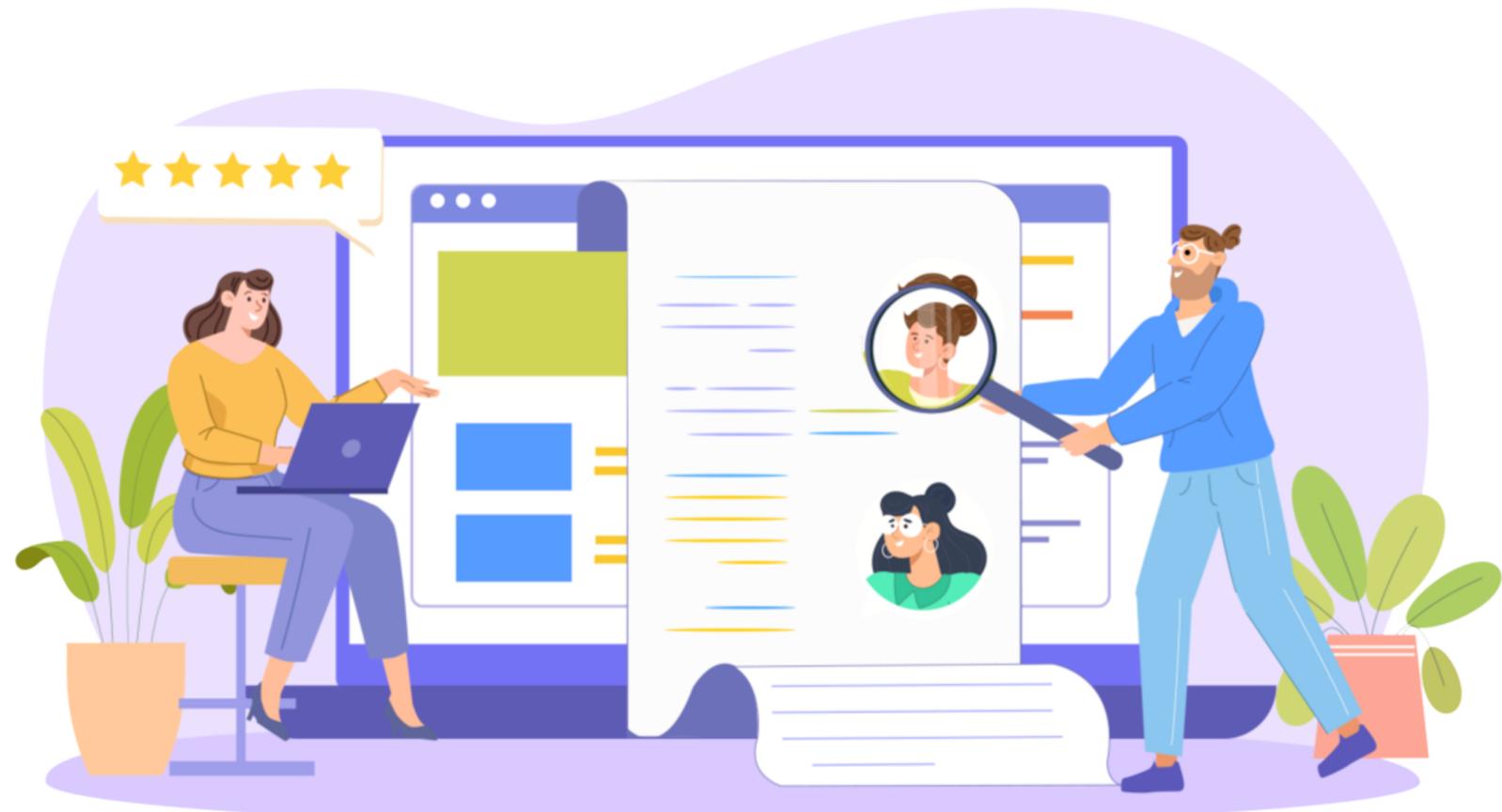


Curriculum Vitae extractor with Langchain

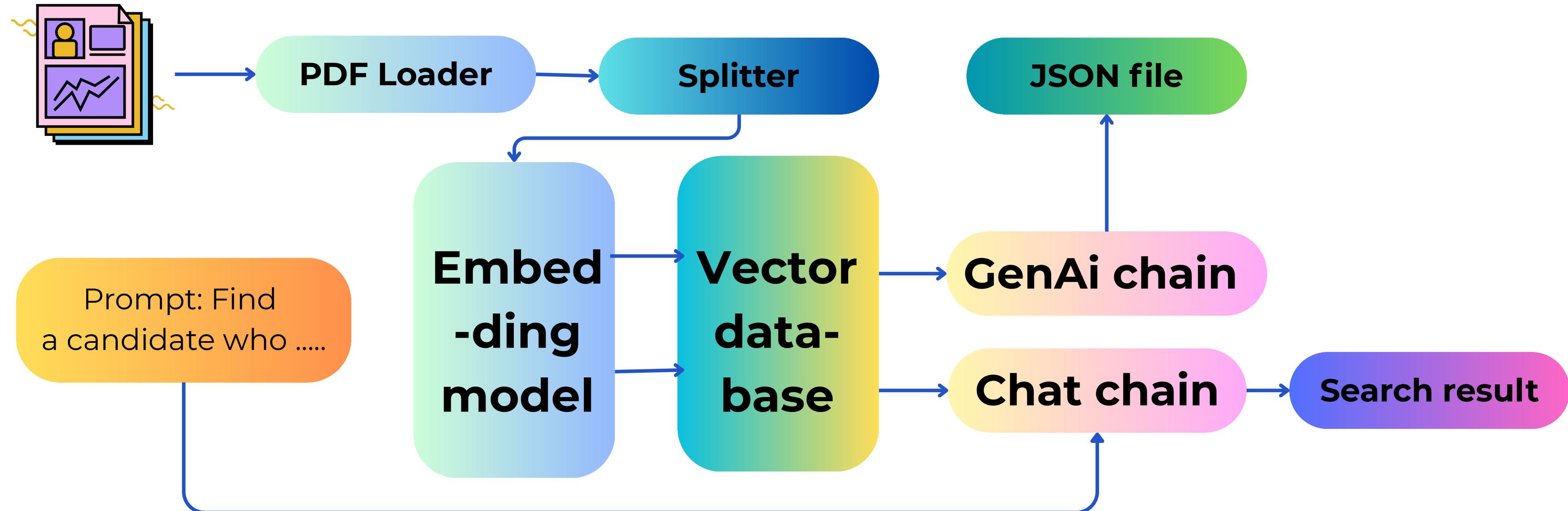
Introduction



The Problem: Recruiters handle thousands of CVs in unstructured formats—manual review is time-

