Certified Tech Developer
The Ultimate Degree

Especialização Back End I

Checkpoint final

Este checkpoint será diferente do anterior, pois iremos construí-lo durante as mesas de trabalho no restante da disciplina.

O objetivo é que você possa aproveitar esses espaços para tirar dúvidas com o professor e completar as instruções.

A entrega dessa avaliação irá acontecer na aula 24.

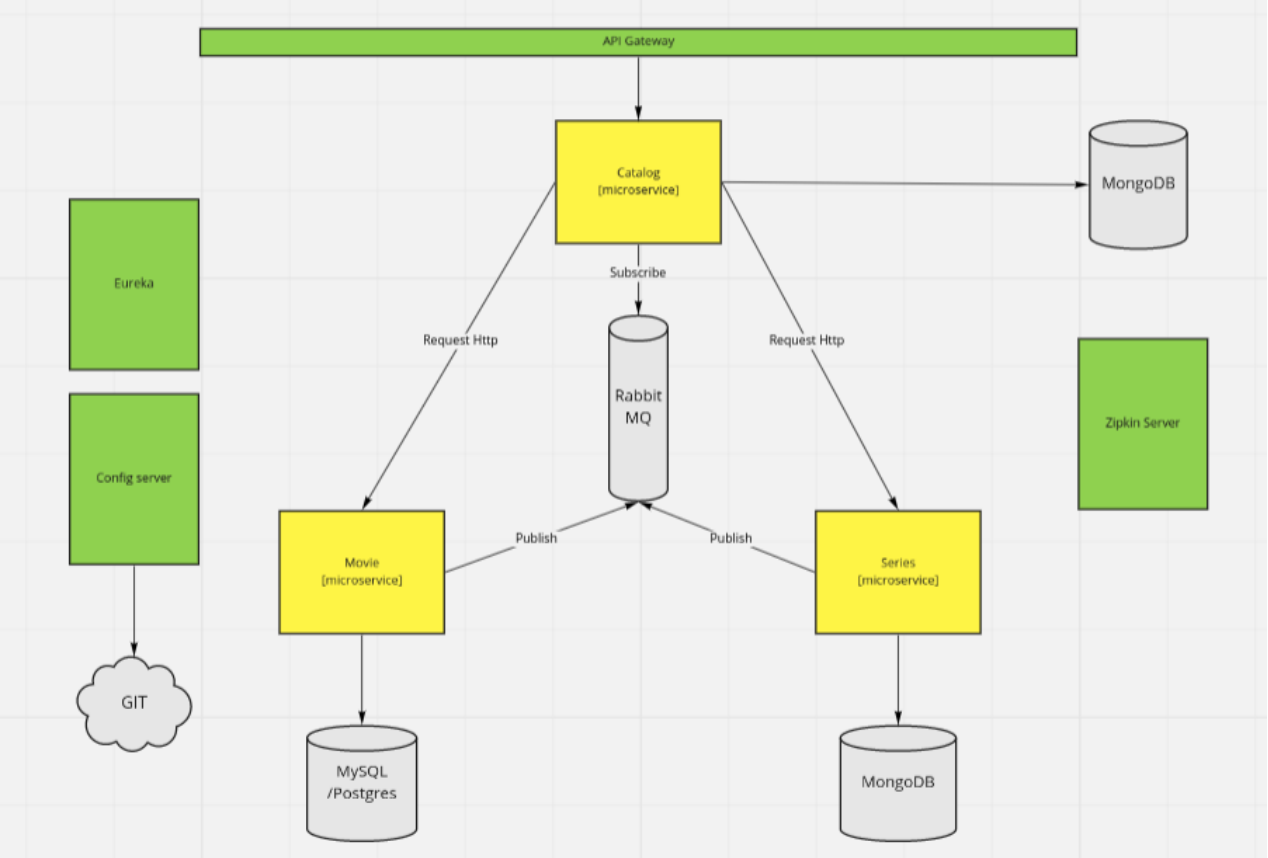
Contextualização

O projeto consiste em 3 micro serviços: **Filme, Série e Catálogo.**

O **Catálogo** é um micro serviço que tem informações de **Filmes e Séries** para enviar um catálogo ao cliente.

