

# Curso Java backend

## Módulo III – Testando a API - Controle de pedidos

### Sumário

Informações Gerais .....	2
Produtos .....	2
Exemplos de JSON para POST E PUT .....	2
POST – Criar um novo produto .....	3
GET – Obter um produto por ID .....	4
GET – Obter todos os produtos .....	5
PUT – Atualizar um produto existente .....	6
DELETE – Excluir um produto .....	7
Pedidos (Order) .....	8
Exemplos de JSON para POST E PUT .....	8
POST – Criar um novo pedido.....	9
GET – Obter pedido pelo ID .....	10
GET – Obter todos os pedidos .....	11
PUT – Atualizar um pedido existente .....	12
DELETE – Excluir um pedido.....	13

# Informações Gerais

Para executar todos os testes na sequência correta, inicie pela criação de produtos, uma vez que para criar um pedido, é obrigatório informar um ou mais produtos nos itens do pedido.

Os testes de exemplos relacionados abaixo, seguem somente o caminho de sucesso, caso você informe valores inválidos, poderá observar mensagens de resposta com erros variados, alguns foram tratados, mas nem todos.

## Produtos

Produtos <small>Contém todas as operações relativas ao cadastro de produtos</small>			^
GET	/api/v1/product/{id}	Obter um produto pelo ID	✓
PUT	/api/v1/product/{id}	Atualizar um produto existente	✓
DELETE	/api/v1/product/{id}	Excluir um produto	✓
GET	/api/v1/product	Obter todos os produtos	✓
POST	/api/v1/product	Criar um novo produto	✓

## Exemplos de JSON para POST E PUT

```
{
  "name": "Geladeira Brastemp",
  "price": 3150
}
```

```
{
  "name": "Geladeira Brastemp",
  "price": 3000
}
```

# POST – Criar um novo produto

- A. Clique em Try it out
- B. Informe os dados do produto
- C. Clique em Execute
- D. Observe as possíveis respostas (Response)

POST /api/v1/product Criar um novo produto

Cria um novo produto

Parameters

Cancel

Reset

No parameters

Request body required

application/json

```
{  "name": "Geladeira Brastemp 510 litros",  "price": 3250}
```

Execute

Server response

Code

Details

201

Response body

```
{  "id": 1,  "name": "Geladeira Brastemp 510 litros",  "price": 3250}
```

Download

Response headers

```
connection: keep-alivecontent-type: application/jsondate: Wed, 19 Jun 2024 21:18:07 GMTkeep-alive: timeout=60transfer-encoding: chunked
```

Responses

Code

Description

Links

201

Produto criado com sucesso

No links

Media type

application/json

Controls Accept header.

Example Value | Schema

```
{  "id": 0,  "name": "string",  "price": 0}
```

400

Dados de entrada inválidos

No links

## GET – Obter um produto por ID

- A. Clique em Try it out
- B. Informe os dados do produto
- C. Clique em Execute
- D. Observe as possíveis respostas (Response)

GET

/api/v1/product/{id} Obter um produto pelo ID

Retorna um produto específico pelo ID

Parameters

Cancel

Name	Description
<b>id</b> <small>* required</small> integer(\$int64) (path)	<input type="text" value="1"/>

Execute

Responses

Code	Description	Links
200	Produto encontrado	No links
<div>Media type <div>application/json</div><div>Controls Accept header.</div><div>Example Value   Schema</div><pre>{   "id": 0,   "name": "string",   "price": 0 }</pre></div>		
404	Produto não encontrado	No links

# GET – Obter todos os produtos

- A. Clique em Try it out
- B. Clique em Execute
- C. Observe as possíveis respostas (Response)

GET /api/v1/product Obter todos os produtos

Retorna uma lista de todos os produtos

Parameters

No parameters

Cancel

Execute

Server response

Code	Details
200	<div>Response body<pre>{  "id": 1,  "name": "Geladeira Brastemp 510 litros",  "price": 3250}</pre></div> <div>Response headers<pre>connection: keep-alivecontent-type: application/jsondate: Wed,19 Jun 2024 21:22:27 GMTkeep-alive: timeout=60transfer-encoding: chunked</pre></div>

Responses

Code	Description	Links
200	Produtos encontrados	No links
<div>Media type<div>application/json</div></div> <div>Controls Accept header.</div> <div>Example Value   Schema<pre>{  "id": 0,  "name": "string",  "price": 0}</pre></div>		
500	Erro interno do servidor	No links

