

Claud.IA

By
NTJ.TECH



PROBLEMA

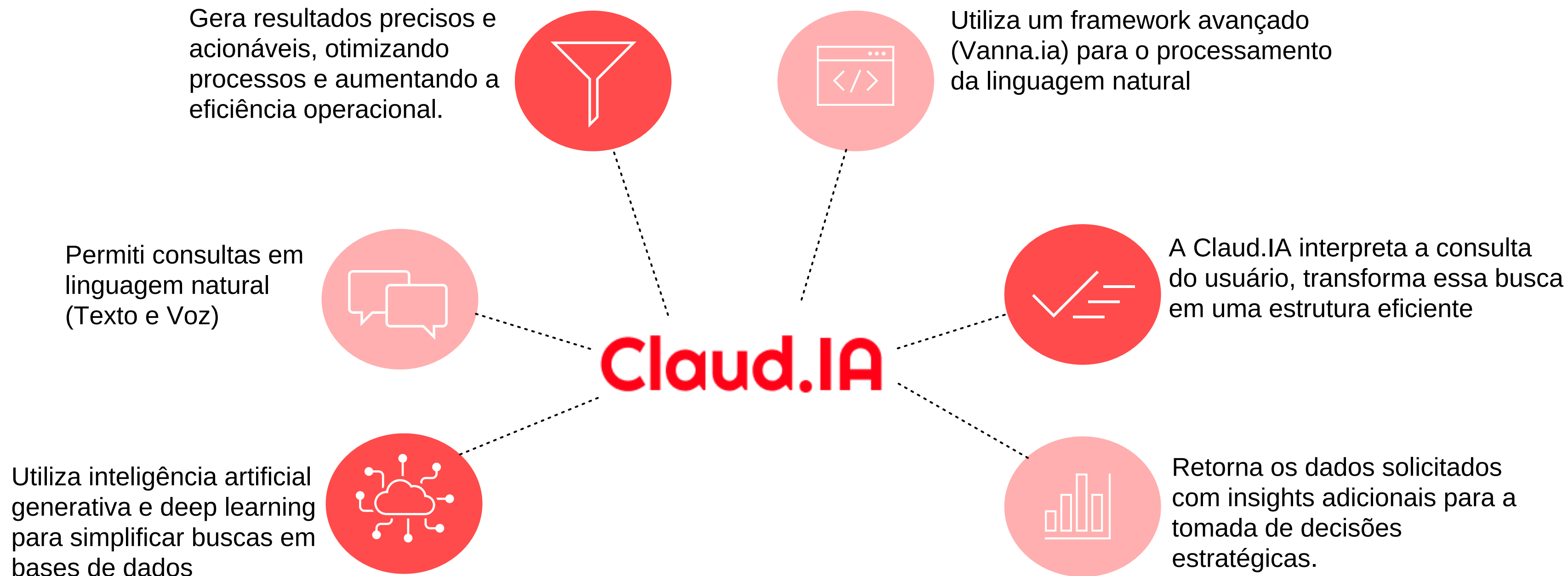


Consultas Complexas

Alto nível de habilidades técnicas

Tempo necessário para escrever e executar consultas

Impacta diretamente na agilidade de tomada de decisões



SOLUÇÃO



Treinamento

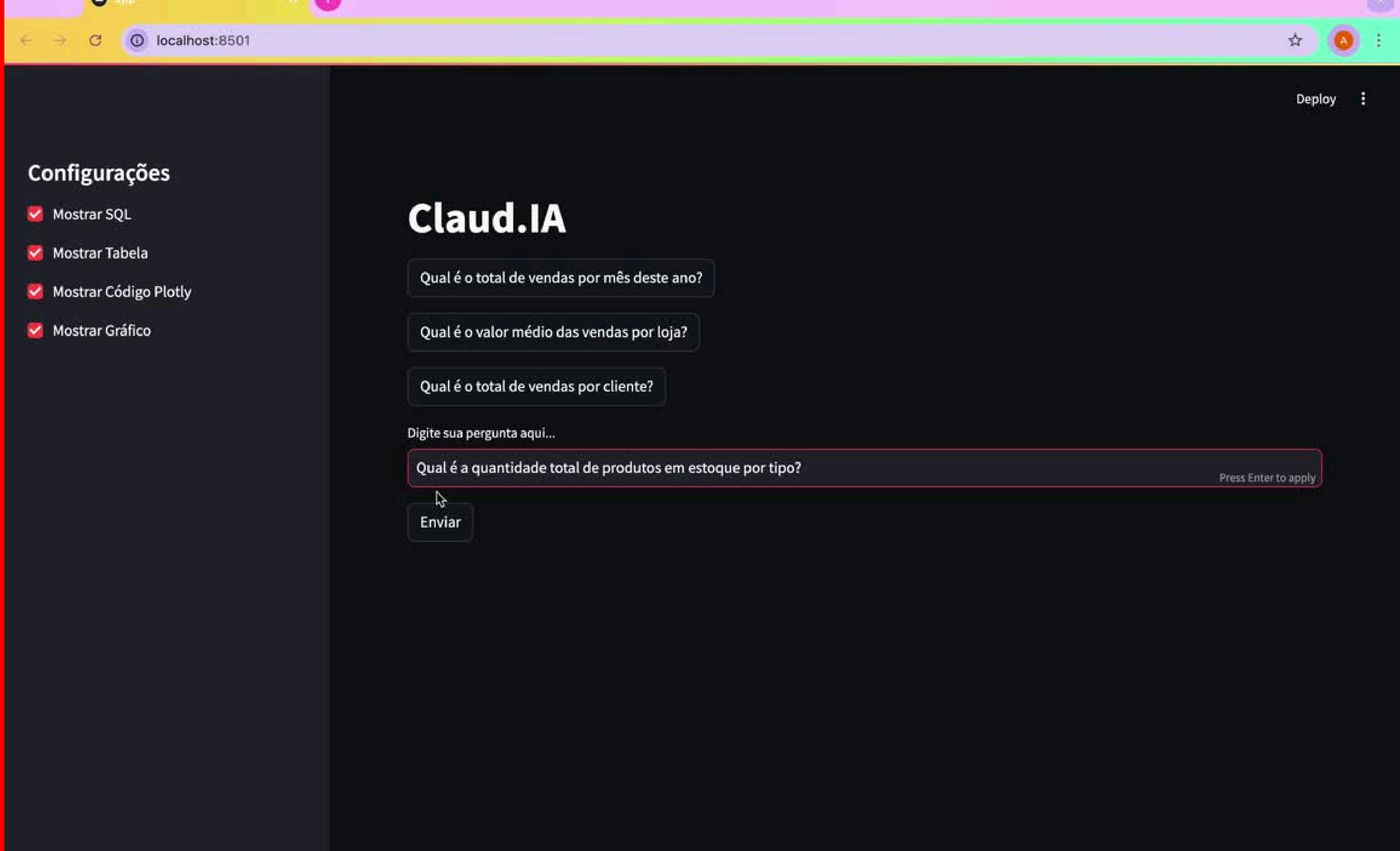
train_vanna_model.py

```
model_training > train_vanna_model.py > ...
165     DATA_REEMBOLSO DATE,
166     VALOR_REEMBOLSO NUMBER,
167     STATUS_REEMBOLSO VARCHAR2(20),
168     DESCRICAO VARCHAR2(255),
169     FOREIGN KEY (ID_PEDIDO) REFERENCES Pedido(ID_PEDIDO)
170 );
171
172 -- Tabela PedidoCancelamento
173 CREATE TABLE PedidoCancelamento (
174     ID_CANCELAMENTO NUMBER PRIMARY KEY,
175     ID_PEDIDO NUMBER,
176     DATA_CANCELAMENTO DATE,
177     STATUS_CANCELAMENTO VARCHAR2(20),
178     DESCRICAO VARCHAR2(255),
179     FOREIGN KEY (ID_PEDIDO) REFERENCES Pedido(ID_PEDIDO)
180 );
181
182 -- Tabela Estoque
183 CREATE TABLE Estoque (
184     ID_ESTOQUE NUMBER PRIMARY KEY,
