

Análise de Arquitetura de Microsserviços Utilizando CQRS: Desenvolvimento de um Sistema de E-commerce como Exemplo Prático

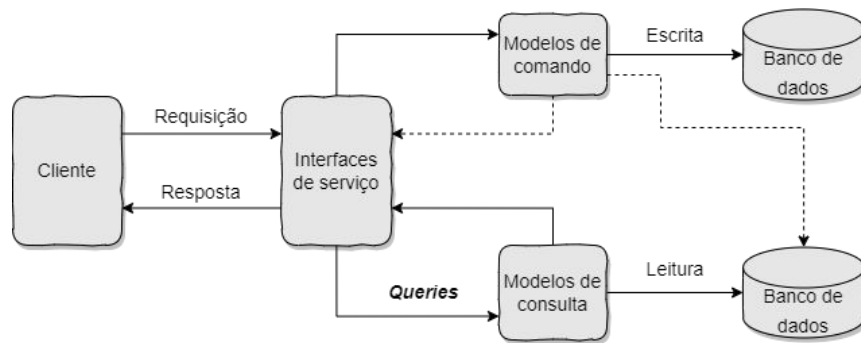
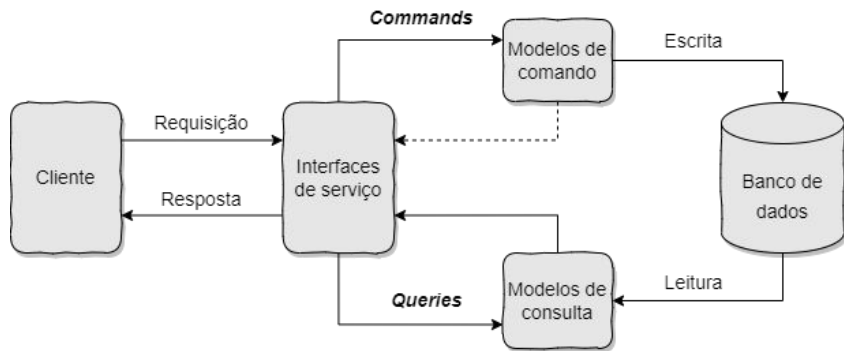
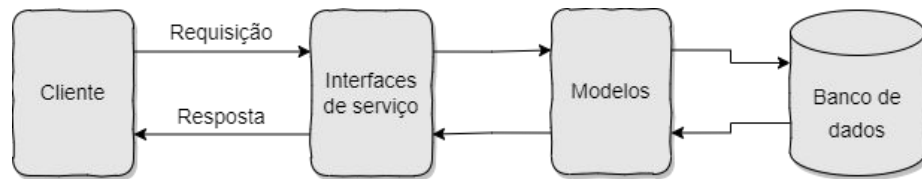
Gabrielle Araújo de Souza

Contextualização - Arquitetura de Software

- Bass, Clements e Kazman (2003) definem a arquitetura de software como a estrutura, ou estruturas do sistema, a qual compreende elementos de software, as propriedades visíveis externamente desses elementos, e os relacionamentos entre os elementos.
- Padrões de projeto e **padrões arquiteturais**.
 - **CQRS** (*Command Query Responsibility Segregation* - Segregação de Responsabilidade de Comando e Consulta)
- Estrutura monolítica, estrutura de **microsserviços**

Contextualização - Arquitetura de Software

- Sem CQRS.
- Com CQRS, banco de dados único.
- Com CQRS, dois bancos de dados.



Problema

- Crescente complexidade dos sistemas de software
- Busca por maior eficiência e agilidade na manutenção
- Impactos negativos tanto para o desenvolvimento do software quanto para o ambiente de trabalho dos desenvolvedores.
- Prejuízo tanto para as empresas quanto para os desenvolvedores

Objetivo

- Compreender como a arquitetura de sistemas de software pode influenciar positivamente na qualidade.
 - Facilidade de manutenção do código-fonte
 - Eficiência de desempenho do sistema.
- Fornecer informações, insights e recomendações para desenvolvedores de software, acerca da qualidade e arquitetura de software.
 - Contribuir para a criação de sistemas mais confiáveis, fáceis de manter e com melhor desempenho.

