Documento de Arquitetura - Sistema XSA Uniformes

1. Visão Geral do Sistema

O **sistema XSA Uniformes** é um aplicativo desenvolvido para permitir que empresas e usuários solicitem orçamentos personalizados de uniformes, escolham o método de pagamento (Pix ou Cartão de Crédito) e finalizem a compra. O sistema oferece uma interface fácil de usar com diversas etapas: solicitação de orçamento, autenticação, escolha do método de pagamento e confirmação do pagamento.

---

2. Arquitetura do Sistema

A arquitetura do sistema é baseada em uma arquitetura **cliente-servidor**, com a parte de frontend (interface do usuário) comunicando-se com um backend responsável por processar as solicitações e interagir com o banco de dados.

2.1 Componentes Principais

- **Frontend (Cliente)**:
  - Aplicativo web/mobile com as interfaces de usuário.
  - Responsável por interagir com o usuário, coletar informações e enviar dados ao backend.
  - Tecnologias: HTML, CSS, JavaScript, React/Angular (para Web) ou React Native (para Mobile).
- **Backend (Servidor)**:
  - Servidor responsável pela lógica de negócios, processamento de orçamentos e controle de autenticação.
  - Tecnologias: Node.js, Express (para API REST), ou Java com Spring Boot.
  - Integração com gateways de pagamento (Pix, Cartão de Crédito).
- **Banco de Dados**:
  - Banco de dados relacional para armazenar dados dos usuários, orçamentos, transações e contratos.
  - Tecnologias: MySQL, PostgreSQL ou MongoDB (caso prefira banco de dados NoSQL).
- **Serviços de Pagamento**:
  - Integração com provedores de pagamento para processamento de Pix e Cartão de Crédito.
  - Tecnologias: APIs de pagamento (como API do Banco para Pix ou Stripe/PagSeguro para Cartão de Crédito).

---

3. Fluxo de Dados

3.1 Processo de Orçamento

1. **Página Inicial**: O usuário acessa o app e visualiza a descrição da empresa. O botão "Eu quero" inicia o processo de orçamento.
2. **Formulário de Orçamento**: O usuário preenche os dados de quantidade, tamanhos, cores, material, etc. Após preencher, clica no botão "Fazer orçamento" para enviar os dados.
3. **Processamento do Orçamento**: O backend recebe as informações e realiza o cálculo do valor do orçamento. O orçamento é então retornado ao frontend.
4. **Exibição do Orçamento Finalizado**: O orçamento calculado é exibido com a opção de prosseguir para o pagamento ou revisar os dados preenchidos.
5. **Autenticação e Seleção do Método de Pagamento**: O usuário insere seus dados pessoais e escolhe entre pagamento via Pix ou Cartão de Crédito.

3.2 Fluxo de Pagamento

1. **Pagamento via Pix**: O backend gera uma chave de pagamento baseada no valor do orçamento e a exibe ao usuário. O usuário copia a chave e a insere no app de seu banco para realizar o pagamento.
2. **Pagamento via Cartão de Crédito**: O usuário insere as informações do cartão de crédito e o backend processa a transação utilizando a API do provedor de pagamento.
3. **Verificação do Pagamento**: O backend verifica a aprovação ou recusa do pagamento, informando o usuário do status da transação.

3.3 Notificação e Confirmação

- Após o pagamento ser aprovado, o sistema envia um e-mail de confirmação com os detalhes do pedido e contrato.
- Caso o pagamento seja recusado, o usuário é informado e pode tentar novamente com outro método de pagamento.

---

4. Tecnologias Envolvidas

Frontend

- **Tecnologias**: React, React Native (para mobile), HTML, CSS, JavaScript.
- **Frameworks**: React Router (para navegação), Axios (para chamadas API).

Backend

- **Tecnologias**: Node.js com Express ou Java com Spring Boot.
- **Banco de Dados**: MySQL/PostgreSQL (para dados relacionais), ou MongoDB (se preferir NoSQL).

- **Autenticação**: JWT (JSON Web Token) para autenticação e autorização.
- **APIs de Pagamento**:
    - **Pix**: Integração com bancos via API (ou um serviço de pagamento que suporte Pix).
    - **Cartão de Crédito**: Integração com APIs de pagamento como Stripe ou PagSeguro.

Serviços de E-mail

- **Tecnologias**: NodeMailer (para envio de e-mails no backend), Amazon SES ou SendGrid.

---

5. Integrações

5.1 Integração com Gateway de Pagamento

- **Pix**: Integração com o banco ou API de pagamento que suporte a geração de chaves Pix.
- **Cartão de Crédito**: Integração com o **Stripe** ou **PagSeguro** para processar pagamentos via cartão de crédito.

5.2 Integração com Servidor de E-mails

- O sistema envia e-mails de confirmação de pagamento e contrato utilizando um serviço de e-mail como **Amazon SES** ou **SendGrid**.

---

6. Estrutura de Banco de Dados

6.1 Entidades Principais

- **Usuário**: Armazena informações de clientes (nome, e-mail, telefone, etc.).
- **Orçamento**: Armazena dados do orçamento solicitado (quantidade, tamanhos, cores, valores, etc.).
- **Transação de Pagamento**: Armazena informações sobre o pagamento (método de pagamento, status, valor pago, etc.).
- **Contrato**: Armazena os detalhes do contrato gerado para o cliente após o pagamento.

6.2 Relacionamentos

- Um **usuário** pode ter vários **orçamentos**.
- Um **orçamento** pode gerar uma ou mais **transações de pagamento**.
- Cada **transação de pagamento** corresponde a um **contrato**.

---

7. Segurança

7.1 Proteção de Dados Pessoais

- Uso de HTTPS para criptografar todas as comunicações entre o cliente e o servidor.
- Armazenamento seguro de senhas utilizando hashing (bcrypt).
- Autenticação via JWT para garantir que apenas usuários autenticados possam acessar dados sensíveis.

7.2 Proteção no Processo de Pagamento

- Implementação de práticas de segurança para transações financeiras, como verificação de autenticação de dois fatores (2FA) e monitoramento de fraudes.

---

8. Escalabilidade e Manutenção

8.1 Escalabilidade

- O backend será construído de forma modular para permitir que novos recursos sejam adicionados de maneira simples e eficiente.
- O banco de dados será dimensionado para suportar grandes volumes de dados, com possibilidade de escalabilidade horizontal.

8.2 Manutenção

- O sistema será projetado com alta modularidade e testes automatizados para facilitar a manutenção contínua e atualizações do sistema.

---

9. Diagramas de Arquitetura

Diagrama de Componentes:

- O frontend interage com o backend via APIs REST.
- O backend se comunica com o banco de dados para armazenar e recuperar informações.
- O backend também se conecta a APIs de pagamento para processar transações financeiras.