Course Syllabus: Agentic AI Workflow Engineer

Course Title: Agentic AI Engineering: Designing and Orchestrating Autonomous Workflows

Target Audience: This course is for advanced developers, MLOps engineers, and software architects who have a strong foundation in Python and a solid understanding of Large Language Models (LLMs).

Course Level: Expert.

Duration: 10 Weeks

Course Description: This course provides a deep, hands-on dive into the world of agentic workflows and autonomous AI. You will learn to architect and build systems where multiple agents collaborate to achieve a goal, utilizing tools, memory, and advanced reasoning. The curriculum emphasizes practical application using leading frameworks and MLOps principles to ensure your agents are not just functional, but also robust, scalable, and safe in production.

Learning Objectives

Upon successful completion of this course, students will be able to:

- Design and implement the core components of a single AI agent, including planning, memory, and tool use.
- Master leading agentic frameworks like CrewAI, LangGraph, and AutoGen.
- Architect and build multi-agent systems where agents collaborate and delegate tasks.
- Integrate agents with external APIs and databases to extend their capabilities.
- Apply advanced reasoning and self-correction techniques (e.g., Reflection, ReAct) to enhance agent performance.
- Implement MLOps practices for deploying, monitoring, and managing agentic workflows.
- Understand the ethical considerations, safety protocols, and governance required for autonomous AI systems.

Course Structure: A Step-by-Step Learning Path

Part 1: Foundational Agent Architectures (Weeks 1-3)

This section establishes the core theory and practical skills for building the building blocks of any agentic system.

Week 1: The Anatomy of an AI Agent

- The Agentic Paradigm: from static programs to autonomous systems.
- Core components of an agent: Perception, Planning, Memory, Reasoning, and Action.
- The Agent Loop: The cycle of Observe, Think, and Act.
- Hands-on Lab: Build a simple, single-task agent from scratch in Python to understand the agent loop.

Week 2: Advanced Reasoning & Self-Correction

- **ReAct** (Reasoning and Acting): A core pattern for multi-step tasks.
- **Reflection:** Teaching an agent to critique its own work and correct its mistakes.
- **Planning:** How agents break down a complex goal into a sequence of executable steps.
- Hands-on Lab: Build an agent that uses a reflective loop to improve its response to a complex query.

Week 3: Tool Use & External Integration

- The importance of **tool use** for extending an agent's capabilities beyond its LLM.
- Creating custom tools and wrappers for APIs (e.g., web search, calculators, database queries).
- Integrating agents with external data sources and services.
- Hands-on Project: Develop an agent that can access an external weather API to provide real-time information.

Part 2: Orchestration with Modern Frameworks (Weeks 4-7)

This section focuses on using specialized frameworks to build and coordinate complex, multi-agent workflows.

Week 4: Agent Orchestration with LangGraph

- Introduction to **LangGraph:** A framework for building robust, stateful agentic systems.
- Defining states, nodes, and edges in a graph-based workflow.
- Creating flexible, looping, and branching workflows for dynamic tasks.
- Hands-on Lab: Build a multi-step agent workflow using LangGraph to automate a simple research task.

Week 5: Collaborative Agents with CrewAI

- Introduction to **CrewAI:** A framework for building multi-agent systems where agents have specific roles.
- Designing agent roles, tasks, and collaboration protocols.
- The concept of a "crew" and how agents delegate and cooperate.
- Hands-on Project: Build a crew of agents to perform a collaborative task, such as writing a blog post (one agent for research, one for writing, and one for editing).

Week 6: Building Scalable Teams with AutoGen

- