Curser

Stop embarrassing names before they launch

Motivation I:

I LOVE annoying my sister

- Problem: I'm currently 80 mi away…
  - Soln: Creative nicknames

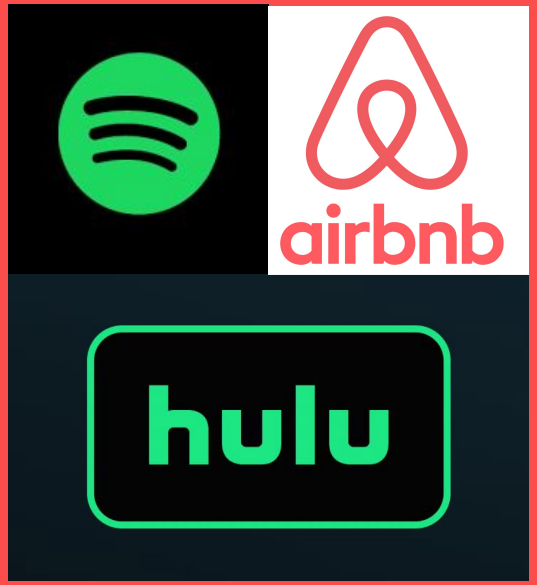
- Problem II: My Bengali is lacking in some departments

Motivation II:

A bad names sets you back before you even start

- Pharma: abstract, legally safe words
- Tech startups: same risk, just with less lawyers
- Abstract names are fragile across languages and cultures

- Curser asks: "Does this sound bad somewhere else?"



How It Works

Big Picture

audio -> text -> phonemes -> matches -> UI

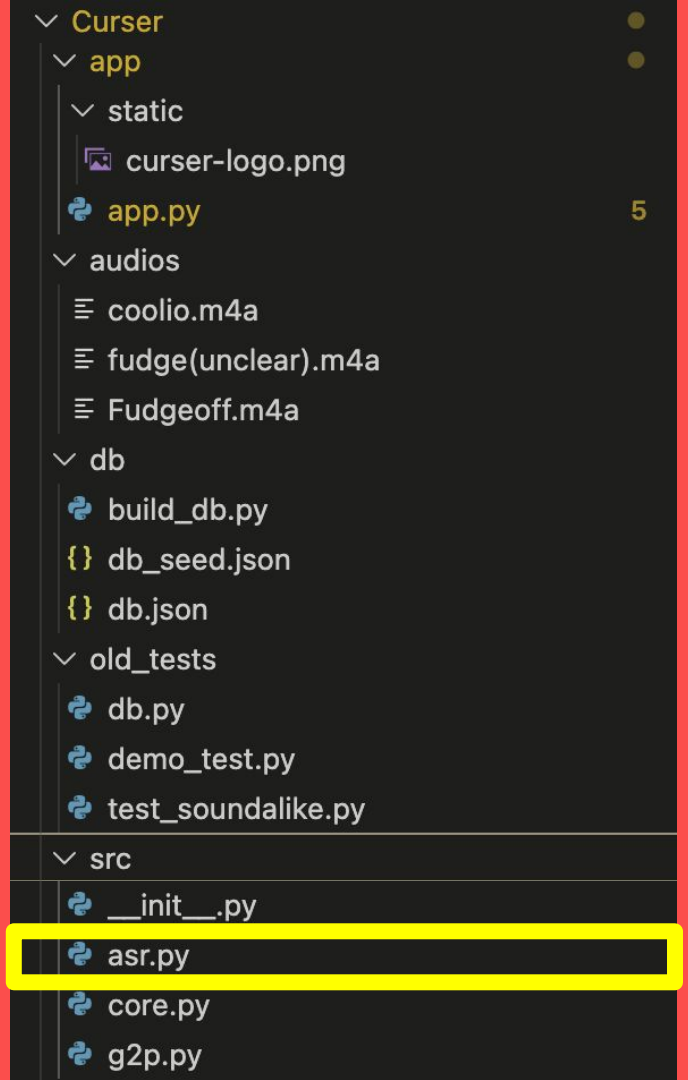
How It Works:

The weeds

- asr.py:

Speech In, Language Handling

- Open-AI Whisper
- Detects or infers language from audio
- Outputs clean text spans for analysis



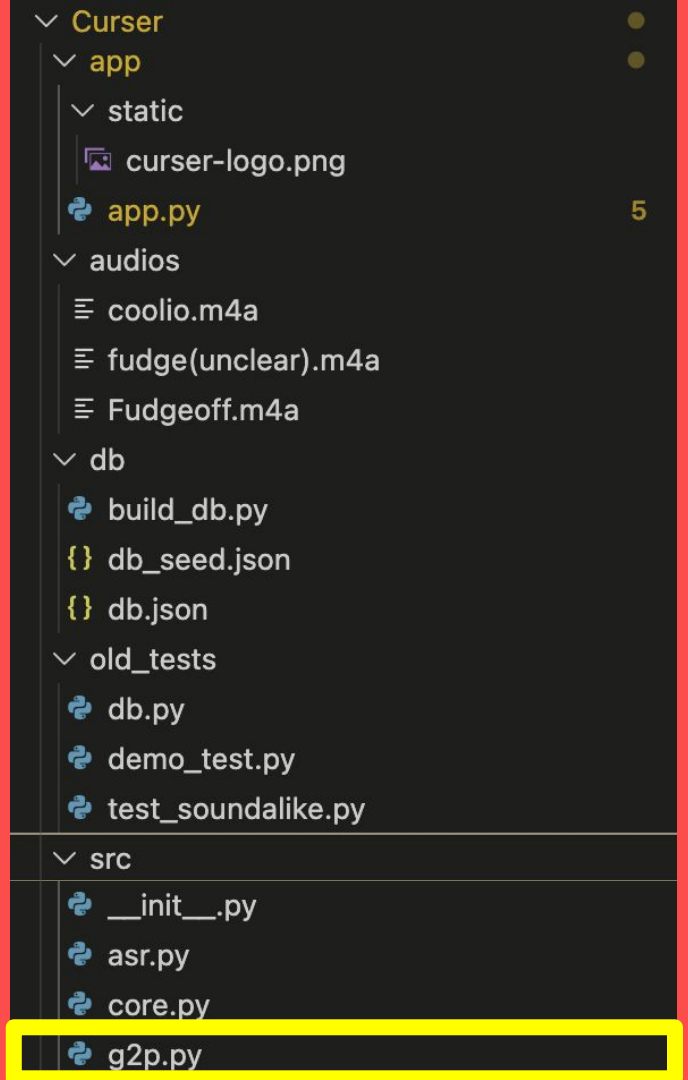
How It Works:

The weeds

- g2p.py:

Text to Phonemes Handling

- Converts text into IPA using eSpeak
- Supports multiple languages with fallbacks
- Normalizes IPA for robust matching

