Contexto:
Plataforma de gestão

Propósito do sistema:
Sistema para pequenas empresas de baixo custo e fácil implementação.

Enunciado:
Pequenas empresas com problemas de gerenciamento de estoques, o impacto esperado é a otimização do trabalho interno e facilidade no gerenciamento de estoque.

Requisitos Não Funcionais:
1. Confiabilidade

Explicação: A plataforma deve estar operacional e acessível pelo menos 99,5% do tempo, especialmente durante o horário comercial. Isso significa pouquíssima indisponibilidade não programada.

Justificativa Prática: Se o sistema fica fora do ar com frequência, a empresa não consegue realizar vendas, consultar estoque ou registrar entradas. Um Ponto de Venda travado significa vendas perdidas e clientes insatisfeitos na hora.

2. Performance

Explicação: As operações críticas devem ser rápidas. Por exemplo: consultar um produto no estoque deve levar menos de 2 segundos, e finalizar uma venda (com a baixa no estoque) deve levar menos de 3 segundos.

Justificativa Prática: Em um ambiente de venda, seja no balcão da loja ou no e-commerce, a agilidade é crucial. Lentidão no atendimento gera filas, estresse e má impressão para o cliente. No e-commerce, um site lento aumenta a taxa de abandono do carrinho.

3. Escalabilidade

Explicação: A arquitetura do sistema deve ser capaz de lidar com um aumento gradual no número de usuários, produtos e transações sem uma queda brusca de performance ou a necessidade de uma reengenharia completa.

Justificativa Prática: A pequena empresa de hoje pode ser a média empresa de amanhã. O sistema precisa crescer junto com o negócio, suportando, por exemplo, um pico de vendas durante uma promoção ou a entrada em um novo marketplace sem travar.

4. Usabilidade

Explicação: A interface deve ser intuitiva, requerendo um treinamento mínimo. Operações comuns, como cadastrar um produto ou registrar uma venda, devem ser realizadas com o menor número de cliques possível. A curva de aprendizado deve ser curta.

Justificativa Prática: O usuário final muitas vezes é o próprio dono ou um funcionário sem treinamento técnico. Uma interface complexa leva à rejeição da ferramenta, erros de registro e retorno a métodos manuais ineficientes, anulando o investimento no software.

5. Segurança de Dados

Explicação: O sistema deve implementar criptografia de dados (em trânsito e em repouso), autenticação robusta e backups regulares e automatizados. Deve estar em conformidade com a LGPD.

Justificativa Prática: A plataforma armazena dados sensíveis do negócio: lista de clientes, custos dos produtos, margem de lucro e histórico de vendas. Um vazamento ou perda desses dados por falta de backup seria catastrófico para a continuidade do negócio.

6. Capacidade de Integração

Explicação: O sistema deve ser projetado com APIs bem documentadas e estáveis para permitir a integração com outros sistemas essenciais, como marketplaces, ERPs financeiros e soluções de e-commerce.

Justificativa Prática: Pequenas empresas usam um ecossistema de ferramentas. A falta de uma API robusta torna impossível automatizar a sincronização de estoque entre a loja física e o online, forçando o retrabalho manual e aumentando o risco de erros.

7. Portabilidade e Acessibilidade Móvel

Explicação: A plataforma deve ser acessível de forma funcional e consistente em diferentes dispositivos (computador, smartphone, tablet) e navegadores, sem perda de funcionalidades críticas.

Justificativa Prática: O gestor precisa de mobilidade para verificar o estoque no depósito, aprovar uma ordem de compra em viagem ou mostrar um relatório rápido a um parceiro. Depender exclusivamente de um computador de mesa limita severamente a agilidade operacional.

8. Mantenabilidade e Suportabilidade

Explicação: O sistema deve ser projetado para que correções, atualizações e a adição de pequenas novas funcionalidades possam ser feitas de forma rápida e sem causar grandes interrupções.

Justificativa Prática: Para o fornecedor do software, isso reduz custos de suporte e permite evoluir o produto rapidamente. Para o cliente final (a pequena empresa), significa que os problemas são resolvidos rapidamente e que o sistema se adapta às suas necessidades em constante mudança.

Linguagens, frameworks, bancos de dados, APIs, serviços em nuvem, bibliotecas e ferramentas de integração:
Java, Mockito, Spring(e Spring Boot), Hibernate, MySQL, JDBC

Papéis Técnicos no Grupo:
Gustavo André Klahold(Desenvolvedor, Analista)
Kauã Rieper Ribeiro(Apresentador Técnico)
Heitor Pereira Emmerich(Gestor de Configuração)
Arthur Luiz Segalla Rapozo(Designer)

utilizaremos para o versionamento do nosso projeto o github que usaremos com a pratica de git flow de um jeito simplificado, deixaremos nosso código pronto estável na main e disso criaremos a Branch chamada develop que nela estaria tudo que estamos alterando ou criando, desta branch teremos duas branchs que serão criadas sempre que necessário uma chamada feature que seria todas as novas criações e uma chamada hotfix que são as coisas que precisam ser arrumadas RÁPIDO, depois disso quando terminar a etapa de desenvolvimento, daremos merge nas hotfix e feature para dentro da develop e depois da develop para a main assim concluindo um ciclo de trabalho.
nisso tudo teremos a assistência de um trello que funcionará como um kanban para nossa equipe no qual utilizaremos de forma simples sem muitas colunas somente com backlog, to do, doing e done.