

Tech Challenge Fase 01 – Taste Food

Grupo G13

Cristiano de Barros – RM355052 - Discord: cristiano.barros13

Graziela Goedert 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.

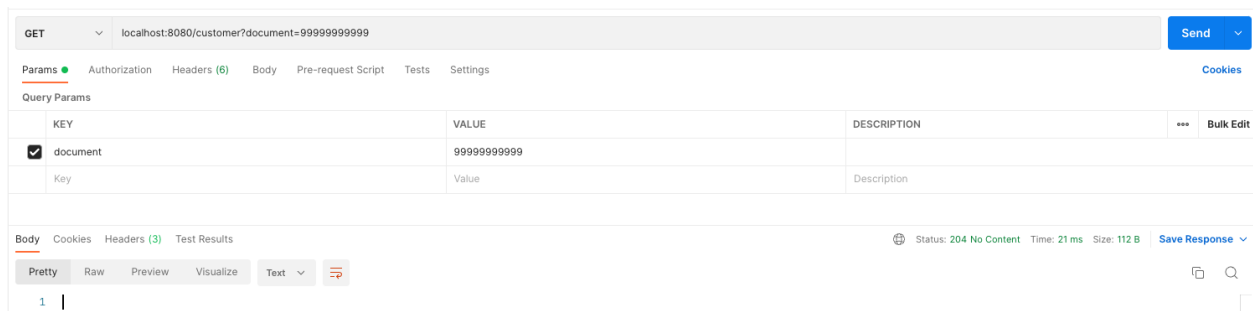
3. Contextos

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=999999999999'
```



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

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

OET localhost:8080/customer?document=33037955090 Send

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	***	Bulk Edit
<input checked="" type="checkbox"/> document	33037955090			
Key	Value	Description		

Body Cookies Headers (5) Test Results Status: 200 OK Time: 22 ms Size: 254 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 1,
3   "name": "Homer Simpson",
4   "email": "homer.simpson@teste.com",
5   "document": "33037955090"
6 }
```

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"
}'
```

POST localhost:8080/customer Send

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL JSON

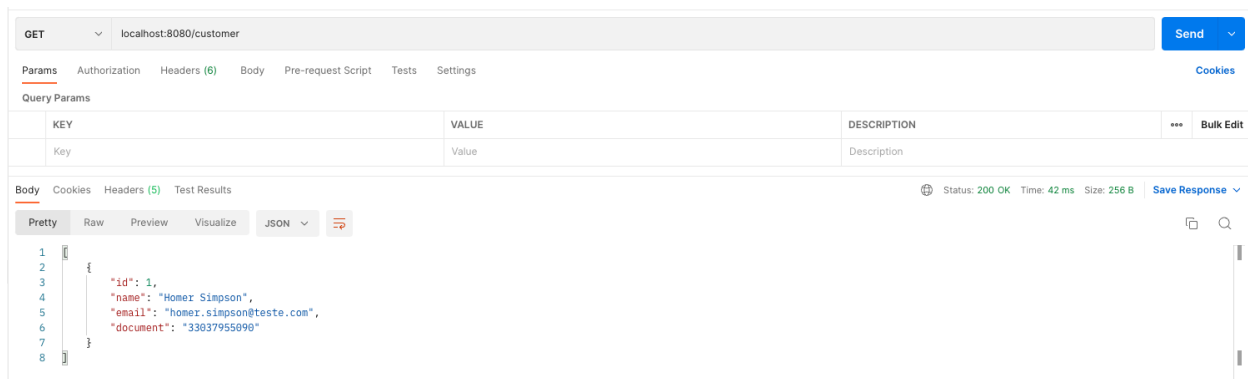
Body Cookies Headers (4) Test Results Status: 200 OK Time: 208 ms Size: 123 B Save Response

Pretty Raw Preview Visualize Text

```
1 {
2   "name": "Homer Simpson",
3   "email": "homer.simpson@teste.com",
4   "document": "33037955090"
5 }
```

Lista de Clientes

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

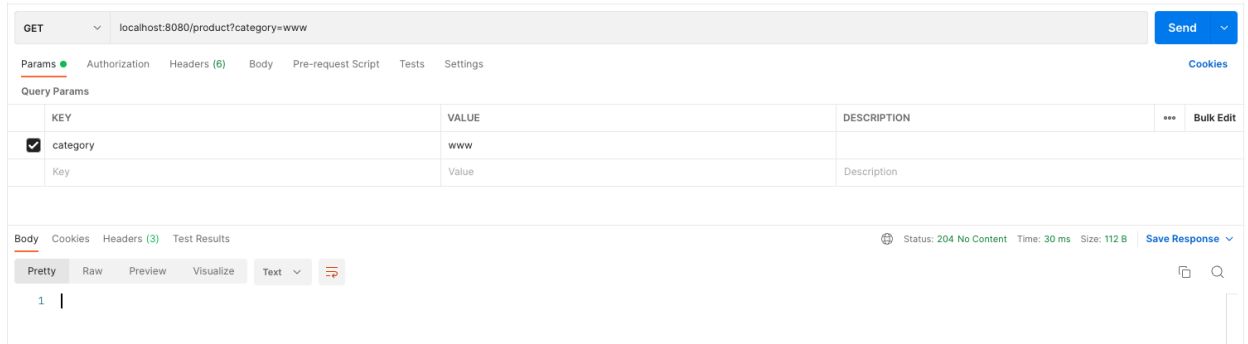


3.2. Contexto do Produto:

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 existe (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'

GET localhost:8080/product?category=BEBIDA

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION
category	BEBIDA	
Key	Value	Description

Body Cookies Headers (5) Test Results Status: 200 OK Time: 80 ms Size: 346 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   {
3     "id": 2,
4     "name": "Duff Beer 350 ml",
5     "description": "Cerveja Duff Long",
6     "category": "BEBIDA",
7     "price": 9.99,
8     "images": [
9       "https://encurtador.com.br/9HLAn",
10      "https://encurtador.com.br/Jly4y"
11    ]
12  }
13 }
```

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"
  ]
}'
```

POST localhost:8080/product

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL JSON

Body Cookies Headers (4) Test Results Status: 200 OK Time: 314 ms Size: 123 B Save Response

Pretty Raw Preview Visualize Text

```
1 {
2   {
3     "name": "Duff Beer 350 ml",
4     "description": "Cerveja Duff Long",
5     "category": "BEBIDA",
6     "price": 9.99,
7     "images": [
8       "https://encurtador.com.br/9HLAn",
9       "https://encurtador.com.br/Jly4y"
10    ]
11  }
12 }
```

Lista de Produtos

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

GET localhost:8080/product Send

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	***	Bulk Edit
Key	Value	Description		

Body Cookies Headers (5) Test Results Status: 200 OK Time: 133 ms Size: 346 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 1,
3   "name": "Duff Beer 350 ml",
4   "description": "Cerveja Duff Long",
5   "category": "BEBIDA",
6   "price": 9.99,
7   "images": [
8     "https://encurtador.com.br/9HLAn",
9     "https://encurtador.com.br/Jly4y"
10  ]
11 }
12
13
```

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"
  ]
}'
```

PUT localhost:8080/product/1 Send

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL JSON

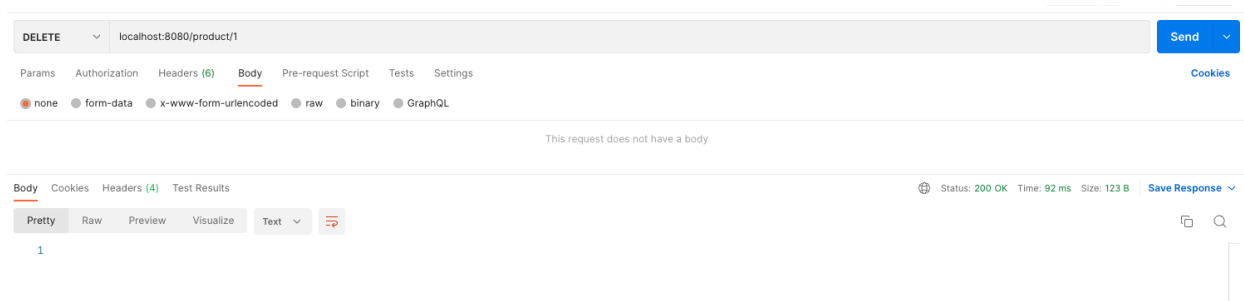
Body Cookies Headers (4) Test Results Status: 200 OK Time: 467 ms Size: 123 B Save Response

Pretty Raw Preview Visualize Text

```
1 {
2   "name": "Duff Beer 500 ml",
3   "description": "Cerveja Duff Long",
4   "category": "BEBIDA",
5   "price": 14.99,
6   "images": [
7     "https://encurtador.com.br/9HLAn",
8     "https://encurtador.com.br/Jly4y"
9   ]
10 }
```

Remoção de Produto

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

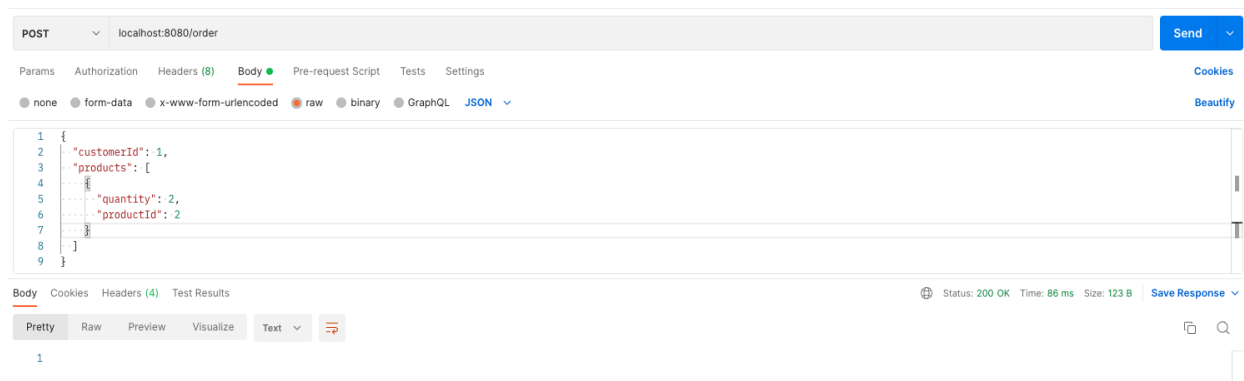



3.3. Contexto: Pedido

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'
```

GET localhost:8080/order

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body Cookies Headers (5) Test Results

Status: 200 OK Time: 55 ms Size: 119 KB Save Response

Pretty Raw Preview Visualize JSON

```
{
  "id": 2,
  "status": "RECEBIDO",
  "createdAt": "2024-05-28T01:13:51.321329",
  "products": [
    {
      "quantity": 2,
      "price": 9.99,
      "product": {
        "id": 2,
        "name": "Duff Beer 350 ml",
        "description": "Cerveja Duff Long",
        "category": "BEBIDA",
        "price": 9.99,
        "images": [
          "https://encurtador.com.br/9HLAn",
          "https://encurtador.com.br/3ly4y"
        ]
      }
    }
  ],
  "customer": {
    "id": 1,
    "name": "Homer Simpson",
    "email": "homer.simpson@teste.com",
    "document": "33037955990"
  },
  "total": 19.98
}
```

Lista de Pedidos Recebidos ou Em Preparação

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

GET localhost:8080/order/queue

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body Cookies Headers (5) Test Results

