# Desafio Final - Caio Reimberg

A aplicação escolhida foi uma API REST para manipulação e consulta de Pedidos utilizando SpringBoot Java e um repositório H2 como base de dados para persistência.

Foi utilizado o site Spring Initializr para gerar um pacote Maven que posteriormente foi importado em uma IDE (Eclipse IDE) a fim de gerar a aplicação Java com repositório, e permitir validar as APIs geradas e resultados esperados.

Abaixo podemos ver os diagramas C4 da aplicação

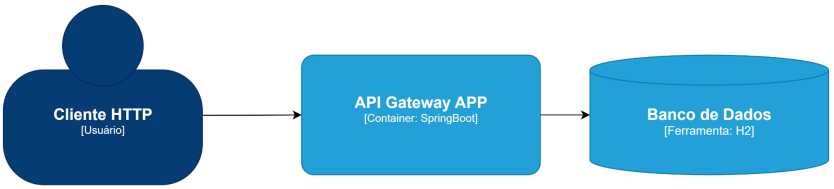



Imagem 1: Diagrama de Contexto

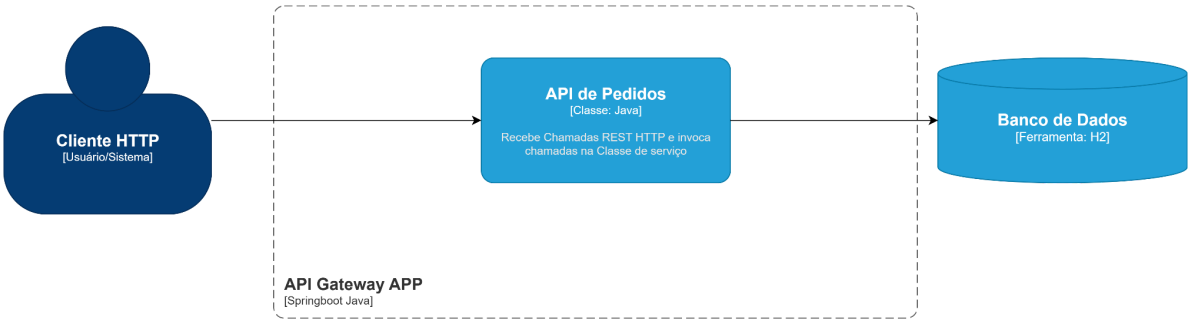



Imagem 2: Diagrama de Contêineres

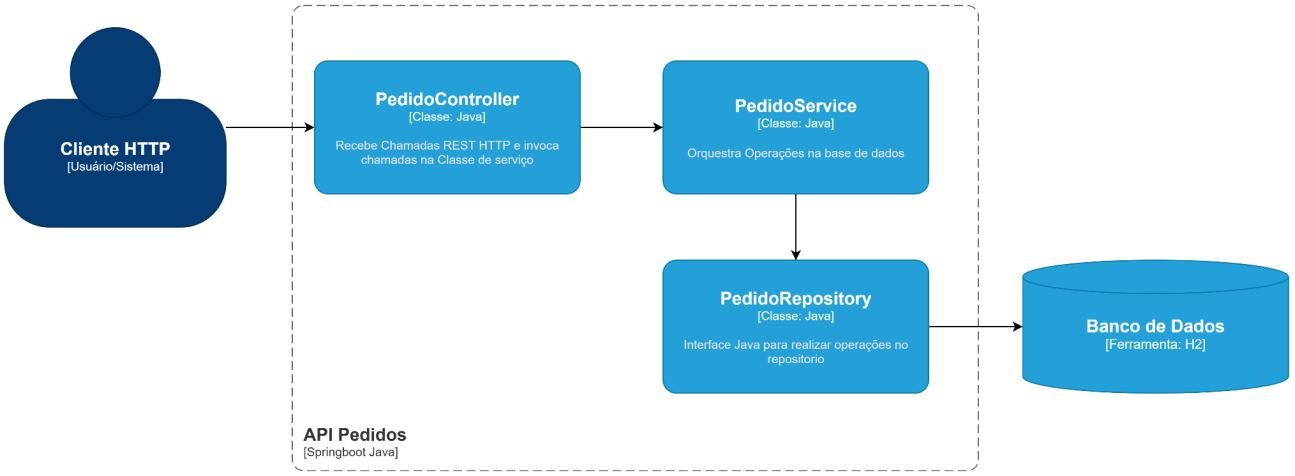



