

Generative AI, Agentic AI & RAG Engineering

10-Day Training Curriculum | 30 Hours Total | 3 Hours / Day

Covers: AI/ML/DL/GenAI/Agents foundations · Python · Prompt Engineering · Transformers & LLMs · GenAI Applications · LangChain · RAG (all types) · Agentic AI · LangGraph · CrewAI · Agentic RAG.. Hands-on labs use OpenAI / Gemini API keys with Python, LangChain & LangGraph.

Day	Topic	Detailed Content	Project / Hands-on Practical
Day 1 3 Hours	Introduction to AI, ML, DL, GenAI, Agents & Agentic AI + Python Environment Setup	Conceptual Foundations • Introduction to AI vs ML vs Deep Learning vs GenAI • Overview of AI-assisted metadata management • Generative AI: foundation models, transformers overview, capabilities & limitations • Agentic AI: autonomous decision-making, tool use, planning vs simple GenAI response generation • GenAI applications deep-dive: Text Generation (support replies), Image Generation (network diagrams/tower imagery), Audio Generation (IVR voice synthesis), Video Generation (training content), Code Generation (automation scripts) Python Environment Setup • Installing Python, Anaconda/venv, VS Code and Jupyter Notebook • Setting up a virtual environment and installing packages with pip • Introduction to Google Colab as an alternative sandbox	
Day 2 3 Hours	Python Fundamentals — Functions, OOP, File Handling & APIs	Core Python • Python basics: variables, data types, operators • Python control flow (if/else, loops) • Data structures deep-dive: lists, tuples, dictionaries, sets • List/Dict Comprehensions • Functions, *args/**kwargs, lambda functions • Exception handling (try/except/finally) and custom exceptions	• Hands-on: Configure local Python + Jupyter/Colab environment end-to-end • Hands-on: Write basic Python scripts (calculator, temperature converter, grade evaluator)
Day 3 1.5 Hours	Prompt Engineering	Prompting Foundations • Anatomy of a good prompt: instruction, context, input data, output format • Zero-shot, one-shot and few-shot prompting techniques • Chain-of-Thought (CoT), Self-Consistency and Tree-of-Thought prompting • Role-based prompting, system vs user vs assistant messages • ReAct prompting pattern (reasoning + acting) — foundation for agents	• Setup: Generate Gemini API key (Google AI Studio) and OpenAI API key • Hands-on: Make first API calls to Gemini & OpenAI from Python

Day	Topic	Detailed Content	Project / Hands-on Practical
		- Controlling output: temperature, top-p, max tokens, stop sequences - Prompt templates and reusable prompt design patterns	- Hands-on: Notebook demonstrating 5+ prompting patterns (zero-shot, few-shot, CoT, role-based, ReAct) - Mini-Project: Build a Prompt-based Text Summarizer & Q&A CLI Bot using OpenAI/Gemini API
Day 3 1.5 Hours	Transformers Architecture & Large Language Models (LLMs)	Transformer Architecture - LLM, Transformer architecture 'Attention is All You Need' — architecture walkthrough - Self-attention and multi-head attention explained intuitively - Encoder-decoder architecture; encoder-only (BERT) vs decoder-only (GPT) models - Tokenization techniques and embeddings - Positional encoding and why order matters Large Language Models - How LLMs are trained: pre-training, fine-tuning, RLHF/alignment - Context window, parameters, and scaling laws (brief overview) - Comparing popular LLMs: GPT-4/4o, Gemini, Claude, Llama, Mistral - LLM limitations: hallucination, bias, knowledge cutoff, cost/latency	- Hands-on: Visualize tokenization using tiktoken / Gemini tokenizer - Hands-on: Generate embeddings and compute cosine similarity between text pairs - Mini-Project: Build a simple Semantic Search tool over a small text corpus using embeddings
Day 4 3 Hours	Generative AI Applications (OpenAI & Gemini SDKs)	GenAI Application Landscape - Text, image, audio and code generation use cases across industries - Multi-modal AI: combining text + image + voice SDKs & Advanced API Usage - OpenAI Python SDK deep-dive: chat completions, streaming - Google Gemini Python SDK deep-dive: generate_content, chat sessions - Function calling / tool calling for structured outputs - Structured output with JSON schema / Pydantic models - Token usage, cost estimation and response streaming for better UX	- Hands-on: Build a streaming AI Chatbot (console/Streamlit) using OpenAI/Gemini - Hands-on: Build a Resume/Invoice Parser using function calling for structured JSON extraction - Mini-Project: AI-powered Blog/Content Generator App with tone & length controls
Day 5 3 Hours	LangChain Framework	LangChain Core Concepts - Why LangChain — orchestrating LLM applications - Models: LLMs vs ChatModels; integrating OpenAI/Gemini via LangChain - PromptTemplates and ChatPromptTemplates - Chains: LLMChain, SequentialChain, and LangChain Expression Language (LCEL)	- Hands-on: Build multi-step LCEL chains (translate → summarize → format) - Hands-on: Build a Conversational Chatbot with memory using LangChain + Gemini/OpenAI

