Baskt Api

Algumas instruções  
*Nesse documento irei deixar algumas configurações que foram realizadas.*

Irei deixar algumas configurações para docker e a execução de outros scripts.

# Comunicando-se com o container do Redis

Esses passos são para executar o container do redis do qual irá ser acessado por uma aplicação que está rodando localmente, sem ter um container para o app.

1. Baixar e executar a imagem.
2. docker run --name local-redis -p 6379:6379 -d redis
3. Aplicação fora do container acessar o redis
4. Só apontar o webconfig para o redis.

"CacheSeetings": {

"ConnectionString": "localhost:6379"

},

# Configuração do DockerFile

O dockerfile pode ser criado mediante a ferramenta de suporte do visual studio e a imagem e o container pode ser criado executando Docker.

Também é possível criar a imagem:

docker build -t apibasket:1.0 -f Basket.Api/Dockerfile .

Para criar o container:

docker container run -it --rm -p 3000:80 --name apibasketcontainer apicatalogo:1.0

Lembrando que vai ser o docker-compose que irá gerenciar o docker file.

# Configuração do Docker Compose

O docker compose pode ser criado mediante a ferramenta do visual studio.

version: '3.4'

# ~~> DEFININDO OS VOLUMES

volumes:

mongo\_data:

# ~~> DEFININDO CONTAINER DO REDIS

services:

basketredis:

container\_name: basketdb

restart: always

image: redis:alpine

ports:

- "6379:6379"

# ~~> DEFININDO CONTAINER DA BASKET.API

catalog.api:

container\_name: basket.api

environment:

- ASPNETCORE\_ENVIRONMENT=Development

image: basketapi

build:

context: .

dockerfile: Api.Basket/Dockerfile

depends\_on:

- basketredis

ports:

- "8001:80"

O comando cria os dois containers, mongo e api catalog.

docker-compose -f docker-compose.yml up -d

# Client Grpc

Foi criando na Api.Baskt uma chamada para o Grpc Service.

Tela de computador com texto preto sobre fundo branco

Descrição gerada automaticamente

Adicionando uma referência de serviço, do tipo grpc, obtendo o arquivo .proto do server, foi gerado o cliente.