# **📄 GenAI Challenge: Build a Q/A Bot from Company Document**

## **Problem Statement**

You are provided with a **sample company document** (approx. 3 pages of text).  
 Your task is to **design prompts** and **build a basic LLM-powered Q/A system** that can answer user questions about the document content.

The goal is to **simulate an intelligent document assistant** that can pull relevant answers from the text.

## **Dataset Information**

## 

**Use this as a sample data to test your agent :**[**Link**](https://drive.google.com/drive/folders/1g-A_CaiikTanU1RF43PR3ydM2oRVUuOQ?usp=sharing)

## 

## **Tasks to Complete**

### **1. Prompt Engineering**

* **Design prompts** that can help extract Q/A from the document:  
  + Example: “Based on the document, what are the services offered?”
  + Focus on **clear, precise prompt writing** to get accurate answers.
* Cover **at least 5 sample questions**.

### **2. Build Basic Q/A System**

* Using **LangChain**, **Gemini**, or **any LLM API (OpenAI, Gemini, Cohere)**:  
  + Load the document.
  + Create a retriever (basic split + retrieval if using LangChain or custom logic).
  + Use the LLM to answer user questions based on document content.

### **3. Jupyter Notebook**

* Build the Q/A system in a **clean Jupyter Notebook**.
* Include:  
  + Code to load document.
  + Code to split document into chunks (if needed).
  + Code to query the LLM.
  + Sample input questions and outputs.

## **🚀 (Optional Bonus) Streamlit App**

* Create a **simple Streamlit web app** with the following:  
  + Upload document (or preloaded file).
  + User input box for questions.
  + Output area displaying answers from the bot.

**Deliverables**

| **Deliverable** | **Description** |
| --- | --- |
| **Jupyter Notebook** | Full working code to build and run the Q/A system. |
| **README.md** | Document with setup instructions, LLM used, and approach. |
| **Streamlit App** | (Optional) Streamlit app code for UI deployment. |

**Time Limit**

* Suggested maximum time: **2 hours**.
* Focus on **functionality**, not heavy optimization.

## **Evaluation Criteria**

| **Criteria** | **Weightage** |
| --- | --- |
| Prompt design quality (clear and context-aware) | 30% |
| Correctness of answers from LLM | 30% |
| Code modularity and clarity | 20% |
| (Bonus) Streamlit app functionality | 10% |
| (Bonus) Creativity in UX or UI | 10% |

# **Suggested Folder Structure**

.

├── README.md (Approach, setup steps)

├── notebooks/

│ └── qa\_bot.ipynb

├── app/

│ └── streamlit\_app.py (optional)

├── data/

│ └── company\_document.txt

└── requirements.txt (optional, for LLMs and Streamlit)

# **Notes**

* Feel free to use **OpenAI’s GPT-3.5/4**, **Gemini Pro**, or any other model you're comfortable with.
* Basic document chunking (e.g., 500 characters per chunk) is acceptable if needed.
* Responses must strictly be based only on the document content (no hallucinations).
* Keep prompts **specific and instructive**, not vague.