Tech Challenge Fase 01 – Taste Food

Grupo G13

Cristiano de Barros – **RM355052** - Discord: **cristiano.barros13**

Graziela Göedert de Souza – **RM355051** - Discord: **graziela_goedert**

Repositório: https://github.com/grazielags/fiap-tech-challenge

Miro: https://miro.com/app/board/uXjVKCp5B74=/

Swagger: http://localhost:8080/swagger-ui/index.html#/

27 de Maio de 2024

Sumário

1. Tecnologias ... 3
2. Como inicializar a aplicação ... 3
2.1. Dependências ... 3
2.1.1. Dependências obrigatórias ... 3
2.1.2. Dependências opcionais ... 3
2.2. Inicialização do projeto ... 3
2.3. Setup do Projeto ... 4
3. Contextos ... 4
3.1. Contexto do Cliente: ... 4
Cliente não existente (Retorna 204 No Content) ... 4
Cliente existente (Retorna 200 e no body os dados do cliente) ... 4
Cadastro de Cliente ... 5
Lista de Clientes ... 5
3.2. Contexto do Produto: ... 6
Produto para uma determinada categoria que não existente (Retorna 204 No Content) ... 6
Produto para uma determinada categoria ... 6
Cadastro de Produto ... 7
Lista de Produtos ... 7
Edição de Produto ... 8
Remoção de Produto ... 8
3.3. Contexto: Pedido ... 9
Realizando o Pedido ... 9
Listando os Pedidos ... 9
Lista de Pedidos Recebidos ou Em Preparação ... 10
Alterar status do Pedido ... 10
3.4. Contexto: Checkout ... 11
Realiza checkout do Pedido ... 11
4. Complementares ... 11
4.1. Swagger ... 12
4.2. Endpoints Complementares ... 12
Buscar todas as categorias de produtos ... 12
Buscar todos os status possíveis de pedidos ... 13

Fluxo de Eventos
Taste Food

1. Tecnologias

Para o desenvolvimento do projeto, foram escolhidas as seguintes tecnologias:

- Java - Linguagem de Programação
- Postgres – Banco de Dados Relacional
- Docker - Containers

2. Como inicializar a aplicação

2.1. Dependências

Para a execução da aplicação, é necessário que algumas dependências (algumas delas opcionais) sejam instaladas em sua máquina, sendo elas:

2.1.1. Dependências obrigatórias

- **Docker**

2.1.2. Dependências opcionais

- **Postman** – REST Client utilizado nos testes das APIs, podendo utilizar como substituto o Swagger da aplicação para realizar as requisições necessárias.

2.2. Inicialização do projeto

A inicialização do projeto ocorre em poucos passos:

- Baixe o **repositório**: https://github.com/grazielags/fiap-tech-challenge
- Abra a pasta via linha de comando. Ex.: ***c:\> cd "c:/projetos/fiap-tech-challenge"***
- Dentro da pasta do projeto ***fiap-tech-challenge*** execute "***docker-compose up***"

2.3. Setup do Projeto

O projeto não conta com dados predefinidos, dessa forma deve ser criado os usuários, produtos e pedidos para a realização dos testes, seja pelo Postman ou pelo Swagger.

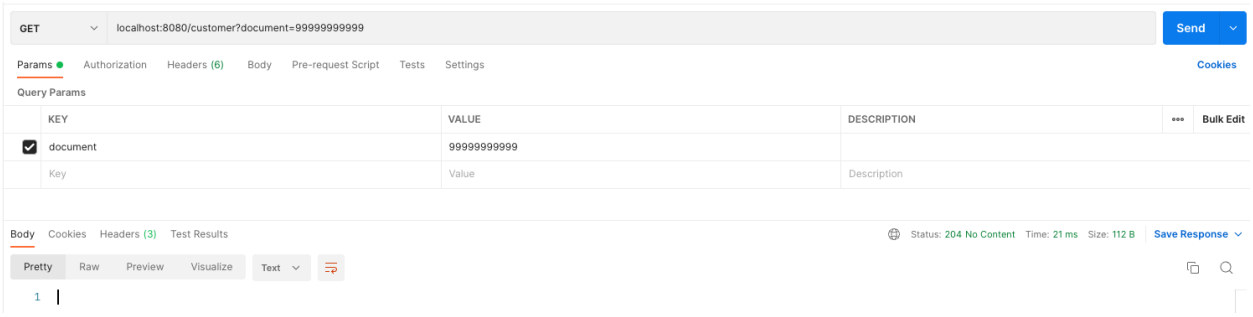
3. Contextos

3.1. Contexto do Cliente:

Nesse contexto, temos a manipulação de dados dos clientes (Cadastro, Listagem de todos os clientes, Busca de Clientes pelo documento).