## PUT – Atualizar um produto existente

- Clique em Try it out
- Informe os dados do produto
- Clique em Execute
- Observe as possíveis respostas (Response)

PUT /api/v1/product/{id} Atualizar um produto existente

Atualiza os dados de um produto existente

Parameters

Name	Description
id <small>required</small>	<input type="text" value="1"/> <small>integer(\$int64) (path)</small>

Cancel Reset

Request body required

application/json

```
{  "name": "Geladeira Brastemp 510 litros",  "price": 3000}
```

Execute

Server response

Code	Details
200	<div>Response body<pre>{  "id": 1,  "name": "Geladeira Brastemp 510 litros",  "price": 3000}</pre><div>Download</div></div> <div>Response headers<pre>connection: keep-alivecontent-type: application/jsondate: Wed, 19 Jun 2024 21:25:31 GMTkeep-alive: timeout=60transfer-encoding: chunked</pre></div>

Responses

Code	Description	Links
200	Produto atualizado com sucesso	No links
400	Dados de entrada inválidos	No links
404	Produto não encontrado	No links

6

Autor: Ubirajara Portela Campos

# DELETE – Excluir um produto

- A. Clique em Try it out
- B. Informe o Id do produto
- C. Clique em Execute
- D. Observe as possíveis respostas (Response)

DELETE

/api/v1/product/{id}

Excluir um produto

Excluir um produto pelo ID

Parameters

Name	Description
id * required	
integer(\$int64)	
(path)	

4

Execute

Server response

Code	Details
204	<div>Response headers</div> <div>connection: keep-alive date: Wed, 19 Jun 2024 21:29:59 GMT keep-alive: timeout=60</div>

Responses

Code	Description	Links
204	Produto excluído com sucesso	No links
404	Produto não encontrado	No links

# Pedidos (Order)

Pedidos (Order) <small>Contém todas as operações relativas ao cadastro de pedidos (Orders)</small>			^
GET	/api/v1/order/{id}	Obter um pedido pelo ID	✓
PUT	/api/v1/order/{id}	Atualizar um pedido existente	✓
DELETE	/api/v1/order/{id}	Excluir um pedido	✓
GET	/api/v1/order	Obter todos os pedidos	✓
POST	/api/v1/order	Criar um novo pedido	✓

## Exemplos de JSON para POST E PUT

```
{
  "customerName": "Pedro da Silva",
  "items": [
    {
      "product": {
        "id": 1
      },
      "quantity": 2
    }
  ]
}
```

```
{
  "customerName": "Pedro da Silva",
  "items": [
    {
      "product": {
        "id": 1
      },
      "quantity": 3
    }
  ]
}
```



# POST – Criar um novo pedido

- A. Clique em Try it out
- B. Informe os dados do pedido
- C. Clique em Execute
- D. Observe as possíveis respostas (Response)

POST

/api/v1/order

Criar um novo pedido

Cria um novo pedido

Parameters

No parameters

Cancel

Reset

Request body

required

application/json

```
{  "customerName": "Pedro da Silva",  "items": [    {      "product": {        "id": 1      },      "quantity": 2    },    {      "product": {        "id": 2      },      "quantity": 1    }  ]}
```

Execute

Server response

Code

Details

201

Response body

```
{  "orderId": 2,  "customerName": "Pedro da Silva",  "items": [    {      "itemId": 2,      "productId": 1,      "quantity": 2    },    {      "itemId": 2,      "productId": 2,      "quantity": 1    }  ]}
```

Download

Response headers

```
connection: keep-alive
content-type: application/json
date: Wed, 19 Jun 2024 22:07:31 GMT
keep-alive: timeout=60
location: http://localhost:8080/api/v1/order/2
transfer-encoding: chunked
```

Responses

Code	Description	Links
201	Pedido criado com sucesso	No links

Media type

application/json

Controls Accept header

Example Value

Schema

400

Dados de entrada inválidos

No links

# GET – Obter pedido pelo ID

- A. Clique em Try it out
- B. Informe o ID do pedido
- C. Clique em Execute
- D. Observe as possíveis respostas (Response)

GET /api/v1/order/{id} Obter um pedido pelo ID

Retorna um pedido específico pelo ID

Parameters

Name	Description
id <small>* required</small>	
integer(\$int64)	
(path)	

