

GenAI & Agentic AI – Day 1 Detailed Revision

1. Purpose of This Project

This project is designed to help transition from being an AI tool user to becoming an AI system builder. The goal is to deeply understand how Generative AI systems work internally and how to architect scalable AI applications.

2. What is a Large Language Model (LLM)?

An LLM is a next-token prediction machine. It does not think or understand like humans. It predicts the next word based on patterns learned during training. Its intelligence is emergent from large-scale statistical learning.

3. What Happened in Day 1 Technically?

We installed Ollama (local model runner), downloaded LLaMA3, connected Python to the local model, sent prompts, and observed how output changes based on instructions.

4. What is a Token?

LLMs do not process full sentences. They process tokens, which are chunks of text. Every response is generated token by token. This is why token limits and cost matter in cloud APIs.

5. Prompt Engineering Basics

The model follows patterns. Clear instructions lead to controlled outputs. Constraints such as 'Return JSON', 'One sentence only', or 'Explain like a 5-year-old' guide behavior.

6. Local LLM Architecture

Python Script -> Ollama Server (localhost) -> LLaMA3 Model -> Token Generation -> Output Displayed. Everything runs locally without internet dependency.

7. Why We Avoid Hardcoding in Production

Hardcoding prompts is for learning only. In production systems, input comes from APIs, users, or other services. The architecture becomes modular and scalable.

8. Difference Between Basic LLM and Agentic AI

Basic LLM: Prompt -> Response. Agentic AI: Goal -> Plan -> Tool Usage -> Memory -> Iteration -> Final Response.

9. Security Practice Learned

Never expose API keys. Use .env files. Never push secrets to GitHub. If exposed, immediately revoke the key.

10. Outcome After Completing Entire Study Plan

By the end of this study plan, you will be able to: - Build GenAI applications using LLMs - Implement Retrieval-Augmented Generation (RAG) - Create AI agents with tool usage - Structure AI backend using FastAPI - Deploy AI systems professionally - Explain AI architecture confidently in interviews