Imagem 3: Diagrama de Componentes

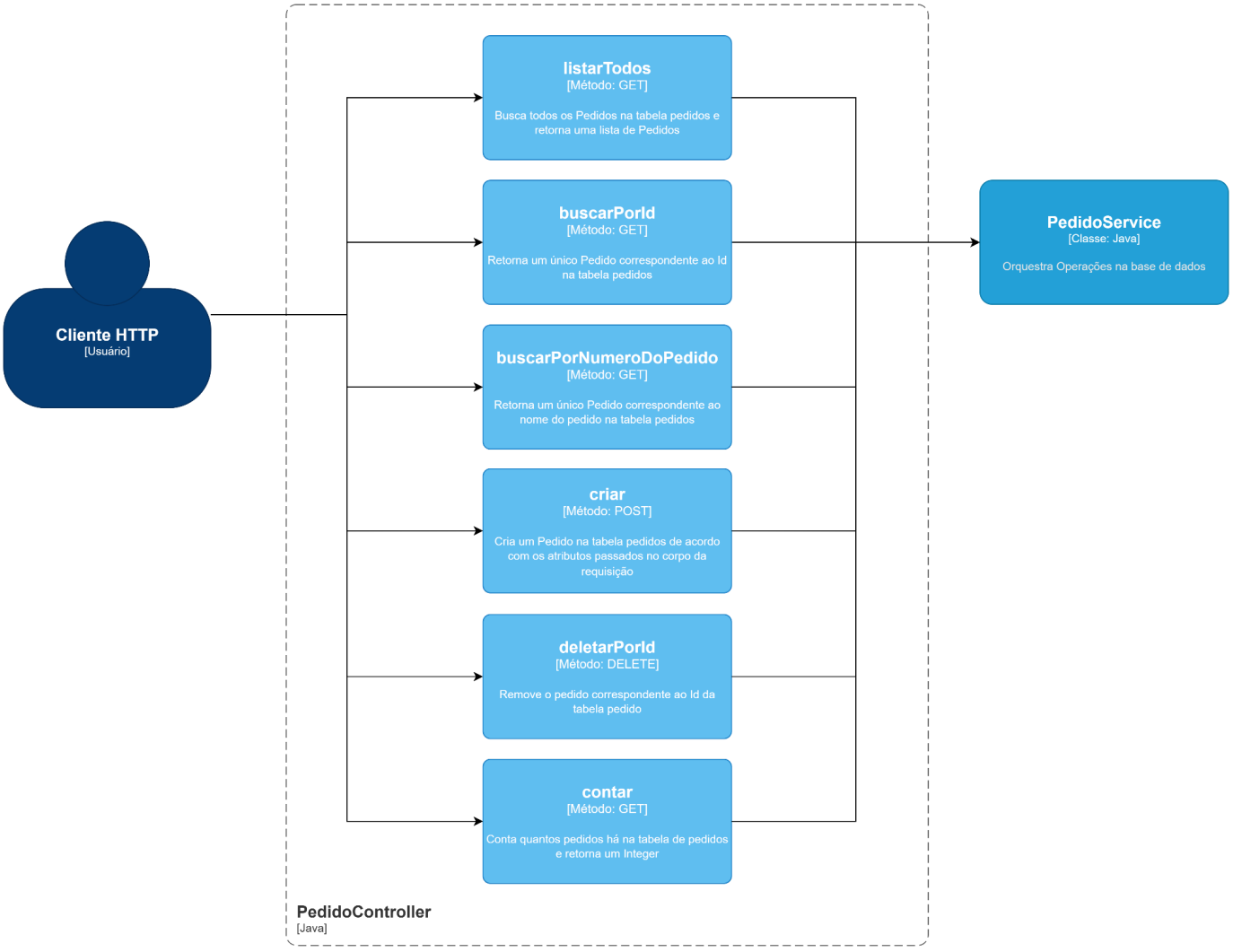



Imagem 4: Diagrama de Código - PedidoController

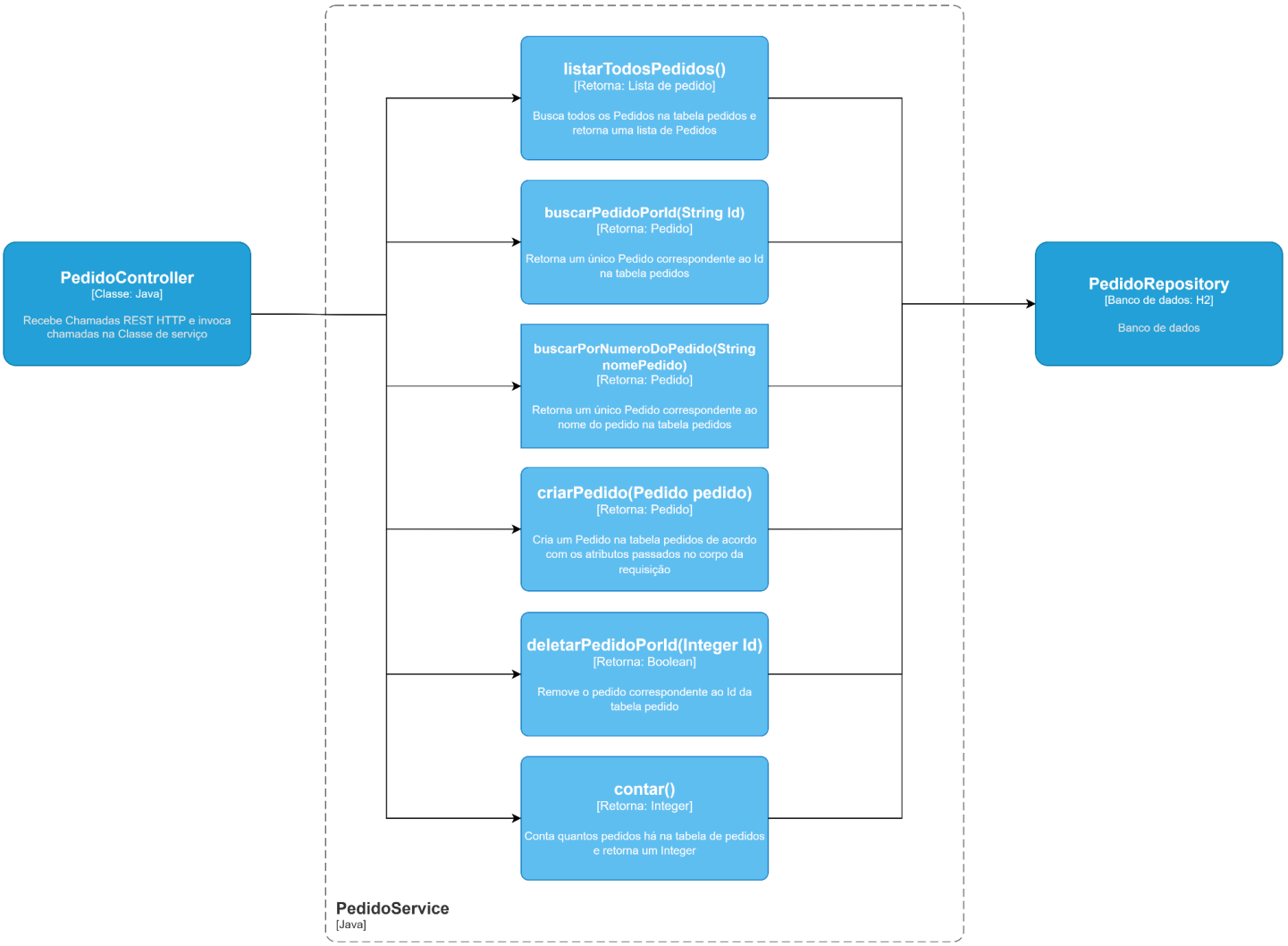
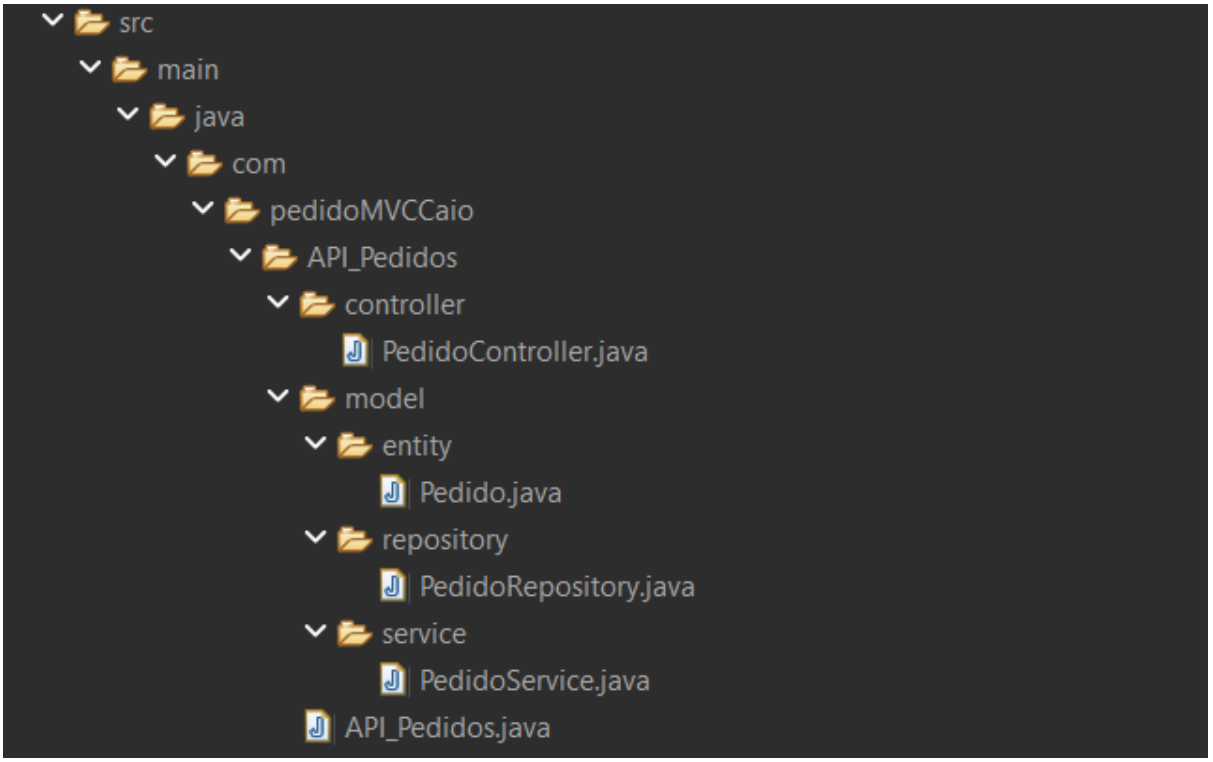



Imagem 4: Diagrama de Código - PedidoService

A estrutura do projeto foi organizado seguindo as diretrizes do MVC clássico, separando as responsabilidades entre a camada de Model e Controller (a camada de view não foi utilizada nesse caso já que se trata apenas de uma API REST).



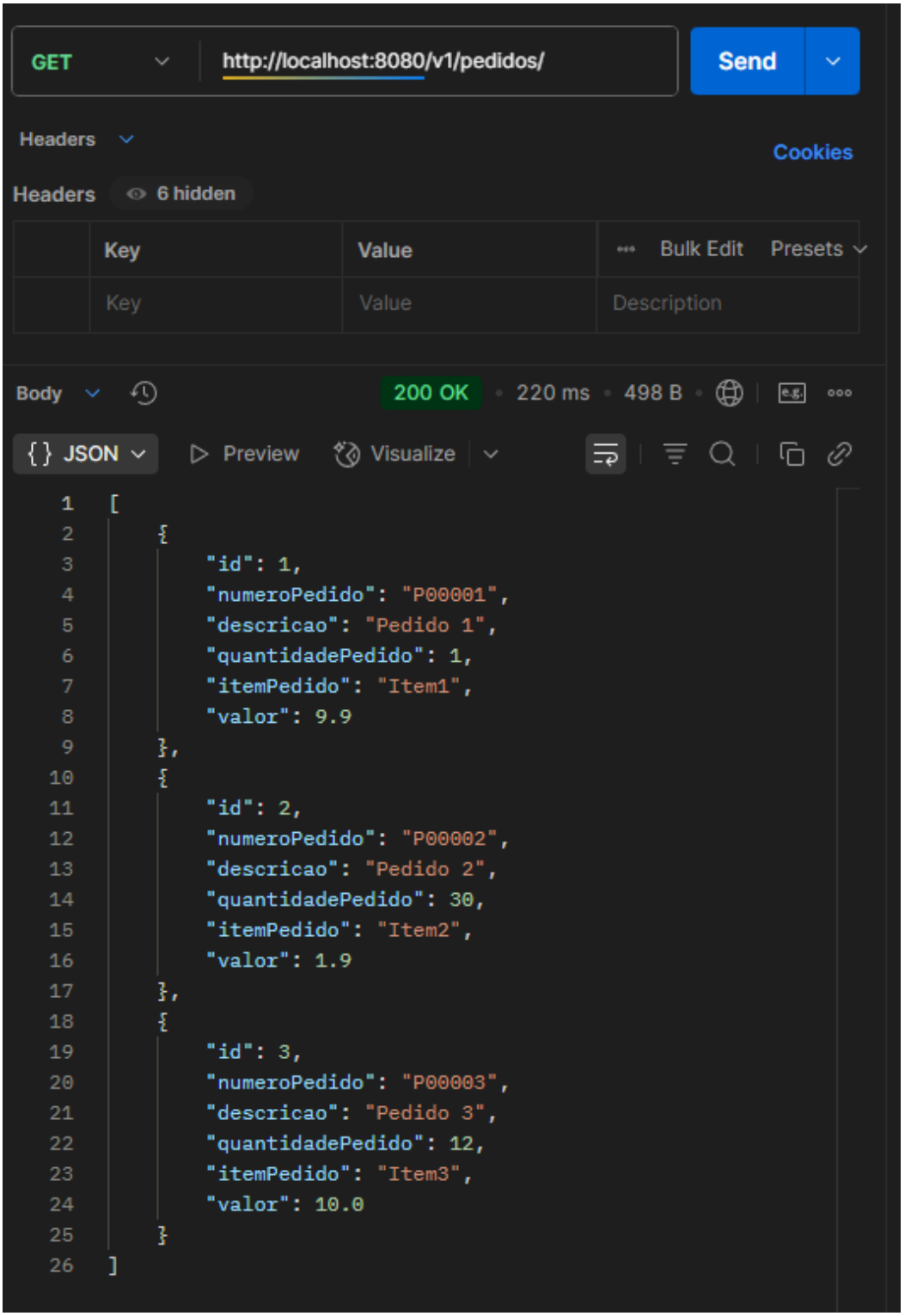

Na tabela abaixo podemos ver as responsabilidades que buscamos alcançar com cada camada do repositório:

| Camada | Pasta | Responsabilidade |
|---|---|---|
| Controller | Controller | Camada que descreve os métodos HTTP REST recebidos, traduz os parâmetros, retorna uma resposta de acordo com a operação realizada |
| Model | Entity | Aqui estão as entidades de domínio, descrevendo os campos pertencentes a cada uma. |
| Model | Service | Essa camada concentra a lógica de negócio e orquestra as operações a serem realizadas no repositório. |
| Model | Repository | Abstrai acesso a dados e permite trabalhar com persistência de dados (Spring Data JPA, JDBC por exemplo). |

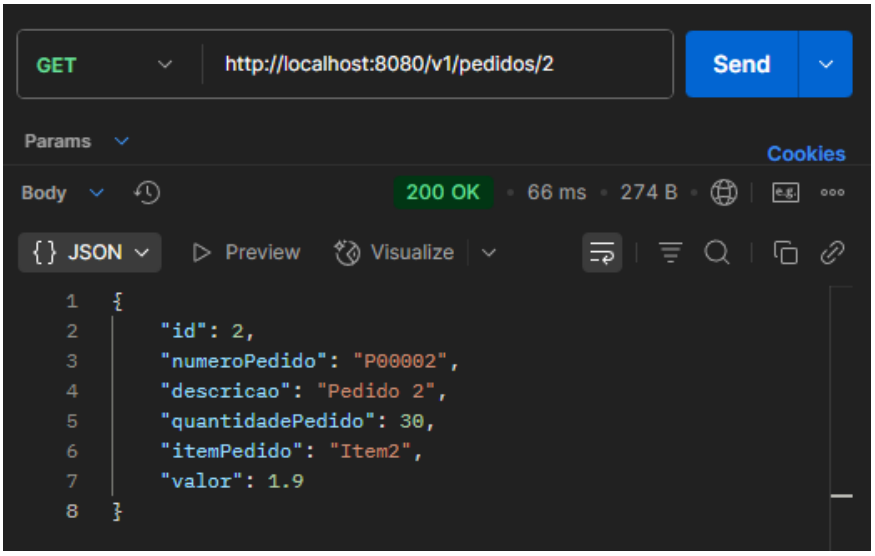
Tabela 1: Descrição das pastas e responsabilidades

Após montar as classes e os métodos sugeridos no enunciado, foi possível testar as chamadas utilizando o Postman e validar a interação entre as chamadas realizadas e o banco de dados.

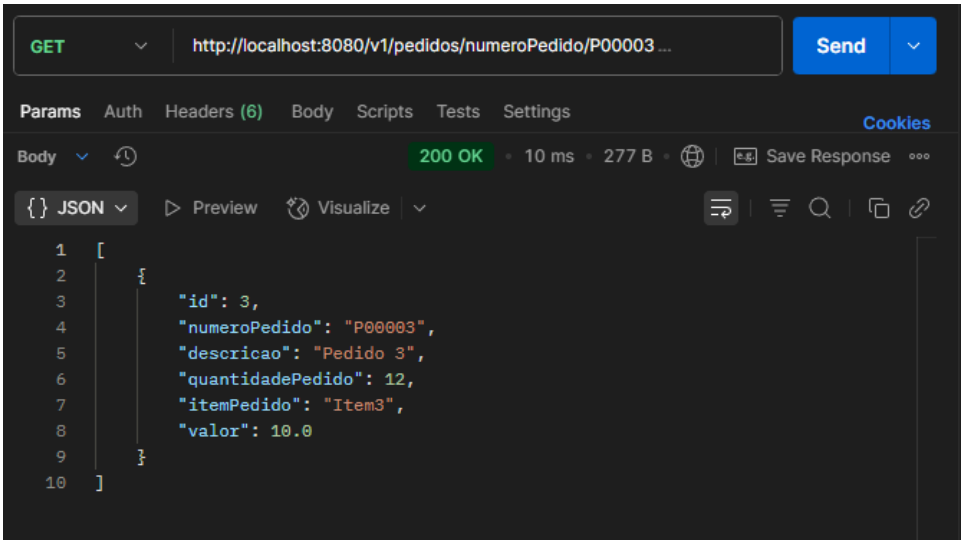
1. **Listar Todos os Pedidos**
   Endpoint: /v1/pedidos/
   Método: GET
   Resposta: List<Pedido>



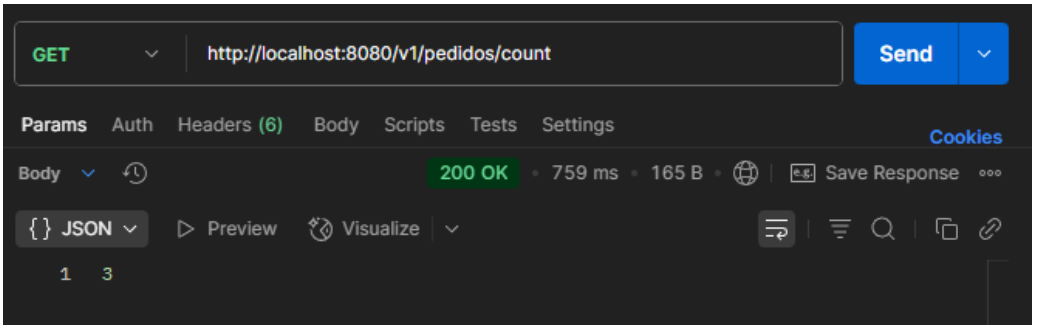
2. **Buscar Por Id**
   Endpoint: /v1/pedidos/{id}
   Método: GET
   Resposta: Pedido