Objetivo

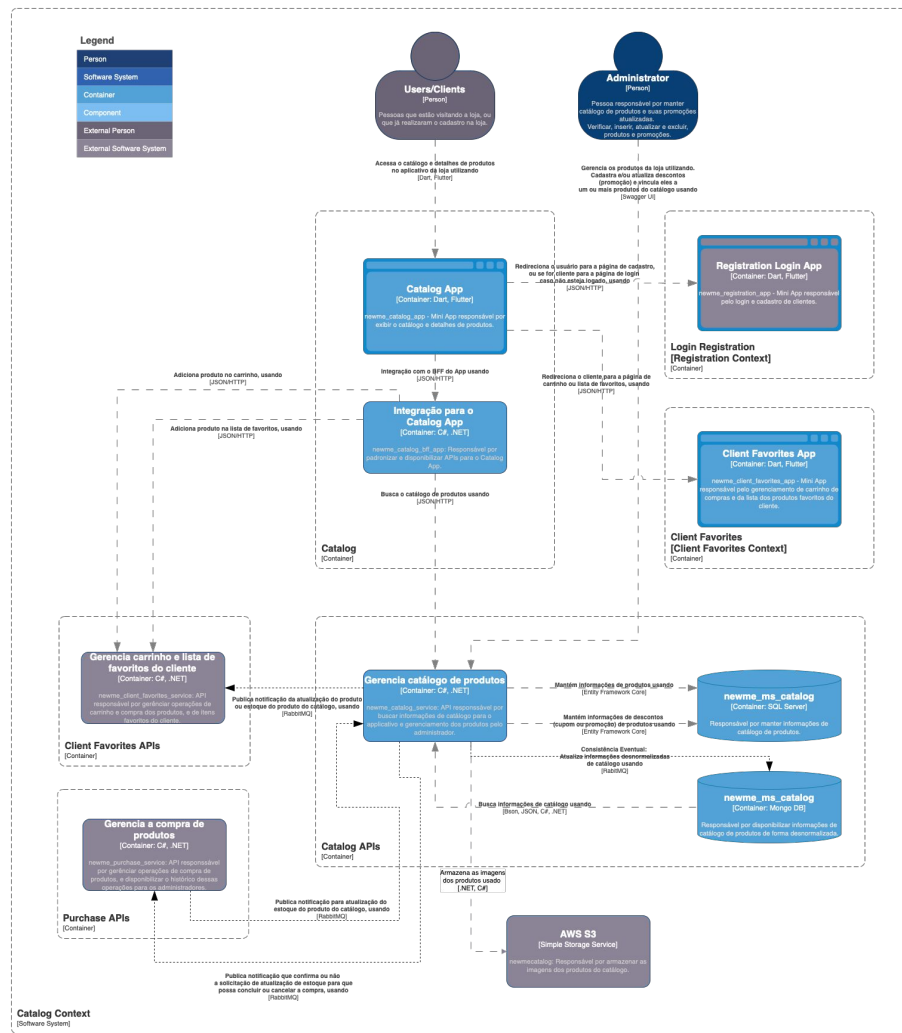
- Pesquisa bibliográfica a partir de fontes em engenharia de software, qualidade e arquitetura de software.
- Desenvolvimento de partes de um sistema de e-commerce utilizando a arquitetura de microsserviços e o padrão arquitetural CQRS.
 - Apresentar um exemplo prático e aplicável de alguns conceitos apresentados ao decorrer do trabalho.

