**📌 Documentação do Aplicativo de Assinatura Digital de Notas Fiscais**

**1. Visão Geral**

Este projeto tem como objetivo desenvolver um **aplicativo móvel** para coletar assinaturas digitais e fotos dos destinatários no momento do recebimento de notas fiscais. O sistema funcionará mesmo sem conexão com a internet, armazenando as informações localmente e sincronizando com o servidor quando a rede estiver disponível.

**2. Funcionalidades Principais**

**📱 Aplicativo Mobile (React Native)**

✅ **Listar notas fiscais pendentes** de assinatura.  
✅ **Exibir detalhes da nota fiscal** (número, valor, data, empresa emissora).  
✅ **Capturar assinatura digital** do destinatário diretamente na tela do dispositivo.  
✅ **Tirar uma foto do destinatário** no momento da assinatura (obrigatório).  
✅ **Registrar a localização (GPS)** do local onde a nota foi assinada.  
✅ **Armazenamento offline**, permitindo que a assinatura e a foto sejam salvas localmente se não houver internet.  
✅ **Sincronização automática**, enviando os dados para o servidor assim que a conexão for restabelecida.

**🖥️ Backend (Flask + MySQL)**

✅ **Receber notas fiscais** de um sistema externo via API.  
✅ **Armazenar assinaturas, fotos e localização** no banco de dados.  
✅ **Painel administrativo** para visualizar e baixar as notas assinadas.  
✅ **Histórico e auditoria**, registrando quando e onde a nota foi assinada.

**3. Tecnologias Utilizadas**

**Frontend (App Mobile)**

* **React Native** → Para o desenvolvimento do aplicativo.
* **Expo (opcional)** → Para facilitar a criação e testes do app.
* **AsyncStorage** → Para armazenamento offline dos dados.
* **React Navigation** → Para navegação entre telas.
* **React Native Vision Camera** → Para capturar fotos.
* **React Native Signature Canvas** → Para capturar assinaturas digitais.
* **Geolocation API** → Para capturar latitude e longitude no momento da assinatura.

**Backend**

* **Python + Flask** → Para criar as APIs do sistema.
* **Flask-RESTful** → Para estruturação da API.
* **MySQL** → Para armazenar os dados das notas, assinaturas e fotos.
* **SQLAlchemy** → ORM para manipulação do banco de dados.
* **Flask-CORS** → Para permitir o acesso do app React Native ao backend.

**Infraestrutura**

* **Servidor VPS (ou AWS, Heroku, Render)** → Para hospedar a API Flask.
* **S3 ou Firebase Storage** → Para armazenar as fotos de forma segura.
* **Banco de Dados MySQL (RDS ou DigitalOcean)** → Para manter as informações centralizadas.

**4. Possíveis Expansões Futuras**

🔹 **Integração com outros sistemas de ERP** para obter notas fiscais automaticamente.  
🔹 **Envio automático das notas assinadas por e-mail** para o destinatário e empresa emissora.  
🔹 **Reconhecimento facial (IA) para validar o destinatário** no momento da assinatura.  
🔹 **Suporte para múltiplos usuários e perfis de acesso**.  
🔹 **Versão Web para acessar os dados e baixar relatórios online**.

**5. Desafios Técnicos e Soluções**

| **Desafio** | **Solução** |
| --- | --- |
| Sem internet no momento da entrega | Armazenamento local (AsyncStorage) e sincronização posterior. |
| Garantia de que a foto foi tirada na hora | Uso da câmera nativa sem opção de importar imagens. |
| Segurança dos dados | Criptografia das assinaturas e restrição de acesso aos arquivos. |
| Integração com outro sistema de notas fiscais | Desenvolvimento de uma API que consome os dados automaticamente. |

**6. Fluxo do Sistema**

1️⃣ **O sistema externo envia as notas fiscais para o backend Flask**.  
2️⃣ **O app móvel exibe a lista de notas pendentes para assinatura**.  
3️⃣ **O entregador/destinatário seleciona a nota e realiza a assinatura**.  
4️⃣ **O app captura a foto do destinatário e a localização (GPS)**.  
5️⃣ **Se não houver internet, os dados são armazenados localmente**.  
6️⃣ **Assim que a internet for restabelecida, os dados são enviados para o servidor**.  
7️⃣ **A empresa pode acessar o painel administrativo e baixar as notas assinadas**.

**7. Conclusão**

Este projeto visa otimizar e digitalizar o processo de confirmação de recebimento de notas fiscais, garantindo **segurança, rastreabilidade e eficiência**. A combinação de **React Native e Flask** permite criar uma solução moderna, flexível e escalável.

**📌 Caminho para Desenvolver o Sistema**

**🔹 1. Planejamento e Estrutura Inicial**

✅ **Definir o fluxo do sistema** (já fizemos isso na documentação).  
✅ **Criar um repositório no GitHub/GitLab** para versionamento do código.  
✅ **Escolher os serviços de hospedagem** (banco de dados, API, armazenamento de imagens).  
✅ **Configurar um Trello (ou Notion)** para acompanhar tarefas do projeto.

**🔹 2. Desenvolvimento do Backend (Flask + MySQL)**

O backend será responsável por **armazenar e processar as notas fiscais, assinaturas, fotos e localização GPS**.

**2.1 - Criar o banco de dados**

* Criar o banco de dados MySQL com tabelas:
  + **empresas** (ID, nome, CNPJ, e-mail, telefone).
  + **notas\_fiscais** (ID, número, data emissão, valor, empresa\_id).
  + **assinaturas** (ID, nota\_id, nome\_recebedor, assinatura, foto, latitude, longitude, data\_assinatura).
* Definir relacionamentos e constraints.

**2.2 - Criar o backend Flask**

* Instalar e configurar Flask.
* Criar APIs para:  
  ✅ Receber as notas fiscais (dados importados de outro sistema).  
  ✅ Listar as notas fiscais pendentes para o app.  
  ✅ Receber assinaturas e fotos do app.  
  ✅ Sincronizar assinaturas armazenadas offline.  
  ✅ Permitir login de administradores.

**2.3 - Testar o backend**

* Testar as APIs com **Postman** ou **Insomnia** antes de conectar ao app.

**🔹 3. Desenvolvimento do Aplicativo (React Native + Expo)**

Agora que o backend está pronto, podemos criar o app para coletar assinaturas e fotos.

**3.1 - Configurar o projeto no Expo**

* Criar um novo projeto React Native com Expo.
* Instalar dependências necessárias:  
  ✅ **React Navigation** (para navegação entre telas).  
  ✅ **Axios** (para comunicação com a API Flask).  
  ✅ **React Native Signature Canvas** (para capturar assinatura).  
  ✅ **React Native Vision Camera** (para tirar foto do recebedor).  
  ✅ **AsyncStorage** (para armazenamento offline).  
  ✅ **Geolocation API** (para capturar GPS no momento da assinatura).

**3.2 - Criar as telas do aplicativo**

✅ **Tela de Login** (caso precise login no app).  
✅ **Tela de Listagem de Notas Fiscais** (notas pendentes de assinatura).  
✅ **Tela de Assinatura e Foto** (onde o usuário assina e tira foto).  
✅ **Tela de Histórico** (mostra notas já assinadas e sincronizadas).

**3.3 - Implementar funcionalidades**

✅ Conectar o app ao backend Flask via **Axios**.  
✅ Implementar **captura de assinatura e foto**.  
✅ Implementar **armazenamento offline** para funcionamento sem internet.  
✅ Implementar **sincronização automática** quando a conexão for restabelecida.

**3.4 - Testar o app**

* Testar o aplicativo no **Expo Go** no celular.
* Simular cenários offline para garantir que os dados são armazenados corretamente.
* Enviar assinaturas e fotos para a API Flask e verificar no banco de dados.

**🔹 4. Implantação e Hospedagem**

Agora que o sistema está funcionando, é hora de disponibilizá-lo para uso.

**4.1 - Implantação do Backend**

* Hospedar a API Flask em um **servidor VPS** (DigitalOcean, Linode, AWS, etc.) ou usar **Render/Heroku**.
* Configurar um **banco de dados MySQL na nuvem**.
* Configurar **armazenamento de fotos** no **S3, Firebase Storage ou servidor próprio**.

**4.2 - Publicação do App**

* Criar conta na **Google Play Store e Apple App Store** (caso vá publicar).
* Gerar APK/IPA com Expo EAS e testar em dispositivos reais.
* Lançar o app para uso interno antes de publicar oficialmente.

**🔹 5. Melhorias e Expansões Futuras**

Após a primeira versão estar rodando, podemos adicionar melhorias:  
🔹 **Gerar relatórios com assinaturas e fotos**.  
🔹 **Enviar e-mails automáticos com as notas assinadas**.  
🔹 **Criar um painel web para visualizar as assinaturas e baixar PDFs**.  
🔹 **Otimizar o desempenho do backend e melhorar segurança**.  
🔹 **Adicionar notificações push para avisar sobre novas notas fiscais**.

**📌 Resumo do Caminho a Seguir**

✅ **Passo 1:** Criar o banco de dados MySQL.  
✅ **Passo 2:** Criar o backend Flask e testar APIs.  
✅ **Passo 3:** Criar o app React Native com Expo.  
✅ **Passo 4:** Conectar o app ao backend e testar.  
✅ **Passo 5:** Implantar backend e publicar o app.  
✅ **Passo 6:** Melhorar e expandir o sistema.