==================================================  
Contexto Técnico e Detalhado do Projeto: Insight Finder AI  
==================================================  
  
  
1. Objetivo do Projeto  
-----------------------  
  
O objetivo é desenvolver uma Prova de Conceito (PoC) funcional de uma plataforma web chamada "Insight Finder AI". A plataforma servirá como um assistente inteligente para jornalistas e criadores de conteúdo, automatizando tarefas de pesquisa e apuração para aumentar a produtividade e a profundidade das matérias. O projeto será desenvolvido em 4 sprints de 15 dias, totalizando 60 dias de desenvolvimento.  
  
  
2. Funcionalidades Principais (Core Features)  
---------------------------------------------  
  
A plataforma é composta por três módulos principais que se integram:  
  
- Agregador de Notícias via RSS: O sistema coletará notícias de múltiplas fontes de mídia (G1, Folha, Estadão, etc.) através de seus feeds RSS. Os artigos serão armazenados e categorizados em um banco de dados central.  
  
- Sumarizador de Tópicos com IA: Diariamente, a IA processará as notícias coletadas e gerará resumos por categorias (ex: "Resumo do dia em Política"). Isso dará ao jornalista uma visão rápida e consolidada dos acontecimentos mais importantes.  
  
- Assistente de Entrevista com IA:  
 - Transcrição de Áudio: O usuário poderá fazer o upload de um arquivo de áudio de uma entrevista, e a IA irá transcrevê-lo para texto.  
 - Busca Contextual: Após a transcrição, a IA identificará o nome do entrevistado e os principais temas da conversa. O sistema então buscará em toda a sua base de notícias por menções e citações anteriores feitas por essa pessoa sobre os mesmos temas, fornecendo um histórico valioso para o jornalista.  
  
  
3. Principais Desafios Técnicos  
--------------------------------  
  
- Processamento Assíncrono: As tarefas de coleta de notícias, sumarização e transcrição de áudio são computacionalmente intensivas e demoradas. A arquitetura deve garantir que essas tarefas rodem em background (de forma assíncrona) sem nunca bloquear a interface do usuário ou a API principal. A aplicação deve permanecer rápida e responsiva o tempo todo.  
  
- Heterogeneidade de Dados: As fontes de notícias (RSS) possuem estruturas e nomes de categorias diferentes. É necessário criar uma camada de abstração para normalizar e padronizar essas categorias (ex: "g1 > Economia" e "Estadão Economia" devem ser mapeadas para a categoria interna "Economia").  
  
- Gerenciamento de Dados Históricos: Coletar anos de notícias é inviável para uma PoC. A solução será popular o banco de dados com notícias "ao vivo" a partir do início do projeto e, adicionalmente, importar um dataset estático e controlado de notícias antigas sobre um tema específico para simular um arquivo histórico e validar a funcionalidade de busca contextual.  
  
  
4. Arquitetura e Tecnologias Definidas  
---------------------------------------  
  
- Frontend:  
 - Tecnologia: React  
 - Descrição: Uma Single Page Application (SPA) com design responsivo, funcionando como um Progressive Web App (PWA).  
  
- Backend (API Principal):  
 - Tecnologia: NestJS (framework Node.js)  
 - Responsabilidades: Gerenciar API RESTful, autenticação de usuários (JWT), servir dados já processados e, crucialmente, delegar tarefas pesadas para a fila de processamento.  
  
- Sistema de Fila e Agendamento:  
 - Tecnologia: Redis como message broker e a biblioteca BullMQ para gerenciamento da fila em Node.js.  
 - Funcionamento: A API principal adiciona tarefas (jobs) na fila do Redis. Um CRON Job será configurado para disparar as tarefas de coleta e sumarização diária.  
  
- Worker (Serviço de Processamento em Background):  
 - Tecnologia: Um processo Node.js separado.  
 - Funcionamento: Este serviço roda de forma independente (em seu próprio container Docker). Ele se conecta ao Redis, "escuta" a fila, pega as tarefas uma a uma, executa o trabalho pesado (chama os containers das IAs, processa os dados) e atualiza o banco de dados com os resultados. Ele é o "operário" do sistema.  
  