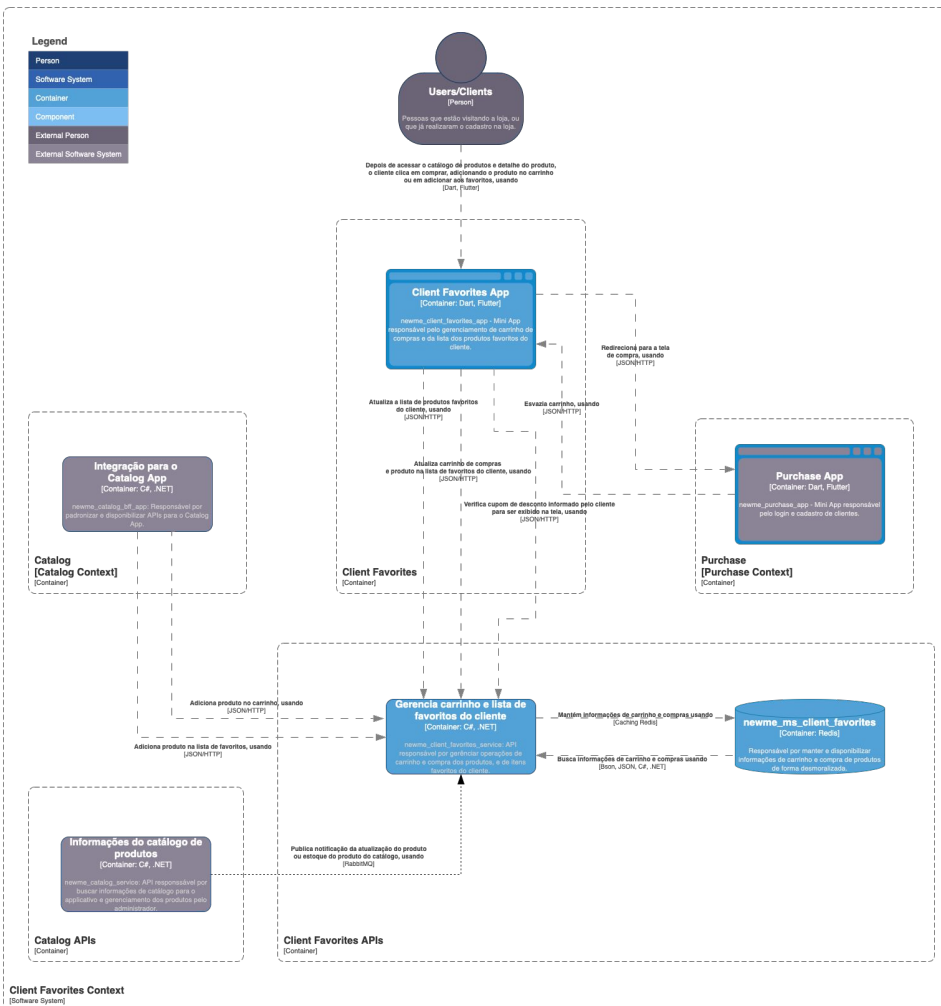
Exemplos de Requisitos Funcionais

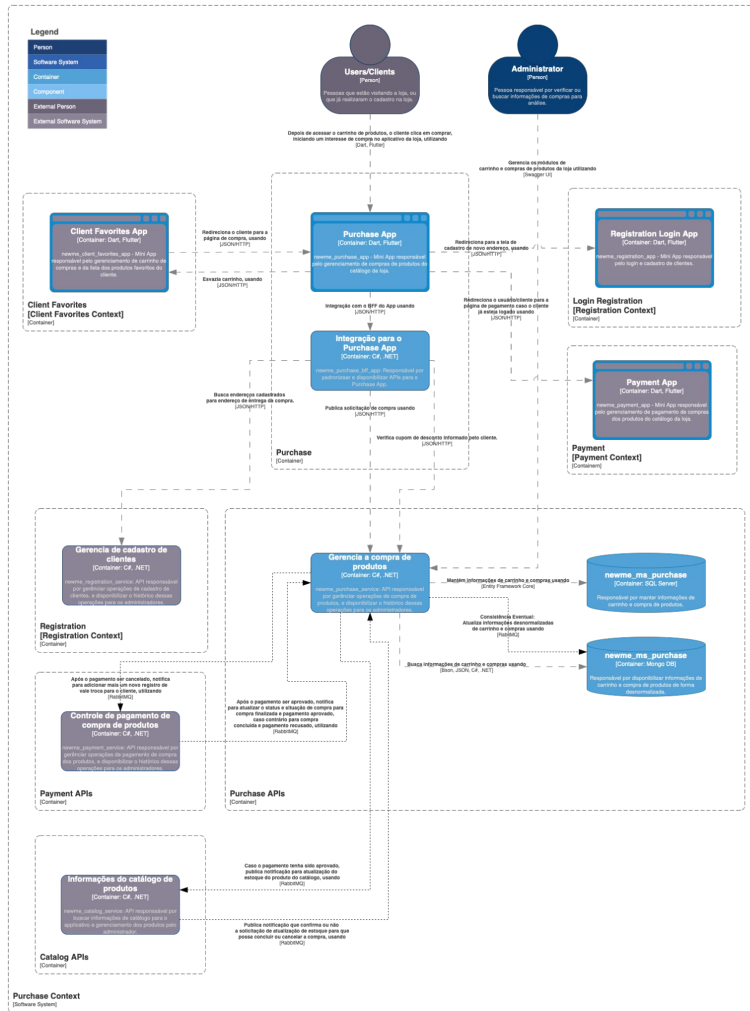
| ID | Nome | Como demonstrar |
|-----|-------------------|--|
| F01 | Cadastrar produto | Como administrador quero conseguir cadastrar produtos no sistema de modo que os clientes possam visualizar no aplicativo. |
| F02 | Buscar produto | Como administrador quero conseguir buscar um produto específico no sistema de modo que possa visualizá-los ou realizar operações. |

