Building AI Agents with LLMs

Duration: 40 hours

Course Overview

Large language models (LLMs) have opened new possibilities for building **agentic systems**—software that can reason, use tools, integrate external data, and collaborate with humans (and other agents) to accomplish complex tasks. This course focuses on building **highly controllable, robust, and evaluable** AI agents using cutting-edge frameworks like **LangGraph, AutoGen, LangChain,** and **LlamaIndex**.

By the end of this course, learners will have a practical understanding of how to **design**, **build**, and **evaluate** AI agents that can:

- Use **search, retrieval, and indexing** to enhance their knowledge.
- Incorporate **long-term memory** and **human feedback**.
- **Collaborate** with other agents and human operators.
- **Evaluate**, debug, and improve their performance systematically.

Course Audience

- **Intermediate Python developers** who are comfortable writing scripts, installing packages, and using Python data structures.
- Anyone who has **basic familiarity** with large language models (e.g., ChatGPT or GPT-4) or has done simple prompt engineering before.
- **Developers or data scientists** looking to integrate AI agents into real-world applications or research projects.

Prerequisites

- **Python** programming skills (comfortable with virtual environments, pip/conda installs).
- Basic understanding of **LLMs**, prompt engineering, and standard **API usage** (e.g., OpenAI, Anthropic, or local LLM APIs).
- Familiarity with **Git** or version control recommended, for managing course projects.

Course Modules

Module 01: Agentic Fundamentals

Explore the basics of AI agents, their architecture, and how they differ from single-call LLM applications

- Understand what an AI agent is and how it differs from standard single-call LLM applications.
- Learn to decompose an agent-based system into tasks, tools, memory, and a decision-making loop.

Module 02: Building Agents from Scratch

Learn to manually invoke LLMs, parse reasoning, and separate responsibilities between language models and environment code

- See how to manually call an LLM to parse, reason, and perform actions.
- Understand the division of tasks between the LLM ("brain") and your code ("environment").

Module 03: Working with LangGraph

Discover LangGraph's components to build flow-based LLM applications, orchestrate branching logic, and enable advanced agentic search

- Explore **LangGraph**'s core components for designing AI workflows: nodes, edges, agent controllers.
- Learn how to build a **flow-based application** that orchestrates LLM calls, tools, and branching logic.
- Add advanced capabilities like **agentic search** for returning multiple structured answers.

Module 04: Multi-Agent Systems with AutoGen

Create collaborative AI agents with specialized roles, employing reflection, tool-use, planning, and code-validation for complex tasks

- Create agents that can **collaborate**, each with different roles (e.g., coder, reviewer, editor).
- Employ design patterns such as **reflection**, **tool-use**, **planning**, and **multi-agent conversation**.
- Integrate code-generation and validation steps for tasks like building custom plots or analyzing data.

Module 05: Functions, Tools & Routing in LangChain

Utilize OpenAI's function-calling and LCEL for simplified tool usage, building a conversational agent with structured function calls

- Use **OpenAI's function-calling** and the **LangChain Expression Language (LCEL)** to simplify tool usage and chain construction.
- Build a **conversational agent** capable of responding with structured tool calls (e.g., function calling) for tasks like extraction, classification, or other tool-based workflows.

Module 06: Event-Driven Document Workflows with LlamaIndex

Implement asynchronous agentic workflows with LlamaIndex, parsing documents, retrieving relevant text, and incorporating human-in-the-loop interactions

- Understand how to build **event-driven agentic workflows** that respond to triggers or tasks asynchronously.
- Use **LlamaIndex** to parse documents, embed them, and retrieve relevant text for a question-answer loop (RAG).
- Implement **human-in-the-loop** steps for a document processing agent (e.g., form-filling, summarization).

Module 07: Long-Term Memory & Continual Learning

Add semantic, episodic, and procedural memory to agents, enabling dynamic prompt refinement and adaptive behavior over time

- Distinguish between **semantic, episodic, and procedural** memory in an agent.
- Add a **long-term memory** store (e.g., vector DB or knowledge graph) that grows with user interactions.
- Incorporate memory retrieval to refine system prompts and adapt behaviors.

Module 08: Evaluating & Monitoring AI Agents

Learn observability techniques, systematic evaluation methods, and experiment tracking to debug, improve, and deploy production-ready AI agents

- Add **observability**: trace each step of the agent's thought process, tool usage, and final outputs.
- Configure systematic **evaluation** of correctness, efficiency, consistency, and user satisfaction.
- Compare **code-based metrics** vs. "LLM-as-a-judge" evaluations vs. **human feedback**.

    - Structure evaluations into **experiments** for iterative improvement and deploy **monitoring** to track performance in production.