185     ID_PRODUTO NUMBER,
186     TIPO_PRODUTO VARCHAR2(20),
187     QUANTIDADE NUMBER,
188     ID_LOJA NUMBER,
189     FOREIGN KEY (ID_LOJA) REFERENCES Loja(ID_LOJA)
190 );
191
192 -- Tabela PedidoProduto
193 CREATE TABLE PedidoProduto (
194     ID_PEDIDO_PRODUTO NUMBER PRIMARY KEY,
195     ID_PEDIDO NUMBER,
196     ID_PRODUTO NUMBER,
197     TIPO_PRODUTO VARCHAR2(20),
198     QUANTIDADE NUMBER,
199     VALOR_UNITARIO NUMBER,
200     FOREIGN KEY (ID_PEDIDO) REFERENCES Pedido(ID_PEDIDO)
201 );
202
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

zsh + ...

anajuliavianna@Anas-MacBook-Air vanna_project Sprint3 %



←

→

↺


🔒

localhost:8501

☆

A

⋮

 RUNNING... Stop Deploy ⋮

Configurações


- ✓ Mostrar SQL
- ✓ Mostrar Tabela
- ✓ Mostrar Código Plotly
- ✓ Mostrar Gráfico


Claud.IA

Qual é o total de vendas por mês deste ano?

Qual é o valor médio das vendas por loja?

Qual é o total de vendas por cliente?

 Qual é o valor médio das vendas por loja?