Status: 200 OK Time: 220 ms Size: 832 B Save Response

Pretty Raw Preview Visualize JSON

```
{
  "id": 1,
  "status": "EM_PREPARACAO",
  "createdAt": "2024-05-28T18:47:43.569531",
  "products": [
    {
      "quantity": 2,
      "price": 9.99,
      "product": {
        "id": 2,
        "name": "Hamburguer",
        "description": "Hamburguer artesanal",
        "category": "LANCHE",
        "price": 9.99,
        "images": [
          "https://encurtador.com.br/9HLAn",
          "https://encurtador.com.br/3ly4y"
        ]
      }
    }
  ],
  "customer": null,
  "total": 19.98
},
{
  "id": 2,
  "status": "RECEBIDO",
  "createdAt": "2024-05-28T18:47:48.477083",
  "products": [
    {
      "quantity": 10,
      "price": 9.99,
      "product": {
        "id": 3,
        "name": "Duff Beer 350 ml",

```

Alterar status do Pedido

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

PUT localhost:8080/order/1/status?status=EM_PREPARACAO Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/> status	EM_PREPARACAO			
Key	Value	Description		

Body Cookies Headers (4) Test Results Status: 200 OK Time: 83 ms Size: 123 B Save Response

Pretty Raw Preview Visualize Text

1

3.4. Contexto: Checkout

Realiza checkout do Pedido

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

POST localhost:8080/order/checkout/2 Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (5) Test Results Status: 200 OK Time: 156 ms Size: 496 B Save Response

