

Relatório Técnico

Desafio BTG-Pactual por Gustavo Ferreira

SUMÁRIO

- I. Plano de trabalho.
- II. Tecnologias utilizadas.
- III. Diagrama de arquitetura.
- IV. Modelagem de Banco de Dados.
- V. Diagrama da implantação.
- VI. Diagrama de Infra.
- VII. Evidências de testes funcionais.
- VIII. Publicação.
- IX. Referências.
- X. Técnicas.
- XI. Docker.

I.Plano de trabalho:

Previsto:

1h para criação de arquitetura DDD na solução

3h para criação de web API

3h para criação e Worker Consumer

Realizado:

1h para criação de arquitetura DDD na solução

2h para criação de web API

1:30h para criação e Worker Consumer

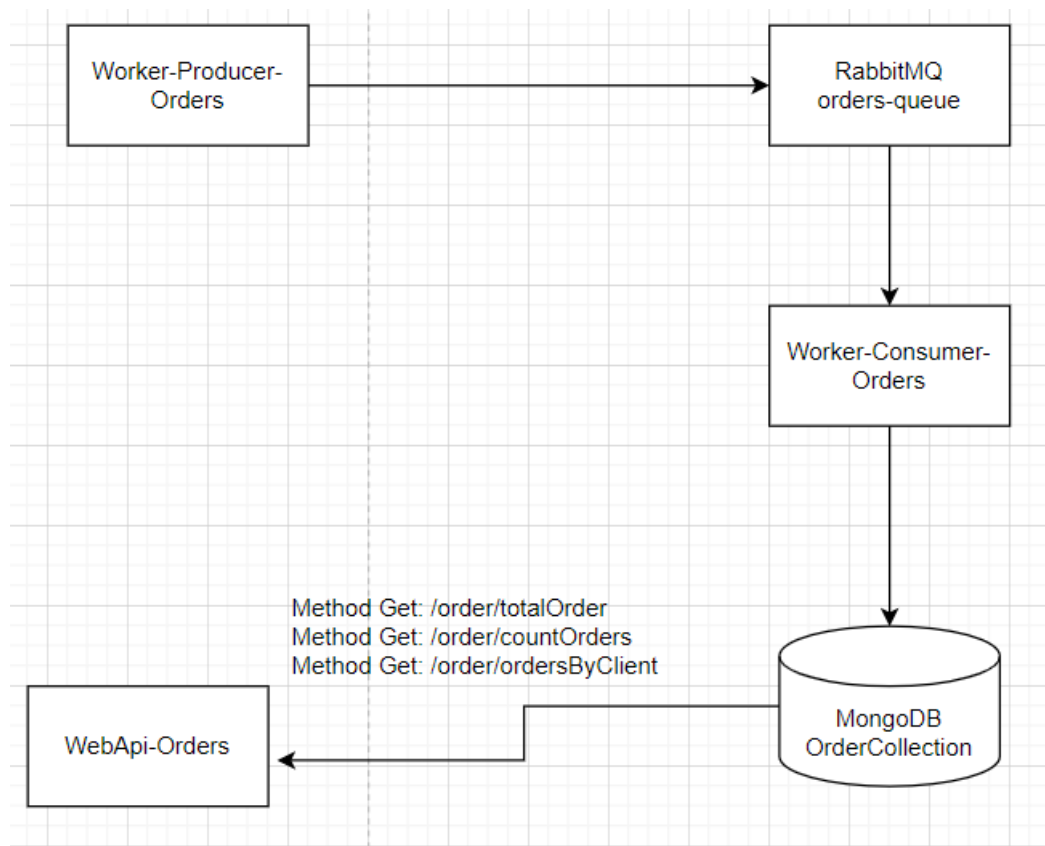
Observações:

Sem desvios no plano de trabalho.