```
SELECT l.NOME_LOJA, AVG(p.VALOR_TOTAL) AS VALOR_MEDIO_VENDA
FROM Pedido p
JOIN Loja l ON p.ID_LOJA = l.ID_LOJA
GROUP BY l.NOME_LOJA
ORDER BY VALOR_MEDIO_VENDA DESC
```


Modelo Reebendo Pergunta

EXPLORER

... vanna_calls.py X

VANNA-STREAMLIT-MAIN

> __pycache__

> .streamlit

> assets

> venv

> .gitignore

app.py

Chinook.sqlite

LICENSE

README.md

requirements.txt

vanna_calls.py

> OUTLINE

> TIMELINE

vanna_calls.py > ...

```
1 import streamlit as st
2 from vanna.remote import VannaDefault
3 import jaydebeapi
4
5 def setup_vanna():
6     # Conectar ao Vanna
7     vn = VannaDefault(api_key=st.secrets.get("VANNA_API_KEY"), model=st.secrets.get("VANNA_MODEL"))
8
9     # Configurar a conexão com Oracle
10    dsn = f"{st.secrets.get('DB_HOST')}:{st.secrets.get('DB_PORT')}/{st.secrets.get('DB_SERVICE')}"
11    vn.connect_to_oracle(dsn=dsn, user=st.secrets.get("DB_USER"), password=st.secrets.get("DB_PASSWORD"))
12
13    return vn
14
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

o (venv) anajuliavianna@Anas-MacBook-Air vanna-streamlit-main % streamlit run app.py

You can now view your Streamlit app in your browser.

