ChatGPT

Developer README: Auto-Analyst (Free Version)

Welcome to **Auto-Analyst**, a fully free and open-source Autonomous Research Assistant built with open-source LLMs and RAG principles. This README provides installation steps, usage instructions and development guidelines.

Prerequisites

- **Python ≥ 3.11**
- **Git** for version control
- **Streamlit** for the UI
- **Docker** (optional) for containerised deployment

Installation

1. **Clone the repository**

```
git clone https://github.com/<your-username>/auto-analyst.git
cd auto-analyst
```

2. **Set up a virtual environment** (recommended)

```
python -m venv venv
source venv/bin/activate
```

3. **Install dependencies**

```
pip install -r requirements.txt
```

The `requirements.txt` should include free libraries such as:

4. `streamlit`
5. `langchain` and `langgraph`
6. `sentence-transformers`
7. `chromadb` or `faiss-cpu`
8. `playwright` and `beautifulsoup4` for web scraping
9. `pdfplumber` for PDF parsing
10. `huggingface-hub` and `transformers` for loading open-source models

11. **Install Playwright browser**

```
playwright install chromium
```

12. **Download free models** Use `huggingface_hub` or `Ollama` to download a free instruct model and an embedding model. For example:

    ```python
    from transformers import AutoModelForCausalLM, AutoTokenizer
    model_name = "mistral-7b-instruct"
    model = AutoModelForCausalLM.from_pretrained(model_name)
    tokenizer = AutoTokenizer.from_pretrained(model_name)
    ```

    Embeddings can be loaded via:

    ```python
    from sentence_transformers import SentenceTransformer
    embedder = SentenceTransformer("all-MiniLM-L6-v2")
    ```

    These models require no paid API keys.

Running the Application

Local Execution

Run the Streamlit application directly:

```
streamlit run app.py
```

Navigate to `http://localhost:8501` in your browser. You will see a text box to enter your research question, a slider to select the number of sources and a **Run Research** button. The results include a structured answer with numbered citations and an expandable list of sources.

Docker (Optional)

For containerised deployment, use the provided `Dockerfile`:

```
docker build -t auto-analyst .
docker run -p 8501:8501 auto-analyst
```

Customisation

- **Vector store:** Choose between ChromaDB (default) and FAISS. Modify the `vector_store.py` helper to switch.
- **Search API:** Adjust the `search_tool.py` to use different free search providers. When using SearxNG, self-host your own instance or configure an available public instance.
- **Models:** Swap in another free LLM or embedding model by changing the model identifiers in the configuration file.

Project Structure

```
auto-analyst/
├── api/               # Orchestration and agent definitions
├── tools/             # Planner, search, fetch, parse, embed, retrieve,
generate, verify
├── ui/app.py          # Streamlit UI
├── vector_store/      # Helper classes for ChromaDB/FAISS
├── requirements.txt   # Python dependencies
├── Dockerfile         # Container configuration (optional)
└── tests/             # Unit tests for each component
```

Usage Notes

- **Ethics and compliance:** Follow legal and ethical guidelines when scraping websites. Do not bypass paywalls or ignore `robots.txt` policies. Always attribute content with citations.
- **Evaluation:** Use the evaluation metrics defined in the technical design document—context relevance, context sufficiency, answer relevance, answer correctness and answer hallucination—to measure and improve system performance [1].
- **Resource directory:** Place additional documentation and reference materials in `/home/guillaume/code/GuiPro0408/auto_analyst/ressources`. This directory will serve as input for automation tools such as Codex.

Contributing

Contributions are welcome! Please open issues or pull requests to discuss bugs or enhancements. When contributing code, follow PEP 8 formatting and include descriptive docstrings. Use SOLID principles and keep components modular. Write unit tests for any new functionality.

License

Auto-Analyst is released under the MIT License. See `LICENSE` for details.

---

[1] RAG Evaluation Metrics: Best Practices for Evaluating RAG Systems
https://www.patronus.ai/llm-testing/rag-evaluation-metrics