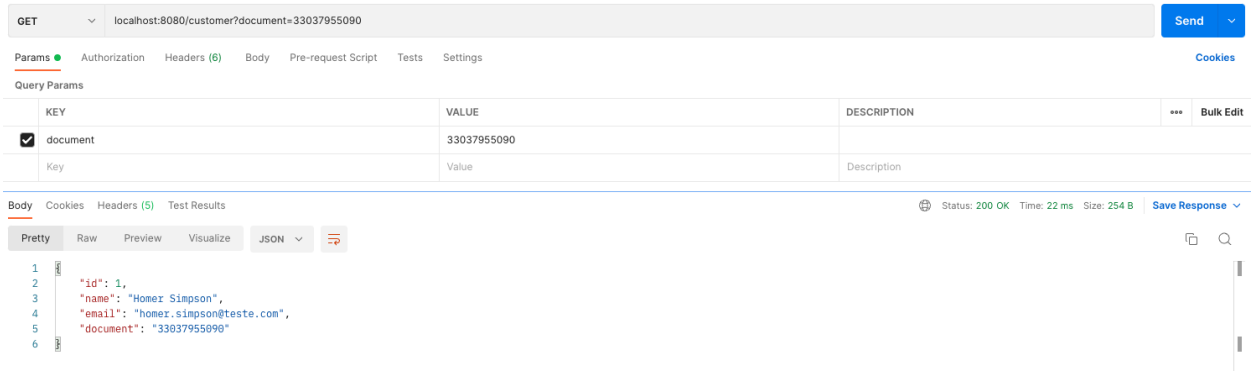
Cliente não existente (Retorna 204 No Content)

curl --location --request GET 'localhost:8080/customer?document=99999999999'



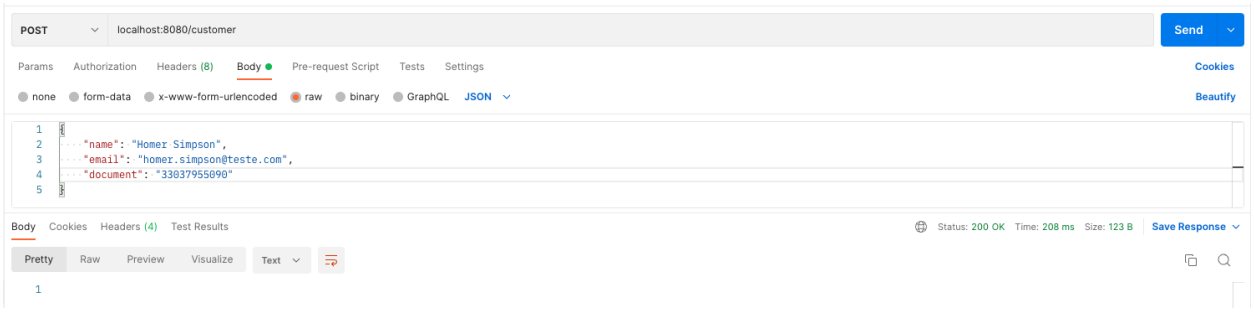
Cliente existente (Retorna 200 e no body os dados do cliente)

curl --location --request GET 'localhost:8080/customer?document=33037955090'



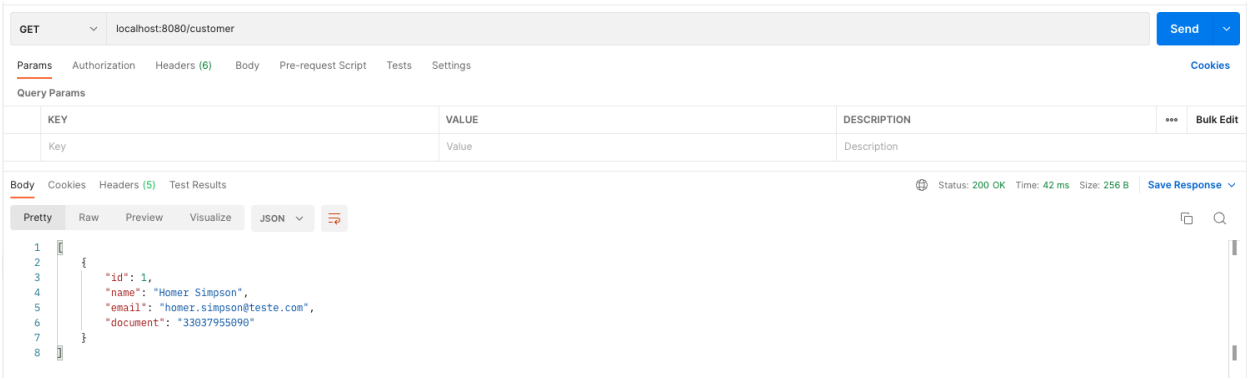
Cadastro de Cliente

```
curl --location --request POST 'localhost:8080/customer' \
--header 'Content-Type: application/json' \
--data-raw '{
    "name": "Homer Simpson",
    "email": "homer.simpson@teste.com",
    "document": "33037955090"
}'
```



