USDA LLM Model – Analyst Walkthrough & Setup Guide

This script processes public comments from JSON files (with optional PDFs), extracts and combines their text, and uses an LLM (GPT) to return structured summaries of:

- who made the comment
- what was requested
- why it was requested
- what issues were raised
- whether scientific or legal support was present

The result is saved as a CSV file for analyst review.

SECTION 1: ENVIRONMENT & SETUP (DO THIS FIRST)

1.1. Install Python Packages (Command Line)

Run the following in your terminal or command prompt:

```
pip install openai pandas python-dotenv pdfplumber pytesseract pdf2image Pillow
```

Additionally, install Poppler for PDF-to-image conversion on Windows:

Download: http://blog.alivate.com.au/poppler-windows/

After extraction, add the bin/ folder to your PATH environment variable.

1.2. Install & Configure Tesseract OCR

Required for extracting text from scanned PDFs.

1. Download Tesseract for Windows:

   https://github.com/UB-Mannheim/tesseract/wiki

2. Install it, then locate the install path (typically):

   C:\Program Files\Tesseract-OCR

3. In the script, update:

   pytesseract.pytesseract.tesseract_cmd = r"C:\\Program Files\\Tesseract-OCR"

1.3. Setup Your `.env` File for API Access

Create a file called `.env` in the same directory as the script with:

```
OPENAI_API_KEY=sk-...your_key_here...
```

This keeps your OpenAI API key secure and outside the source code. You can do this by making a text file, insert your API key after ‘OPENAI_API_KEY=’, and then save it as ‘.env’ ensuring ‘all types’ is selected for file types.

1.4. Define Input/Output Paths in Script

Set the following variables at the top of the script to point to your local files:

```
JSON_FOLDER = r"C:\\Path\\To\\JSON\\Files"
PDF_FOLDER = r"C:\\Path\\To\\Attached\\PDFs"
OUTPUT_FILE = "processed_comments.csv"
```

You must have all PDFs already downloaded and saved locally in the attachments folder using the preprocessing script.

SECTION 2: CODE FUNCTIONALITY – HOW IT WORKS

2.1. `extract_text_from_pdf(pdf_path)`

Attempts to extract text from PDFs using:

1. pdfplumber (for text-based PDFs)
2. pytesseract OCR (for scanned/image PDFs)

Returns the full extracted text.

2.2. `classify_comment_by_issue(comment_text, pdf_attached=False)`

This function sends the combined comment + PDF text to GPT via the OpenAI API.

Extracted outputs (as JSON fields):

- who_type: individual, organization, or anonymous
- who_name: name of the commenter (if inferable)
- what: requested changes
- why: reasons for the request
- issues: a list of specific issues mentioned
- scientific_legal_support: "Yes" or "No"

load_dotenv & API Key

- Securely loads your OpenAI key from the `.env` file.

🔷 OpenAI Client Initialization

- Connects to OpenAI using your API key: `client = OpenAI(api_key=API_KEY)`

CUSTOMIZATIONS:

Change GPT Model:

model = "gpt-4-turbo"

Tweak Prompt:

Inside classify_comment_by_issue(), update the `prompt` string to change how GPT interprets the comment.

process_json_comments()

- Loops through a single JSON file
- Pulls comment text + PDF text
- Calls GPT to classify
- Formats a record with metadata

process_all_comments()

- Loops through all JSON files in your folder
- Aggregates all results and saves them to the CSV

extract_json_block()

- Ensures GPT responses are parsed cleanly as JSON
- Uses regex to pull the structured block