- Inteligência Artificial (Containerizada):  
 - Modelo de Transcrição: Whisper (modelo `base` ou `small`) rodando em um container Docker.  
 - Modelo de Análise de Texto: Flan-T5 (modelo `base`) rodando em um container Docker para sumarização e extração de entidades (NER).  
  
- Banco de Dados:  
 - Tecnologia: PostgreSQL  
  
  
5. Estrutura do Banco de Dados (Schema SQL)  
--------------------------------------------  
  
Abaixo está a estrutura de tabelas planejada para o PostgreSQL.  
  
```sql  
-- Tabela para gerenciar os usuários da plataforma  
CREATE TABLE users (  
 id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),  
 name VARCHAR(255) NOT NULL,  
 email VARCHAR(255) UNIQUE NOT NULL,  
 password\_hash VARCHAR(255) NOT NULL,  
 created\_at TIMESTAMP WITH TIME ZONE DEFAULT timezone('America/Sao\_Paulo', now()),  
 updated\_at TIMESTAMP WITH TIME ZONE DEFAULT timezone('America/Sao\_Paulo', now())  
);  
  
-- Tabela para armazenar as fontes de notícias (RSS feeds)  
CREATE TABLE sources (  
 id SERIAL PRIMARY KEY,  
 name VARCHAR(255) NOT NULL, -- Ex: 'G1', 'Folha de S.Paulo'  
 rss\_url VARCHAR(255) UNIQUE NOT NULL,  
 is\_active BOOLEAN DEFAULT true  
);  
  
-- Nossas categorias internas e padronizadas  
CREATE TABLE global\_categories (  
 id SERIAL PRIMARY KEY,  
 name VARCHAR(100) UNIQUE NOT NULL -- Ex: 'Política', 'Economia', 'Tecnologia'  
);  
  
-- Tabela para mapear as categorias "sujas" de cada fonte para as nossas categorias globais  
CREATE TABLE category\_mappings (  
 id SERIAL PRIMARY KEY,  
 source\_id INTEGER REFERENCES sources(id) ON DELETE CASCADE,  
 source\_category\_name VARCHAR(255) NOT NULL, -- O nome exato que vem no RSS. Ex: 'g1 > Economia'  
 global\_category\_id INTEGER REFERENCES global\_categories(id) ON DELETE CASCADE,  
 UNIQUE(source\_id, source\_category\_name)  
);  
  
-- Tabela para armazenar os artigos coletados  
CREATE TABLE articles (  
 id SERIAL PRIMARY KEY,  
 source\_id INTEGER REFERENCES sources(id),  
 global\_category\_id INTEGER REFERENCES global\_categories(id),  
 title TEXT NOT NULL,  
 link TEXT UNIQUE NOT NULL,  
 content TEXT,  
 publication\_date TIMESTAMP WITH TIME ZONE,  
 created\_at TIMESTAMP WITH TIME ZONE DEFAULT timezone('America/Sao\_Paulo', now())  
);  
  
-- Tabela para guardar os resumos diários gerados pela IA  
CREATE TABLE daily\_summaries (  
 id SERIAL PRIMARY KEY,  
 global\_category\_id INTEGER REFERENCES global\_categories(id),  
 summary\_date DATE NOT NULL,  
 summary\_content TEXT NOT NULL,  
 created\_at TIMESTAMP WITH TIME ZONE DEFAULT timezone('America/Sao\_Paulo', now()),  
 UNIQUE(global\_category\_id, summary\_date)  
);  
  
-- Tabela para as transcrições de entrevistas dos usuários  
CREATE TABLE transcriptions (  
 id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),  
 user\_id UUID REFERENCES users(id) ON DELETE CASCADE,  
 original\_audio\_filename VARCHAR(255),  
 status VARCHAR(50) DEFAULT 'PENDING', -- PENDING, PROCESSING, COMPLETED, FAILED  
 transcribed\_text TEXT,  
 extracted\_entities JSONB, -- Para salvar nomes, empresas, etc., que a IA identificar  
 created\_at TIMESTAMP WITH TIME ZONE DEFAULT timezone('America/Sao\_Paulo', now()),  
 completed\_at TIMESTAMP WITH TIME ZONE  
);