--> **How to automatically extract the structured data from CVs?**

System overview

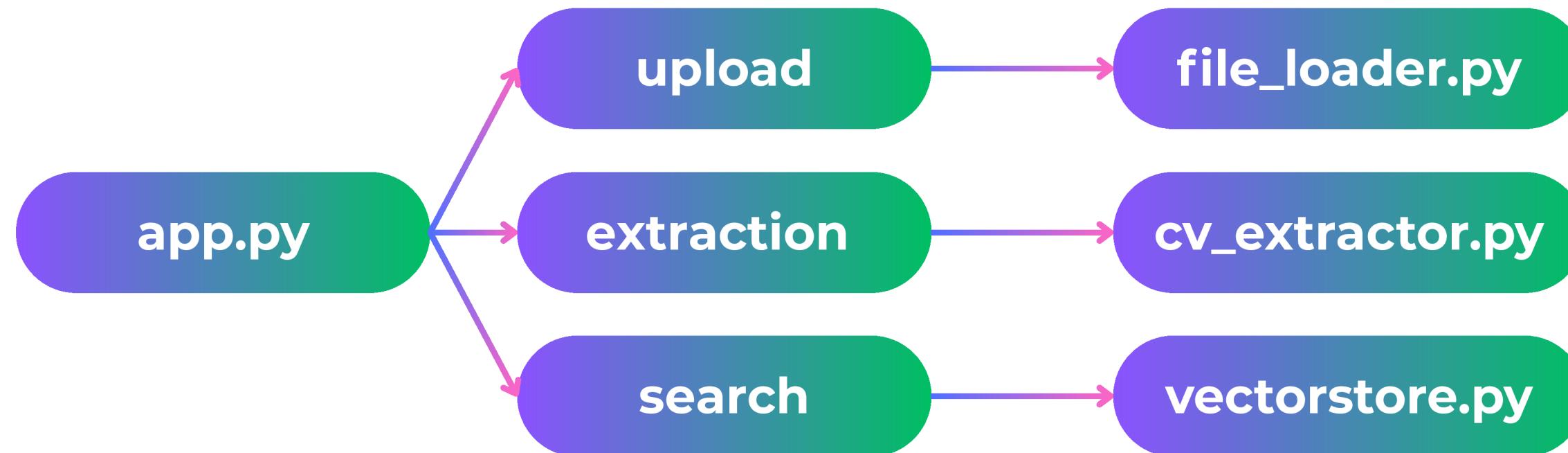


Directory structure

```
CV_EXTRACTOR_LANGCHAIN/
├── chat_histories/
│   └── .gitkeep
├── data_source/
│   └── generative_ai/
│       ├── download.sh
│       └── PhungMinhChiCV.pdf
├── docker/
│   └── Dockerfile
└── src/
    ├── base/
    │   ├── llm_model.py
    │   └── output_parser.py
    ├── chat/
    │   ├── history.py
    │   └── main.py
    └── rag/
        ├── cv_extractor.py
        │   ├── file_loader.py
        │   └── main.py
        └── offline_rag.py
            └── utils.py
    └── vectorstore.py
        ├── app.py
        └── streamlit.py
    └── .dockerignore
    └── .gitignore
    └── docker-
        compose.yml
    └── README.md
    └── requirements.txt
```

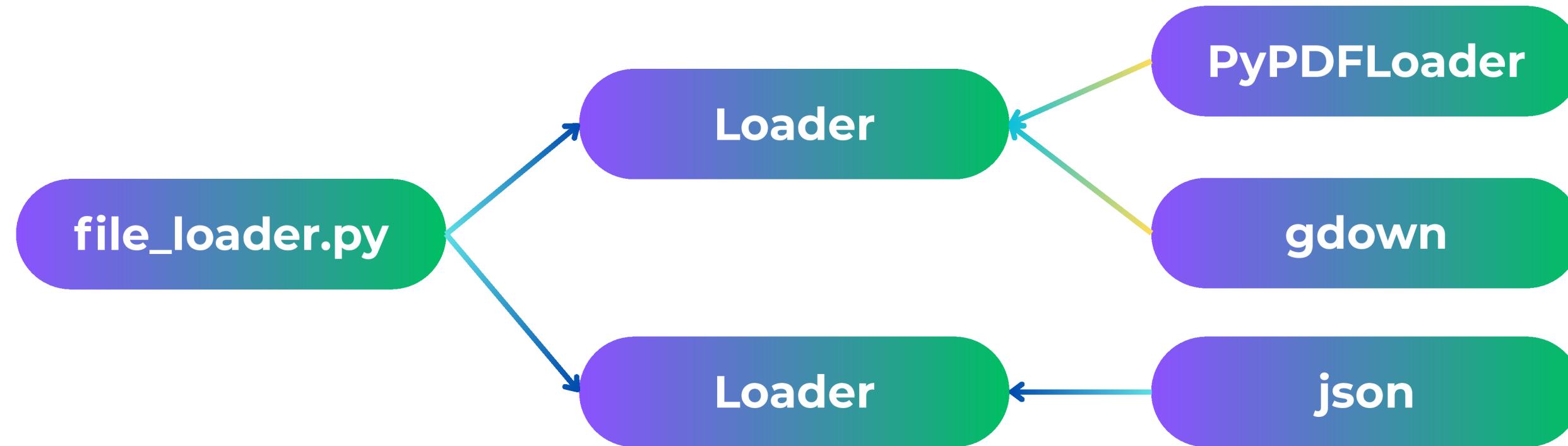
Solution architecture

- **API Layer:** FastAPI server in **app.py** handles **upload, extraction, search, and QA**.
- **Processing Flow:**
 - **Uploaded CVs** are **loaded** and **chunked**.
 - **LLM parses** the content **into structured JSON**.
 - **Extracted information** is **embedded** and **indexed**.
- **Storage:** FAISS vector store enables similarity search over CV chunks.



