# Teste NodeJS - Pay

**Objetivo**: O desafio é desenvolver um sistema de processamento de pagamentos integrado ao ambiente de homologação do **Asaas**, utilizando **Node.js** para o backend. O cliente deve acessar uma página simples onde poderá selecionar a opção de pagamento entre Boleto, Cartão ou Pix.

**Documentação do Asaas**: https://asaasv3.docs.apiary.io/#

**Credenciais de Sandbox**:
Crie uma conta no Asaas Sandbox (https://sandbox.asaas.com/), e na parte de **Configuração de Conta -> Integrações**, você encontrará a API Key de Sandbox para iniciar a integração.

---

## Requisitos do Sistema:

1. **Backend em Node.js**:
    - Desenvolver o backend utilizando **Node.js** com um framework de sua escolha (preferencialmente **Express.js** ou **NestJS**).
    - O backend será responsável por processar os pagamentos e retornar as informações necessárias para o frontend.

2. **Processamento de Pagamentos**:
    - Implementar as opções de pagamento com **Boleto**, **Cartão de Crédito** e **Pix** integradas ao ambiente de homologação do Asaas.
    - Tratar adequadamente os dados recebidos no corpo da requisição, garantindo validações para evitar dados incompletos ou incorretos.

3. **Respostas Padronizadas**:
    - As respostas das requisições devem seguir um padrão (por exemplo, utilizando o formato JSON), incluindo mensagens claras para erros e sucessos.
    - Em caso de erro na requisição ou recusa do cartão, retornar mensagens amigáveis que expliquem o motivo do problema.

4. **Frontend Simples (Diferencial)**:
    - Criar uma página básica com HTML e Bootstrap (ou utilizar um framework de frontend simples como React, Vue.js ou Angular, caso prefira).
    - Esta página será usada para:
        - Selecionar o método de pagamento.
        - Preencher os dados necessários.

        - Exibir os resultados (como o link do boleto, QRCode ou status do pagamento).

5. **Página de Obrigado (Diferencial)**:
    - Após o processamento do pagamento:
        - Se o pagamento for **Boleto**, exibir um botão com o link para o boleto.
        - Se o pagamento for **Pix**, exibir o QRCode e o código **Copia e Cola**.
        - Em caso de recusa ou erro, exibir uma mensagem amigável explicando o problema.

6. **Boas Práticas de Desenvolvimento**:
    - Seguir boas práticas de codificação e organização de projetos em Node.js.
    - Implementar abstrações e camadas para os serviços de integração com a API do Asaas.
    - Utilizar boas práticas de versionamento de código com Git.

7. **Documentação**:
    - Documentar o projeto com instruções claras de como configurá-lo e rodá-lo localmente.

## Opcionais:

1. **Testes Automatizados**:
    - Escrever testes automatizados (unitários e/ou de integração) para as funcionalidades principais.
    - Informar a cobertura de testes no projeto.
2. **Persistência dos Dados**:
    - Utilizar um banco de dados relacional (preferencialmente **MySQL**) para armazenar as informações de pagamentos processados e logs de transações.