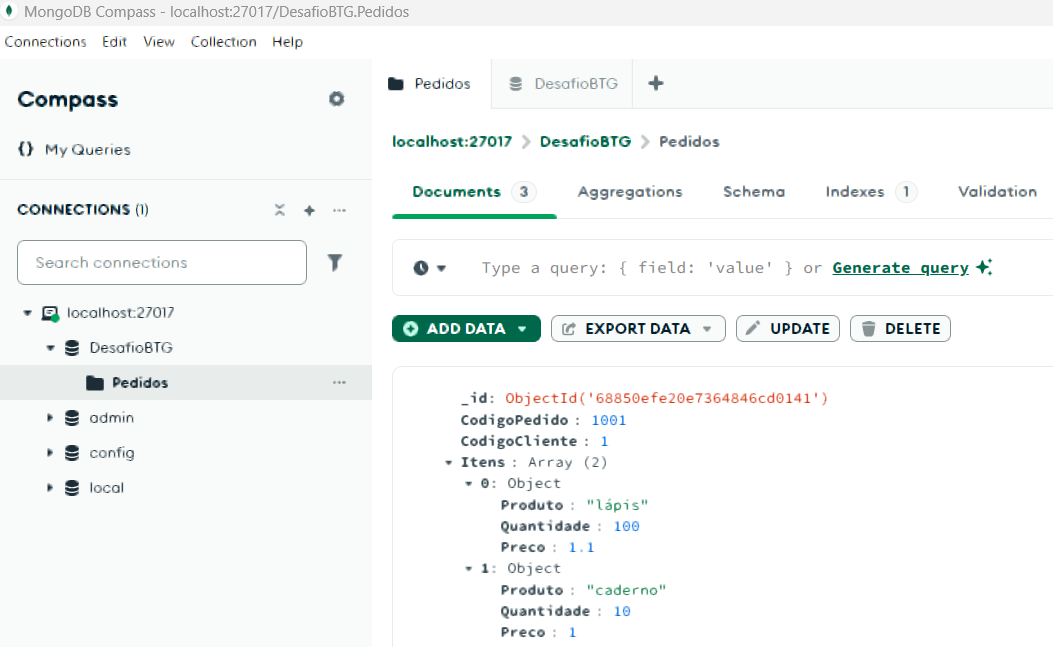
Os dados devem ser salvos no banco de dados sem erros e a resposta do MS deve indicar sucesso.

## Evidências:









# 2. Testes dos Endpoints da API

## Endpoint 1: GET /api - Confirmar que a API está em funcionamento

### Cenário de Teste:

Confirmar que a API está em funcionamento.

### Pré-condições:

* A API deve estar em execução.

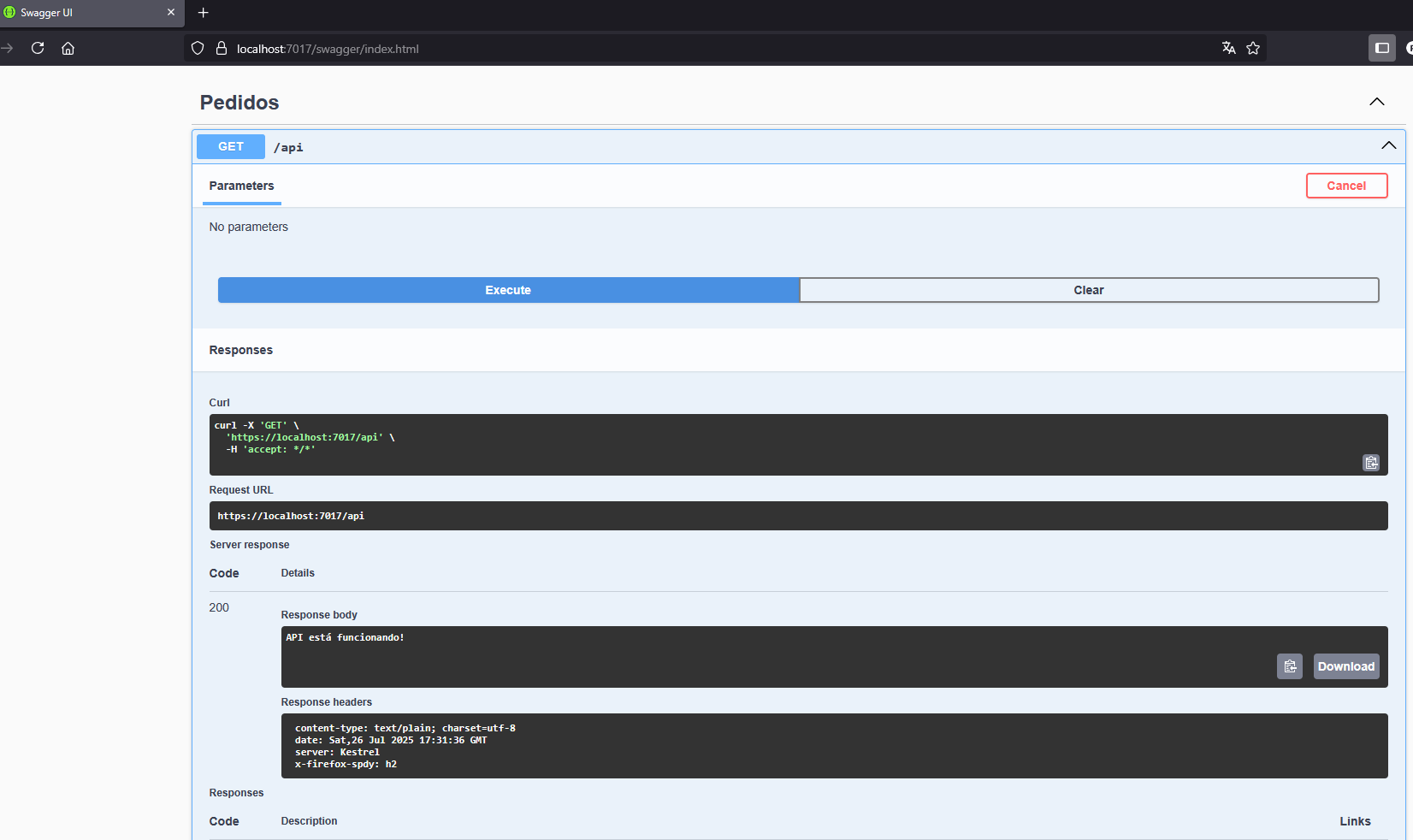
### Passos de Teste:

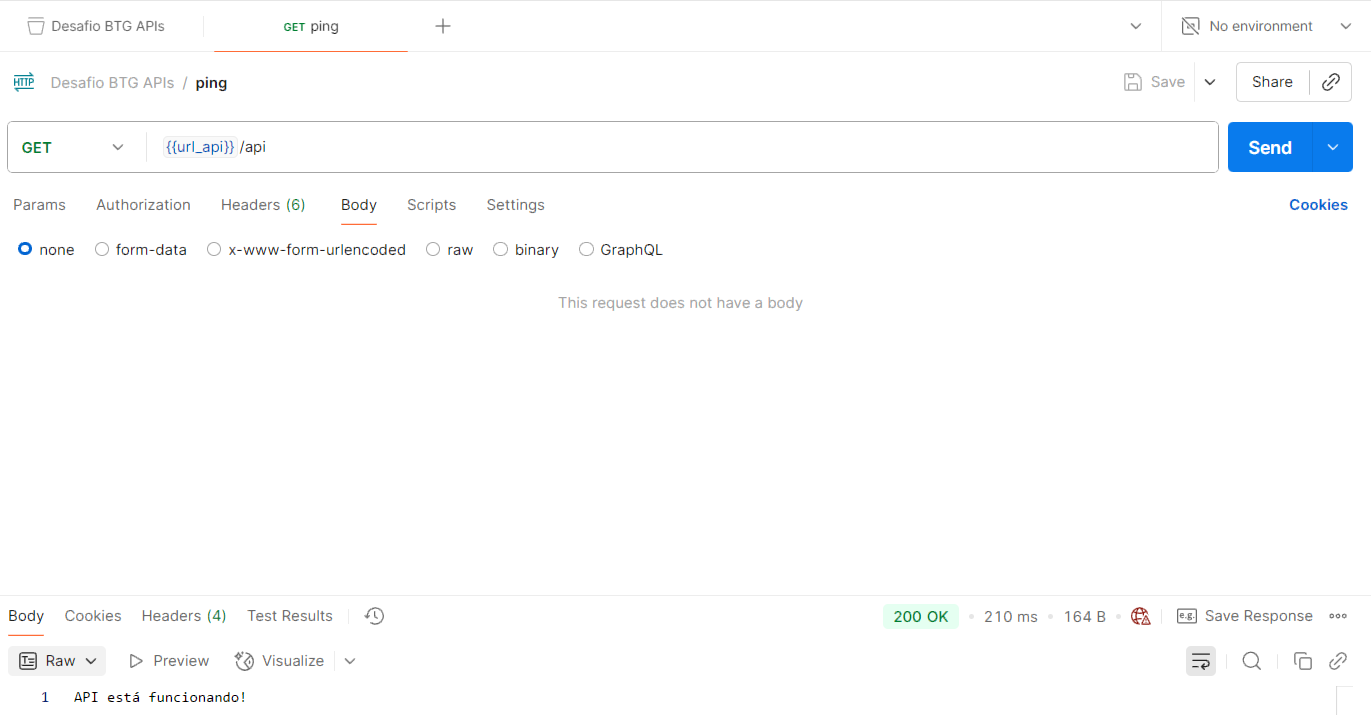
1. Enviar uma requisição GET para /api.
2. Verificar o código de status da resposta (esperado: 200 OK).

### Resultado Esperado:

A API deve retornar a mensagem “API em funcionamento!”.

### Evidências:





## Endpoint 2: POST /api/pedidos - Incluir um pedido na fila de pedidos do RabbitMQ

### Cenário de Teste:

Incluir um pedido na fila de pedidos do RabbitMQ para agilizar a inclusão de pedidos.

### Pré-condições:

* A API deve estar em execução.
* RabbitMQ em execução e fila de pedidos criada.
* A mensagem a ser consumida deve estar no padrão JSON definido.