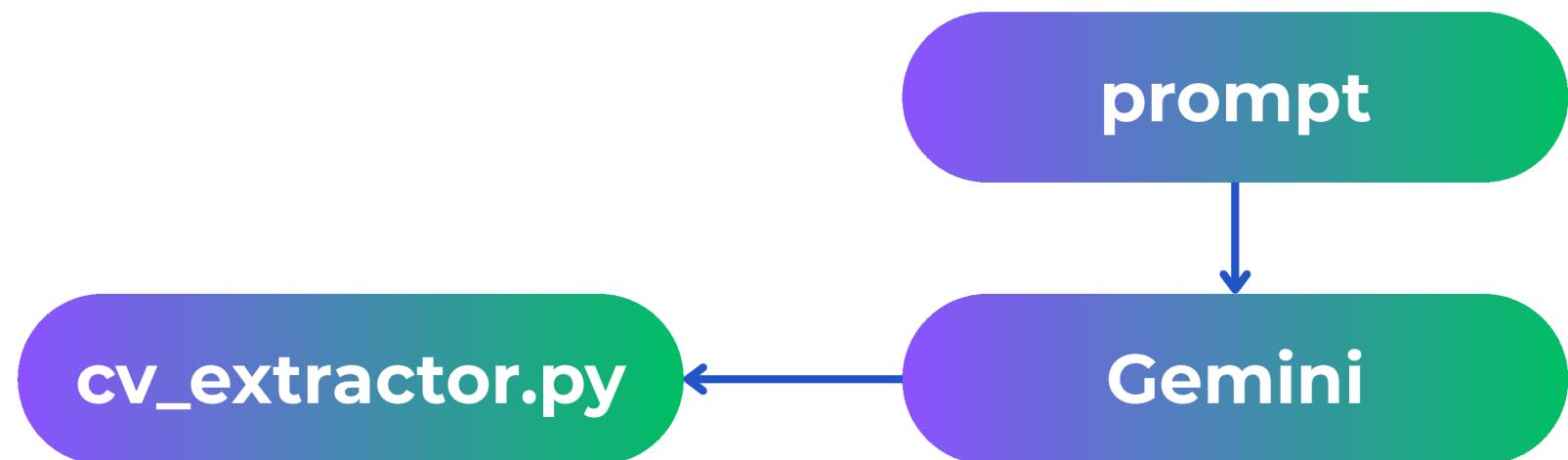
Module breakdown

- **file_loader.py:** Reads PDFs/HTML, splits into chunks.
- **cv_extractor.py:** Uses LLM to extract fields (e.g., name, skills) from each chunk.
- **vectorstore.py:** Creates and queries the FAISS index.
- **app.py:** Connects all modules, defines REST endpoints.



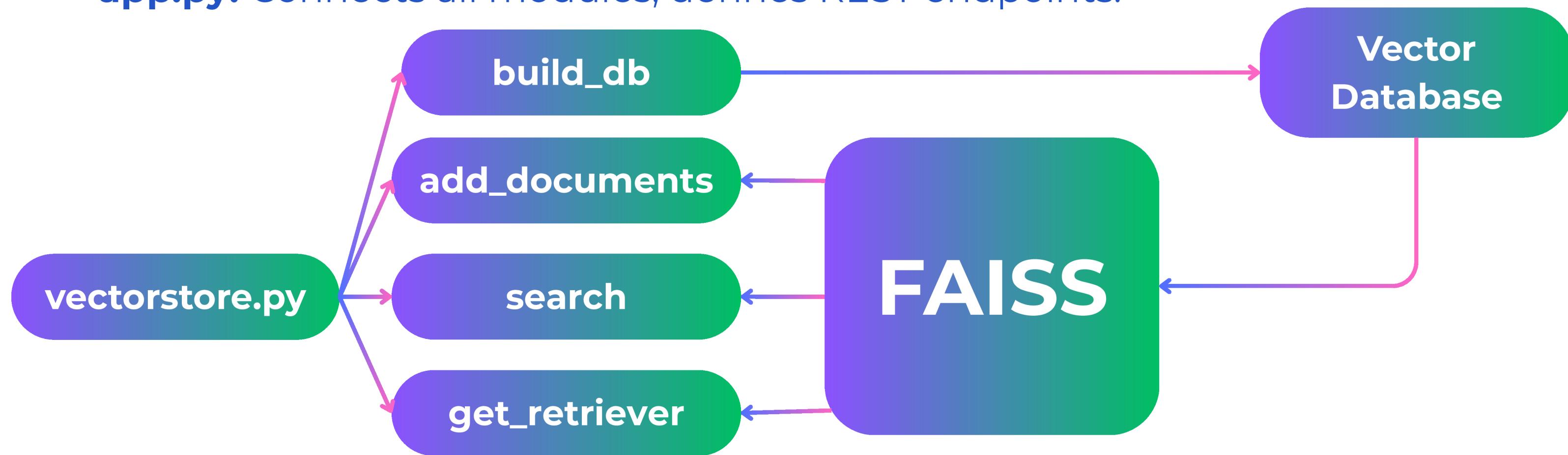
Module breakdown

- **file_loader.py:** Reads PDFs/HTML, splits into chunks.
- **cv_extractor.py:** Uses LLM to extract fields (e.g., name, skills) from each chunk.
- **vectorstore.py:** Creates and queries the FAISS index.
- **app.py:** Connects all modules, defines REST endpoints.



