Skyro AI Engineer Case Study: AI-Powered Knowledge Access

Background

Skyro is exploring how to improve internal knowledge access using AI. The company has semi-structured (e.g., Confluence, Google Docs, FAQs, Meetings notes), and unstructured (e.g., PDFs, emails, Slack threads) data scattered across systems. The end goal is to enable employees and users to query this internal knowledge using natural language (via an LLM interface) while ensuring privacy, relevance, and accuracy.

We want to assess how you would:

- Understand and frame the problem
- Build a small but real solution (or part of it) in a short time
- Enable adoption across technical and business teams
- Communicate your thinking clearly to both technical and non-technical stakeholders

---

Your Task

Design and implement a lightweight prototype of an **AI-powered internal knowledge assistant** that demonstrates your ability to:

1. Use a publicly available LLM (open-source or API-based) in a retrieval-augmented generation (RAG) setup.
2. Ingest a small dataset of internal-style documents (provided below).
3. Allow natural language questions to be answered by combining LLM reasoning with relevant content snippets.

4. Document your approach with clarity, including architecture, trade-offs, and extension ideas.

What You Should Do

To simulate a Skyro like environment, create your own small dataset of 10 to 15 internal-style documents that might exist in an fintech platform:

- Confluence-style Markdown notes (e.g., feature specs, OKRs, Experiments, business etc.)
- Meeting summaries or transcripts
- Product specs (PDF, JSON, or plain text)

The idea is to build a *realistic test bed* for an internal knowledge assistant.

Requirements

You must:

- Build a simple RAG pipeline (can use LangChain / LlamaIndex / LangGraph / Haystack)
- Use any LLM (e.g., Mistral, Phi-2) or an API (e.g., OpenAI, Anthropic, etc.)
- Support 5 – 7 example questions and show that the assistant retrieves the correct context and answers the question.
- Ensure modularity and clarity in your code.
- Write a short readme explaining:
    - Your thought process and choices
    - Architecture (include a diagram)
    - How this can be extended to a Skyro-scale solution
    - A brief example of how this could be integrated into a Slack bot or web UI
    - How you would involve product and engineering teams to roll this out in a real scenario

---

Bonus (Optional):

- Show how different LLMs affect performance.
- Add document-level access control simulation.
- Implement basic feedback capture (e.g., “Was this answer helpful?”)

Deliverables

- GitHub repo
- markdown readme with flow diagram
- Deadline: [3 calendar days from assignment]

Evaluation Criteria

Area	**What We're Looking For**
Problem Understanding	Clarity of assumptions, scope framing, understanding of use cases
Technical Design & Implementation	Clean code, modularity, use of relevant tools (e.g., vector DBs, LangChain, etc.)
Product Thinking	How usable is this for internal users? Can it scale?
Communication & Documentation	Clear readme, logical explanations, ability to speak to tech and business audiences
Initiative & Pragmatism	Sensible scope choices, prioritization, balancing trade-offs under time constraints