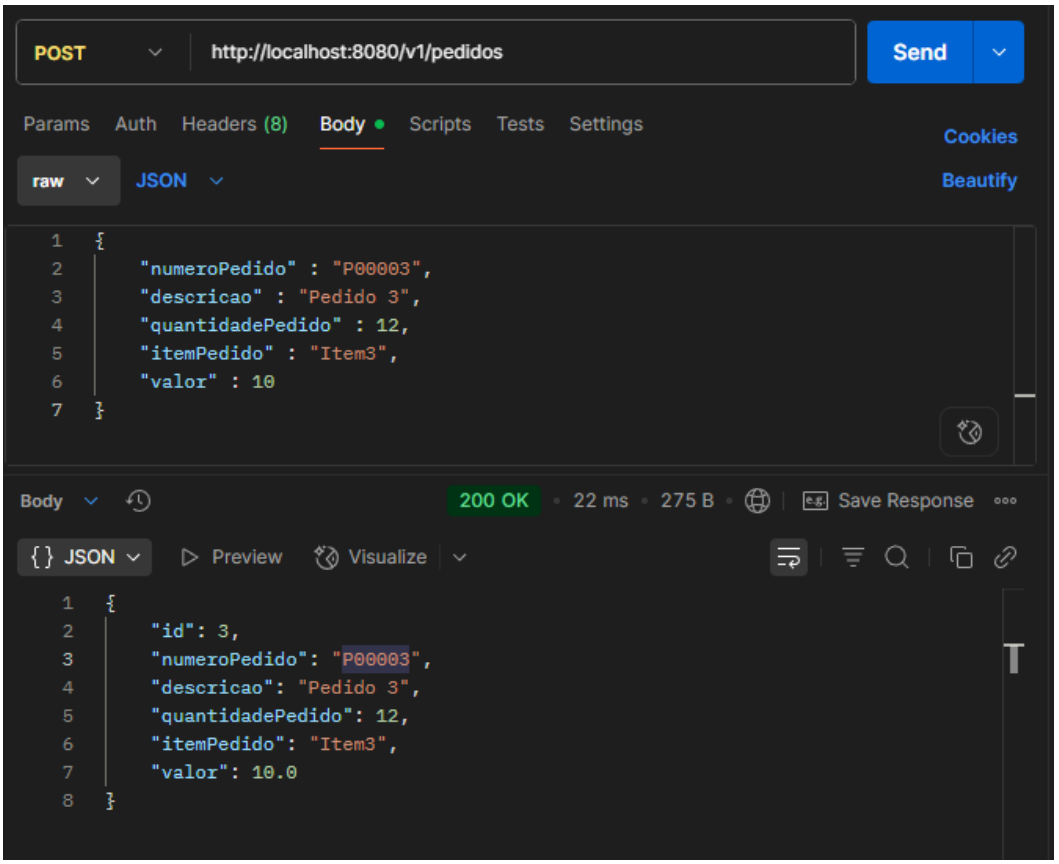

3. **Buscar Por Numero Do Pedido**
   Endpoint: /v1/pedidos/numeroPedido/{numeroPedido}
   Método: GET
   Resposta: List<Pedido>



4. **Contagem de pedidos**
   Endpoint: /v1/pedidos/numeroPedido/{numeroPedido}
   Método: GET
   Resposta: Integer




5. **Criar pedido**
   Endpoint: /v1/pedidos
   Método: POST
   Resposta: Integer



6. **Deletar Pedido**
   Endpoint: /v1/pedidos/{id}
   Método: DELETE
   Resposta: Boolean