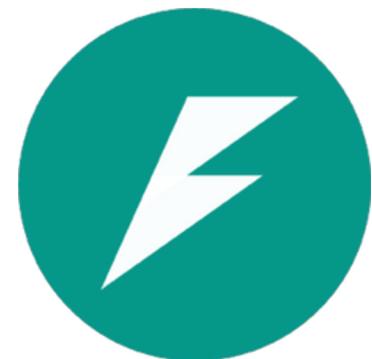
Module breakdown

- **file_loader.py:** Reads PDFs/HTML, splits into chunks.
- **cv_extractor.py:** Uses LLM to extract fields (e.g., name, skills) from each chunk.
- **vectorstore.py:** Creates and queries the FAISS index.
- **app.py:** Connects all modules, defines REST endpoints.



Technologies Used

- **Backend:** Python 3.11, FastAPI
- **LLM Orchestration:** LangChain, Google Generative AI
- **Search:** FAISS for embedding-based similarity queries
- **Front-End:**
 - Streamlit UI for uploading/searching/exporting results
 - Langchain playground for searching
- **Other:** PyPDFLoader



Gemini

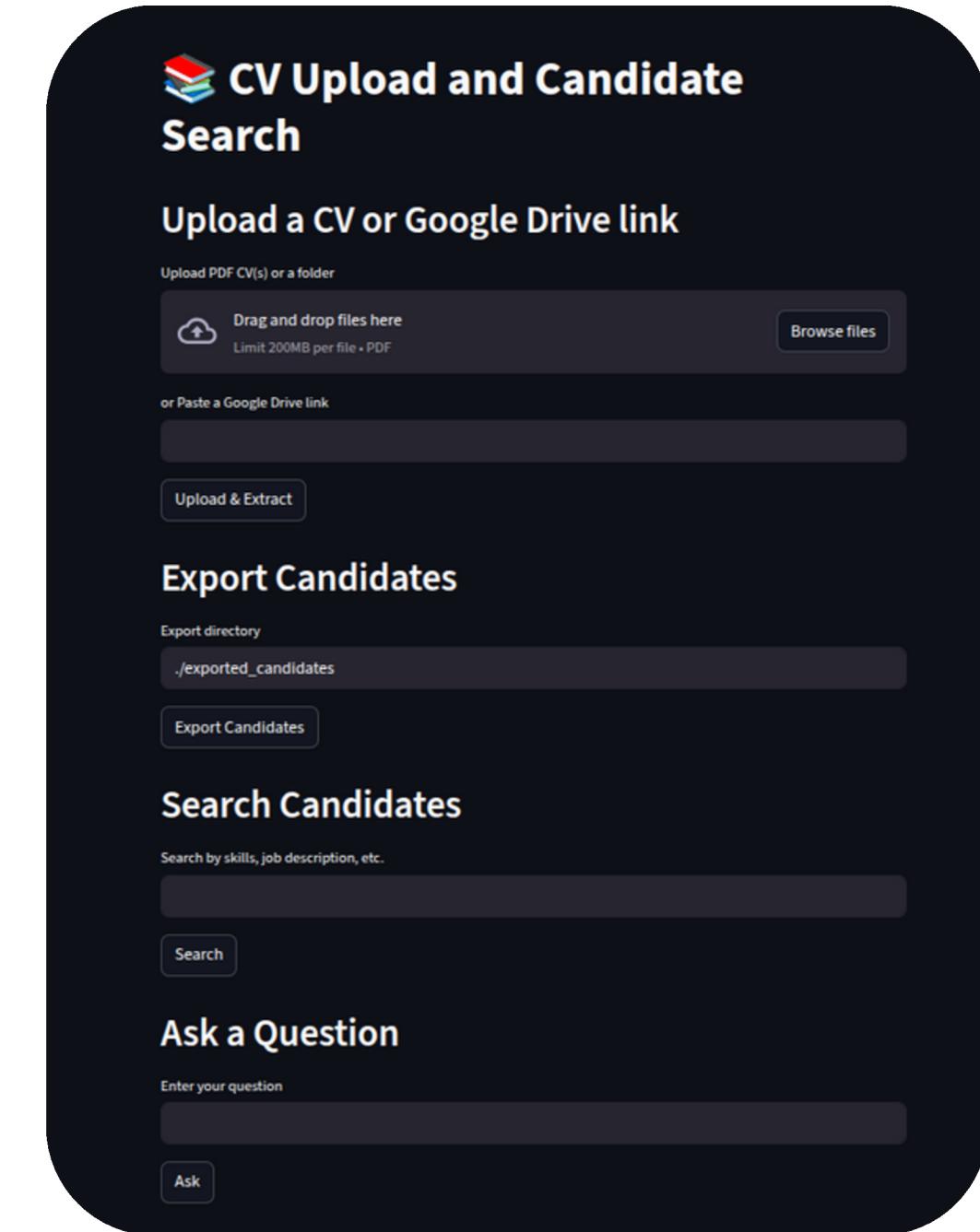


Demo

```
CV processed successfully!
```

```
{  
  "message" : "CVs processed"  
  "extracted" : [  
    0 : ````json  
    {  
      "Full Name": "ARDRA PRASAD",  
      "Email": "ardraprasad93@gmail.com",  
      "Phone Number": [  
        "9891423551",  
        "9650459770"  
      ],  
      "Education": [],  
      "Work Experience": [],  
      "Skills": [],  