Lista de Clientes

```
curl --location --request GET 'localhost:8080/customer'
```

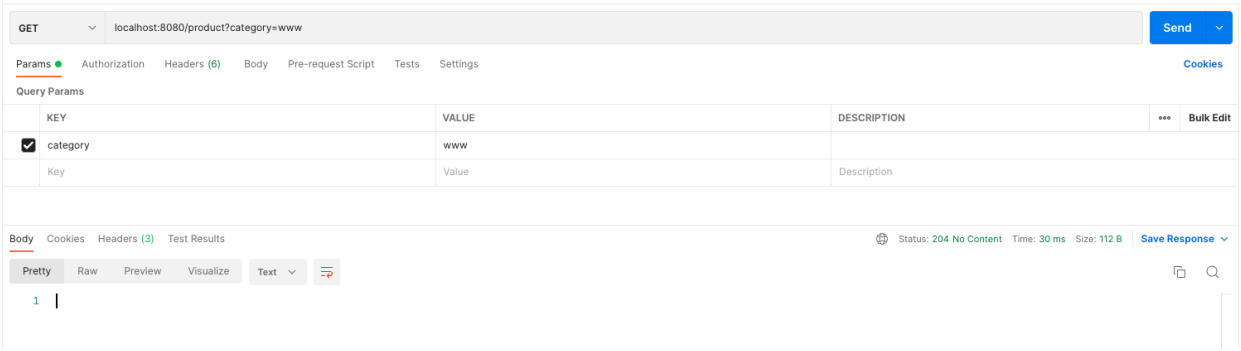


3.2. Contexto do Produto:

Nesse contexto, temos a manipulação de dados dos produtos (Cadastro, Edição, Remoção e Listagem de todos os Produtos).

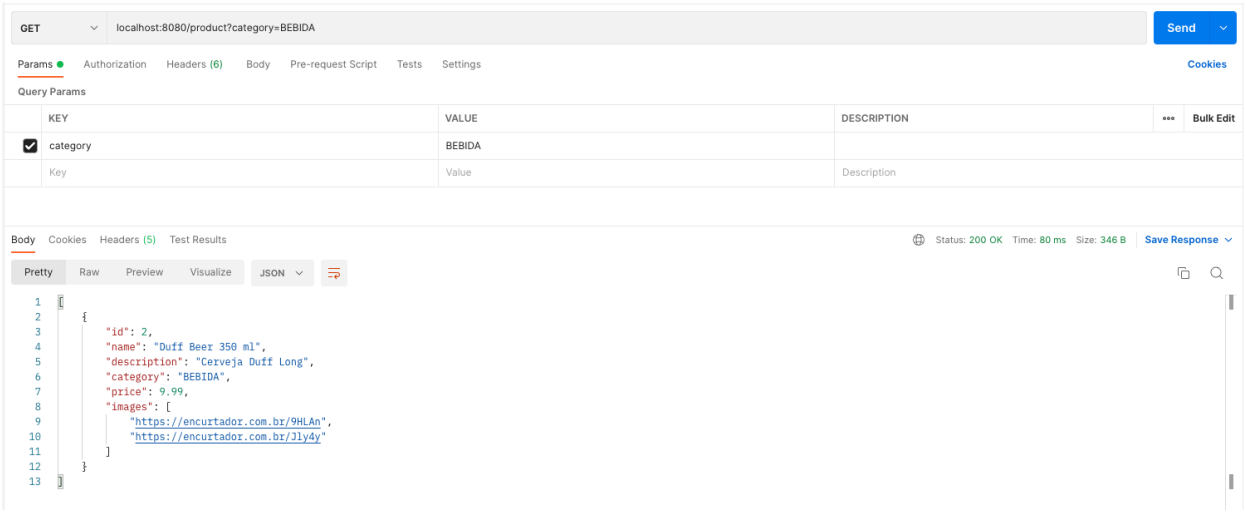
Produto para uma determinada categoria que não existente (Retorna 204 No Content)

curl --location --request GET 'localhost:8080/product?category=www'



