Objective

Design and implement a **mini AI-powered document intelligence backend** that demonstrates your ability to build **multi-agent workflows**, REST APIs, AI integrations, and file orchestration.

This task focuses on **system design, clean architecture, and real-world problem solving**, not UI or polish.

Time Constraint

- **30 hours** from the time this task is shared

Scope

You are required to build a backend system that:

- Accepts document uploads (PDFs / Images)
- Processes them using AI agents
- Stores structured knowledge
- Answers user questions based on uploaded content

Frontend is NOT required.

Functional Requirements

You should implement **at least the following**:

1. **API Layer**

- REST API for document upload
- REST API to ask questions from processed documents
- Clear request/response structure

2. **File Handling**

- Store uploaded files using **local storage**
- Proper handling of file types (PDF / Image)

3. **AI Processing**

- Text extraction (OCR for images, PDF parsing)
- Chunking & embeddings
- Vector storage (FAISS or Pinecone)

4. Multi-Agent Workflow (Minimum 3 Agents)

Example agents (you may rename/design your own):

- **Ingestion Agent** – handles file parsing & cleanup
- **Indexing Agent** – embeddings & vector storage
- **QA Agent** – retrieves context & generates answers

Agents should be **logically separated** and orchestrated via a controller/service layer.

5. Orchestration

- Clear flow showing how agents communicate
- Can be synchronous or async

Tech Stack (Flexible)

You may use:

- **Backend:** Django / FastAPI
- **AI:** OpenAI / HuggingFace / LangChain
- **Vector DB:** FAISS / Pinecone
- **Storage:** local
- **DB:** PostgreSQL / SQLite

Docker, async processing, and caching are **bonus**, not mandatory.

Deliverables

Submit:

1. GitHub repository
2. README.md including:
    - System architecture overview
    - Agent responsibilities
    - API endpoints
    - Setup instructions
3. Sample API calls (Postman or curl)
4. Notes on trade-offs & future improvements

Evaluation Criteria

We will evaluate:

- Agent-based design & separation of concerns
- Orchestration logic
- Code quality & structure
- Real-world thinking (errors, scalability, extensibility)
- Ability to clearly explain decisions

Clarifications

- You may use your own API keys or mock credentials
- You may ask **one clarification question** within the first 6 hours

Good luck focus on **clarity over completeness**.