FACULDADE DE INFORMÁTICA E ADMINISTRAÇÃO PAULISTA

Carlos André Oliveira RM347921
Igor Salvati RM348362
Leandro Dias RM350115
Rafael Gobbi RM348229
Wagner Ferreira RM348060

Trabalho de Arquitetura da Escolha (UX, DESIGN THINKING E MODERN WEB)

Respostas do questionario

1 - O que esperamos aprender com esse projeto?

- Esperamos aprender como melhorar a escalabilidade e eficiência da integração de pedidos em nosso sistema ERP, identificando e implementando soluções tecnológicas adequadas. Além disso, esperamos aprender sobre as melhores práticas de arquitetura de microserviços, Event-Driven Architecture (EDA), CDC (Change Data Capture), SAGA (compensating transactions) e CQRS (Command Query Responsibility Segregation).

2 - Que perguntas precisamos que sejam respondidas?

- Precisamos responder perguntas como:
    - Como podemos otimizar a integração de pedidos para lidar com picos de carga?
    - Quais são as tecnologias e estratégias mais adequadas para melhorar a escalabilidade do nosso sistema?
    - Como podemos garantir a consistência das transações e a segurança dos dados durante o processo de integração?

3 - Quais são os nossos principais riscos?

- Custos
- Implementar uma solução que não resolva o problema e que não seja escalavel
- Parar a operação da Empresa
- Prejudicar nossos clientes no processo de venda de seus produtos em nossa plataforma
- Integrar todas as plataformas externas a nova solução proposta

4 - Crie um plano para aprender o que precisamos para responder a perguntas específicas

- Realizar pesquisas sobre as melhores práticas de arquitetura de microserviços e estratégias de escalabilidade.
- Conduzir análises detalhadas da infraestrutura existente e dos requisitos de integração de pedidos.
- Realizar testes piloto e prototipagem para validar a eficácia das soluções propostas.
- Investir em treinamento e capacitação da equipe para adquirir habilidades necessárias

5 - Crie um plano para reduzir riscos.

- Implementar uma estratégia de transição gradual para minimizar interrupções no serviço.
- Realizar testes abrangentes de integração para identificar e resolver

quaisquer problemas antes da implementação completa.

- Estabelecer comunicação clara e transparente com todas as partes interessadas para garantir o alinhamento de expectativas e mitigar resistências.

6 - Quem são as partes interessadas?

- Logistica
- Faturamento
- Clientes
- Cliente(Final)

7 - O que eles esperam ganhar?

- Uma aplicação que atenda a demanda de pedidos atual e o aumento que possa ocorrer, não prejudicando a operação da empresa e a eficiencia no processo de logistica.

8 - Quem são os usuários?

- Clientes que contratam o nosso serviço para vender seus produtos.
- Cliente Final

9 - O que eles estão tentando realizar?

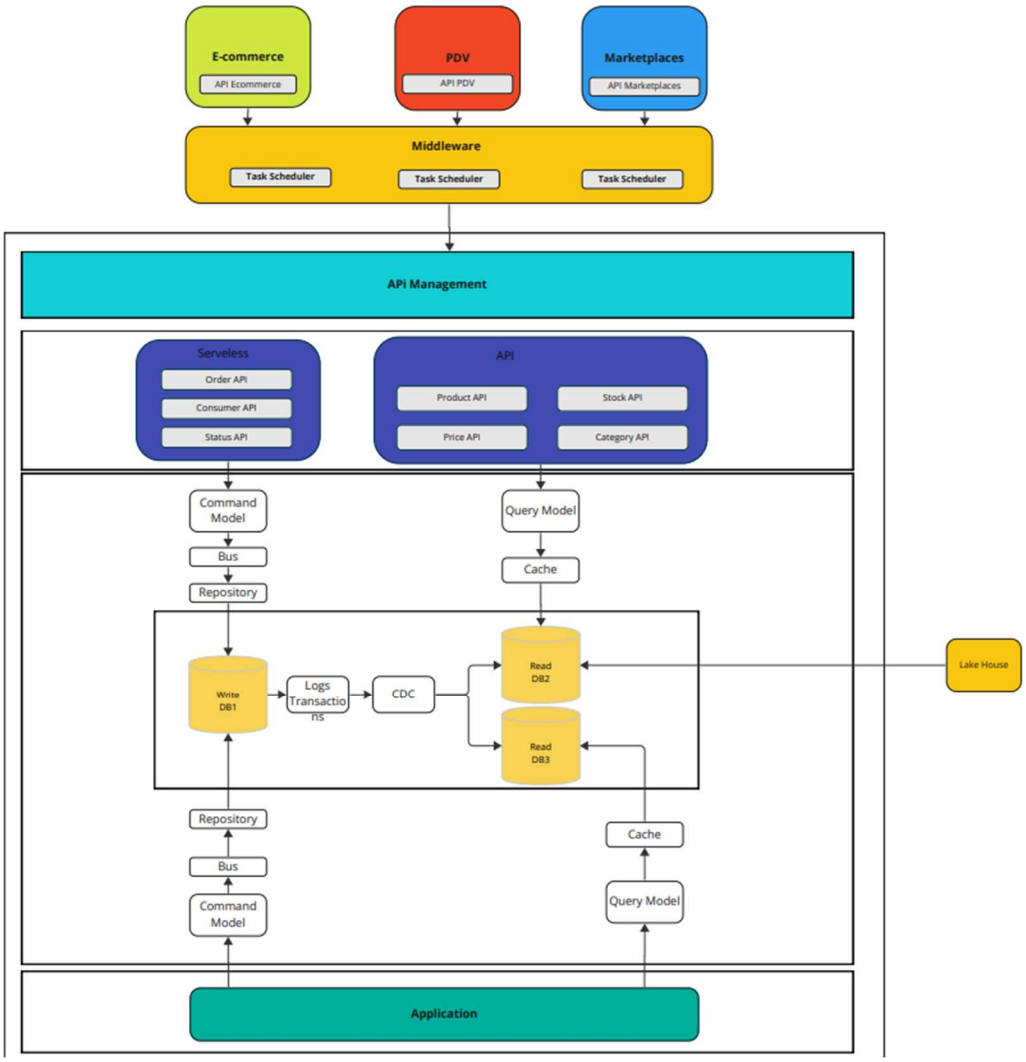
- Expor seus produtos para venda na plataforma
- Comprar os produtos disponivel em nossa plataforma de ecommerce

10 - Qual o pior que pode acontecer?

- Integrar todas as plataformas externas a nova solução proposta

operação da empresa

11 - **Desenhe uma arquitetura;**



12 - **Faça uma descrição de cada um dos componentes que você desenhou;**

- Microserviços:

Os microserviços dividem o sistema em componentes independentes, cada um responsável por uma função específica, como processamento de pedidos, gerenciamento de estoque, gestão de pagamentos, entre outros. Cada microserviço pode ser escalado individualmente para lidar com picos de carga e é mais fácil de manter e evoluir do que um monólito.

- Event-Driven Architecture (EDA):

A EDA baseia-se em eventos e reações a esses eventos. Os eventos são gerados quando algo significativo acontece no sistema, como a criação de um novo pedido, a atualização do estoque ou a confirmação de pagamento.

Os microserviços podem ser projetados para serem acionados por eventos específicos, realizando ações conforme necessário. Isso permite uma comunicação assíncrona entre os componentes, reduzindo a dependência de chamadas síncronas que podem causar gargalos.

- SAGA (Compensating Transactions):

A SAGA é uma técnica para gerenciar transações distribuídas e longas, garantindo a consistência mesmo em caso de falhas em etapas individuais.

No contexto da integração de pedidos, a SAGA pode ser utilizada para coordenar transações que envolvem vários microserviços. Por exemplo, ao processar um pedido, várias etapas podem ser necessárias (verificação de estoque, processamento de pagamento, etc.), e a SAGA garante que todas as etapas sejam executadas com sucesso ou que falhem de maneira consistente, revertendo operações conforme necessário.

- CQRS (Command Query Responsibility Segregation):

O CQRS separa a leitura dos dados (queries) da gravação dos dados (commands), permitindo otimizações específicas para cada tipo de operação.

No contexto da integração de pedidos, o CQRS pode ser aplicado para separar as operações de leitura (consultar informações de pedidos, por exemplo) das operações de escrita (criação, atualização, cancelamento de pedidos). Isso permite otimizar o desempenho e a escalabilidade de cada tipo de operação de forma independente.

- CDC (Change Data Capture)

Change Data Capture (CDC) é uma técnica usada em sistemas de gerenciamento de bancos de dados para rastrear e capturar mudanças nos dados em um banco de dados. A ideia principal é identificar e registrar as alterações feitas nos dados para que elas possam ser propagadas para outros sistemas ou processos de maneira eficiente e em tempo real ou quase em tempo real.

Ao combinar esses quatro conceitos, parar criar uma arquitetura distribuída, reativa a eventos, capaz de lidar com transações complexas e otimizada para operações de leitura e escrita de dados. Cujo objetivo a resolver o problema de gargalo na integração de pedidos no ERP KPL, melhorando a escalabilidade, a confiabilidade e o desempenho do sistema como um todo.

13 - **Descreva requisitos que você considera importante e por quê? (Mínimo 5)**

- Escalabilidade Horizontal: O sistema deve ser capaz de lidar com o aumento do volume de pedidos sem degradação significativa do desempenho. A escalabilidade horizontal permite adicionar mais recursos, como servidores, para distribuir a carga e atender à demanda crescente.
- Desempenho: O sistema deve processar os pedidos de forma rápida e eficiente, garantindo tempos de resposta baixos mesmo durante picos de carga. Isso proporciona uma experiência positiva para os usuários e evita a perda de vendas devido a lentidão no processamento.
- Confiabilidade: O sistema deve ser altamente confiável, garantindo que todos os pedidos sejam processados corretamente e sem falhas. Isso inclui a implementação de mecanismos de recuperação em caso de falhas e a garantia da integridade dos dados.
- Segurança: Deve-se garantir a segurança dos dados durante todo o

processo de integração, incluindo a transmissão de dados entre sistemas e o armazenamento seguro no banco de dados. Isso envolve criptografia de dados, autenticação e autorização adequadas e conformidade com regulamentos de privacidade de dados.

- Monitoramento e Alertas: É essencial ter um sistema de monitoramento em vigor para acompanhar o desempenho do sistema, identificar problemas em potencial e tomar medidas proativas para mitigar problemas antes que impactem os usuários finais. Alertas automatizados podem notificar a equipe de TI sobre eventos críticos que exigem intervenção imediata

14 - **Sobre o que o diagrama ajuda você a raciocinar/pensar?**

- O diagrama ajuda a visualizar a estrutura do sistema, seus componentes, suas interações e como eles se relacionam entre si. Ele auxilia no entendimento do fluxo de dados, na identificação de pontos críticos e na análise da arquitetura como um todo. Além disso, ajuda a comunicar ideias e conceitos complexos de forma clara e concisa para outras partes interessadas.

15 - **Quais são os padrões essenciais no diagrama?**

- Microserviços
- Event-Driven Architecture (EDA)
- SAGA
- CQRS (Command Query Responsibility Segregation)
- CDC(Change Data Capture)

16 - **Existem padrões ocultos?**

Não

17 - **Qual é o Metamodelo?**

- O metamodelo é a estrutura ou modelo conceitual subjacente que define os elementos, relacionamentos e restrições que podem ser representados no diagrama. Ele define as regras e convenções para criar e interpretar o diagrama, fornecendo uma base comum para comunicação e entendimento entre os membros da equipe.

18 - **Pode ser discernido no diagrama único?**

- O metamodelo pode ser parcialmente discernido no diagrama único, dependendo da clareza e da detalhe da representação dos elementos e relacionamentos. No entanto, o metamodelo completo pode exigir uma análise mais aprofundada da estrutura do sistema e das convenções de modelagem utilizadas.

19 - **O diagrama está completo?**

Sim

20 - **Poderia ser simplificado e ainda assim ser eficaz?**

- Sim, o diagrama pode ser simplificado sem comprometer sua eficácia, especialmente se for usado para comunicação de alto nível ou para apresentações para partes interessadas não técnicas. Remover detalhes excessivos e focar nos aspectos mais importantes da arquitetura pode tornar o diagrama mais claro e fácil de entender. No

entanto, é importante garantir que a simplificação não leve à perda de informações importantes ou à distorção do contexto do sistema.

21 - **Houve alguma discussão importante que vocês tiveram como equipe?**

- Sim. Discutimos quais patterns de arquitetura iriam usar e tambem a questão de manipulação dos erros ococrridos na plataforma.

22 - **Que decisões sua equipe teve dificuldade para tomar?**

- Escolha dos patterns de arquitetura que solucionasse o problema**.**

23 - **Que decisões foram tomadas sob incerteza?**

- A sincronização das bases de dados dos microserviços, se todo o fluxo do dados e disponibilidade dos das iriam funcionar.

24 - **Houve algum ponto de decisão sem retorno que o forçou a desistir de um determinado**

- Não

Link repositório github: https://github.com/isalvati/fiap-60aso-Trabalho-Final-Design-de-Arquiteturas