Day	Topic	Detailed Content	Project / Hands-on Practical
		• Output Parsers (String, JSON, Pydantic parsers) • Memory: ConversationBufferMemory, ConversationSummaryMemory • Document Loaders (PDF, text, web, CSV) and Text Splitters • Introduction to Vector Stores (FAISS/Chroma) — preparing for RAG • Debugging & tracing with LangSmith (overview)	• Mini-Project: Document Q&A Pipeline (load PDF → split → chain → answer questions)
Day 6 3 Hours	Retrieval-Augmented Generation (RAG) — Types of RAG	RAG Fundamentals • What is RAG and why it solves hallucination & knowledge-cutoff problems • RAG architecture: retriever + generator; embeddings & vector databases • Vector databases: FAISS, ChromaDB, Pinecone — setup and comparison • Chunking strategies (fixed size, recursive, semantic chunking) Types of RAG • Naive RAG vs Advanced RAG (re-ranking, hybrid search — keyword + vector) • Corrective RAG (CRAG) and Self-RAG (self-reflection on retrieved content) • Adaptive RAG and Multi-hop RAG for complex, multi-step questions • Graph RAG (knowledge-graph based retrieval) and HyDE (Hypothetical Document Embeddings) • Evaluating RAG pipelines with RAGAS (faithfulness, relevancy, precision/recall)	• Hands-on: Build an end-to-end RAG pipeline for PDF Q&A using LangChain + Gemini/OpenAI + ChromaDB/FAISS • Hands-on: Add re-ranking and hybrid search to improve retrieval quality • Mini-Project: 'Chat with your Documents' RAG App with source citation and RAGAS-based evaluation
Day 7 3 Hours	AI Agents & Agentic AI Fundamentals + Building Agents with LangChain	Agentic AI Fundamentals • What is an AI Agent — difference between a chatbot and an agent • Agent components: planning, memory, tools, reasoning loop (ReAct pattern) • Types of agents: reactive, deliberative, hybrid; single-agent vs multi-agent systems • Autonomy, goal-directed behavior and decision-making in Agentic AI • LangChain Agent types: Zero-shot ReAct Agent, Tool-Calling / OpenAI Functions Agent • Tools & Toolkits: built-in tools (search, calculator, Python REPL) • Creating custom tools with @tool decorator • Agent executors, intermediate steps and handling agent errors	• Hands-on: Build a LangChain agent with custom tools (calculator, web search, weather API) • Hands-on: Build a ReAct-based Research Assistant Agent that reasons and calls tools step-by-step
Day 8 1.5 Hours	Agentic RAG	Agentic RAG • What is Agentic RAG — agent decides when, how and where to retrieve • Self-correcting/reflective RAG agents (query rewriting, retry on poor retrieval) • Combining LangGraph + RAG for dynamic, multi-source retrieval routing	• Mini-Project: Personal Task Automation Agent (e.g., email drafter + calculator + search agent)

Day	Topic	Detailed Content	Project / Hands-on Practical
Day 8 1.5 Hours	Agent Orchestration using LangGraph	LangGraph - Multi-agent architectures (supervisor, hierarchical, collaborative) - Agent communication protocols, shared state management - Parallel execution - subgraph composition. - Build supervisor + worker agent system - Implement parallel tool execution - Add agent handoff protocols - Integrate with external APIs - Build market analysis multi-agent	Hands-on: Build a multi-agent LangGraph workflow (Researcher → Writer → Reviewer)
Day 9 3 Hours	Agent Orchestration using CrewAI	CrewAI - CrewAI framework concepts — crews, agents, tasks, processes (sequential, hierarchical) - Delegation, backstory design, memory systems - Tools sharing between agents - Build Single-Agent - Multi-Tool Agent crew - Human feedback loop, YouTube content pipeline	- Hands-on: Build the same workflow using CrewAI and compare developer experience - Mini-Project: Automated Content Pipeline — LangGraph/CrewAI multi-agent system producing a research-backed article
Day 10 3 Hours	Performance Fine-Tuning	Fine tuning on LLM model - Fine-tuning paradigms: full fine-tuning vs Parameter-Efficient Fine-Tuning (PEFT) - LoRA mechanics: low-rank adaptation matrices, rank selection trade-offs - QLoRA mechanics: 4-bit quantization combined with LoRA for memory-efficient fine-tuning - Dataset preparation: instruction format, chat template structuring, data cleaning for fine-tuning - RLHF overview: reward modeling and policy optimization at a conceptual level - Fine-tuning LLaMA with QLoRA on a custom dataset using HuggingFace (Transformers + PEFT + TRL libraries) - Preparing an instruction dataset from raw domain text - Evaluating fine-tuned models with benchmark metrics	- Fine-tune LLaMA-3-8B with QLoRA (4-bit) to specialize in telecom troubleshooting responses - Evaluate fine-tuned model with ROUGE/BLEU and human evaluation against actual support-agent replies

Prerequisites & Tools Required

- Laptop with Python 3.11+, VS Code / Jupyter / Google Colab access
- Free API keys: OpenAI (platform.openai.com) and Google Gemini (Google AI Studio) — some usage may incur cost beyond free tier
- Basic programming familiarity is helpful but not mandatory — Day 1-2 covers Python fundamentals from scratch
- Key libraries used: langchain, langgraph, crewai, openai, google-generativeai, chromadb/faiss-cpu, streamlit, python-dotenv