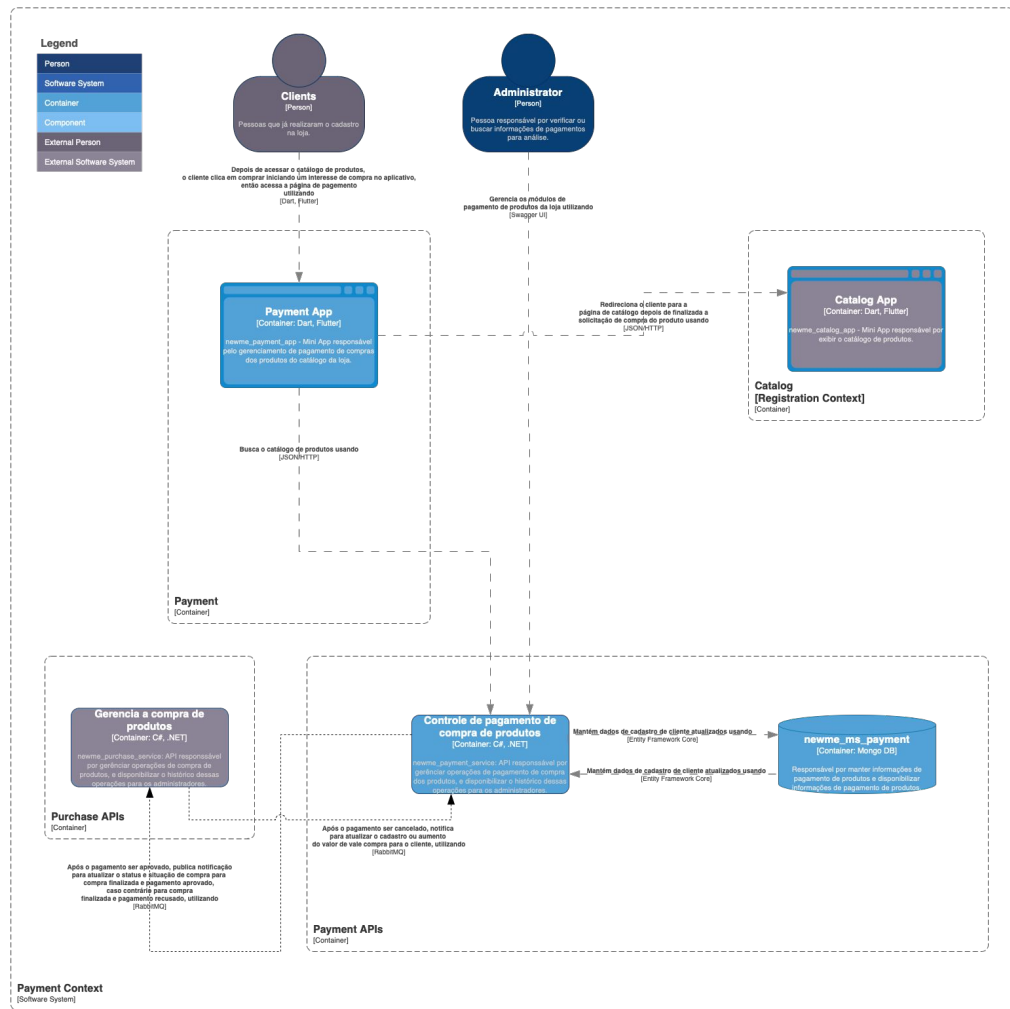
| | | |
|-----|-------------------------------|--|
| C11 | Realizar compra de produtos | Como cliente quero conseguir realizar compras no sistema e escolher se quero pagar com cartão de crédito, cartão de débito, ou pix de modo que possa adquirir os produtos de meu interesse da forma que melhor me atenda. |
| C13 | Visualizar compras realizadas | Como cliente quero conseguir visualizar as compras realizadas por mim no sistema de modo que possa acompanhar meus gastos na loja. |
| C14 | Filtrar produtos | Como usuário quero conseguir aplicar filtros de busca de produtos por categoria, gênero, tamanho e/ou cor no sistema de modo que possa encontrar o que busco com mais facilidade. |

