Sentra Interview - Final Round

For this final round of interview, we'd like you to build an AI resume screener.

Please feel free to use whatever tool that might be helpful, e.g. web searches, AI coding tools.

Problem Statement

For engineering team hiring for open roles, the challenges at the beginning stage of the interview - resume screening, come from 3 angles:

1. Non-technical people (recruiters, head of talent, etc.) don't know in-depth what high-value signals to look for.
2. Engineering managers themselves often can't clearly 100% articulate what they are looking for. Usually it's a "if you see it you know it" type of resume screening strategy.
3. There are an enormous number of resumes for each open role. Manually screening them takes up valuable engineering bandwidth.

To solve all 3 aspects of this problem, please design and implement a system that utilizes LLM to screen inbound resumes.

Functional Requirements

1. The system should be able to automatically ingest new resumes from a job posting on the Ashby platform.

   For the sake of this interview, first mock this step by reading directly from the pdf files provided. If there is extra time, implement the actual API integration.

2. The system should be able to do an initial round of screening based on natural language instructions (e.g. job description and other) given, present the passing resumes to a human reviewer for further review.

   Expect the reviewer to only spend ~10 seconds on each resume here, so please give extra thoughts on what information you present.

3. The system should be able to take feedback from the human reviewer in natural language, memorize it, and implement it for future screenings.
4. The system should be able to push candidates down the Ashby pipeline after the human reviewer approves (i.e. move them to the next step of the interview).

For the sake of this interview, first mock this step by writing into a csv file. If there is extra time, implement the actual API integration.

Non-functional Requirements

1. The system should be able to perform initial screening at a rate of >100 resumes/minute.
2. The system should have a lean but intuitive UI.
3. The system should be able to deploy locally via a docker container.

Other Requirements

1. Write clean and maintainable code - document well, separate code into modular and logical functions and files. It's ok to not finish in the given amount of time, focus on | quality over quantity.

Use this API Key:
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

Submission

- Please record a demo video under 5 minutes, and send the link to andrey@sentra.app and jonathan@sentra.app.
- Please submit the code via a Github repo, and invite andrey@sentra.app and jonathan@sentra.app as collaborators.