Projeto e Arquitetura de Software
Prof. Bernardo Copstein

Trabalho Final – Parte 2

Backend do Sistema de uma Tele Pizza

Enunciado Geral da “Parte 1”

Uma pizzaria online utiliza um sistema de gestão de pedidos automatizados. O cliente acessa o sistema usando um aplicativo no celular. Inicialmente o cliente deve entrar no aplicativo. Se ainda não tiver cadastro deve se cadastrar. O cadastro do cliente compreende nome, cpf, celular, endereço, email e senha de acesso. O "usuário" será sempre o email.

Depois de "entrar" no sistema, o cliente pode consultar o cardápio e montar seu pedido. No cardápio o cliente visualiza a descrição de cada item e seu preço unitário. Um pedido pode incluir quantos itens de cardápio desejar. Para cada item deve indicar a quantidade desejada. No final do pedido o cliente deve informar também o endereço de entrega.

Quando o pedido está completo o cliente submete o pedido para aprovação (neste momento o status do pedido é marcado como "NOVO"). Para cada item o sistema conhece a relação de ingredientes necessários. Então é feita uma verificação no estoque para saber se existem ingredientes suficientes para atender a todos os itens do pedido. Em caso negativo o pedido é retornado destacando os itens que não podem ser atendidos. Automaticamente os itens de cardápio correspondentes são marcados como indisponíveis (essa situação só se altera quando chegarem mais ingredientes no estoque). Se tudo estiver ok o status do pedido é alterado para "APROVADO" e o custo do pedido é calculado. Importante notar que os itens do estoque são organizados nas porções necessárias para atender as receitas.

O cálculo do custo compreende somar o custo dos itens individuais, aplicar o desconto - se for o caso - e calcular os impostos. Atualmente a pizzaria paga um imposto único de 10% sobre o valor do somatório do custo dos itens. Para clientes que fizeram mais de 3 pedidos nos últimos 20 dias é fornecido um desconto de 7% no custo de cada item. O custo final é calculado subtraindo-se o desconto do custo dos itens, e somando-se o imposto ao resultado da subtração.

O pedido "APROVADO" é então retornado para o cliente efetuar o pagamento. Neste momento o cliente pode cancelar o pedido ou efetuar o pagamento. Se o

pagamento for efetuado o pedido é encaminhado para a fila da cozinha com o status "PAGO" (a partir deste momento não pode mais ser cancelado) e uma mensagem com o número do pedido é informada ao cliente. Caso contrário o pedido é abandonado.

Na cozinha, o pedido recebe o status "AGUARDANDO". Quando começa a ser preparado o pedido passa para o status “PREPARAÇÃO”. Quando o pedido estiver pronto o status passa para “PRONTO”. Neste momento é encaminhado para a fila do setor de entregas.

No setor de entregas o pedido aguarda ser atribuído a um entregador. Quando um entregador recebe o pedido este passa para o estado "TRANSPORTE" e, finalmente, quando é entregue passa para o estado "ENTREGUE". Pedidos entregues são então arquivados no sistema associados ao cliente. Durante todo o tempo, as mudanças de status do pedido podem ser acompanhadas pelo aplicativo a partir do número do pedido.

Casos de uso e serviços definidos na “Parte 1”

Na parte 1 do trabalho foram definidos os seguintes casos de uso:

- Registrar cliente no sistema (UC1)
    - Cliente se registra no sistema para poder se logar futuramente
- Autenticar no sistema (UC2)
    - Cliente se autentica no Sistema para poder seguir operando
- Carregar cardápio (UC3)
    - Cliente solicita cardápio para poder montar pedido (nesta versão é devolvido sempre o cardápio de código 1 – prever uma forma de controlar qual é o cardápio corrente).
- Submeter pedido para aprovação (A) (UC4)
    - Cliente submete um pedido (lista de itens) para aprovação. O retorno deste caso de uso é o pedido aprovado com o preço calculado ou o pedido negado por falta de ingredientes.
- Solicitar status de pedido (A) (UC5)
    - Cliente solicita o status de seu pedido
- Cancelar pedido (A) (UC6)
    - Cliente solicita o cancelamento de um pedido aprovado, mas não pago.
- Pagar pedido (A) (UC7)
    - O pagamento do pedido implica que ele é encaminhado para a cozinha para ser elaborado sendo que as mudanças de estado devem ser registradas junto ao pedido. Depois de elaborado o

pedido deve ser encaminhado para o setor de entregas, sendo que as mudanças de estado devem ser registradas junto ao pedido.
- Listar os pedidos entregues entre duas datas (UC8)
- Listar os pedidos de um determinado cliente entregues entre duas datas (UC9)

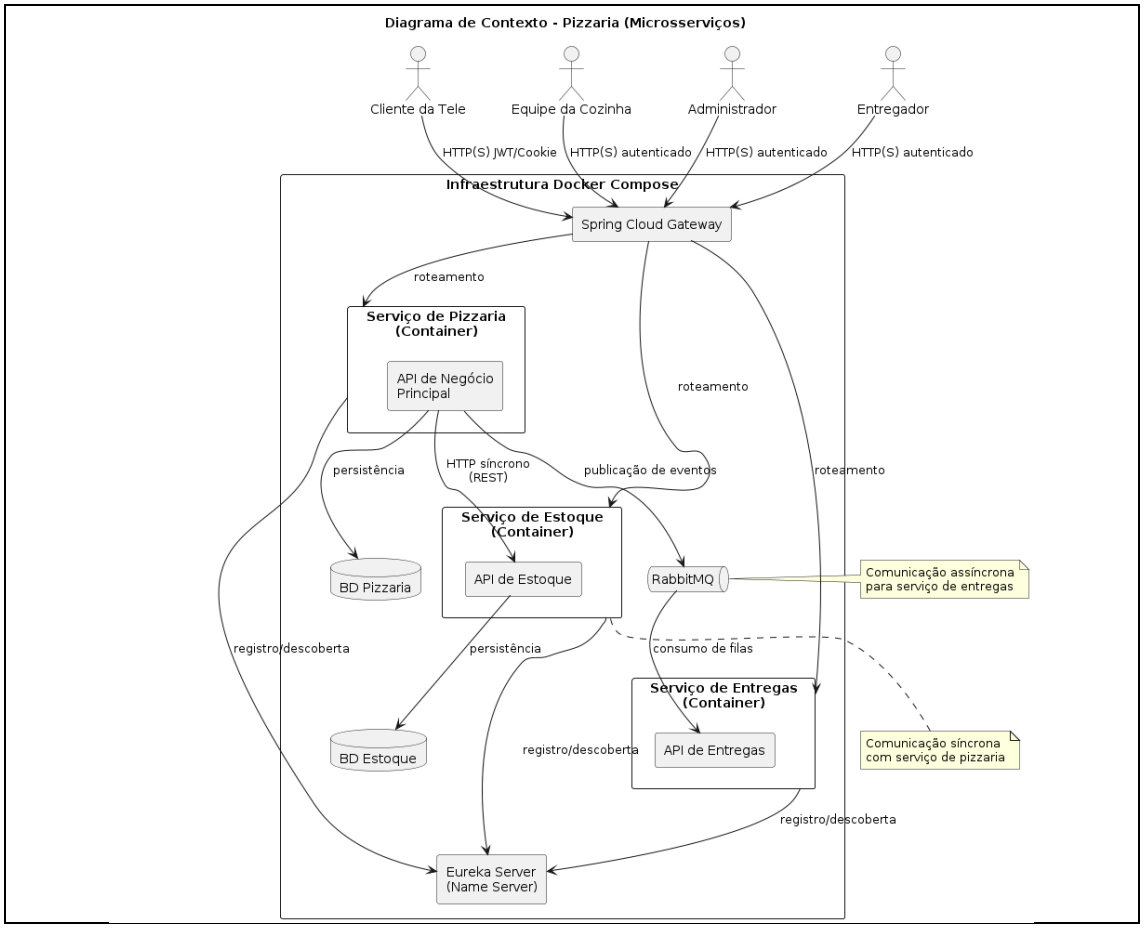
Deverão ser implementados os seguintes serviços de domínio:

- Cadastro de clientes: mantém o cadastro dos clientes
- Autenticação: responsável por autenticação e autorização
- Pedidos: responsável por verificar a consistência do pedido, calcular valores etc. Para o cálculo dos valores acionar o serviço de impostos e o serviço de descontos.
- Estoque: mantém a relação de itens disponíveis para o preparo dos itens do cardápio (incluindo bebidas)
- Cardápio: mantém a lista de itens que podem compor um pedido com os respectivos preços e receitas
- Cozinha: mantém a fila de pedidos e acompanha o preparo (versão simulada)
- Entrega: mantém a fila de entregas, atribui entregadores e acompanha a entrega (versão simulada)
- Pagamento: responsável pelos meios de pagamento
- Impostos: responsável pelo cálculo de impostos
- Descontos: responsável pelas políticas de descontos e fidelidade se houverem

Objetivos da "Parte 2"

Na parte 2 do trabalho o objetivo principal é experimentar uma arquitetura de microsserviços. Para tanto o monolito desenvolvido na etapa 1 deverá ser incluído em uma infraestrutura de microsserviços. Nesta cada microserviço deverá ser "conteinerizado". O acesso aos mesmos deve ser feito a partir de um "gateway" que devem se registrar em um "name server". Segue o detalhamento dos itens que devem fazer parte da solução final:

- A aplicação deve ser composta pelos seguintes microserviços:
    - "Name server", conforme exemplo apresentado em aula
    - "Gateway", conforme exemplo apresentado em aula acrescido das funcionalidades necessárias para autenticação dos usuários



- “Serviço de pizzaria” corresponde ao monolito da “parte 1” com as alterações propostas no restante deste enunciado. Entre estas alterações está a troca do mecanismo de autenticação que deve ser retirado do “serviço principal” e passado para junto do “gateway”.
- O “serviço de estoque” agora deve ser implementado como um microsserviço a parte, com banco de dados próprio. O acesso a este banco de dados deve ser feito usando a tecnologia **JPA** obrigatoriamente. A comunicação entre o “serviço principal” e o “estoque” deve ser síncrona.
- O “módulo de entregas” também deve ser implementado como um microserviço a parte. Seu funcionamento pode continuar sendo simulado, mas em um microserviço a parte. As requisições de entrega devem ser recebidas através de uma fila de mensagens. Devem ser providenciadas pelo menos 3 instancias desse microserviço que devem consumir as demandas a partir de uma fila única.
- “RabbitMQ”: deve existir uma instancia do “RabbitMQ” para servir de brocker de mensagens entre o monolito e o módulo de entregas.

Cronograma de entregas

Data	Casos de uso a serem entregues
10/11/2025	Aplicação contendo gateway, name server, serviço principal e serviço de estoque implementado como um microserviço independente
17/11/2025	Serviço de entregas se comunicando por filas
18/11/2025	Entrega do trabalho completo no Moodle
19/11/2025	Apresentação em sala de aula

Sobre o desenvolvimento em equipe

Será avaliada a participação de cada um dos membros da equipe no desenvolvimento do trabalho. Isso significa que todos os membros da equipe devem ter um número equilibrado de "commits" ou "pull-requests" dependendo da forma como a equipe se organizar. Membros que não conseguirem comprovar suas contribuições efetivas não terão nota atribuída.

Importante definir o "líder do projeto" que irá criar o repositório base onde as versões serão consolidadas.

DICAS E SUGESTÕES

Dica 1: Como disparar várias instâncias de um serviço:

Sintaxe:
docker compose up - -scale <nome-do-serviço>=<qtdade-instancias>
Exemplo: Docker compose up –scale currency-exchange=3
Obs: na hora de indicar o mapeamento das portas no compose.yaml, indicar apenas a porta interna. Ao invés de "ports: 8000:8000", indicar apenas "ports: 8000". Isso permite que o número das portas das diferentes instancias seja ajustado pelo compose.

- É possível verificar que instancia está respondendo através da tela da console.

Dica 2: como fazer cada instancia do serviço de entregas se registrar em uma fila diferente

Na classe de configuração do RabbitMQ criar o nome da fila usando um gerador de números aleatórios para tentar garantir que cada fila tem um nome diferente.

```
@Configuration
public class RabbitMQConfig {
   ...
   public static final String QUEUENAME =
                  "scaa.v1.subscription-update.save-signature" + Math.random() * 1000;
   ...
}
```

Na hora de identificar o método que “escuta” as requisições na fila, recuperar o nome da fila na anotação:

```
   @RabbitListener(queues = "#{rabbitMQConfig.getQueueName()}")
   public void receive(Subscription entity) {
      logger.info("Mensagem recebida com a atualização da assinatura: {}", entity);

      ... // Codigo de atualização da assinatura ...

      logger.info("Data do fim da assinatura guardada na cache.");
   }
```