Desafio Técnico - Desenvolvedor Fullstack (Nível Sênior)

Olá!

Obrigado pelo seu interesse em nossa oportunidade. Neste desafio, você construirá uma versão simplificada de uma plataforma de pedidos para um café. A proposta é avaliar sua capacidade técnica, organização de código, boas práticas e clareza de raciocínio.

Este desafio é voltado para desenvolvedores de nível sênior, por isso, esperamos atenção especial a aspectos como: arquitetura, testes, documentação, e qualidade geral do projeto.

Temos como objetivo avaliar suas habilidades técnicas e raciocínio individual. Por isso, pedimos que não utilize ferramentas de inteligência artificial (como ChatGPT ou similares) na elaboração da solução.

Queremos entender seu domínio prático e sua forma de pensar — e não o resultado de assistentes automatizados.

📃 Cenário

A empresa Café Ltda está se digitalizando e passará a oferecer a seus clientes um QR Code para que realizem seus pedidos diretamente pelo celular. Você foi contratado para desenvolver essa aplicação.

🎯 Objetivo

Criar uma aplicação fullstack onde o cliente possa:

- Visualizar o cardápio do café
- Selecionar produtos e personalizá-los com observações e quantidades
- Visualizar o valor total da compra
- Preencher um formulário com dados de pagamento (simulado)
- Registrar a transação

🧩 Requisitos

Funcionalidades mínimas

Frontend

- [ ] Listar produtos do cardápio com nome, imagem e preço
- [ ] Permitir seleção de produtos, com quantidade e observações
- [ ] Calcular e exibir o valor total da compra
- [ ] Formulário de pagamento (não precisa integrar com gateway)
- [ ] Feedback de sucesso após pedido finalizado

Backend

- [ ] Middleware de autenticação (JWT, básica ou outro mecanismo)
- [ ] Endpoints para CRUD de produtos
- [ ] Endpoint para registrar pedido

💻 Stack obrigatória

Você deverá utilizar:

- Frontend: React.js v17+
- Backend: Node.js v16+
  - Express.js ou NestJS
- Banco de dados: MySQL v8+

🧪 Qualidade de código

- [ ] Boas práticas de codificação e componentização
- [ ] Nomenclatura clara e semântica
- [ ] Boas práticas RESTful na API
- [ ] Validações de entrada e resposta com status adequado
- [ ] Consistência na formatação de código
- [ ] Documentação da API (Swagger ou Postman Collection)

📐 System design

- [ ] Elabore um system design que considere robusto e escalável para a solução proposta no mundo real
- [ ] Inclua o desenho da arquitetura ao README

⚠️ Atenção: a codificação deste desafio não deve necessariamente refletir o system design proposto

---

🧠 Modelagem de dados

- [ ] Crie e utilize um modelo relacional no MySQL
- [ ] Inclua diagrama ER e DDL ao README

🧰 Diferenciais (opcional)

Estes itens não são obrigatórios, mas contarão positivamente:

- [ ] TypeScript
- [ ] CI/CD (Bitbucket, Gitlab ou Github)
- [ ] Docker compose
- [ ] Responsividade mobile
- [ ] Uso de design system (MUI, Tailwind, etc)
- [ ] Testes unitários e/ou de integração

📦 Entregáveis

- [ ] Link do repositório com o código-fonte
- [ ] URL pública com a aplicação funcionando ou instruções para rodar o projeto localmente (README)

📊 Critérios de avaliação

- Clareza e organização do código
- Estrutura da aplicação (backend e frontend)
- Qualidade da modelagem de dados
- Boas práticas de desenvolvimento
- UX e usabilidade da interface
- Documentação do projeto

Desejamos sucesso na sua jornada! Se tiver dúvidas, entre em contato com o time de recrutamento. Estamos ansiosos para ver sua solução 🚀