**Documentação do Sistema de Recomendação de Filmes com Supabase e Gemini**

1. Objetivo do Projeto

O objetivo deste projeto é construir um sistema de recomendação de filmes que utiliza:

* Supabase: para armazenar e consultar dados de filmes.
* Gemini 2.0 Flash (Google Generative AI): para interpretar perguntas dos usuários em linguagem natural e gerar respostas com base no banco de dados.

O sistema permite que o usuário faça perguntas abertas, como por exemplo:

* "Filmes com Tom Cruise"
* "Quero filmes de aventura com nota acima de 7"
* "Me recomende algo dirigido por Peter Hastings com um cachorro policial"
* "Me vê filmes com Jason Momoa"
* "Me fala filmes do diretor David Ayer"
* "Me fale bons filmes com avaliações maiores que 8"
* "Me fale um filme dirigido por Lowell Dean, com Steven Ogg no elenco e a avaliação maior que 4"

A IA então interpreta a intenção do usuário e recomenda filmes relevantes que estão armazenados no banco.

1. Tecnologias utilizadas

* Python 3.11+
* Supabase (com SDK supabase-py)
* Google Generative AI (modelo gemini-1.5-flash)
* Terminal (modo de execução via linha de comando)

1. Estrutura do Script

3.1. Variáveis de configuração

O script utiliza três constantes para configurar os acessos:

SUPABASE\_URL = "[https://SEU-PROJETO.supabase.co](https://seu-projeto.supabase.co)" SUPABASE\_KEY = "SUA\_CHAVE\_SUPABASE" GEMINI\_API\_KEY = "SUA\_CHAVE\_GEMINI"

Essas chaves devem ser definidas com os valores do seu projeto no Supabase e na Google AI Platform.

3.2. Inicialização dos serviços

* Supabase é inicializado com create\_client
* Gemini é configurado com genai.configure(api\_key=...)

3.3. Função principal: recomendar\_com\_gemini(pergunta\_usuario, filmes)

Responsável por montar um prompt para a IA Gemini e retornar a resposta.

Entrada:

* pergunta\_usuario: string com o texto digitado pelo usuário
* filmes: lista de dicionários contendo os dados dos filmes do banco

Processamento:

* Concatena os filmes em uma string contendo título, nota, diretor, elenco, gênero e sinopse
* Monta um prompt em linguagem natural contendo a pergunta e a lista de filmes
* Envia o prompt ao modelo Gemini, pedindo que ele interprete e recomende com base nos dados fornecidos

Saída:

* Texto gerado pela IA como resposta à pergunta

3.4. Fluxo principal: responder\_pergunta\_usuario(pergunta)

Esta função executa o seguinte:

* Verifica se a conexão com o banco está ativa
* Carrega os 200 filmes com maior nota do Supabase
* Envia a pergunta e os dados dos filmes para a IA via recomendar\_com\_gemini
* Retorna o texto gerado como resposta

3.5. Execução direta no terminal

A parte final do script executa o seguinte:

if **name** == "**main**": pergunta = input("O que você gostaria de assistir?\n> ") resposta = responder\_pergunta\_usuario(pergunta) print("\nRecomendações da IA:\n") print(resposta)

1. Dados esperados no Supabase

A tabela Filme deve conter os seguintes campos:

* titulo: texto (obrigatório)
* avaliacaoMedia: numérico (float, ex: 7.4)
* diretor: texto
* elenco: array de texto (TEXT[])
* genero: array de texto (TEXT[])
* sinopse: texto

1. Comportamento Inteligente da IA

A IA é instruída a interpretar livremente a pergunta do usuário. Exemplos:

* Identifica atores ou diretores citados no campo correspondente
* Considera valores de nota se citados pelo usuário (ex: "nota 7")
* Interpreta temas ou descrições presentes apenas na sinopse (ex: "filme com cachorro policial")
* Gera respostas naturais, úteis e sem inventar filmes fora da lista

1. Melhorias realizadas no código

Durante a evolução do projeto, foram feitas as seguintes alterações importantes:

* Substituição de regex e filtros manuais por interpretação livre pela IA
* Envio estruturado de campos relevantes (diretor, elenco, gênero, sinopse) no prompt
* Ajuste do prompt para orientar a IA a interpretar com base em todos os campos
* Aumento do limite de filmes enviados à IA de 50 para 200
* Remoção de funções de parsing por regex, tornando o código mais limpo e confiável

1. Possibilidades futuras de extensão

* Adicionar paginação ou busca por lotes para grandes bancos
* Incluir campos adicionais como ano, país, duração
* Criar uma interface web ou API para integração com frontend
* Permitir feedback do usuário (“Gostei / Não gostei”)

1. Exemplo de prompt gerado (parcial)

O usuário perguntou: "Filmes com avaliação 7"

Filmes disponíveis no banco de dados:

* Título: Exemplo (7.1) Diretor: Fulano Elenco: Ator A, Ator B Gênero: Drama, Aventura Sinopse: Uma emocionante história sobre amizade...

Sua tarefa é responder à pergunta do usuário de forma inteligente e relevante.