      "Certifications": []  
    }  
    ````  
 1 :
    ````  
  ]  
}
```

extraction result



The image shows a Streamlit application interface titled "CV Upload and Candidate Search". The interface is dark-themed with white text and light-colored input fields.

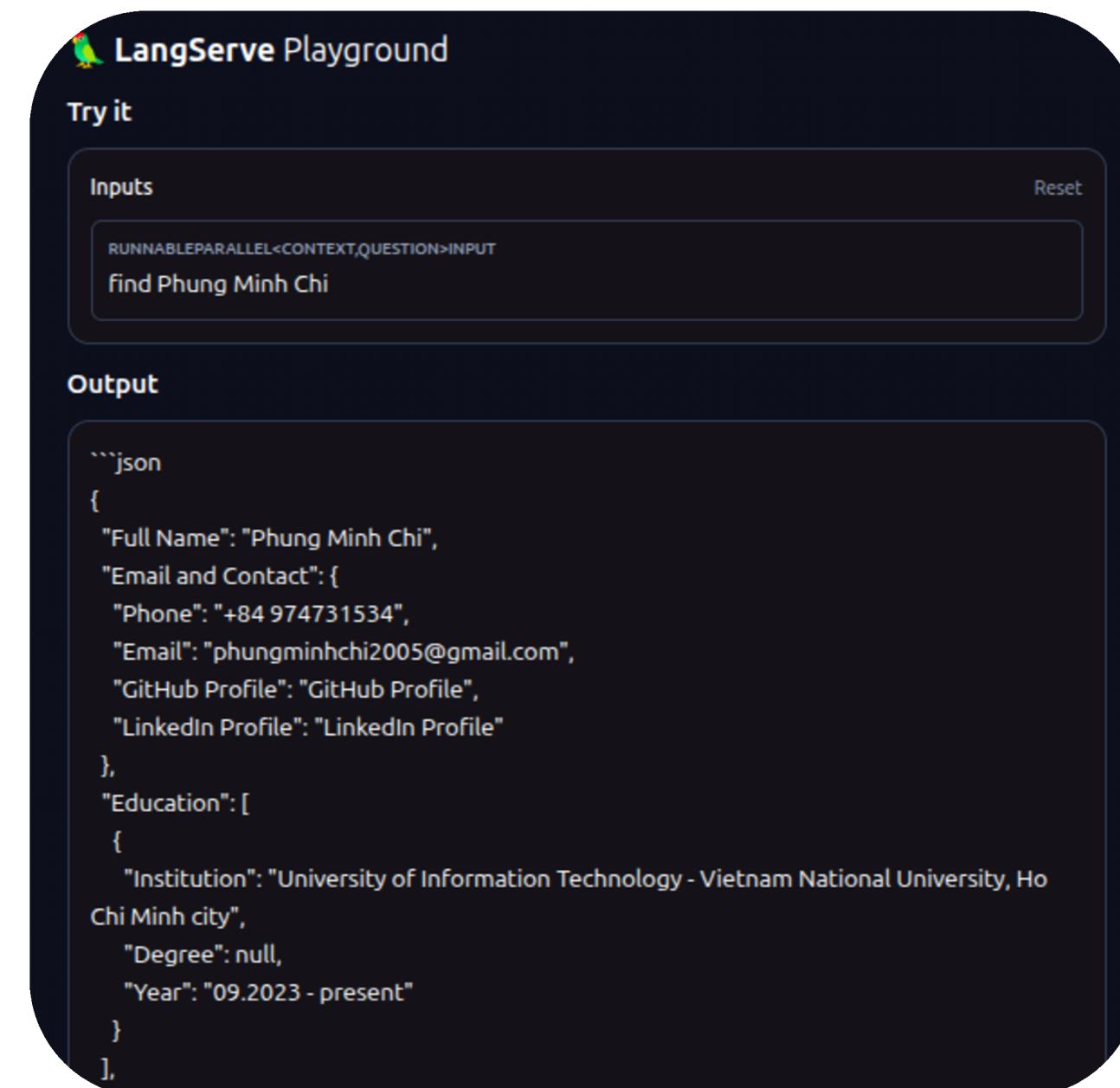
- Upload a CV or Google Drive link:** A section for uploading files. It includes a "Drag and drop files here" area with a file limit of 200MB per file + PDF, a "Browse files" button, and a "or Paste a Google Drive link" input field. A "Upload & Extract" button is located below these fields.
- Export Candidates:** A section for exporting candidate data. It includes an "Export directory" input field containing the value "/exported_candidates" and a "Export Candidates" button.
- Search Candidates:** A section for searching candidates by skills or job descriptions. It includes a "Search by skills, job description, etc." input field and a "Search" button.
- Ask a Question:** A section for asking questions. It includes an "Enter your question" input field and an "Ask" button.