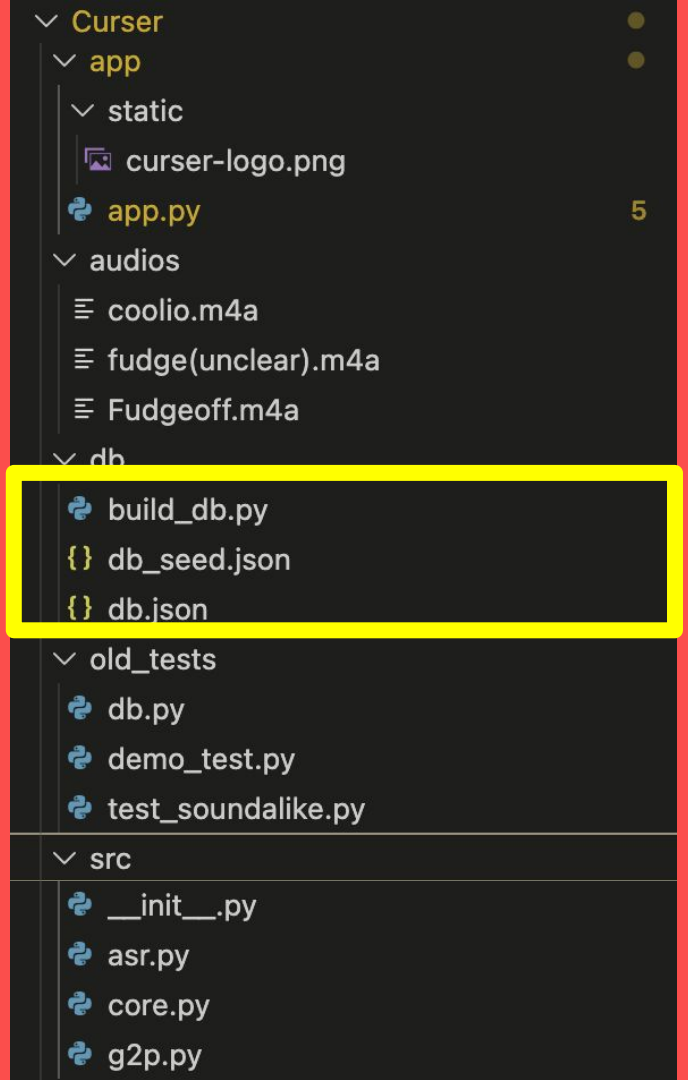


How It Works:

The weeds

db_seed.json + build_db.py: The Database

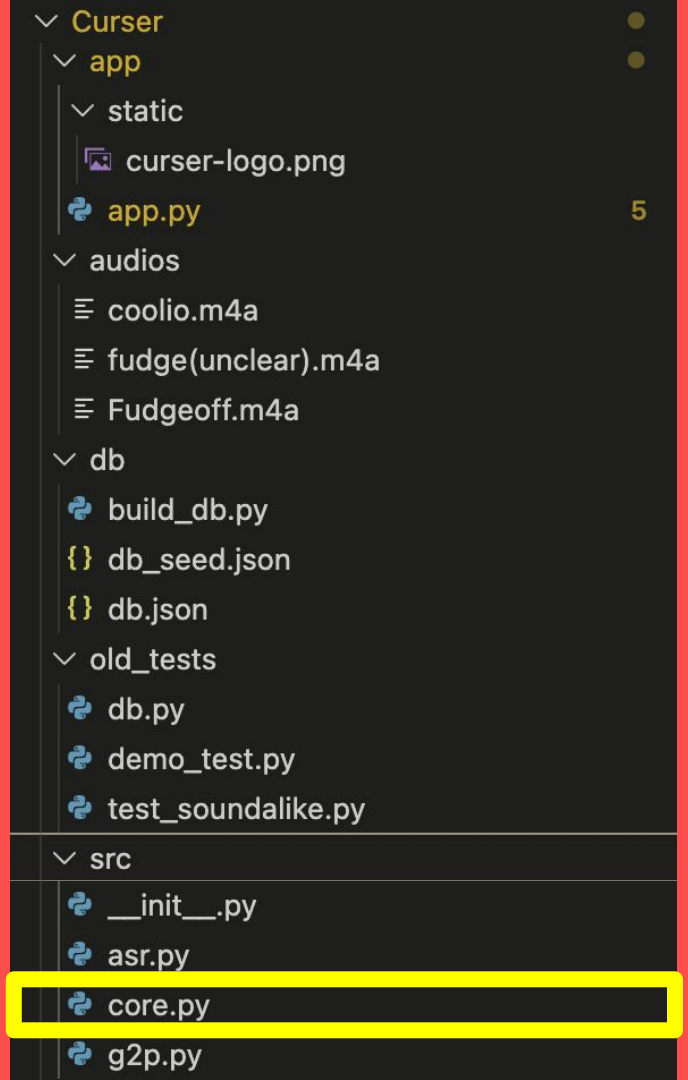
- Human-editable seed file defines words and metadata
- build_db.py converts words to IPA and normalized IPA
- Produces a reproducible phonetic database



