AI Engineer Take-Home Test

Objective:

Build a prototype system for **document categorisation** and **content extraction (OCR)** using an LLM / VLM, with a focus on **fraud prevention**. The solution should help determine the most relevant category for the file, and extract content from it.

Scenario

We receive diverse unstructured files from customers. Your task is to process these files to:

1. **Categorise** them into meaningful predefined categories.
2. **Extract content** (e.g., key entities, dates, relevant metadata).

We will provide **API_KEY** for LLM models [via **OpenRouter**] with $100 credits so that you can use them for your experiments. We will also share several example files to give you an idea of what to focus on.

File categories: *invoice, marketplace_listing_screenshot, chat_screenshot, website_screenshot, other*.

Requirements

1. Core functionality

- Accept multiple file types (at minimum: PDF, PNG/JPEG).
- Use an LLM / VLM to:
  - Assign a category (either from a predefined set or dynamically inferred).
  - Extract key structured attributes.
- Provide a standardised, structured output format (e.g., JSON) across different files.
- Provide an interface for uploading files and displaying results (we recommend using a framework such as **Streamlit**)

2. Non-functional concerns

- **Upload speed**: minimize latency from file selection to file processing..
- **Processing speed**: minimise latency to result availability after upload.
- **Robustness:** demonstrate how you defend against both user errors and the probabilistic nature of the system.

Implementation guidelines

- Use **basic, readable code** (clarity over complexity).
- Clearly separate **data ingestion**, **preprocessing**, **model calls**, and **postprocessing**.
- You may use open-source libraries for file handling and model interaction.
- We also strongly encourage the use of development tools such as Cursor, Copilot, and others to quickly prototype a working solution.

Deliverables

- A Git repository containing:
  - The source code.
  - A README explaining:
    - How to run the code.
    - Assumptions and design decisions.
    - Known limitations and possible improvements.

- Example input files (we will provide several, but you may add your own for testing).
- Example output files in JSON format.

Evaluation criteria

We will assess (ordered by priority):

1. **Correctness** – Does the solution categorise and extract features as intended?
2. **Code quality** – Is it clean, modular, and easy to follow?
3. **Performance awareness** – Are speed and stability considered?
4. **Guardrails** – Does the solution fail gracefully?
5. **Extensibility** – Is there a clear path for future improvements?

Time expectations

Please don’t spend more than 3 hours on this. We are not looking for a production-ready system — focus on demonstrating your **problem-solving approach** and ability to work with LLM/VLM tools.