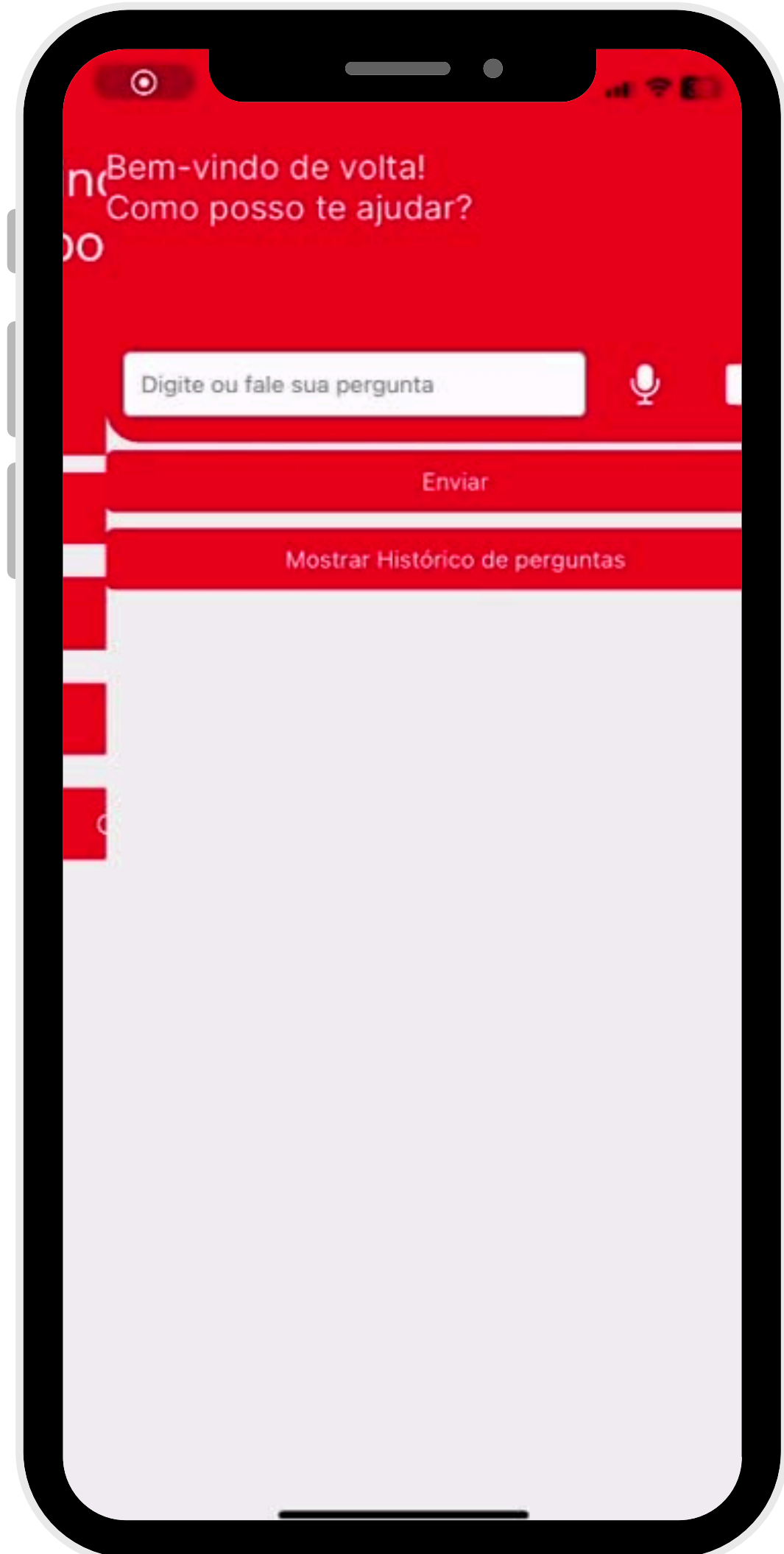
Local URL: <http://localhost:8503>

Network URL: <http://172.16.73.28:8503>

SQL Prompt: [{'role': 'system', 'content': "You are a SQL expert. Please help to generate a SQL query to answer the question. Your response should ONLY be based on the given context and follow the response guidelines and format instructions. \n===Tables \n\n-- Tabela Endereco\nCREATE TABLE Endereco (\n ID_ENDERECO NUMBER PRIMARY KEY,\n RUA VARCHAR2(100),\n NUMERO VARCHAR2(10),\n COMPLEMENTO VARCHAR2(50),\n BAIRRO VARCHAR2(50),\n CIDADE VARCHAR2(50),\n ESTADO VARCHAR2(2),\n CEP VARCHAR2(10)\n);\n\n-- Tabela Loja\nCREATE TABLE Loja (\n ID_LOJA NUMBER PRIMARY KEY,\n NOME_LOJA VARCHAR2(100),\n ENDERECO NUMBER,\n FOREIGN KEY (ENDERECO) REFERENCES Endereco(ID_ENDERECO)\n);\n\n-- Tabela Cliente\nCREATE TABLE Cliente (\n ID_CLIENTE NUMBER PRIMARY KEY,\n NOME_CLIENTE VARCHAR2(100),\n EMAIL VARCHAR2(100),\n TELEFONE VARCHAR2(15),\n ENDERECO NUMBER,\n FOREIGN KEY (ENDERECO) REFERENCES Endereco(ID_ENDERECO)\n);\n\n-- Tabela Vendedor\nCREATE TABLE Vendedor (\n ID_VENDEDOR NUMBER PRIMARY KEY,\n NOME_VENDEDOR VARCHAR2(100),\n HORARIO_ENTRADA VARCHAR2(5),\n HORARIO_SAIDA VARCHAR2(5),\n SALARIO NUMBER\n);\n\n-- Tabela Blusa\nCREATE TABLE Blusa (\n ID_BLUSA NUMBER PRIMARY KEY,\n NOME_BLUSA VARCHAR2(100),\n DESCRICAO VARCHAR2(255),\n PRECO NUMBER,\n MARCA VARCHAR2(50),\n COR VARCHAR2(20),\n TAMANHO VARCHAR2(10)\n);\n\n-- Tabela Calca\nCREATE TABLE Calca (\n ID_CALCA NUMBER PRIMARY KEY,\n NOME_CALCA VARCHAR2(100),\n DESCRICAO VARCHAR2(255),\n PRECO NUMBER,\n MARCA VARCHAR2(50),\n COR VARCHAR2(20),\n TAMANHO VARCHAR2(10)\n);\n\n-- Tabela Saia\nCREATE TABLE Saia (\n ID_SAIA NUMBER PRIMARY KEY,\n NOME_SAIA VARCHAR2(100),\n DESCRICAO VARCHAR2(255),\n PRECO NUMBER,\n MARCA VARCHAR2(50),\n COR VARCHAR2(20),\n TAMANHO VARCHAR2(10)\n);\n\n-- Tabela Acessorio\nCREATE TABLE Acessorio (\n ID_ACESSORIO NUMBER PRIMARY KEY,\n NOME_ACESSORIO VARCHAR2(100),\n DESCRICAO VARCHAR2(255),\n