Exemplo da mensagem que deve ser consumida:

{

"codigoPedido": 1001,

"codigoCliente":1,

"itens": [

{

"produto": "lápis",

"quantidade": 100,

"preco": 1.10

},

{

"produto": "caderno",

"quantidade": 10,

"preco": 1.00

}

]

}

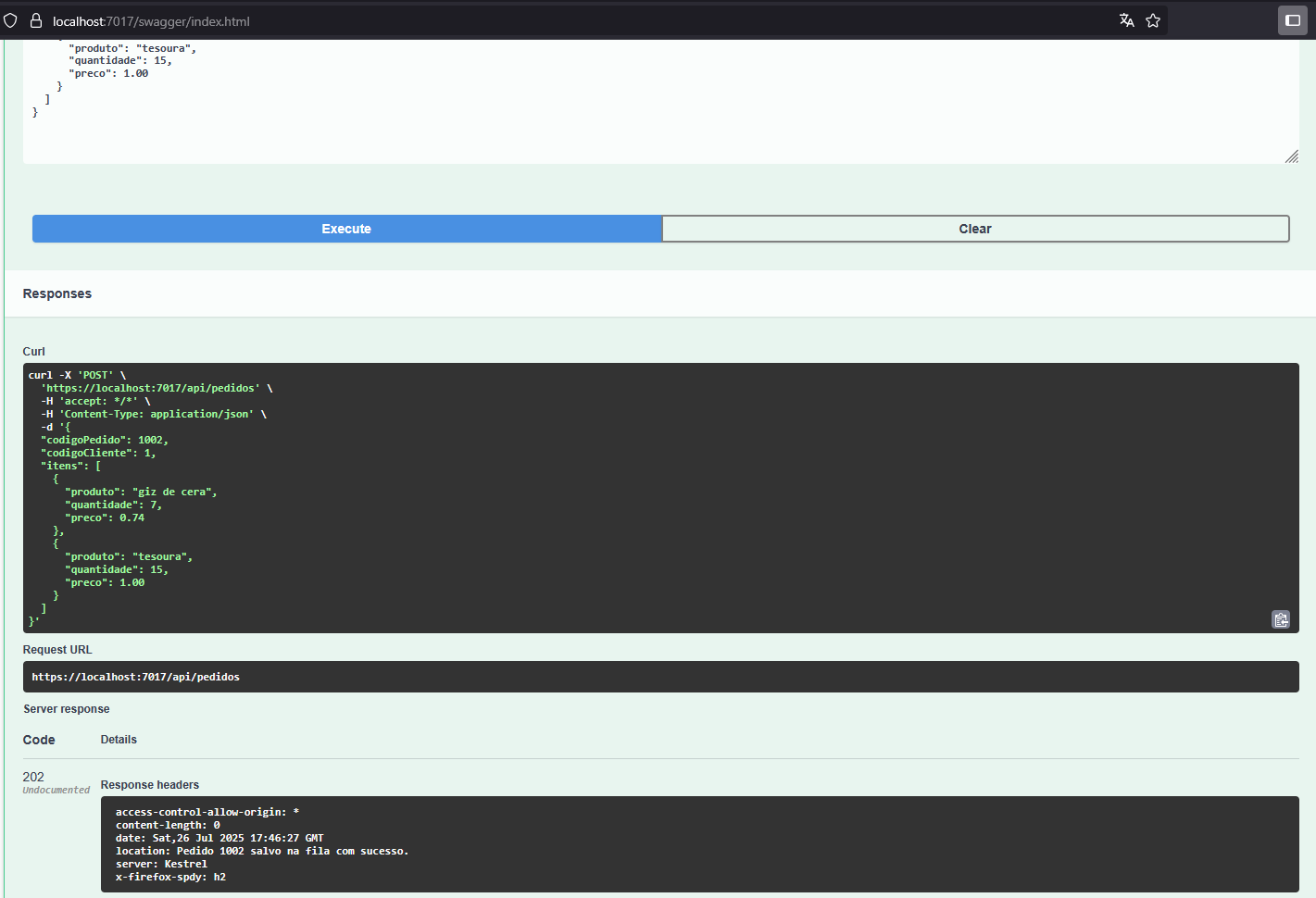
### Passos de Teste:

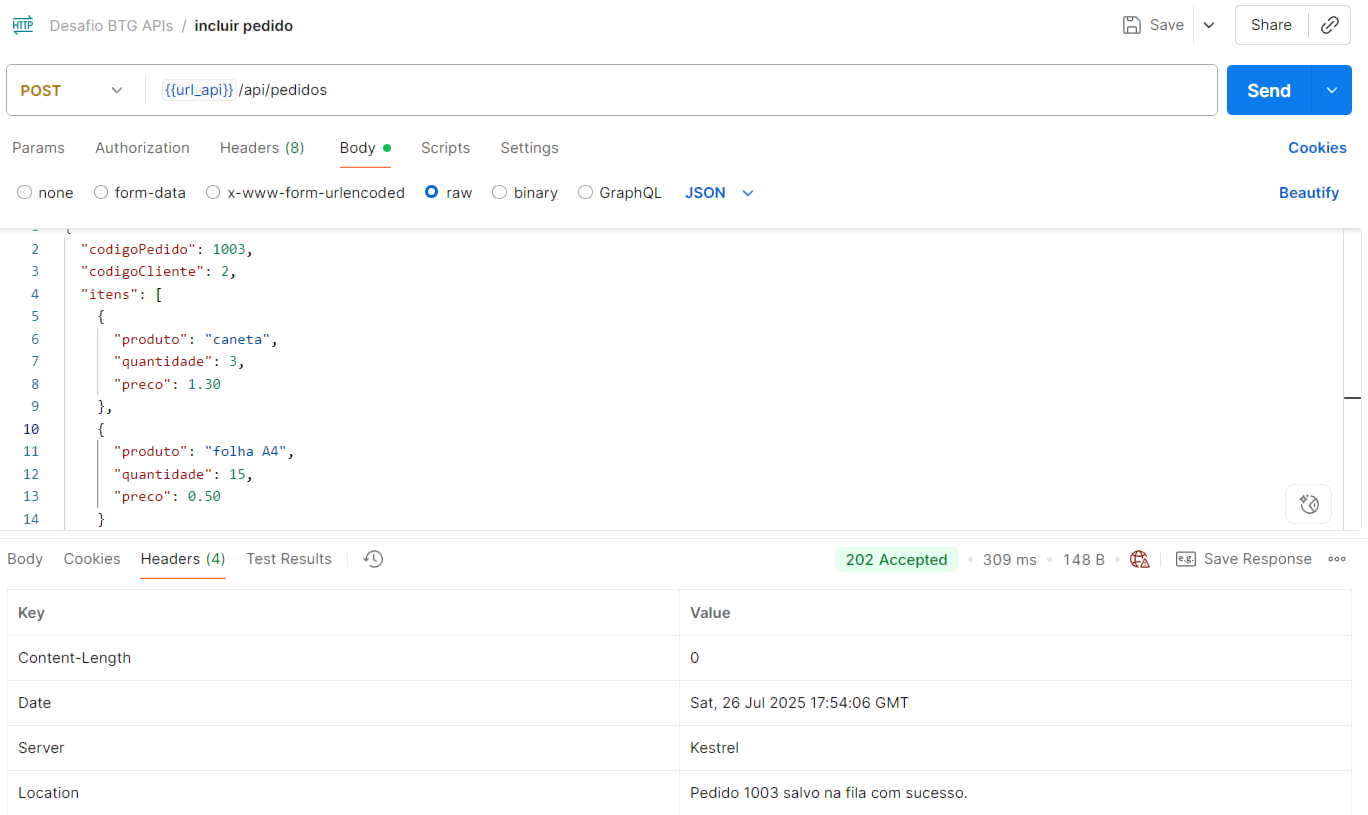
1. Informar mensagem no padrão JSON a ser consumida.
2. Enviar uma requisição POST para /api/pedidos.
3. Verificar o código de status da resposta (esperado: 202 Accepted).

### Resultado Esperado:

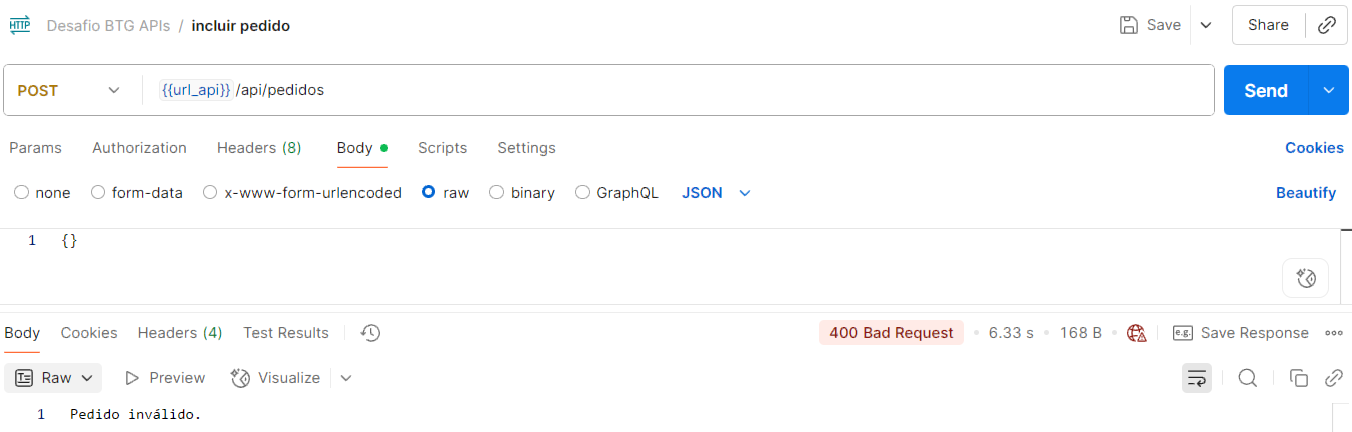
A API deve retornar a mensagem “Pedido XXXX salvo na fila com sucesso.”.

### Evidências:

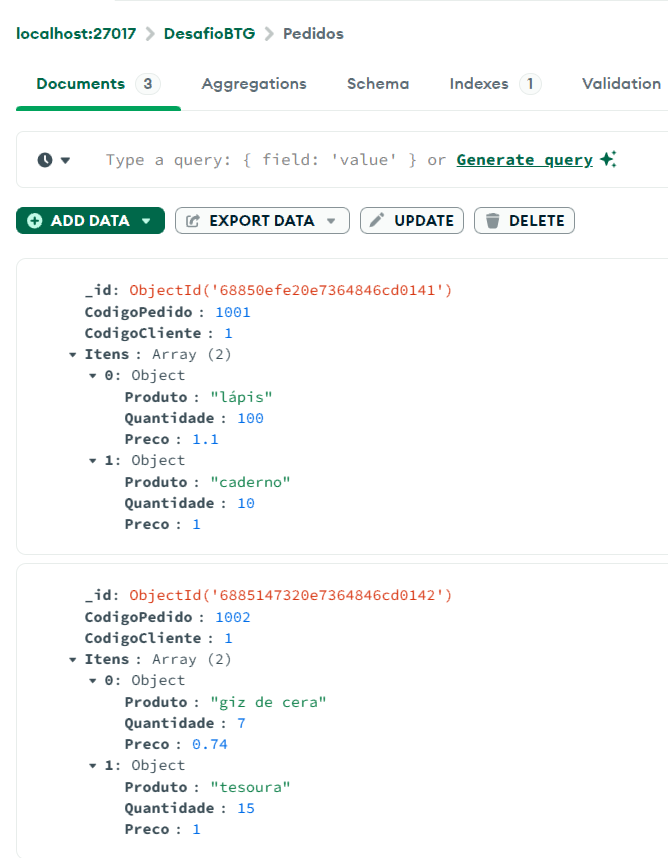
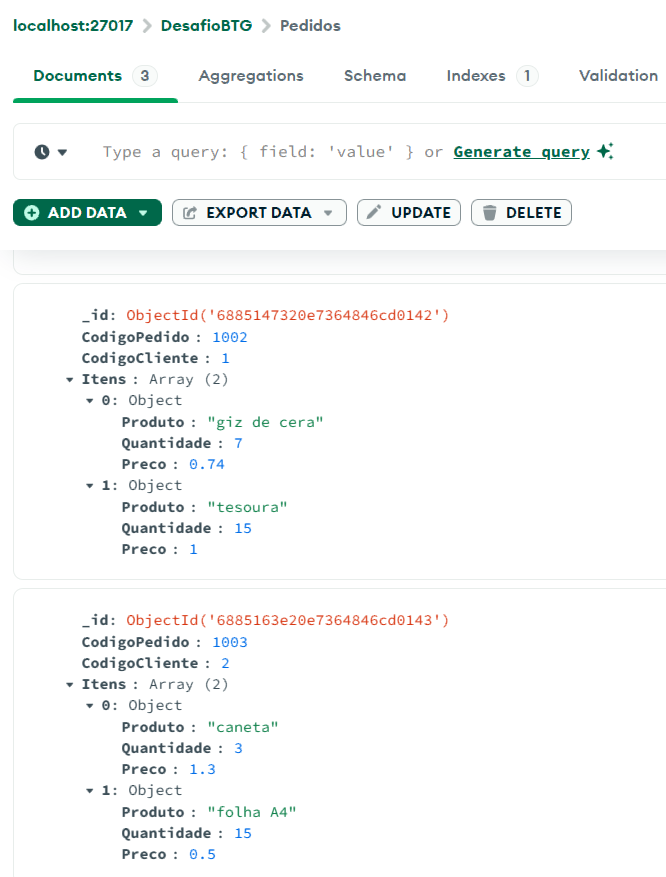




Se o JSON enviado estiver fora do padrão, deve retornar o código de status da resposta 400 Bad Request e a mensagem "Pedido inválido".



Total de 3 pedidos em 2 clientes diferentes na base de dados para os demais testes.

## Endpoint 3: GET /api/pedidos/{codigo\_cliente} - Listar pedidos por cliente

### Cenário de Teste:

Listar todos os pedidos do cliente.

### Pré-condições:

* A API deve estar em execução.
* A base de dados deve conter pedidos do cliente informado.

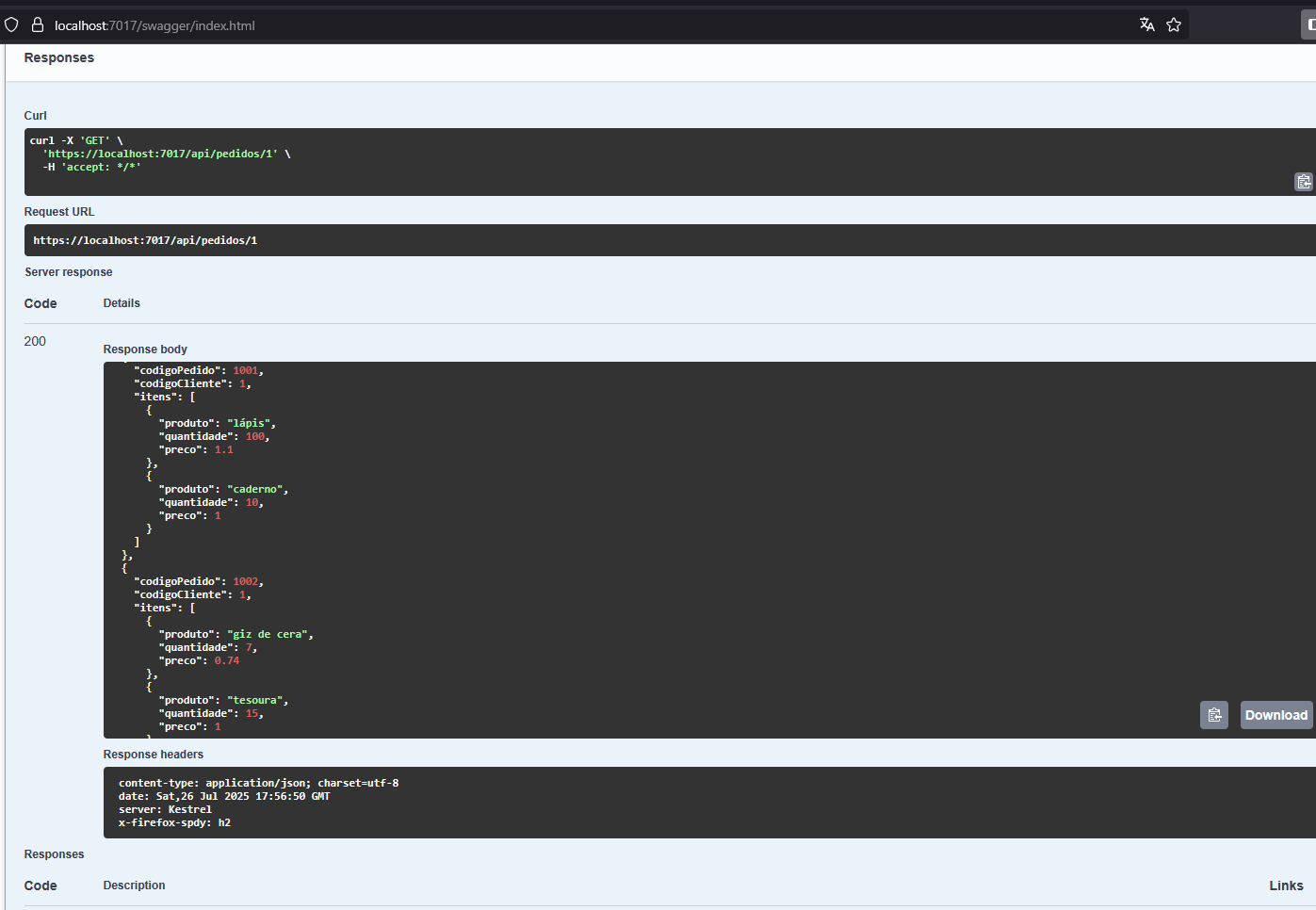
### Passos de Teste:

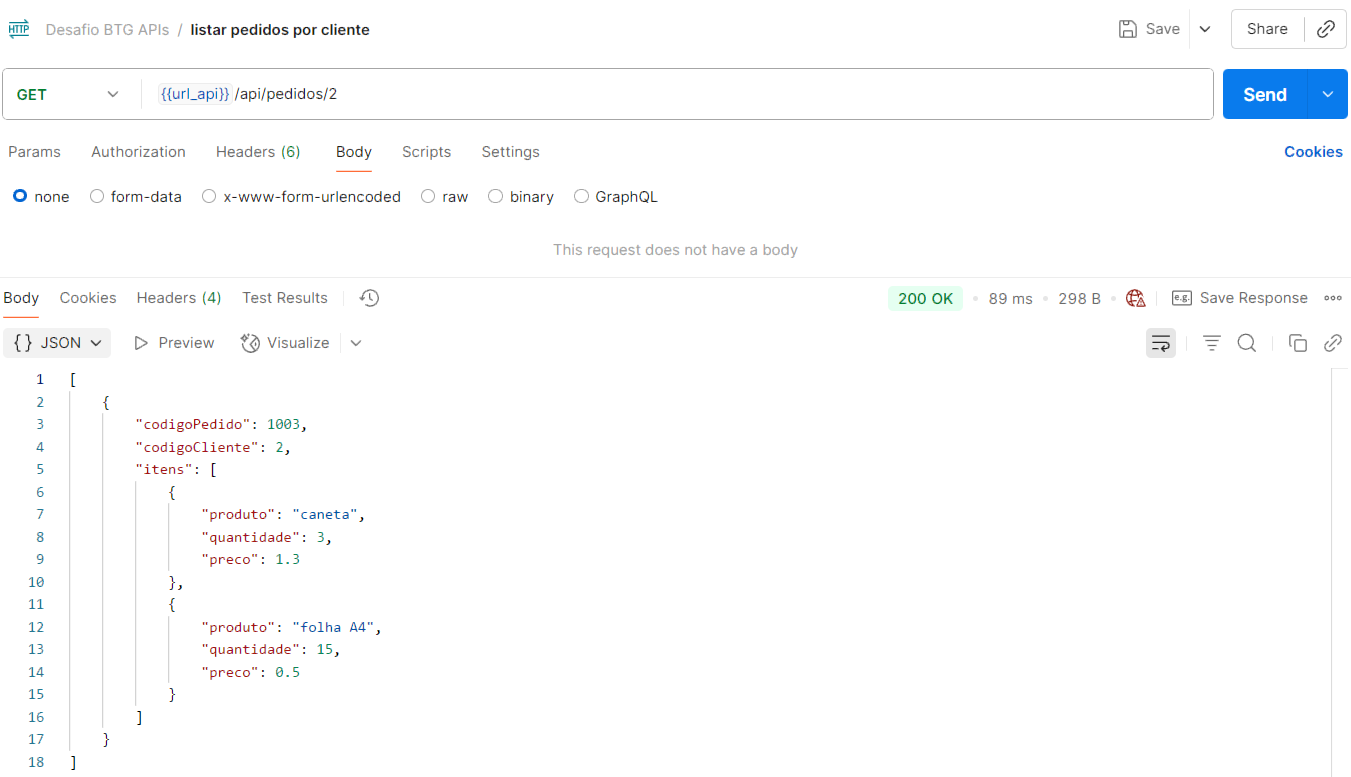
1. Enviar uma requisição GET para /api/pedidos/{codigo\_cliente} onde {codigo\_cliente} deve conter o código do cliente.
2. Verificar o código de status da resposta (esperado: 200 OK).

