End-to-End MLOps Pipeline

Pipeline Stages

1. **Data Ingestion**:
    - Upload raw data (PDFs/Word) through the FastAPI backend or UI (Streamlit).
    - Store the uploaded documents temporarily (e.g., **Blob Storage**, **S3**, or local).
2. **Tools**: FastAPI, Streamlit, Cloud Storage (Azure Blob, S3).

3. **Data Preprocessing**:
    - Extract and chunk the documents into smaller sections.
    - Generate embeddings for text chunks using **Sentence Transformers**.
4. **Tools**: Sentence Transformers, Python NLP libraries (e.g., `pdfminer`, `python-docx`).

5. **Vector Database Integration**:
    - Store generated embeddings and metadata (e.g., document names, sections) into a **vector database**.
    - Perform semantic search for user queries.
6. **Tools**: Milvus (or alternatives like Pinecone, ChromaDB).

7. **Model Serving**:
    - Use **FastAPI** for serving models and handling API endpoints.
    - Query embeddings and perform top-k similarity searches.
8. **Tools**: FastAPI, Uvicorn.

9. **Monitoring**:
    - Monitor system performance (e.g., API latency, query response time) using **Prometheus** and visualize with **Grafana**.
    - Collect feedback on incorrect responses for improvement.
10. **Tools**: Prometheus, Grafana.

11. **Performance Evaluation**:
    - Evaluate accuracy using BLEU Score or other NLP metrics for a pre-defined question-answer set.
    - Log results for future comparisons.
12. **Tools**: BLEU Score, Python Libraries (e.g., `nltk`), MLflow.

13. **Feedback Collection**:
    - Capture user queries and responses to identify incorrect answers.
    - Store this feedback for further processing.
14. **Tools**: FastAPI (to handle queries), Cloud Storage or Databases.

15. **Retraining Pipeline**:
    - Automate retraining pipelines for generating updated embeddings or improving the LLM model.
    - Use **Airflow** or **Kubeflow** to schedule retraining jobs.
    - Register new models in a **model registry**.
16. **Tools**: Apache Airflow / Kubeflow, MLflow for versioning.

17. **CI/CD for Deployment**:
    - Use CI/CD pipelines to automate:
        - Model updates.
        - API testing and deployment using Docker.
    - Monitor deployment health with tools like Kubernetes and Docker Swarm.
18. **Tools**: GitHub Actions, Jenkins, Docker, Kubernetes.

Performance Evaluation

Proposed Method for Production

1. **Accuracy-Based Evaluation**:
    - Use a test set of question-answer pairs for the uploaded documents.
    - Evaluate the chatbot's responses using metrics like:
        - **Accuracy**: Fraction of correct answers.
        - **BLEU Score**: Evaluate text overlap between generated answers and ground truth.
2. **Response Time**:
    - Track latency for document upload and query responses.
3. **System Monitoring**:
    - Use tools like **Prometheus** for metrics collection and **Grafana** for visualization.
    - Metrics include:
        - Query latency
        - API uptime
        - CPU/memory usage