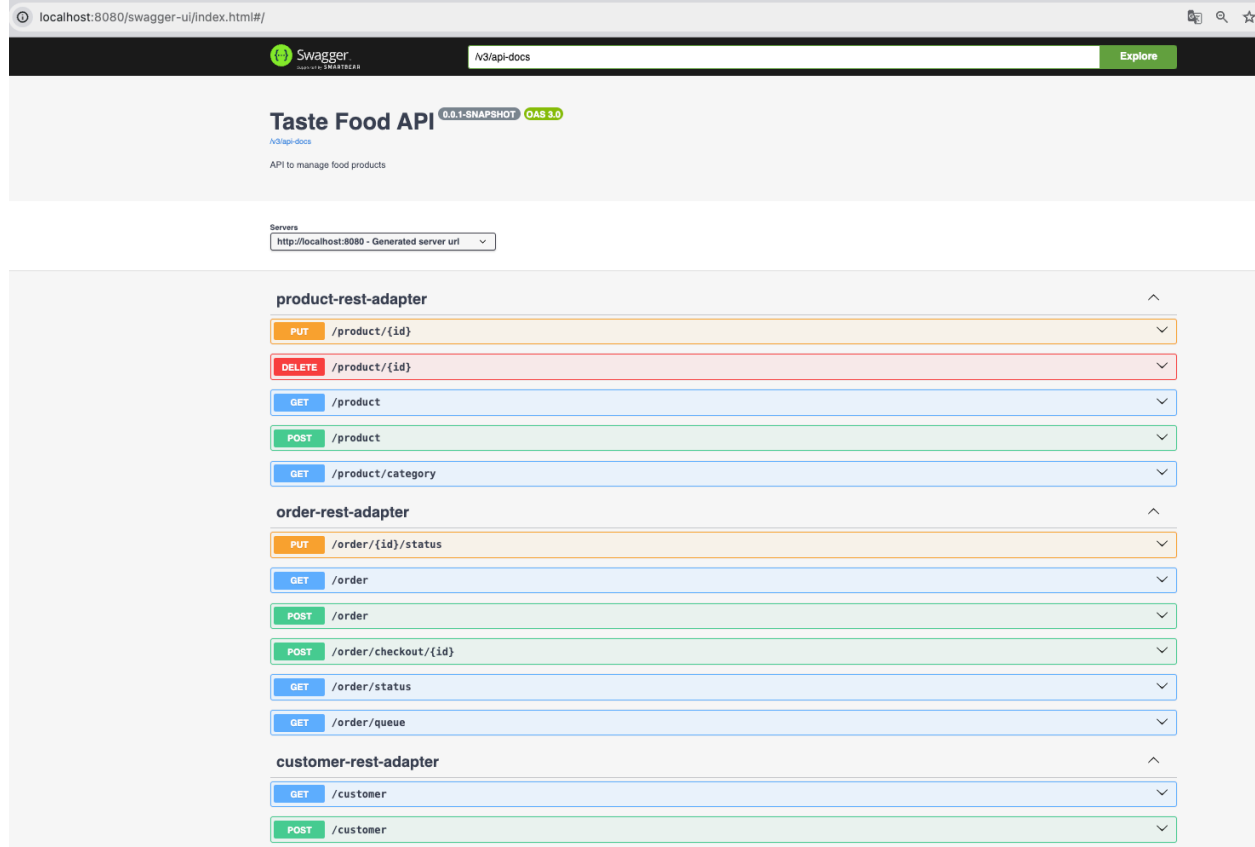
Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 2,
3   "status": "RECEBIDO",
4   "createdAt": "2024-05-28T18:47:48.477883",
5   "products": [
6     {
7       "quantity": 10,
8       "price": 9.99,
9       "product": {
10        "id": 3,
11        "name": "Duff Beer 350 ml",
12        "description": "Cerveja Duff Long",
13        "category": "BEBIDA",
14        "price": 9.99,
15        "images": [
16          "https://encurtador.com.br/9HLAn",
17          "https://encurtador.com.br/3ly4y"
18        ]
19      }
20    }
21  ],
22   "customer": null,
23   "total": 99.90
24 }
```

4. Complementares

4.1. Swagger

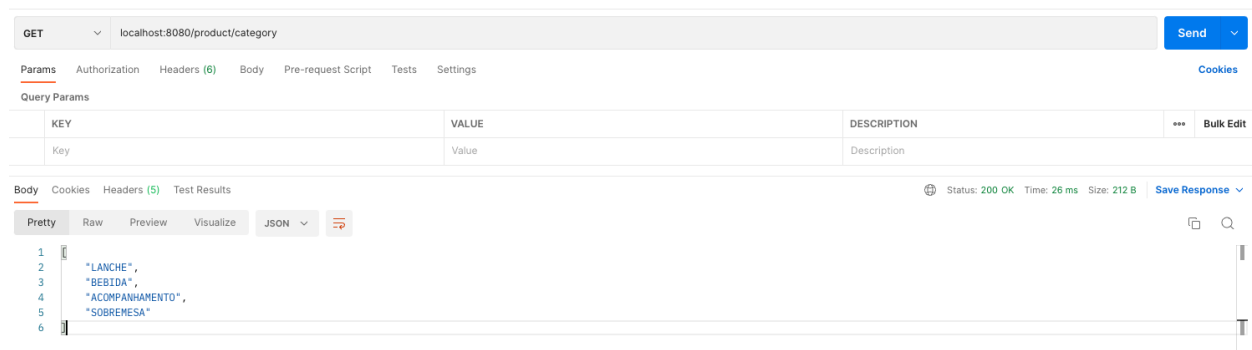
Possuímos o Swagger com uma interface clara sobre cada endpoint disponível: <http://localhost:8080/swagger-ui/index.html#/>



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'

GETlocalhost:8080/order/statusSend

ParamsAuthorizationHeaders (6)BodyPre-request ScriptTestsSettings

Cookies

Query Params

KEY	VALUE	DESCRIPTION		Bulk Edit
Key	Value	Description		

BodyCookiesHeaders (5)Test Results

Status: 200 OKTime: 105 msSize: 235 BSave Response

PrettyRawPreviewVisualizeJSON

1

2

3

4

5

6

7

8

"CRIADO",

"RECEBIDO",

"EM_PREPARACAO",

"PRONTO",

"FINALIZADO",

"CANCELADO"