Diagramas de Arquitetura

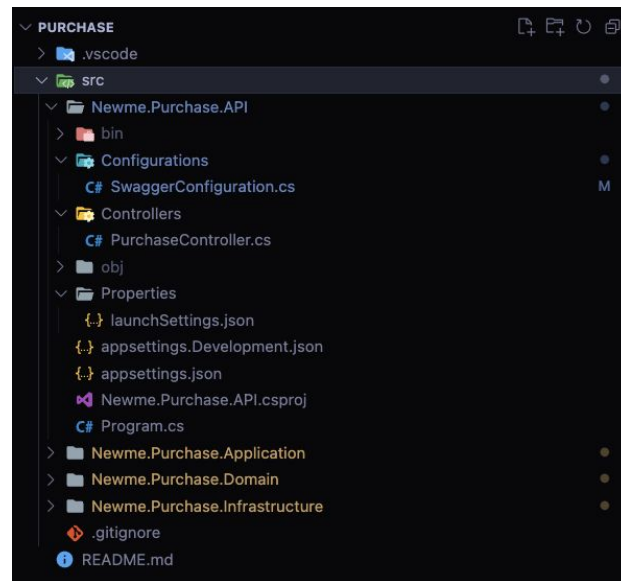
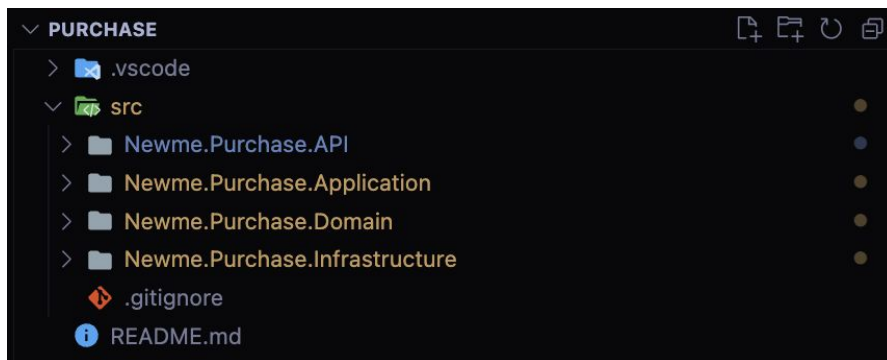


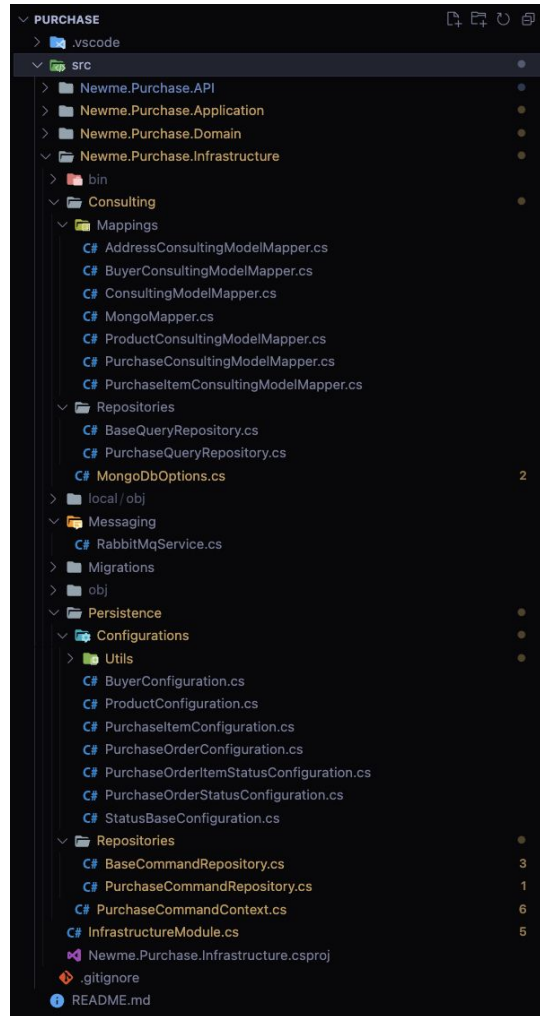
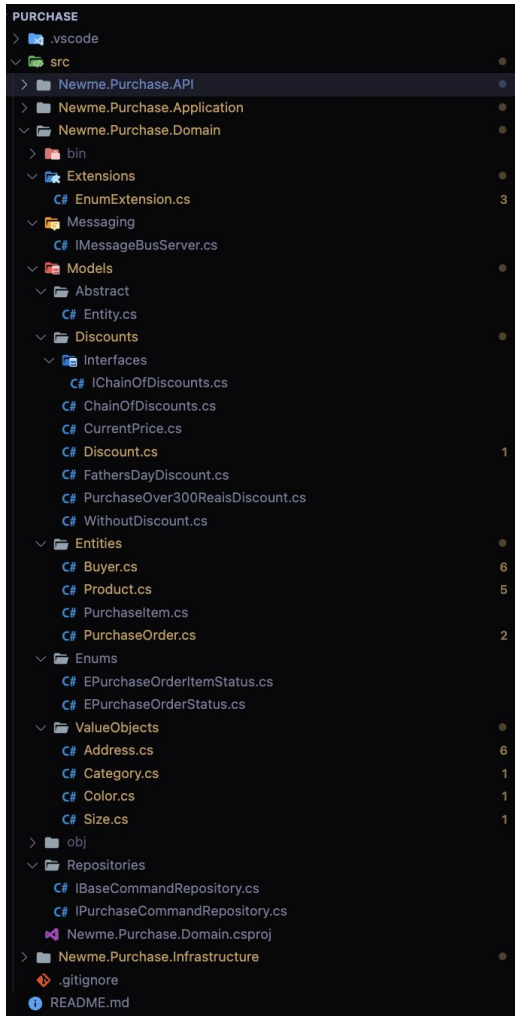
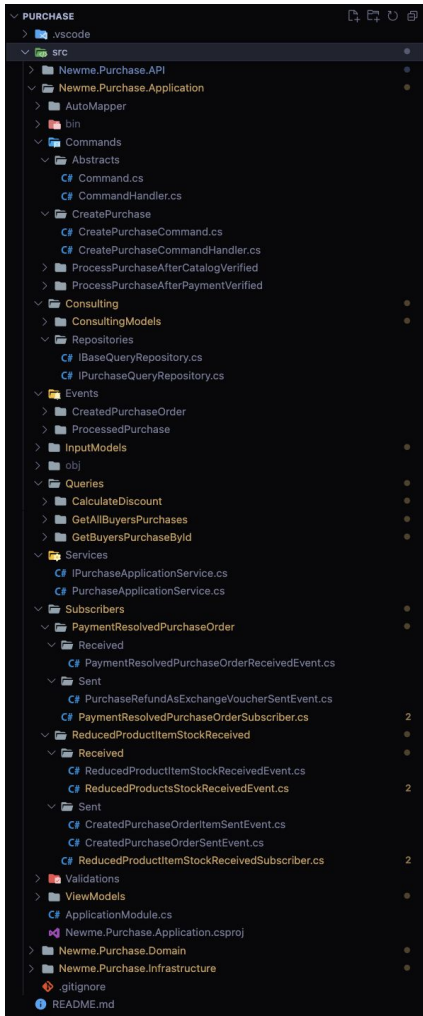






Camadas dos Microserviços





Tecnologias Utilizadas

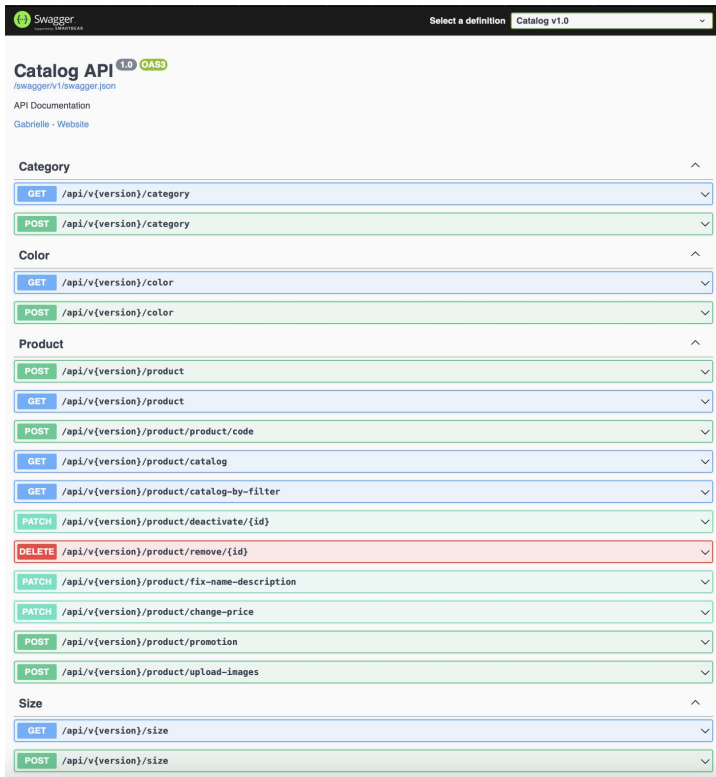
- Linguagens de programação e Frameworks
 - C#, plataforma .NET.
- Armazenamento de Dados:
 - SQL Server, MongoDB
 - Amazon Web Services - AWS S3
- Comunicação entre Microserviços
 - RabbitMQ



Exemplos de Uso



Exemplos de Uso



Swagger
v1.0.0

Select a definition: Catalog v1.0

Catalog API 1.0 OAS3

/swagger/v1/swagger.json
API Documentation
Gabrielle - Website

Category

- GET /api/v(version)/category
- POST /api/v(version)/category

Color

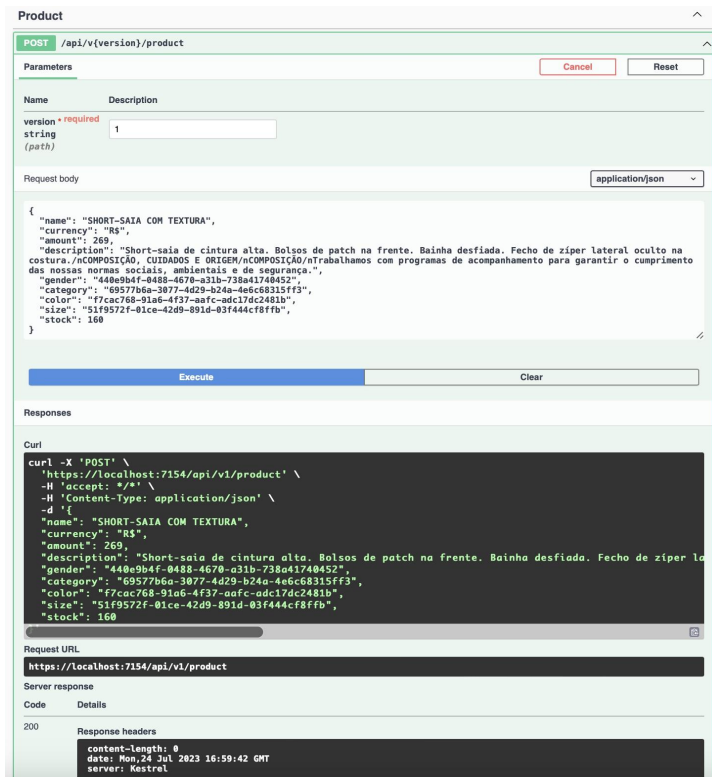
- GET /api/v(version)/color
- POST /api/v(version)/color

Product

- POST /api/v(version)/product
- GET /api/v(version)/product
- POST /api/v(version)/product/product/code
- GET /api/v(version)/product/catalog
- GET /api/v(version)/product/catalog-by-filter
- PATCH /api/v(version)/product/deactivate/{id}
- DELETE /api/v(version)/product/remove/{id}
- PATCH /api/v(version)/product/fix-name-description
- PATCH /api/v(version)/product/change-price
- POST /api/v(version)/product/promotion
- POST /api/v(version)/product/upload-images

Size

- GET /api/v(version)/size
- POST /api/v(version)/size



Product

POST /api/v(version)/product

Parameters

| Name | Description |
|--------------------|-------------|
| version * required | 1 |
| string | (path) |

Request body

application/json

```
{
  "name": "SHORT-SAIA COM TEXTURA",
  "currency": "R$",
  "amount": 269,
  "description": "Short-saia de cintura alta. Bolsos de patch na frente. Bainha desfiada. Fecho de zíper lateral oculto na costura./nCOMPOSICÃO, CUIDADOS E ORIGEM/nCOMPOSICÃO/nTrabalhamos com programas de acompanhamento para garantir o cumprimento das nossas normas sociais, ambientais e de segurança.",
  "gender": "440e9b4f-8488-4670-a31b-738a41740452",
  "category": "69577b6a-3077-4d29-b24a-4a6c68315ff3",
  "color": "f7cac768-91a6-4f37-aafc-adc17dc2481b",
  "size": "51f9572f-01ce-42d9-891d-83f444cf8ffb",
  "stock": 160
}
```

Execute Clear

Responses

Curl

```
curl -X 'POST' \
  'https://localhost:7154/api/v1/product' \
  -H 'accept: */*' \
  -H 'Content-Type: application/json' \
  -d '{
    "name": "SHORT-SAIA COM TEXTURA",
    "currency": "R$",
    "amount": 269,
    "description": "Short-saia de cintura alta. Bolsos de patch na frente. Bainha desfiada. Fecho de zíper lateral oculto na costura./nCOMPOSICÃO, CUIDADOS E ORIGEM/nCOMPOSICÃO/nTrabalhamos com programas de acompanhamento para garantir o cumprimento das nossas normas sociais, ambientais e de segurança.",
    "gender": "440e9b4f-8488-4670-a31b-738a41740452",
    "category": "69577b6a-3077-4d29-b24a-4a6c68315ff3",
    "color": "f7cac768-91a6-4f37-aafc-adc17dc2481b",
    "size": "51f9572f-01ce-42d9-891d-83f444cf8ffb",
    "stock": 160
  }'
```

Request URL

https://localhost:7154/api/v1/product

Server response

Code Details

200

Response headers

```
content-length: 0
date: Mon, 24 Jul 2023 16:59:42 GMT
server: Kestrel
```

- Cadastrando produto.