How It Works:

The weeds

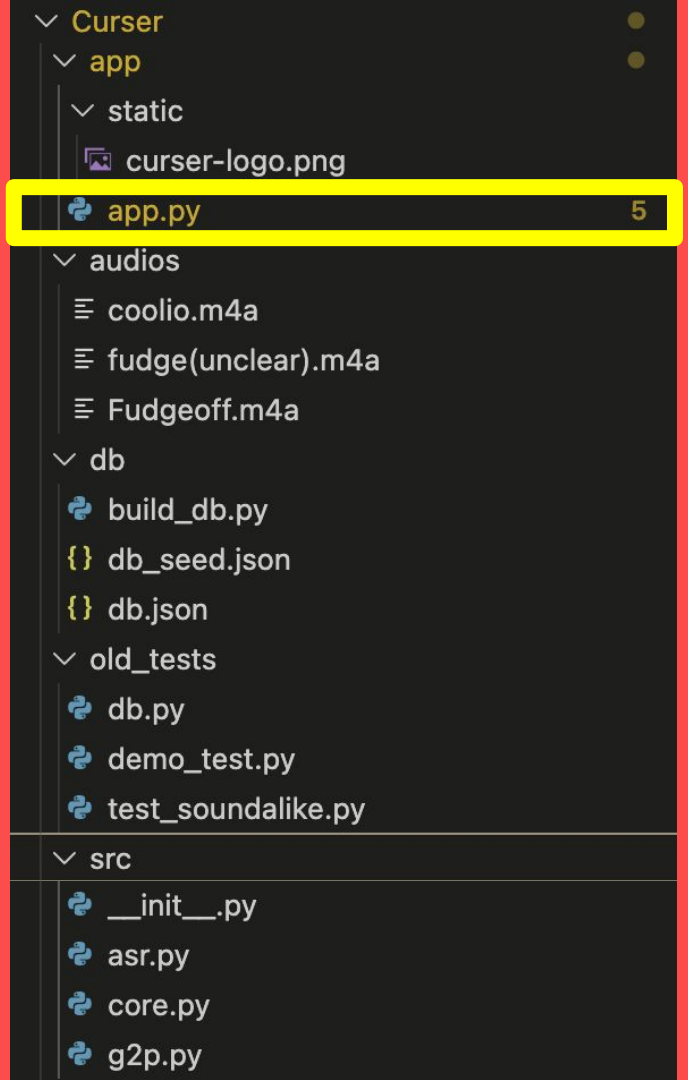
- core.py: Phonetic Matching Engine
- Sliding-window problem for phoneme sequences
- Computes phonetic distances with PanPhon
- Ranks matches by similarity score



How It Works:

The weeds

- app.py: Orchestration and UI
- Orchestrates the full pipeline end-to-end
- Manages state, history, and user interaction
- Surfaces results through Streamlit



Text-to-Speech and ElevenLabs

- Pronunciation playback of matched words
- Default local TTS using eSpeak
- ElevenLabs integration for neural, high-quality voices
- Useful for evaluating spoken ambiguity, not just text

DEMO TIME!!!

Interface and Demo Notes

- Built entirely with Streamlit
- Live mic input, uploads, and one-shot recording
- Sortable result tables with explanations
- Session-level history for comparisons

Challenges

- Environment management and dependency conflicts
- Audio, ASR, and G2P tools had mismatched assumptions
- Many failures were silent, not crashes
- Scope creep before pipeline stability
- Debugging required heavy instrumentation

What I Learned + What's Next

- Learned applied phonetics without formal linguistics training
- First real web app with reproducible setup
- First time using authenticated third-party APIs
- Want customizable scoring and better explanations
- Long-term goal is a hosted, extensible tool

Thank You