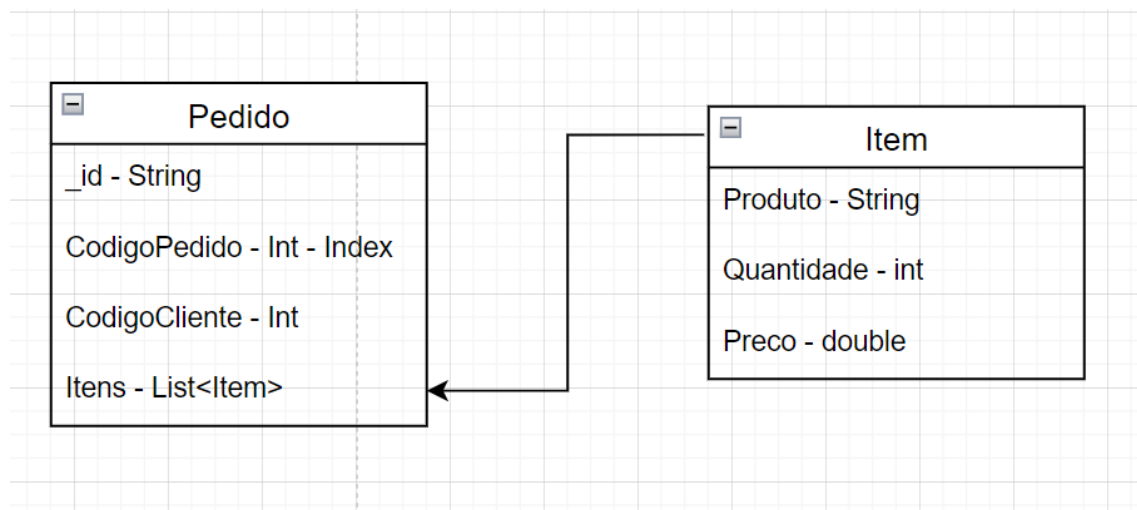
II. Tecnologias utilizadas:

.Net Core, MongoDB, RabbitMQ e Docker.

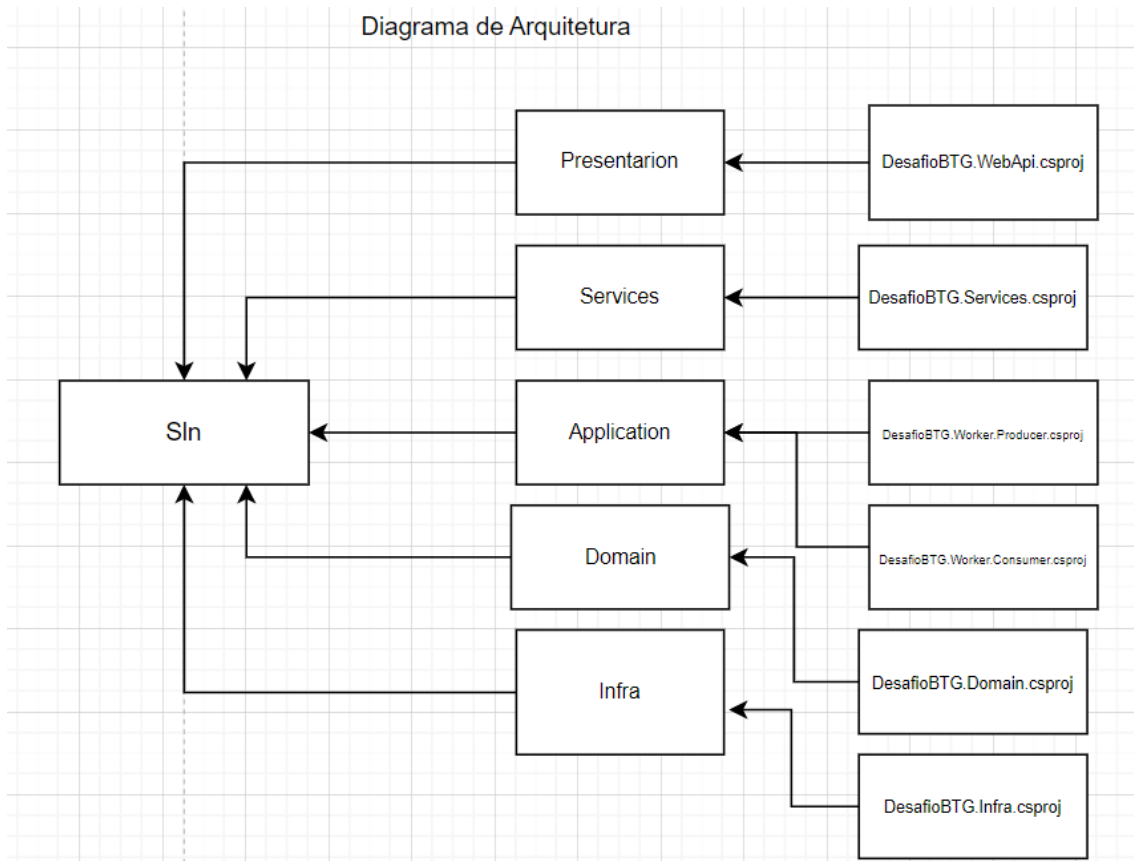
III. Diagrama de Arquitetura:



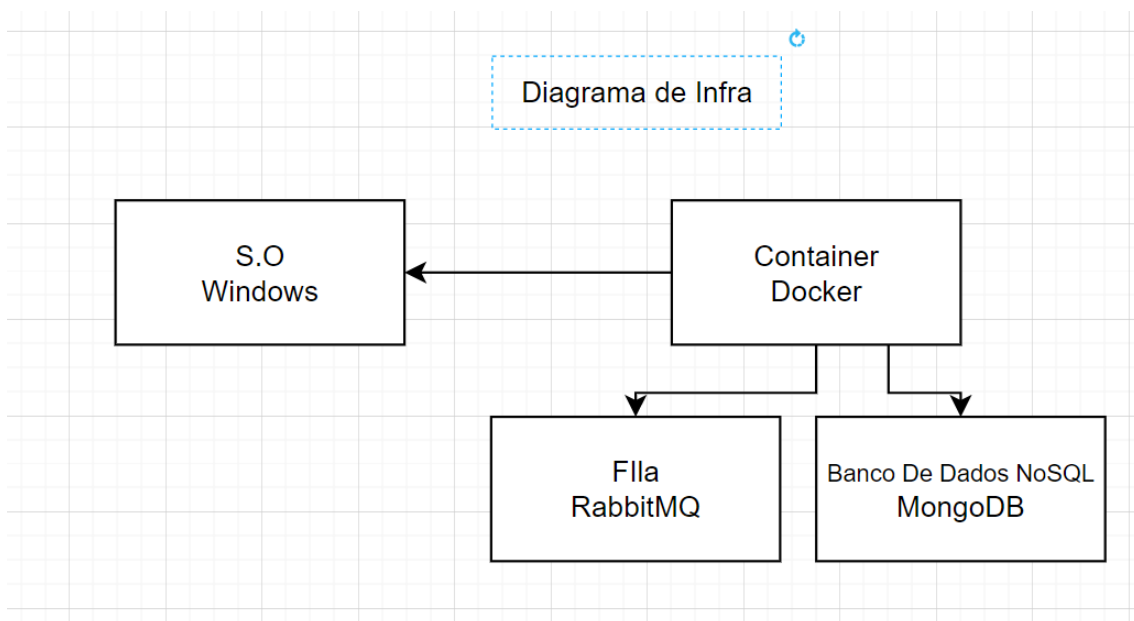
IV. Modelagem de Banco de Dados:



V. Diagrama da implantação:



VI. Diagrama de Infra:



VII. Evidências de testes funcionais:

Evidência do Worker-Consumer:

RabbitMQ

RabbitMQ 3.11.9

Erlang 25.2.3

Retreshed 202

Overview

Connections

Channels

Exchanges

Queues

Admin

Consumers (0)

Bindings (1)

Publish message

Get messages

/arning: getting messages from a queue is a destructive action. ?

ack Mode:

Nack message requeue true

Encoding:

Auto string / base64

Messages:

1

Get Message(s)

Message 1

The server reported 2 messages remaining.