Exemplos de Uso

The screenshot shows a REST client interface with the following sections:

- POST /api/v{version}/product/upload-images**
- Parameters:** A table with columns 'Name' and 'Description'. It contains one parameter: 'version' (required, string, path) with a value of '1'.
- Request body:** A dropdown menu set to 'multipart/form-data'.
- ProductId:** A required string field with the value 'd7778dcd-2855-4c54-bcd6-5bbe08fe52'.
- Images:** A required array field containing three file uploads: 'SHORT-SAIA 1.jpeg', 'SHORT-SAIA 2.jpeg', and 'SHORT-SAIA 3.jpeg'. There is an 'Add string item' button below.
- Execute:** A blue button to execute the request.
- Responses:** A section showing the curl command and the server response.

```
curl -X 'POST' \
'https://localhost:7154/api/v1/product/upload-images' \
-H 'accept: */*' \
-H 'Content-Type: multipart/form-data' \
-F 'ProductId=d7778dcd-2855-4c54-bcd6-5bbe08fe52ba' \
-F 'Images=@SHORT-SAIA 1.jpeg;type=image/jpeg' \
-F 'Images=@SHORT-SAIA 2.jpeg;type=image/jpeg' \
-F 'Images=@SHORT-SAIA 3.jpeg;type=image/jpeg'
```

Request URL
https://localhost:7154/api/v1/product/upload-images

Server response

| Code | Details |
|--------------|--|
| 204 | Response headers |
| Undocumented | date: Mon, 24 Jul 2023 17:03:55 GMT server: Kestrel |

Responses

- Adicionando imagens para um produto cadastrado.

Exemplos de Uso

POST

/api/v{version}/payment/change-status

Parameters

Cancel

Reset

| Name | Description |
|-----------------------------------|-------------|
| version <small>* required</small> | |
| string | 1 |
| <small>(path)</small> | |

Request body

application/json

```
{  "id": "6e5b0166-3c51-4db1-ab48-64204d7d30af",  "type": "Pix",  "status": "AuthorizedPayment"}
```

Execute

Clear

Responses

Curl

```
curl -X 'POST' \
'https://localhost:7189/api/v1/payment/change-status' \
-H 'accept: */*' \
-H 'Content-Type: application/json' \
-d '{
  "id": "6e5b0166-3c51-4db1-ab48-64204d7d30af",
  "type": "Pix",
  "status": "AuthorizedPayment"
}'
```

Request URL

https://localhost:7189/api/v1/payment/change-status

Server response

| Code | Details |
|------|---------|
|------|---------|

- Alterando o status do pagamento de uma compra.

Schemas

```
ChangePaymentStatusInputModel ∨ {
  id      string($uuid)
  type    EPaymentType string
  Enum:
    ∨ [ Pix, DebitCard, CreditCard ]
  status  EPaymentStatus string
  Enum:
    ∨ [ PaymentValidation, AuthorizedPayment, UnauthorizedPayment ]
}
```

```
EPaymentStatus ∨ string
Enum:
  ∨ [ PaymentValidation, AuthorizedPayment, UnauthorizedPayment ]
```

```
EPaymentType ∨ string
Enum:
  ∨ [ Pix, DebitCard, CreditCard ]
```

Contribuições

- Visão geral sobre os principais aspectos de qualidade e arquitetura de software.
- Exemplos de aplicação.
 - Microserviços.
 - Padrão arquitetural CQRS.
 - Clean Architecture (Arquitetura Limpa).
 - SOLID.
 - Padrões de projeto como: repository, factory, chain of responsibility e mediator.

Limitações

- Finalizar a implementação de toda a proposta.
- Abordar observabilidade em microserviços.

Trabalhos Futuros

- Implementar algum microsserviço existente, ou um novo, e integrar ao sistema, utilizando outras tecnologias.
- Aprofundar no tema de micro frontends ou micro apps.

Onde encontrar

- https://1drv.ms/b/s!AkaDfOFQ8sOmgf5kLF_L0oZa1x-p2A?e=258X8T