GENERAL INSTRUCTIONS:

1. Read all instructions carefully before attempting.
3. Use of AI tools and frameworks is encouraged.
4. Submit your solution within the given time limit.
5. Code should be well-documented and explainable.

Q.1 PROBLEM STATEMENT

Design an LLM-Powered Intelligent Query–Retrieval System that can process large documents and make contextual decisions. Your system should handle real-world scenarios in insurance, legal, HR, and compliance domains.

Input Requirements:

- Process PDFs, DOCX, and email documents
- Handle policy/contract data efficiently
- Parse natural language queries

Technical Specifications:

- Use embeddings (FAISS/Pinecone) for semantic search
- Implement clause retrieval and matching
- Provide explainable decision rationale
- Output structured JSON responses

Sample Query:

"Does this policy cover knee surgery, and what are the conditions?"

Q.2 SYSTEM ARCHITECTURE & WORKFLOW

2.1) Design and implement the following system components:

1

Input Documents

PDF Blob URL

2

LLM Parser

Extract structured query

3

Embedding Search

FAISS/Pinecone retrieval

4

Clause Matching

Semantic similarity

5

Logic Evaluation

Decision processing

6

JSON Output

Structured response

Q.3 EVALUATION PARAMETERS

3.1) Your solution will be evaluated based on the following criteria:

a **Accuracy**
Precision of query understanding and clause matching

b **Token Efficiency**
Optimized LLM token usage and cost-effectiveness

c **Latency**
Response speed and real-time performance

d **Reusability**
Code modularity and extensibility

Explainability

Clear decision reasoning and clause traceability

Q.4 RETRIEVAL SYSTEM API DOCUMENTATION

Base URL (Local Development):

```
http://localhost:8000/api/v1
```

Authentication:

```
Authorization: Bearer 5f5247a861f9c5c82791ab8ebdded7644021b6ff466ee2ba5b6a61328f83c9a0
```

✅ Team token loaded successfully

API Endpoints Overview

POST /hackrx/run

Run Submissions

Sample Upload Request:

```
POST /hackrx/run
Content-Type: application/json
Accept: application/json
Authorization: Bearer 5f5247a861f9c5c82791ab8ebdded7644021b6ff466ee2ba5b6a61328f83c9a0

{
    "documents": "https://hackrx.blob.core.windows.net/assets/policy.pdf?sv=2023-01-03
    "questions": [
        "What is the grace period for premium payment under the National Parivar Medic
        "What is the waiting period for pre-existing diseases (PED) to be covered?",
        "Does this policy cover maternity expenses, and what are the conditions?",
        "What is the waiting period for cataract surgery?",
        "Are the medical expenses for an organ donor covered under this policy?",
        "What is the No Claim Discount (NCD) offered in this policy?",
        "Is there a benefit for preventive health check-ups?",
        "How does the policy define a 'Hospital'?",
        "What is the extent of coverage for AYUSH treatments?",
        "Are there any sub-limits on room rent and ICU charges for Plan A?"
    ]
}
```

Sample Response:

```
{
"answers": [
        "A grace period of thirty days is provided for premium payment after the due d
        "There is a waiting period of thirty-six (36) months of continuous coverage fr
        "Yes, the policy covers maternity expenses, including childbirth and lawful me
        "The policy has a specific waiting period of two (2) years for cataract surger
        "Yes, the policy indemnifies the medical expenses for the organ donor's hospit
        "A No Claim Discount of 5% on the base premium is offered on renewal for a one
        "Yes, the policy reimburses expenses for health check-ups at the end of every
        "A hospital is defined as an institution with at least 10 inpatient beds (in t
        "The policy covers medical expenses for inpatient treatment under Ayurveda, Yo
        "Yes, for Plan A, the daily room rent is capped at 1% of the Sum Insured, and
    ]
}
```

Recommended Tech Stack:

FastAPI
Backend

Pinecone
Vector DB

GPT-4
LLM

PostgreSQL
Database

Q.5 SCORING DOCUMENTATION

📘 Scoring System Explanation

1. **Document Types**
   - *Known Documents*: Publicly available
   - *Unknown Documents*: Private & unseen
2. **Document-Level Weightage**
   - *Known Documents*: Low weightage (e.g., 0.5)
   - *Unknown Documents*: High weightage (e.g., 2.0)
3. **Question-Level Weightage**
   - Each question may have its own weight (e.g., some are worth more due to complexity or importance).
4. **Score Calculation**
   - For each correct answer:
     `Score = Question Weight × Document Weight`
   - Final score is the **sum of all such scores** across all documents.

📊 Example

Documents:

- *Doc A (Known)* – Weight: 0.5
- *Doc B (Unknown)* – Weight: 2.0

Questions:

Question	**Document**	**Answered Correctly?**	**Question Weight**	**Score Contribution**
Q1	Doc A	✅ Yes	1.0	1.0 × 0.5 = **0.5**
Q2	Doc A	❌ No	2.0	0
Q3	Doc B	✅ Yes	1.0	1.0 × 2.0 = **2.0**
Q4	Doc B	✅ Yes	2.0	2.0 × 2.0 = **4.0**

🧮 **Total Score:**

- *Doc A Contribution*: 0.5
- *Doc B Contribution*: 2.0 + 4.0 = 6.0
- **Final Score = 0.5 + 6.0 = 6.5**

✅ Key Points

- Correct answers from **unknown documents** contribute more.
- **High-weight questions** boost your score significantly.
- **Incorrect or unattempted questions** contribute 0, regardless of weight.