Exchange

(AMQP default)

Routing Key

orders-queue

Redelivered

0

Properties

Payload

153 bytes













Encoding: string

{"codigoPedido":1001,"codigoCliente":1,"itens":[{"produto":"l\u00E9pis","quantidade":100,"preco":1.1}, {"produto":"caderno","quantidade":10,"preco":5.5}]}

- Mensagem exibida no RabbitMQ

```
info: DesafioBTG.Worker.Consumer.WorkerExecutor[0]
[*] Waiting for messages.
info: DesafioBTG.Worker.Consumer.WorkerExecutor[0]
Finish Worker
info: DesafioBTG.Worker.Consumer.WorkerExecutor[0]
[x] Received: {"codigoPedido":1001,"codigoCliente":1,"itens":[{"produto":"l\u00E9pis","quantidade":100,"preco":1.1}, {"produto":"caderno","quantidade":10,"preco":5.5}]}
info: DesafioBTG.Worker.Consumer.WorkerExecutor[0]
[x] Order Code: 1001 | CodeClient 1
info: DesafioBTG.Worker.Consumer.WorkerExecutor[0]
[x] Received: {"codigoPedido":1002,"codigoCliente":1,"itens":[{"produto":"caneta","quantidade":50,"preco":3.1}, {"produto":"borracha","quantidade":5,"preco":0.5}]}
info: DesafioBTG.Worker.Consumer.WorkerExecutor[0]
[x] Order Code: 1002 | CodeClient 1
info: DesafioBTG.Worker.Consumer.WorkerExecutor[0]
[x] Received: {"codigoPedido":1003,"codigoCliente":2,"itens":[{"produto":"caneta","quantidade":12,"preco":3.1}, {"produto":"borracha","quantidade":6,"preco":0.5}]}
info: DesafioBTG.Worker.Consumer.WorkerExecutor[0]
[x] Order Code: 1003 | CodeClient 2
```

- em destaque amarelo no console a mensagem que estava sendo exibida no RabbitMQ em seguida outras mensagens da fila sendo consumidas pelo Worker-Consumer.

Order > codigoPedido			
_id	codigoPedido	codigoCliente	itens
 63fa4d48bb35209a1c8219b3	 1003	 2	 [2 elements]
 63fa4d47bb35209a1c8219b1	 1001	 1	 [2 elements]
 63fa4d47bb35209a1c8219b2	 1002	 1	 [2 elements]

- evidência do consumo do Worker-Consumer gravando os dados da fila na Collection Order no banco de dados MongoDB

Evidência da WebApi:

GET
▼
http://localhost:5000/api/order/countOrders?codeClient=1

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

Query Params

	KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/>	codeClient	1	
	Key	Value	Description

body
Cookies
Headers (4)
Test Results
200 OK 1257 ms

Pretty
Raw
Preview
Visualize
JSON
▼

1 2

- rota countOrders retornando a quantidade de pedidos por cliente


GET ▼ http://localhost:5000/api/order/totalOrder?codeOrder=1001


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

Query Params

	KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/>	codeOrder	1001	
	Key	Value	Description

Body Cookies Headers (4) Test Results

 200 OK 198 ms

Pretty Raw Preview Visualize JSON ▼ 

1 165

- rota codeOrder retornando a soma do total do pedido


GET ▼ http://localhost:5000/api/order/ordersByClient?codeClient=1


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

Query Params

	KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/>	codeClient	1	
	Key	Value	Description

Body Cookies Headers (4) Test Results

 Status: 200 OK Time: 7 ms Size: 44

Pretty Raw Preview Visualize JSON ▼ 

```
1 {
2   "codigoPedido": 1001,
3   "codigoCliente": 1,
4   "itens": [
5     {
6       "produto": "lápis",
7       "quantidade": 100,
8       "preco": 1.1
9     },
10    {
11      "produto": "caderno"
```

- rota orderByClient retornando a lista de todos os pedidos do cliente.

VIII. Publicação:

Repositório Github: <https://github.com/gustavofvieira/BTG-Challenge>

IX. Referências:

RabbitMq: <https://www.rabbitmq.com/getstarted.html>

X. Técnicas:

Foi implementado neste projeto o DDD (Domain-Driven Design), para design otimizado da arquitetura e boa manutenção, utilizado o S.O.L.I.D para implementação de toda a estrutura do projeto, Injeção de Dependencias de interfaces

XI. Docker.

Imagens utilizadas:

https://hub.docker.com/_/rabbitmq - RabbitMq

https://hub.docker.com/_/mongo - Mongo DB