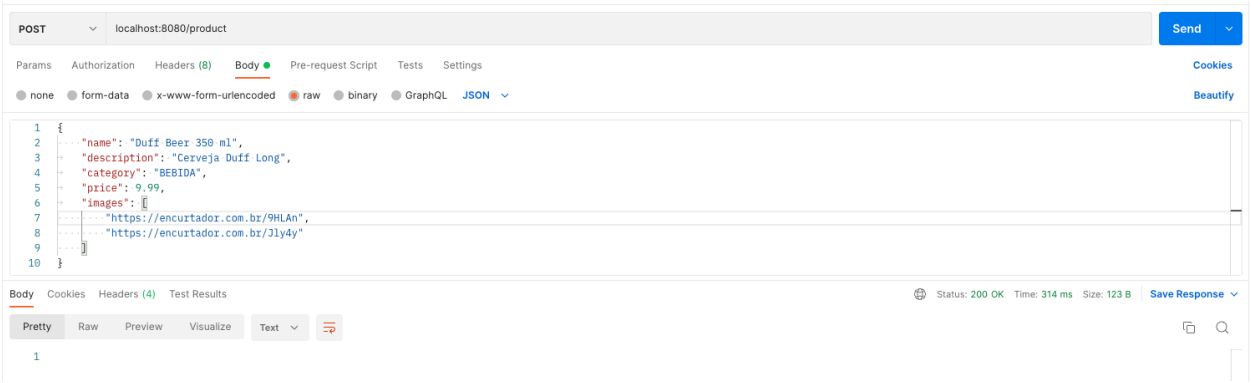
Produto para uma determinada categoria

curl --location --request GET 'localhost:8080/product?category=BEBIDA'



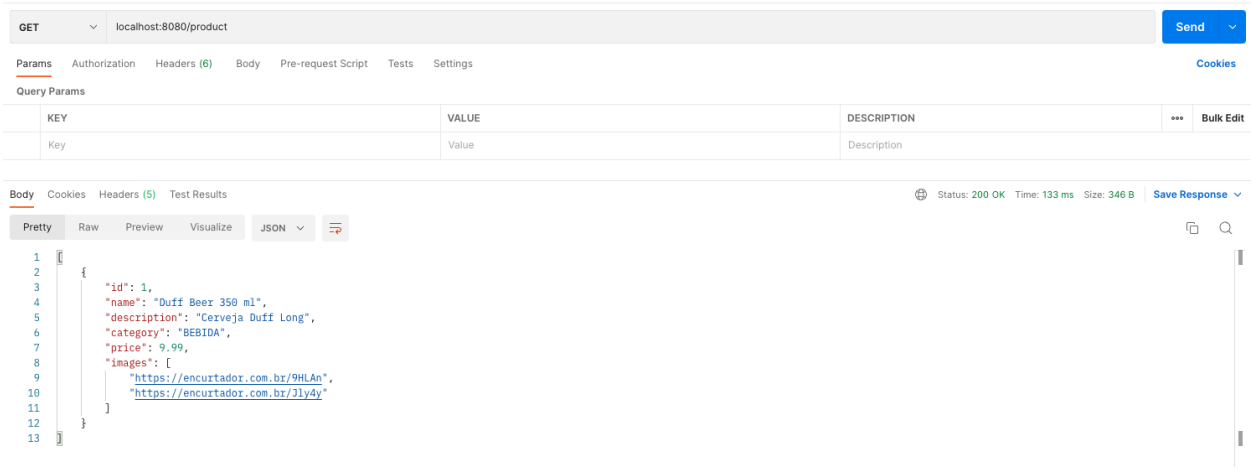
Cadastro de Produto

```
curl --location --request POST 'localhost:8080/product' \
--header 'Content-Type: application/json' \
--data-raw '{
   "name": "Hamburguer",
        "description": "Cerveja Duff Long",
        "category": "LANCHE",
        "price": 9.99,
        "images": [
      "https://encurtador.com.br/9HLAn",
      "https://encurtador.com.br/Jly4y"
   ]
}'
```



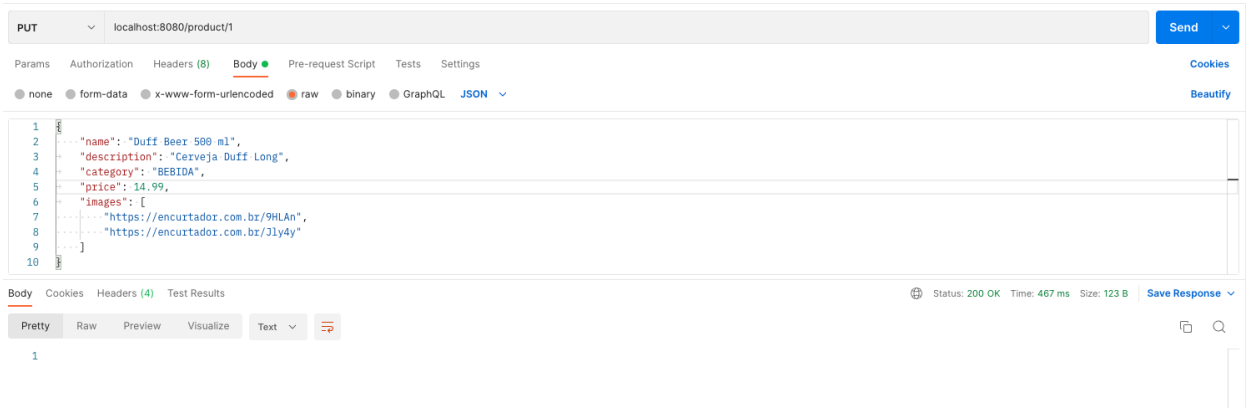
Lista de Produtos

```
curl --location --request GET 'localhost:8080/product?category=BEBIDA'
```



