AI Engineer Intern – Take-Home Assignment

Prompt Engineering & RAG Mini Project

Objective

Build a **small Retrieval-Augmented Generation (RAG) system** and demonstrate your ability to:

- Design **effective prompts**
- Improve response quality through **iteration**
- Evaluate and reason about LLM outputs

Problem Statement

You are given a set of **company policy documents** (e.g., Refund Policy, Cancellation Policy, Shipping Policy).

Your task is to build a **question-answering assistant** that:

1. Retrieves relevant information from the documents
2. Generates **accurate, grounded answers**
3. Avoids hallucinations
4. Uses **clear and well-structured prompts**

Scope & Constraints

- Time limit: **4–6 hours**
- This is **not** a UI task (CLI or simple script is enough)
- Focus on **prompt quality, retrieval, and evaluation**
- You may use **OpenAI, Anthropic, or any open-source model**
- Use **Python**

Core Requirements

1. Data Preparation

- Load the provided policy documents (PDF / TXT / Markdown).
- Clean and chunk the text using a reasonable strategy.
- Explain **why** you chose your chunk size.

---

2. RAG Pipeline

Implement a basic RAG flow:

- Embedding generation
- Vector storage (Chroma / Qdrant / FAISS)
- Semantic retrieval (top-k)
- Pass retrieved context to the LLM

---

3. Prompt Engineering (Very Important)

Design prompts that:

- Clearly instruct the model to answer **only from retrieved context**
- Handle **missing information gracefully**
- Use a **structured format** (headings, bullet points, citations, or JSON)

Include:

- Your **initial prompt**
- At least **one improved iteration**
- A short explanation of **what changed and why**

---

4. Evaluation

Create a small evaluation set (5–8 questions):

- Some answerable from the documents
- Some partially answerable
- Some unanswerable

Evaluate:

- Accuracy
- Hallucination avoidance
- Answer clarity

You can use:

- Manual evaluation
- Simple scoring rubric (✅ / ⚠️ / ❌)

---

5. Edge Case Handling

Show how your system responds when:

- No relevant documents are found
- The question is outside the knowledge base

Deliverables

1. **GitHub repository** containing:
    - Source code
    - `README.md`
2. **README should include**:
    - Setup instructions
    - Architecture overview
    - Prompt(s) used

    - Evaluation results
    - Key trade-offs and improvements you’d make with more time

---

Optional Bonus (Nice to Have)

(Do **not** over-optimize—these are bonus)

- Prompt templating with LangChain / LangGraph
- Simple reranking step
- Output schema validation (JSON)
- Comparison between **two prompt versions**
- Logging or tracing (very basic)

Evaluation Criteria (What We Look For)

Area	**What We Evaluate**
Prompting	Clarity, structure, hallucination control
RAG	Correct retrieval & grounding
Thinking	Trade-offs, explanations
Code	Readability, simplicity
Evaluation	Ability to judge model quality

What We Are *Not* Evaluating

- Fancy UI
- Large datasets
- Perfect accuracy
- Advanced ML math

We care about **reasoning, clarity, and iteration**.

Submission Instructions

- Share a **GitHub repo link**
- Include a short note explaining:
    - What you’re most proud of
    - One thing you’d improve next