streamlit

Demo

```
ported_candidates > {} candidates.json > ...
1  [
2    {
3      "Full Name": "ARDRA PRASAD",
4      "Email": "ardraprasad93@gmail.com",
5      "Phone Number": [
6        "9891423551",
7        "9650459770"
8      ],
9      "Education": [],
10     "Work Experience": [],
11     "Skills": [],
12     "Certifications": []
13   },
14   {
15     "Full Name": null,
16     "Email": null,
17     "Phone Number": null,
18     "Education": [
19       {
20         "school": "Sarabhai Institute of Science and Technology (CUSAT ), Trivandrum",
21         "degree": "B.Tech in Civil Engineering",
22         "years": "2015",
23         "description": "70.5 (til 7thsem; last year result awaiting)"
24       },
25       {
26         "school": null
27       }
28     ]
29   }
30 ]
```

exported json



LangServe Playground

Future work

- Improve **retrieval accuracy**, the current model is **still facing hallucination** with some inaccurate information
- The current model **will be overloaded** if there are **over 100 CVs imported**, consider **batching strategies** or **async processing**.
- Explore **cloud deployment**
- Improve the **frontend site**
- Add **admin dashboard** for **reviewing** and **correcting** parsed results

References

1. AI VIET NAM – AI COURSE 2024 Tutorial: RAG with LangChain for PDF Question Answering
2. Langchain Services - AIO, https://github.com/ThuanNaN/Langchain_Services/
3. Build a Retrieval Augmented Generation (RAG) App, <https://python.langchain.com/docs/tutorials/rag/>
4. ChatGoogleGenerativeAI, https://python.langchain.com/docs/integrations/chat/google_generative_ai/
5. PyPDFLoader, https://python.langchain.com/docs/integrations/document_loaders/pypdfloader/
6. langchain_community.document_loaders.pdf.PyPDFLoader,
https://api.python.langchain.com/en/latest/document_loaders/langchain_community.document_loaders.pdf.PyPDFLoader.html

Thank You

During the project, I needed to balance preparing for my final exams with working on the project. I could not dedicate full-time effort to it, so some parts may be imperfect. However, I truly appreciate the opportunity to learn about the related technologies.



Linh Xuan, Thu Duc, Tp. Ho Chi Minh



0974731534



www.linkedin.com/in/chisphung