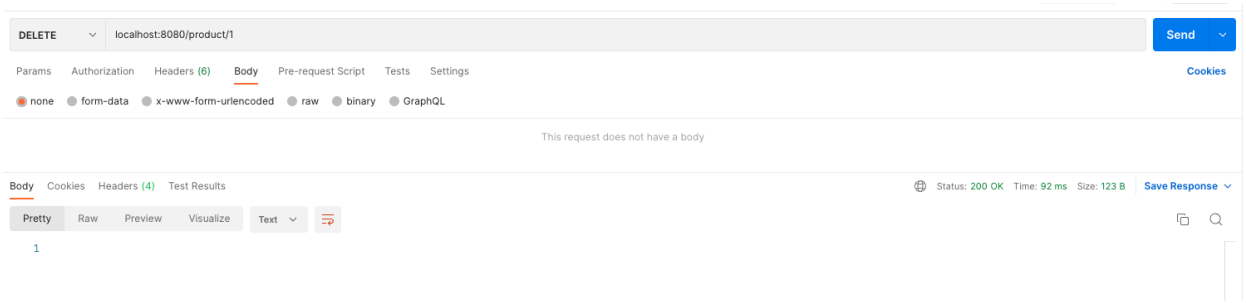
Edição de Produto

```
curl --location --request PUT 'localhost:8080/product/1' \
--header 'Content-Type: application/json' \
--data-raw '{
   "name": "Duff Beer 500 ml",
        "description": "Cerveja Duff Long",
        "category": "BEBIDA",
        "price": 14.99,
        "images": [
      "https://encurtador.com.br/9HLAn",
      "https://encurtador.com.br/Jly4y"
   ]
}'
```



Remoção de Produto

```
curl --location --request DELETE 'localhost:8080/product/1'
```

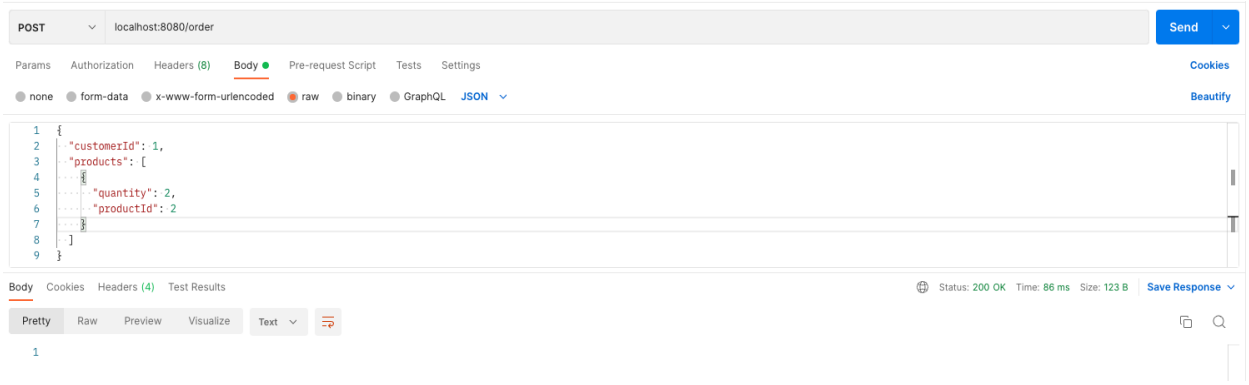


3.3. Contexto: Pedido

Nesse contexto, temos a manipulação de dados dos Pedidos (Cadastro, Alteração de status, Listagem Geral, Listagem de todos os status do Pedido, Checkout do Pedido e Fila de pedidos).

