Assignment 1: Gen AI Application

Problem Statement

You are required to build a **Retrieval-Augmented Generation (RAG)** application with **agentic behavior**, a **Streamlit UI**, and a **FastAPI backend**.

Your tasks:

1. **Create a Knowledge Base**

- Use any publicly available dataset (e.g., airline SOP documents, financial policies, retail product descriptions, etc.).
- Convert documents into embeddings and store them locally (e.g., FAISS, ChromaDB).

---

2. **Build a RAG Pipeline (in FastAPI)**

Your FastAPI application must expose the following endpoints:

1. **/ask**
    - Input: { "query": "..." }
    - Output: A GenAI response grounded in retrieved documents.
2. **/agent-ask**
    - Implement a simple **task-solving agent** that:
        - Breaks a user query into smaller sub-queries
        - Retrieves relevant context
        - Produces a final synthesized answer
    - Example queries the agent should handle:
        - "Compare policies between Document A and B and summarize differences."
        - "Find steps from all SOPs and create a combined checklist."

---

3. **Streamlit Application**

Build a frontend with:

- A text input box with chat interface

- A dropdown to select **RAG** or **Agent Mode**
- Display retrieved context + final answer
- Call your FastAPI backend

---

4. **Deployment**

You must deploy the app so we can test it:

- Provide:
    - Public URL for testing

---

5. **Submission Requirements**

- Submit a **ZIP file of your codebase**, containing:
    - FastAPI backend code
    - Streamlit UI
    - RAG pipeline
    - Agent logic
    - Requirements.txt
- A video explaining the working of the GenAI application

Assignment 2: ML Lifecycle

Problem Statement

Consider any open dataset -
Example: Telco Customer Churn dataset, Loan Default dataset, or any Classification dataset.

Your task:

1.Perform EDA

- Identify missing values, outliers
- Understand feature distributions
- Identify key drivers of the target variable

- Provide visualizations
- Summarize insights in business terms

---

2. Preprocessing & Feature Engineering

- Handle missing values
- Encode categorical features
- Scale numerical variables if needed
- Engineer at least **2 new features**

---

3. Build and Train Models

Train at least **three different models**, e.g.:

- Logistic Regression
- Random Forest
- XGBoost / LightGBM

---

4. Evaluate Model Performance

Include:

- Accuracy
- Precision, Recall, F1
- ROC curve & AUC
- Confusion matrix
- Feature importance
- Business interpretation of results

---

5. Select the Best Model

Explain:

- Why this model performs best
- How it would be deployed

- What monitoring would look like in production

---

6. Deliverables

Submit a ZIP containing:

- Jupyter notebook
- Dataset (or link)
- Model files
- A video explaining the solution & the full life cycle