Disciplina: Sistemas Distribuídos
Professora: Ana Cristina Barreiras Kochem Vendramin

Avaliação (valor 2,5)
Microsserviços, Middleware orientado a Mensagens, REST, SSE, Webhook

Implemente uma aplicação web de Reserva de Cruzeiros (ver Figura 1).

***Frontend* (valor 0,7)**

O ***frontend*** deve ser implementado em uma **linguagem diferente** daquela utilizada no *backend*. Ele deve se comunicar com o *backend* por meio de uma API **REST (valor 0,4)** e receber notificações via **SSE (*Server-Sent Events*) (valor 0,3)**.

Os clientes poderão interagir com a aplicação web para:

- consultar itinerários disponíveis;
- efetuar uma reserva;
- **cancelar uma reserva;**
- **registrar interesse em receber notificações de promoções;**
- **cancelar interesse em receber notificações de promoções.**

***Backend* (valor 1,6)**

O *backend* é composto por **cinco microsserviços (MSs)** desenvolvidos de forma independente, que se comunicam, na maioria das vezes, de maneira assíncrona e desacoplada, através de um serviço de mensageria, como o RabbitMQ.

(valor 0,9) **MS Reserva**

(valor 0,5) Responsável por receber as requisições REST do *frontend* para:

- **Consultar itinerários**: com base nos dados fornecidos pelo cliente (destino, data de embarque, porto de embarque), **o MS Reserva faz uma requisição REST para o MS Itinerários para consultar os itinerários disponíveis** e retornar ao cliente as seguintes informações: datas disponíveis de partida, nome do navio, porto de embarque, lugares visitados, número de noites, valor por pessoa;
- **Efetuar uma Reserva**: o cliente seleciona um cruzeiro disponível para efetuar a reserva, informando a data de embarque escolhida, o número de passageiros e o número de cabines. O MS Reserva cria a reserva, publica a mensagem sobre a reserva na fila *reserva-criada*, **faz uma requisição REST para o MS Pagamento solicitando um link de pagamento para a reserva** e retorna esse link ao cliente.

- **<u>Cancelar uma reserva</u>: o cliente informa o código da reserva.** O **MS Reserva publica uma mensagem na fila *reserva-cancelada***. O cancelamento pode ocorrer por solicitação do cliente ou por recusa do pagamento;
- **<u>Registrar interesse em receber notificações de promoções</u>**: o cliente apenas manifesta interesse em ser notificado sobre promoções de cruzeiros. O MS Reserva deve gerenciar os interesses ativos e os eventos gerados, a fim de enviar notificações apenas aos clientes interessados nesses eventos;
- **<u>Cancelar interesse em receber notificações de promoções</u>**: o MS Reserva não deve mais enviar promoções de cruzeiros aos clientes que cancelaram o registro de interesse.

(valor 0,4) Responsável por consumir eventos das filas *pagamento-aprovado, pagamento-recusado, bilhete-gerado* e *promocoes* **e manter conexões SSE com os clientes que iniciaram uma reserva ou registraram interesse em receber notificações sobre promoções. Ele envia notificações personalizadas (através de canais diferenciados pelo clienteId) com base nos eventos consumidos (status de uma reserva e promoções).**

(valor 0,2) **<u>MS Itinerários</u>**

Responsável por gerenciar os itinerários. Ele consome eventos das filas *reserva-criada* e *reserva-cancelada* para atualizar a disponibilidade de cabines. Ele **expõe uma API REST** que é consumida pelo MS Reserva para consultar a **disponibilidade de itinerários e cabines**.

(valor 0,5) **<u>MS Pagamento</u>**

Expõe uma API REST para que o MS Reserva solicite links de pagamento. Integra-se com um sistema externo de pagamentos, para o qual envia os dados do pagamento (valor, moeda, informações do cliente) e recebe um link de pagamento. Ele define um *endpoint* que recebe notificações assíncronas do sistema externo indicando o status da transação (aprovada ou recusada). Com base nos eventos externos recebidos, ele publica mensagens na fila *pagamento-aprovado* ou *pagamento-recusado*.

<u>MS Bilhete</u>

Escuta a fila *pagamento-aprovado* e, ao confirmar que um pagamento foi aprovado, gera o bilhete da viagem e publica uma mensagem na fila *bilhete-gerado*.

<u>MS Marketing</u>

Publica promoções na fila *promocoes*.

(valor 0,2) Sistema de pagamento externo

Responsável por processar os pagamentos. Após o processamento, o sistema de pagamento externo envia uma notificação assíncrona via webhook (HTTP POST) ao *endpoint* configurado no MS Pagamento. Essa notificação inclui informações sobre o evento: ID da transação, status do pagamento (autorizado ou recusado), valor e dados do comprador.

Observações:

- Não há necessidade de assinar as mensagens digitalmente;
- Não precisa mais diferenciar as promoções por destino;
- Desenvolva uma interface com recursos de interação apropriados.
- É obrigatória a defesa da aplicação para obter a nota.
- O desenvolvimento do sistema pode ser individual ou em dupla.

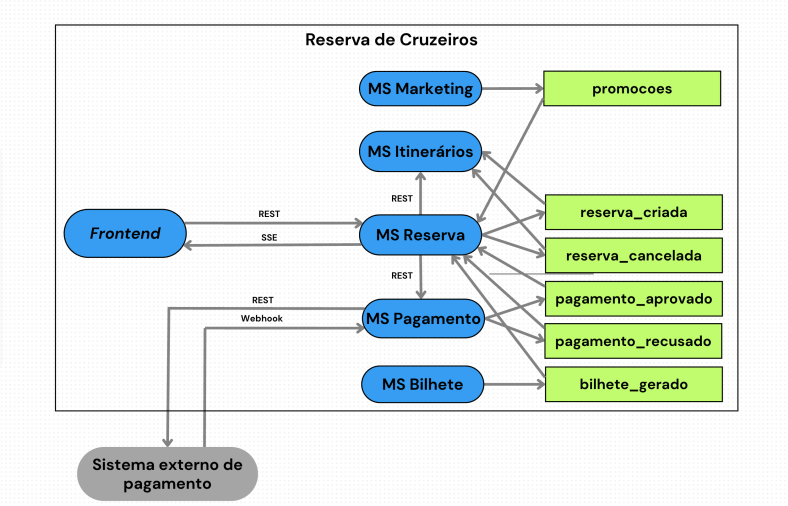

Figura 1 – Arquitetura.

Conceitos:

SSE (*Server-Sent Events*) é uma tecnologia que permite ao servidor enviar atualizações em tempo real para os clientes por meio de uma conexão HTTP, utilizando um canal de comunicação unidirecional. Diferentemente do ***WebSocket***, que estabelece uma comunicação bidirecional, o SSE é projetado especificamente para o envio de dados do servidor para os clientes. Isso o torna especialmente adequado para aplicações que demandam notificações em tempo real ou atualizações contínuas, como alertas de promoções.

Webhooks (ou *APIs reversas* ou *APIs de push)* são mais comumente usados para comunicação entre servidores de forma assíncrona, enquanto REST é um padrão de comunicação cliente-servidor que geralmente envolve interações síncronas, iniciadas e controladas pelo cliente. *Webhooks* são ideais para notificar um servidor sobre a ocorrência de um evento em outro servidor. Isso torna a comunicação mais eficiente e escalável, sem a necessidade de manter conexões abertas ou realizar chamadas repetitivas (*polling*) para verificar se um evento ocorreu.