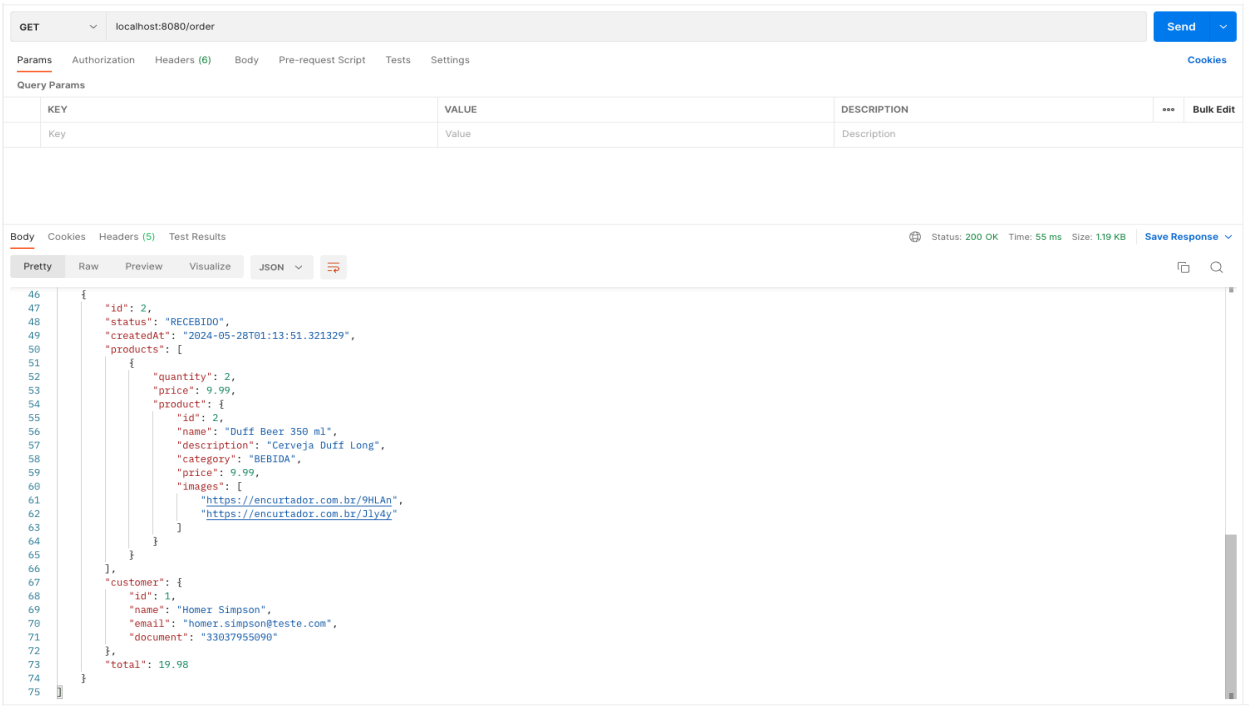
Realizando o Pedido

```
curl --location --request POST 'localhost:8080/order' \
--header 'Content-Type: application/json' \
--data-raw '{
  "products": [
    {
      "quantity": 10,
      "productId": 1
    }
  ]
}'
```



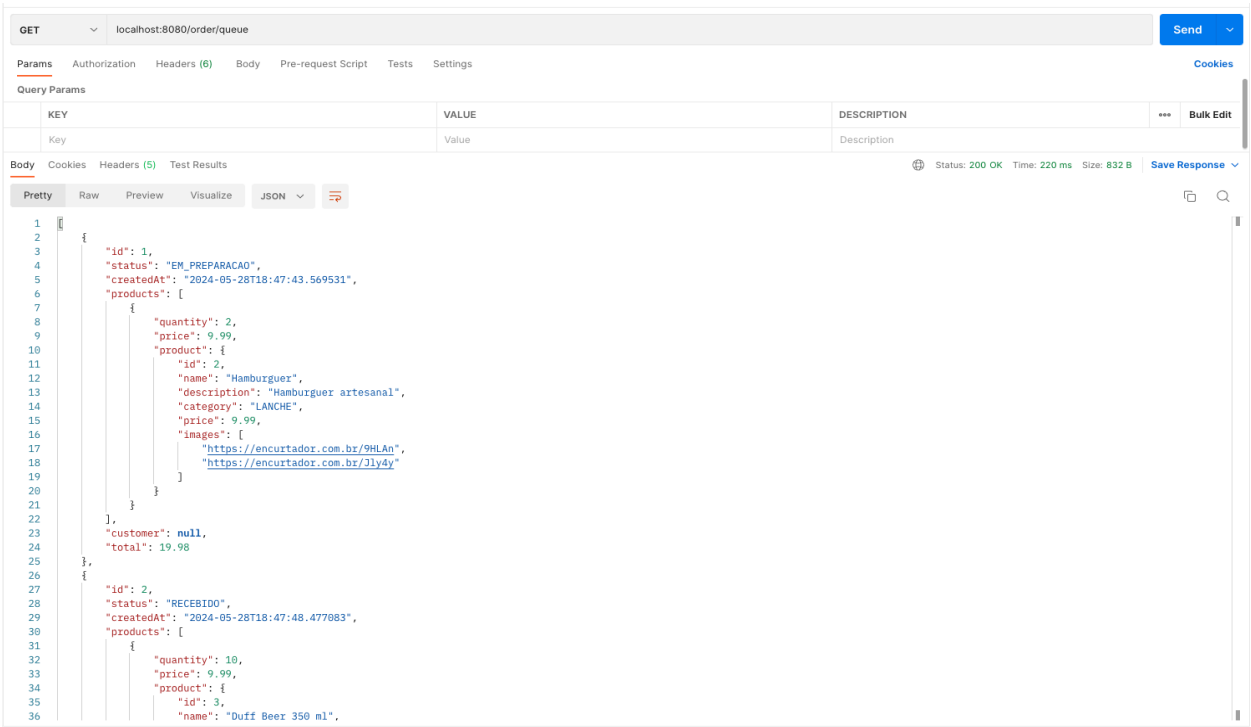
Listando os Pedidos

curl --location --request GET 'localhost:8080/order'