Execute

Server response

Code	Details									
200	<div><div>Response body</div><div><pre>{  "orderId": 2,  "customerName": "Pedro da Silva",  "items": [    {      "itemId": 1,      "productid": 1,      "quantity": 2    },    {      "itemId": 4,      "productid": 2,      "quantity": 1    }  ]}</pre></div><div>Download</div></div>									
<div><div>Response headers</div><div><pre>connection: keep-alive content-type: application/json date: Wed, 19 Jun 2024 22:33:27 GMT keep-alive: timeout=60 transfer-encoding: chunked</pre></div></div>										
<div><div>Responses</div><table><thead><tr><th>Code</th><th>Description</th><th>Links</th></tr></thead><tbody><tr><td>200</td><td>Pedido encontrado</td><td>No links</td></tr><tr><td>404</td><td>Pedido não encontrado</td><td>No links</td></tr></tbody></table></div>		Code	Description	Links	200	Pedido encontrado	No links	404	Pedido não encontrado	No links
Code	Description	Links								
200	Pedido encontrado	No links								
404	Pedido não encontrado	No links								

10  
Autor: Ubirajara Portela Campos

# GET – Obter todos os pedidos

- A. Clique em Try it out
- B. Clique em Execute
- C. Observe as possíveis respostas (Response)

GET /api/v1/order Obter todos os pedidos

Retorna uma lista de todos os pedidos

Parameters

No parameters

Cancel

Execute

Server response

Code

Details

200

Response body

```
{
  "orderId": 1,
  "customerName": "Pedro da Silva",
  "items": [
    {
      "itemId": 1,
      "productId": 1,
      "quantity": 2
    },
    {
      "itemId": 4,
      "productId": 2,
      "quantity": 1
    }
  ]
}
```

Download

Response headers

```
connection: keep-alive
content-type: application/json
date: Wed, 19 Jun 2024 22:36:25 GMT
keep-alive: timeout=60
transfer-encoding: chunked
```

Responses

Code	Description	Links
200	Pedidos encontrados	No links
500	Erro interno do servidor	No links

## PUT – Atualizar um pedido existente

- A. Clique em Try it out
- B. Informe os dados do produto
- C. Clique em Execute
- D. Observe as possíveis respostas (Response)

PUT

/api/v1/order/{id} Atualizar um pedido existente

Atualiza os dados de um pedido existente

Parameters

Cancel

Reset

Name	Description
id <small>* required</small>	<input type="text" value="2"/>
<small>integer(\$int64)</small>	<small>(path)</small>

Request body required

application/json

```
{  "customerName": "Pedro da Silva",  "items": [    {      "product":      {        "id": 1      },      "quantity": 5    },    {      "product":      {        "id": 2      },      "quantity": 3    }  ]}
```

Execute

Server response

Code	Details
200	<div><div>Response body</div><div><pre>{  "orderId": 2,  "customerName": "Pedro da Silva",  "items": [    {      "itemId": 3,      "productid": 1,      "quantity": 5    },    {      "itemId": 4,      "productid": 2,      "quantity": 3    }  ]}</pre></div><div><div>Download</div></div></div> <div><div>Response headers</div><div><pre>connection: keep-alive content-type: application/json date: Wed, 10 Jun 2020 22:41:54 GMT keep-alive: timeout=60 transfer-encoding: chunked</pre></div></div>

Responses

Code	Description	Links
200	Pedido atualizado com sucesso	No links
400	Dados de entrada inválidos	No links
404	Pedido não encontrado	No links

# DELETE – Excluir um pedido

- A. Clique em Try it out
- B. Informe o Id do produto
- C. Clique em Execute
- D. Observe as possíveis respostas (Response)

**DELETE** /api/v1/order/{id} Excluir um pedido

Excluir um pedido pelo ID

Parameters

Name	Description
<b>id</b> <small>required</small>	<input type="text" value="2"/>
<small>integer(\$int64)</small>	
<small>(path)</small>	

Execute

Server response

Code	Details	
204	<div>Response headers<div>connection: keep-alive date: Wed, 19 Jun 2024 22:46:05 GMT keep-alive: timeout=60</div></div>	
Responses		
Code	Description	Links
204	Pedido excluído com sucesso	No links
404	Pedido não encontrado	No links