PRECO NUMBER,\n MARCA VARCHAR2(50),\n TIPO VARCHAR2(50)\n);\n\n-- Tabela Bolsa\nCREATE TABLE Bolsa (\n ID_BOLSA NUMBER PRIMARY KEY,\n NOME_BOLSA VARCHAR2(100),\n DESCRICAO VARCHAR2(255),\n PRECO NUMBER,\n MARCA VARCHAR2(50),\n COR VARCHAR2(20),\n TIPO VARCHAR2(50)\n);\n\n-- Tabela Perfume\nCREATE TABLE Perfume (\n ID_PERFUME NUMBER PRIMARY KEY,\n NOME_PERFUME VARCHAR2(100),\n DESCRICAO VARCHAR2(255),\n PRECO NUMBER,\n MARCA VARCHAR2(50),\n FRAGRANCIA VARCHAR2(50)\n);\n\n-- Tabela Pedido\nCREATE TABLE Pedido (\n ID_PEDIDO NUMBER PRIMARY KEY,\n DATA_PEDIDO DATE,\n HORARIO_PEDIDO VARCHAR2(5),\n VALOR_TOTAL NUMBER,\n ID_CLIENTE NUMBER,\n ID_VENDEDOR NUMBER,\n ID_LOJA NUMBER,\n TIPO_PEDIDO VARCHAR2(20),\n STATUS_PEDIDO VARCHAR2(20),\n FOREIGN KEY (ID_CLIENTE) REFERENCES Cliente(ID_CLIENTE),\n FOREIGN KEY (ID_VENDEDOR) REFERENCES Vendedor(ID_VENDEDOR),\n FOREIGN KEY (ID_LOJA) REFERENCES Loja(ID_LOJA)\n);\n\n-- Tabela PedidoTroca\nCREATE TABLE PedidoTroca (\n ID_TROCA NUMBER PRIMARY KEY,\n ID_PEDIDO NUMBER,\n DATA_TROCA DATE,\n STATUS_TROCA VARCHAR2(20),\n DESCRICAO VARCHAR2(255),\n FOREIGN KEY (ID_PEDIDO) REFERENCES Pedido(ID_PEDIDO)\n);\n\n-- Tabela PedidoDevolucao\nCREATE TABLE PedidoDevolucao (\n ID_DEVOLUCAO NUMBER PRIMARY KEY,\n ID_PEDIDO NUMBER,\n DATA_DEVOLUCAO DATE,\n STATUS_DEVOLUCAO VARCHAR2(20),\n DESCRICAO VARCHAR2(255),\n FOREIGN KEY (ID_PEDIDO) REFERENCES Pedido(ID_PEDIDO)\n);\n\n-- Tabela PedidoReembolso\nCREATE TABLE PedidoReembolso (\n ID_REEMBOLSO NUMBER PRIMARY KEY,\n ID_PEDIDO NUMBER,\n DATA_REEMBOLSO DATE,\n VALOR_REEMBOLSO NUMBER,\n STATUS_REEMBOLSO VARCHAR2(20),\n DESCRICAO VARCHAR2(255),\n FOREIGN KEY (ID_PEDIDO) REFERENCES Pedido(ID_PEDIDO)\n);\n\n-- Tabela PedidoCancelamento\nCREATE TABLE PedidoCancelamento (\n ID_CANCELAMENTO NUMBER PRIMARY KEY,\n ID_PEDIDO NUMBER,\n DATA_CANCELAMENTO DATE,\n STATUS_CANCELAMENTO VARCHAR2(20),\n DESCRICAO VARCHAR2(255),\n FOREIGN KEY (ID_PEDIDO) REFERENCES Pedido(ID_PEDIDO)\n);\n\n-- Tabela Estoque\nCR

Ln 85, Col 1 Spaces: 4 UTF-8 LF {} Python 3.12.2 ('venv': venv)

Versão Final Mobile



```
LOG Resposta da API: {"success": true, "user": {"cpf": "47017614899", "id": 106006, "senha": "2281"}}
LOG Login realizado com Sucesso.
LOG Gravação iniciada
LOG Gravação parada e salva em file:///var/mobile/Containers/Data/Application/5A47ABCC-7CBC-4BCB-909A-F6975EA800F0/Library/Caches/ExponentExperienceData/@
anonymous/MobileChallenge-bb0dc05d-0a53-4b53-b266-c170f56eec0b/AV/recording-AB57D25E-B79D-4E5F-84D0-89DBC84AEB59.m4a
LOG Transcrição do áudio: Qual é o total de vendas por mês deste ano
LOG Enviando pergunta: Qual é o total de vendas por mês deste ano?
LOG Resposta do servidor:
```

Pontos Altos

- **Conexão com o Banco de Dados Oracle:** Integração eficiente para consultas diretas no banco de dados.
- **Treinamento do Modelo com DDL, Queries e Documentação:** O modelo foi treinado utilizando comandos DDL, consultas SQL e documentações específicas, garantindo maior precisão e contextualização nas respostas.
- **Interpretação de Perguntas em Linguagem Natural:** Identificação e interpretação das perguntas do usuário em linguagem natural, convertendo-as em consultas SQL.
- **Consulta Direta no Banco de Dados:** Consultas são realizadas diretamente no banco de dados, com o retorno de dados atualizado em tempo real.
- **Visualização dos Resultados e Dashboards:** O resultado das queries é apresentado junto a gráficos e dashboards para facilitar a interpretação dos dados.
- **Aplicação Web Intuitiva com Streamlit:** Interface simplificada e fácil de usar, permitindo uma navegação rápida e intuitiva.
- **Aplicação Mobile com Insights pela OpenAI:** A versão mobile permite que o usuário obtenha insights adicionais com a API da OpenAI, respondendo perguntas sobre os resultados das queries.
- **Funcionalidade de Áudio na Aplicação Mobile:** Além de digitar, o usuário pode enviar perguntas por áudio, facilitando ainda mais a interação.
- **Ganho de Conhecimento em IA Generativa:** O projeto proporcionou um aprendizado significativo sobre tecnologias de IA generativa e sua aplicação prática.

Pontos de Melhoria

- **Limitação de Acesso Direto da OpenAI ao Banco de Dados:** Atualmente, a OpenAI não se conecta diretamente ao banco de dados como o Vanna.AI. Para obter insights, o usuário precisa primeiro visualizar o resultado da query e, em seguida, enviar uma pergunta na aba de insights, onde a OpenAI está integrada.
- **Limitação de Uso do Vanna.AI:** O Vanna.AI possui um limite de testes gratuito. Para utilizá-lo sem restrições, seria necessário contratar uma assinatura. Como alternativa, poderíamos explorar uma solução que utilizasse exclusivamente a OpenAI para geração de queries.

Futuramente