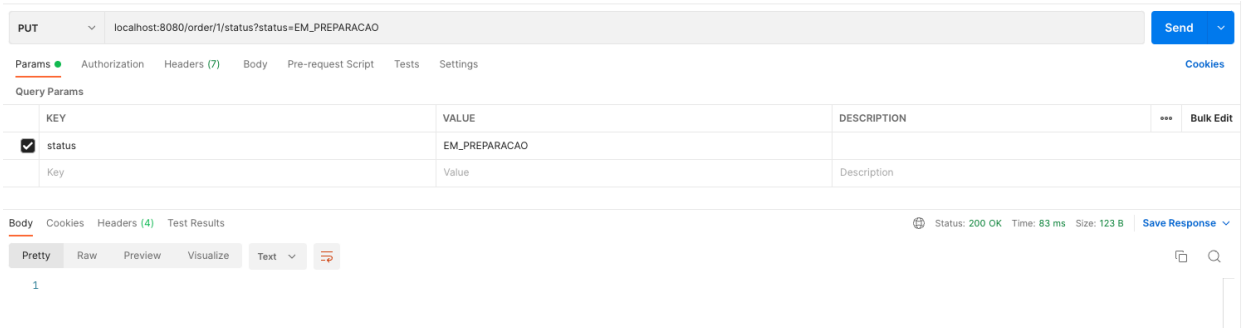
Lista de Pedidos Recebidos ou Em Preparação

curl --location --request GET 'localhost:8080/order/queue'



Alterar status do Pedido

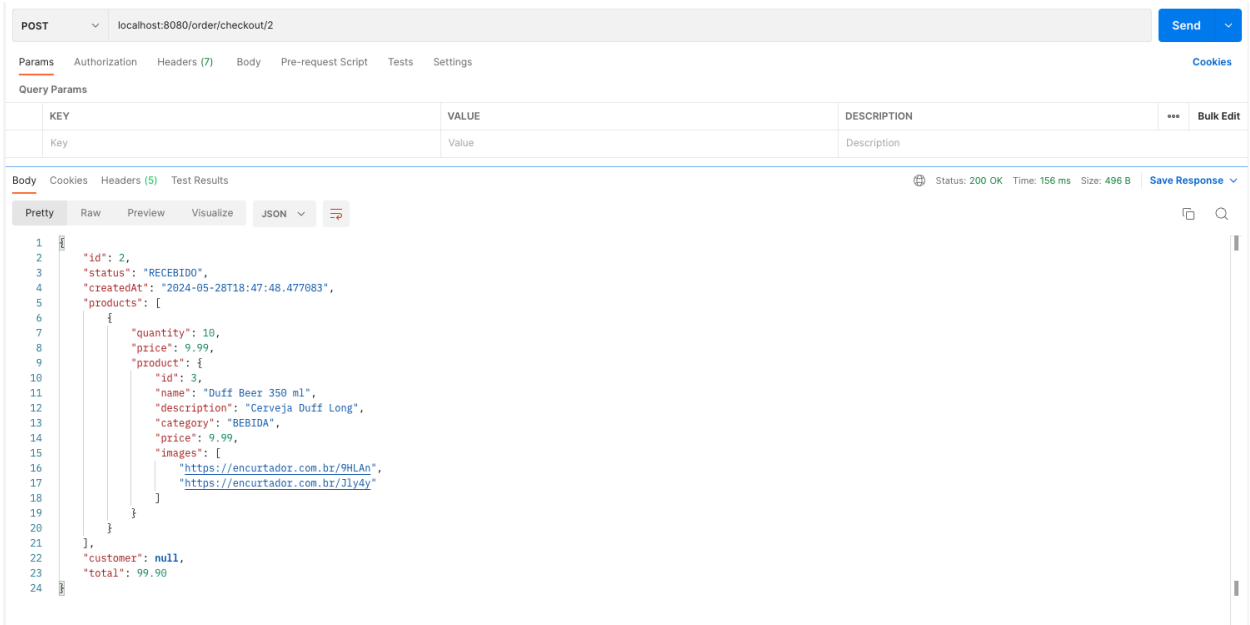
curl --location --request PUT 'localhost:8080/order/1/status?status=EM_PREPARACAO'



