Assignment 2

1. Where would you rate yourself on (LLM, Deep Learning, AI, ML)? A, B, C [A = can code independently; B = can code under supervision; C = have little or no understanding]

Ans: **Machine Learning: A –** I can build end-to-end models and write code independently.

Deep Learning: B – I understand core concepts and can implement models with supervision.

LLMs: B – I understand how LLM-based systems work and can build basic applications using existing frameworks and APIs with guidance.

2. What are the key architectural components to create a chatbot based on LLM? Please explain the approach on a high level.

Ans:1) **User Interface (UI):** The UI is the front layer where users interact with the chatbot. Users type questions and see responses in real time. This can be a web UI, mobile app, or chat platform like Dialogflow or Telegram.

2) **API:** The API acts as the bridge between the UI and the AI system. It receives user messages, manages requests, and sends them to the LLM. Frameworks like FastAPI are commonly used because they are fast.

3) **LLM (Large Language Model):** The LLM is the main component of chatbot. It understands user intent, processes language, and generates responses. Models like BERT (for understanding) and GPT (for generating) are commonly used.

4) **Context Handling:** This component prepares the final input fot the LLM like user's message, system instructions and conversation history. All this is formatted into a single prompt so the LLM can respond accurately.

5) **Database:** Database stores documents and company data. As we know in RAG setup, relevant information is retrieved from the database and added to the prompt. This allows the chatbot to give accurate and required answers instead of relying only on memory.

6) **Processing of Response:** Before sending the response to the user, the output is cleaned and processed. This includes removing unnecessary text, formatting the answer, applying filters. The final polished response is shown to the user via UI.

3. Please explain vector databases. If you were to select a vector database for a hypothetical problem (you may define the problem) which one will you choose, and why?

Ans: Vector databases, in simple terms, are used to store embeddings, and an embedding is just a vector form of representing data. It can be text, images, audio, or even videos. Vector databases make it easy to find relationships between vectors.

For example, if we are making a movie recommendation app and a user has watched Interstellar and the next movie they want to watch is Oppenheimer, it becomes easier if the movies are stored in a vector database because the vector database can easily find relationships between these vectors or movies and automatically recommend the next related movie.

Hypothetical problem:

Suppose we are building a small chatbot or recommendation system where we need to store embeddings and find similar data based on meaning.

Vector database choice:

I would choose FAISS.

Why:

FAISS is simple to use and works well with Python. It can store embeddings and easily find relationships between vectors based on similarity. It also runs locally, so there is no need for complex setup, which makes it good for small to medium projects and learning purposes.