O **catálogo** recebe uma mensagem sempre que um filme ou série é cadastrado e os mantém em um banco de dados MongoDB. Quando uma solicitação vem do cliente, ele pesquisa no banco de dados e responde.

Diagrama básico de microsserviços:



A seguir, veremos as informações detalhadas dos micro serviços.

movie-service

O microsserviço gerencia as operações nos filmes, cada filme tem como atributos:

- id
- name
- genre
- urlStream

serie-service

O microsserviço gerencia as operações na série, cada série tem como atributos:

- Id
- name
- genre
- seasons
    - id
    - seasonNumber
    - chapters

        - id
        - name
        - number
        - urlStream

catalog-service

O microsserviço tem como objetivo invocar os micro serviços de filmes e séries. Esses micro serviços devem ser invocados sempre que um novo filme ou série é carregado e as informações fornecidas por ambos os micro serviços devem ser persistidas em um banco de dados MongoDB não relacional com a seguinte estrutura:

- genre
    - movies
        - id
        - name
        - genre
        - urlStream
    - series
        - Id
        - name
        - genre
        - seasons
            - id
            - seasonNumber
            - chapters
                - id
                - name
                - number
                - urlStream

Requisitos

serie-service

- Criar um micro serviço com o nome "serie".
- Configurar o eureka para o novo micro serviço e utilizar o nome: **serie-service**
- Configurar a rota no gateway para o novo serviço e adicionar ele a segurança e ao oAuth.

- Configurar o server config para obter a configuração desde um repositório do git.
- Criar uma API que permita:
    - Obter uma lista de séries por gênero **Endpoint : /series/{genre} [GET]**
    - Adicionar uma nova série. **Endpoint : /series [POST]**
- Persistencia: Adicionar a dependencia e implementar o MongoRepository para persistir as novas series.
- Adicionar o rabbitMQ e enviar uma mensagem no momento que se adicionar uma nova série.

movie-service

- Adicionar a persistencia: Utilizar o MySql para persistir os filmes.
- Configurar a rota no gateway para adicionar segurança com o oAuth.
- Adicionar o RabbitMQ e enviar uma mensagem no momento que se adicionar um novo filme.

catalog-service

- Atualizar o catálogo utilizando Feign de maneira que adicione a este serviço a busca da serie por género (serie-service) e criar a resposta do **endpoint /catalog/{genre}**
- Adicionar persistência: Assim que receber os filmes e as series por genero, persistiram as mesmas informações no MongoDB.
- Adicionar o rabbitMQ e escutar as mensagens que os serviços **movie-service** e **serie-service** enviam, e atualizar a lista de filmes e séries que o catálogo fornece.

Spring Cloud

- Criar um projeto e configurar utilizando o **zipkin server** para receber as mensagens dos microsserviços. Adicionar a **Zipkin UI** para visualização das informações.
- Configurar o zipkin em cada um dos micro serviços.
- Através da **Zipkin UI** visualizar as comunicações entre os micro serviços**.**

Deployment: Todos os microsserviços precisarão ser implantados em dockers.

Resilience4J

- Do projeto anterior, você deve selecionar um dos serviços, de preferência aquele que você acha que será mais utilizado e adaptá-lo para que seja tolerante a falhas.
  você deve:
  - Defina o esquema de resiliência (por exemplo: redundância dupla, nova tentativa e fallback, balanceamento de carga, tempos de aquecimento, regras de circuito).
  - Modifique o código do seu projeto (aplicando qualquer uma das 3 tecnologias mencionadas) para que o esquema definido seja aplicado dentro do serviço selecionado.
- No mínimo, o serviço deve ter:
  - Redundância dupla.
  - Regras de circuito (podem criar um serviço que retorna ativo / inativo com base na memória disponível / uso do processador / exceções).
  - Descrição da solução de redundância, justificativa. (Um comentário no código.).