### Resultado Esperado:

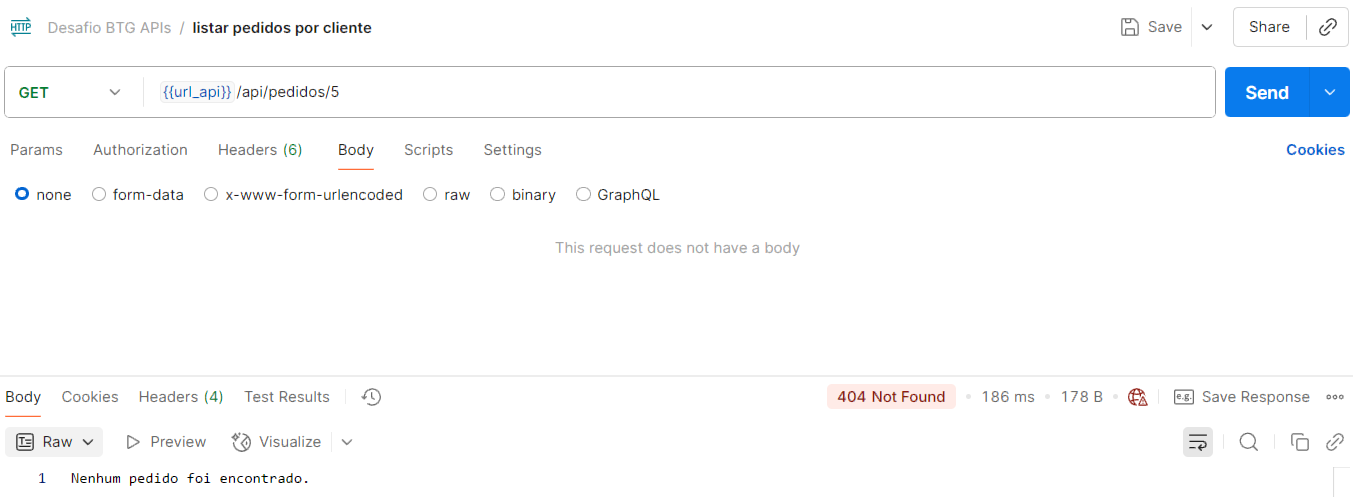
A API deve retornar todos os pedidos do cliente previamente cadastrados no formato JSON.

### Evidências:





Ao informar um código de cliente inexistente na base de dados, é retornado o código de status da resposta 404 e mensagem “Nenhum pedido foi encontrado”.



## Endpoint 4: GET /api/pedidos/{codigo\_cliente}/quantidade - Obter a quantidade de pedidos por cliente

### Cenário de Teste:

Retornar a quantidade de pedidos por cliente.

### Pré-condições:

* A API deve estar em execução.
* A base de dados deve conter pedidos do cliente informado.

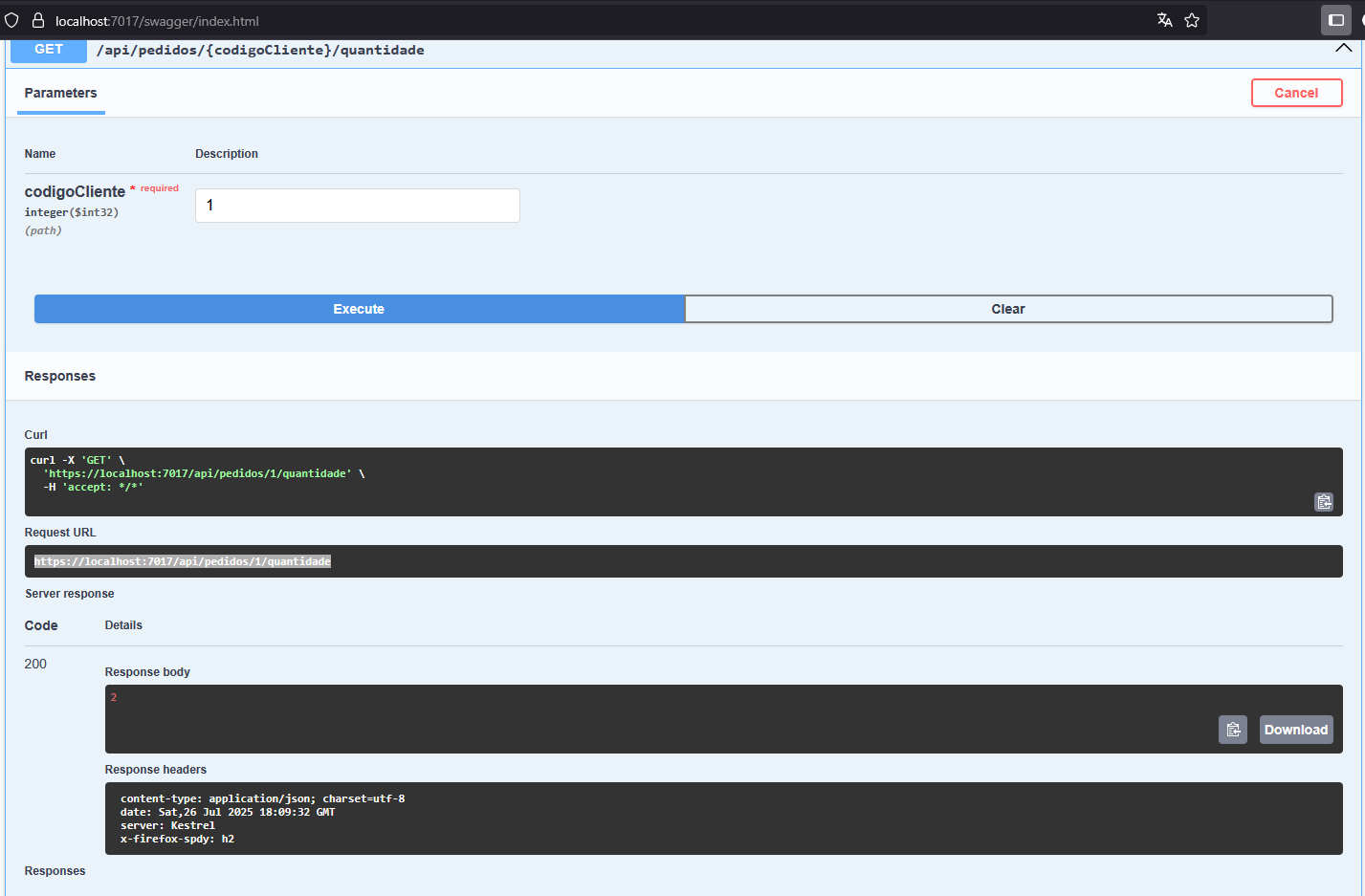
### Passos de Teste:

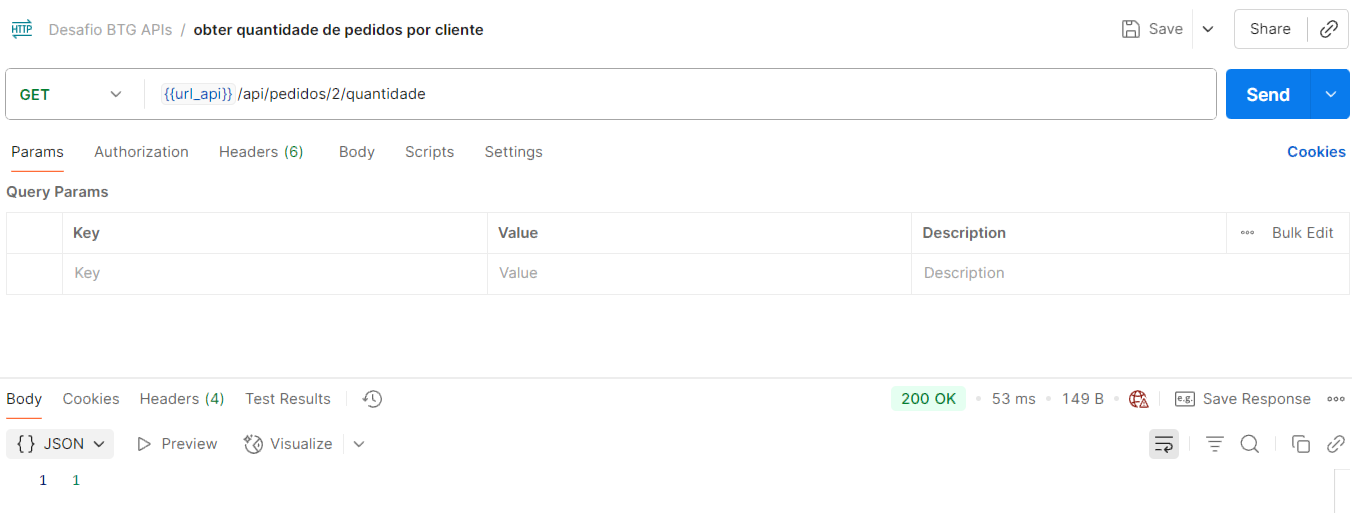
1. Enviar uma requisição GET para /api/pedidos/{codigo\_cliente}/quantidade onde {codigo\_cliente} deve conter o código do cliente.
2. Verificar o código de status da resposta (esperado: 200 OK).

### Resultado Esperado:

A API deve retornar um valor numérico com a quantidade total de pedidos do cliente informado.

### Evidências:





## Endpoint 5: GET /api/pedidos/{codigo\_pedido}/valortotal - Obter o valor total de um pedido

### Cenário de Teste:

Retornar o valor total por pedido.

### Pré-condições:

* A API deve estar em execução.
* A base de dados deve conter itens e o pedido informado.

### Passos de Teste:

1. Enviar uma requisição GET para /api/pedidos/{codigo\_pedido}/valortotal onde {codigo\_pedido} deve conter o código do pedido.
2. Verificar o código de status da resposta (esperado: 200 OK).

### Resultado Esperado:

A API deve retornar um valor total do pedido informado (quantidade x preço de cada item).

### Evidências:

