**Documentação do Projeto Mobile**

**1. Informações Iniciais**

**Nome do Projeto:** Pescados TF

**Dupla:** Giselly Pereira Gondim **(Nº: 13)** e Guido Vidal de Oliveira **(Nº:15)**

**Turma:** 3º Ano

**Professor:** Eduardo Alves

**2. Visão Geral do Projeto**

Pescados TF é uma aplicação web para produtores de camarão, focada no controle eficiente de viveiros. Permite o cadastro e gerenciamento de viveiros com cálculo automático da área, e gramatura. Conta também com um módulo de biometria para registrar dados como peso, quantidade de camarões e data da medição. Apesar de ainda em desenvolvimento, já oferece funcionalidades essenciais para o monitoramento da produção e crescimento dos camarões, com potencial para expansão futura.

**Problema que Resolve / Necessidade que Atende:**

O Viveiro Rei foi criado para resolver a dificuldade que muitos produtores de camarão enfrentam ao tentar organizar e acompanhar suas produções de forma manual. Informações como tamanho dos viveiros, status de uso e crescimento dos camarões acabam se perdendo ou sendo registradas de forma pouco prática. Com o sistema, tudo isso fica centralizado e acessível digitalmente, trazendo mais praticidade, agilidade e confiança no dia a dia do produtor.

**Público-Alvo:**

* Produtores de camarão que desejam melhorar o controle e monitoramento dos seus viveiros de forma digital.
* Pequenas e médias propriedades aquícolas, buscando praticidade e organização na gestão da produção.

**3. Funcionalidades Implementadas**

**Funcionalidades atuais:**

* Tela de Login
* Tela de listagem (Viveiro e Biometria)
* Crud (Create/Read/Update/Delete) Viveiro
* Crud (Create/Read/Update/Delete) Biometria
* Integração com API
* Banco Supabase
* Logout
* Token de acesso

**Funcionalidades em andamento:**

* Autenticação real
* Cadastro com validação
* Integração completa com banco
* Níveis de Acesso
* Crud Uso diário
* Crud Controle de Estoque
* Crud Funcionário
* Crud Cultivos

**4. Como Executar o Projeto**

**O que preciso fazer para executar seu projeto ?**

**4.1. FlutterFlow (Frontend)**

1. Acesse o projeto no FlutterFlow:

https://app.flutterflow.io/project/teste-hc2tlz

2. Telas disponíveis:

* Login
* Dashboard
* formUpdateViveiro
* formViveiro
* listaViveiro
* formUpdateBiometria
* formBiometria
* listaBiometria

**4.2. API Backend**

1. Clone o repositório:

git clone https://github.com/guidoolliveira/pescadosApi2.git

2. Instale as dependências: npm install

3. Execute a API:

4. Link da API via ngrok (se aplicável): https://lemur-valued-commonly.ngrok-free.app/api

**5. Estrutura do Banco de Dados (Supabase)**

**Tabelas Criadas:**

Tabela: **Viveiro**

* id
* name
* width
* length
* area

Tabela: **Biometria**

* id
* shrimp\_weight
* weight
* quantity
* date

Link do projeto Supabase: https://rnifstnnmmtzuauqdksv.supabase.co

**6. Integração App + API**

**Integração**

A integração do aplicativo foi realizada utilizando três ferramentas principais: **FlutterFlow, Supabase e Ngrok.**

O aplicativo foi desenvolvido no FlutterFlow, uma plataforma low-code que permite criar interfaces e fluxos de navegação. O FlutterFlow se conecta ao Supabase, que funciona como backend, banco de dados em tempo real e também gerencia autenticações e permissões de acesso.

A comunicação entre o aplicativo e o Supabase acontece por meio de requisições HTTP (API REST), permitindo o cadastro, a consulta, a atualização e o gerenciamento de livros, autores, gêneros, idiomas, editoras e usuários de forma organizada e dinâmica.

