Plano de Projeto - MVP com IA para Acupuntura

Objetivo Geral

Criar uma aplicação local em Python para gravação de sessões entre acupunturista e paciente, com transcrição automática, separação por falante (diarização) e organização da informação em ficha clínica.

Etapas do Projeto

Etapa 1 – Estrutura Inicial do Projeto

Organizar diretórios e arquivos:

```
acupuntura_mvp/
├── audio/                # Áudios gravados
├── transcricao/          # Transcrições brutas
├── output/               # Fichas geradas
├── gui/                  # Código da interface gráfica
├── models/               # Modelos e LLMs
├── scripts/              # Utilitários
├── requirements.txt      # Dependências
└── main.py               # Script principal
```

Etapa 2 – Interface Gráfica com Gravação, Pausa e Retomada

Nova etapa! Interface Tkinter simples e amigável com:

- Campo para nome do paciente
- Botões: Iniciar, Pausar, ▶ Retomar, Parar
- Status: "Gravando...", "Pausado", tempo corrido

Lógica: cada trecho entre Iniciar e Pausar é salvo separadamente. Após o clique em Parar, todos são concatenados em um único `.wav`.

Etapa 3 – Transcrição com Whisper (com instalação)

- **Instalar:**

```
pip install git+https://github.com/openai/whisper.git
```

- **Requer:** Python 3.8+, Git, ffmpeg (`sudo apt install ffmpeg`)
- **Executar:**

```
whisper audio/arquivo.wav --language Portuguese --model medium
```

- Resultado: arquivos `.txt`, `.srt`, `.json` na mesma pasta

Etapa 4 – Diarização de Falantes com pyannote.audio (com instalação)

- **Instalar:**

```
pip install pyannote.audio
```

- **Pré-requisitos:** conta na Hugging Face + token + `huggingface-cli login`
- **Exemplo:**

```
from pyannote.audio import Pipeline
pipeline = Pipeline.from_pretrained("pyannote/speaker-diarization", u
diarization = pipeline("audio/arquivo.wav")
with open("transcricao/arquivo.rttm", "w") as f:
    diarization.write_rttm(f)
```

- Saída: segmentos com falantes (speaker_1, speaker_2)

Etapa 5 – Organização da Ficha Clínica

Transformar a transcrição em formato estruturado:

```
{
  "nome": "Pedro Silva",
  "data": "2025-06-22",
  "queixa_principal": "Dores nas costas",
  "respostas_do_paciente": [...],
  "observacoes": "Paciente com histórico de estresse."
}
```

Etapa 6 – (Opcional) Resumo com LLM

- Usar LLM local (Ollama, llama.cpp) ou GPT via API
- **Prompt sugerido:**

  > Com base na transcrição abaixo, identifique os principais sintomas, queixas e respostas do paciente, e gere um resumo clínico.

Etapa 7 – Exportação

- Gerar PDF, Markdown ou JSON com a ficha
- Ferramentas: `fpdf`, `reportlab`, `json`

Stack Recomendada

Etapa	Ferramenta
Interface	`Tkinter`
Áudio	`sounddevice`, `numpy`, `scipy`
Transcrição	`Whisper`
Diarização	`pyannote.audio`
LLM	GPT-4, Mistral, Ollama

Etapa	Ferramenta
Exportação	`fpdf`, `reportlab`

Termos para Pesquisa

- **"Tkinter GUI Python passo a passo"**
- **"Python gravação de áudio com pausa"**
- **"Whisper transcrição português"**
- **"pyannote diarization speaker separation"**
- **"Python JSON to PDF"**
- **"LLM summarization local Ollama llama.cpp"**

Fluxo Geral do MVP

```
[GUI] → Grava .WAV → Transcreve (Whisper) → Diariza (pyannote) → Organiza
```