3.4. Contexto: Checkout

Realiza checkout do Pedido

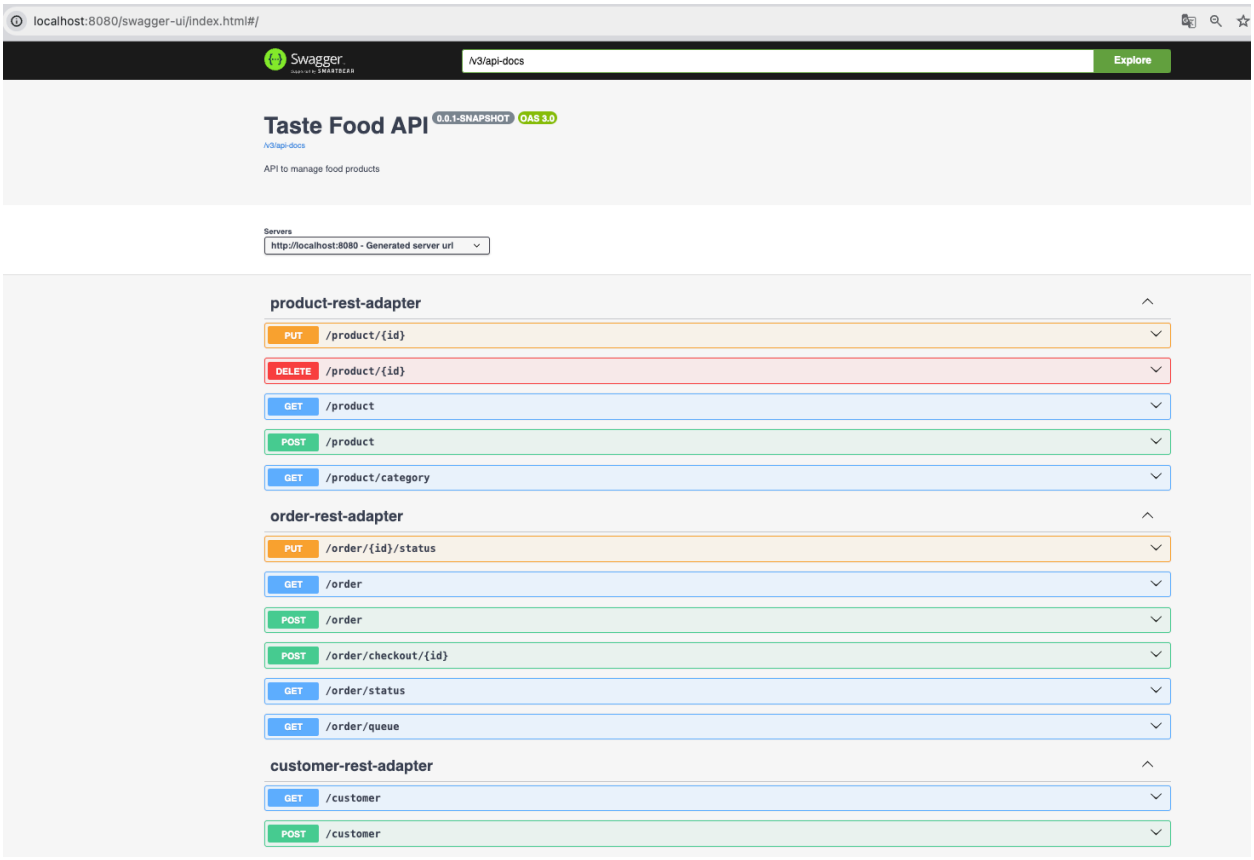
curl --location --request POST 'localhost:8080/order/checkout/2'



4. Complementares

4.1. Swagger

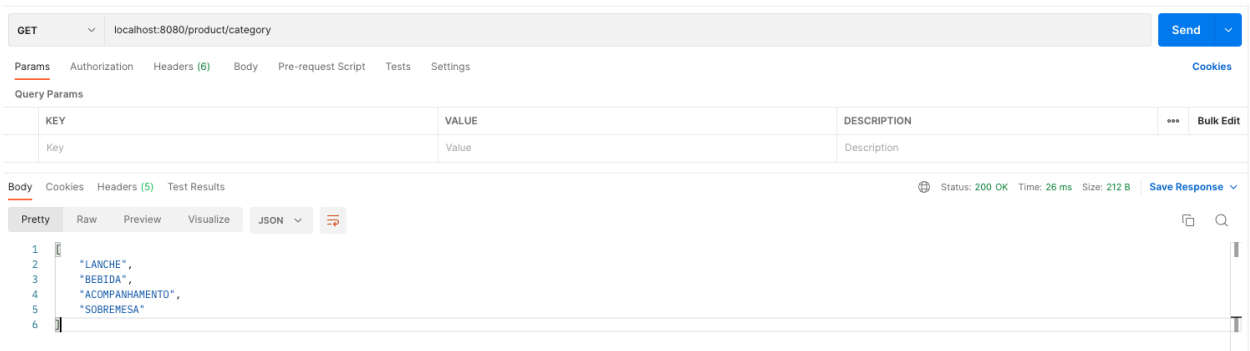
Possuímos o Swagger com uma interface clara sobre cada endpoint disponível: http://localhost:8080/swagger-ui/index.html#/



4.2. Endpoints Complementares

Buscar todas as categorias de produtos

curl --location --request GET 'localhost:8080/product/category'



Buscar todos os status possíveis de pedidos

curl --location --request GET 'localhost:8080/order/status'