Durante o desenvolvimento, foi utilizado o Ngrok para criar um túnel seguro que expõe a API localmente, possibilitando que o FlutterFlow se conecte ao Supabase durante os testes em ambiente local.

**Endpoints Utilizados**

● POST /api/auth/login → autenticação de usuários.

● GET /api/viveiros → read de viveiros.

● DELETE /api/viveiros/[id] → delete de viveiros.

● POST /api/viveiros → cadastro de viveiros.

● PUT /api/viveiros/[id] → update de viveiros.

● GET /api/biometrias → read de biometrias.

● DELETE /api/biometrias/[id] → delete de biometrias.

● POST /api/biometrias → biometrias de biometrias.

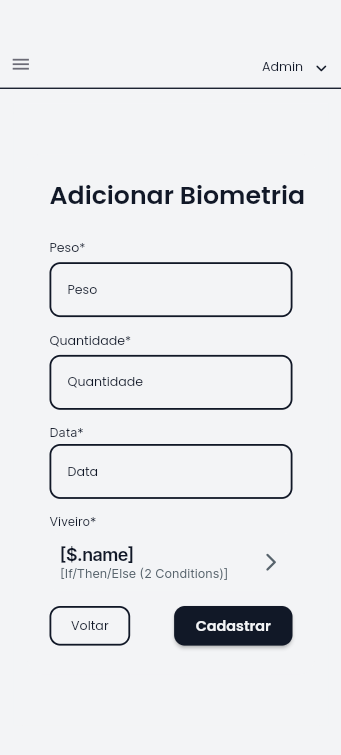
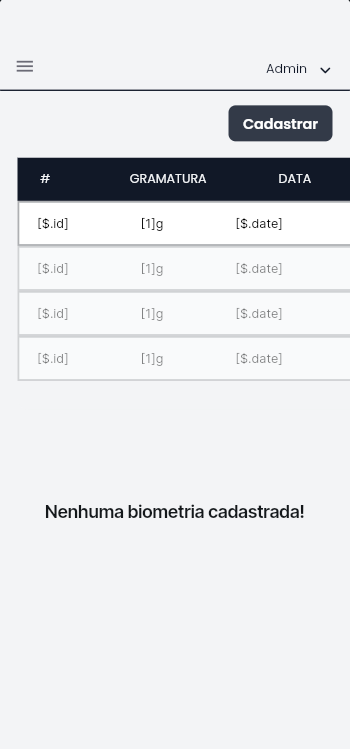
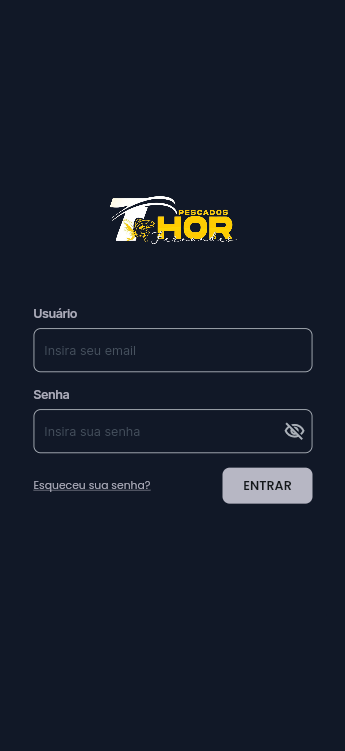
● PUT /api/biometrias/[id] → update de biometrias.

**Segurança**

● Todas as requisições são feitas utilizando HTTPS para garantir a segurança dos dados em trânsito.

● O sistema utiliza autenticação via JWT (JSON Web Token), que protege o acesso às rotas e garante que apenas usuários autenticados possam utilizar os serviços.

**7. Prints do App**



**8. Exportação de Rotas (Postman)**

Arquivo JSON da coleção: api-collection.postman\_collection.jso → Armazenado no GitHub → https://github.com/guidoolliveira/pescadosApi2.git