[Challenge] Machine Learning Engineering - LLM

📄 Case Description

Build a system that allows users to upload PDF documents and later ask questions about their contents. Your solution should extract information from the documents, store it in a way that allows efficient retrieval, and use a large language model (LLM) to answer user questions accurately.

You are free to choose your tools, libraries, and frameworks. We are most interested in how you structure your solution, understand retrieval-augmented generation (RAG), and integrate different components into a working system.

🚀 Features

- Upload PDF documents and extract their contents
- Chunk and embed document text for semantic search
- Store embeddings for later retrieval
- Ask questions based on the uploaded content
- Use an LLM to answer questions with contextual accuracy

🧪 API Specification

POST /documents

Upload one or more PDF documents to be indexed.

Request:

- Content-Type: `multipart/form-data`
- Body: One or more PDF files under the field name `files`

Response

```
{
  "message": "Documents processed successfully",
  "documents_indexed": 2,
  "total_chunks": 128
}
```

POST /question

Request:

- Content-Type: `application/json`

```
{
  "question": "What is the power consumption of the motor?"
}
```

Response

```
{
```

```
  "answer": "The motor's power consumption is 2.3 kW." ,
  "references":  [
      "the motor xxx has requires 2.3kw to operate at a 60hz line frequency"
  ]
}
```

💡 Optional Enhancements

- Basic frontend or Streamlit interface
- Support multiple LLM providers or fallback behavior
- Add logging, stats, or latency metrics
- Dockerized environment or Makefile

📦 Deliverables

Your submission should include:

- A GitHub repository with your complete implementation
- Clear instructions to set up and run your project
- Example requests and expected responses
- Include any extra requirements, environment variables, or API keys needed to run your code.
    - Don't worry about LLM provider keys. List them as necessary, and we will have one to test.

✅ Evaluation Criteria

Criteria	Description
Functionality	The system works as described and returns accurate answers
Retrieval	Relevant document chunks are correctly retrieved
LLM Use	Prompts are constructed effectively for answer generation
Code Quality	Clean, modular, maintainable code
API Design	Clear, documented, intuitive endpoints
Developer UX	Easy to set up, test, and understand your solution

📬 Questions?

If you have any questions or need clarification, feel free to reach out to the hiring team.
Good luck 🚀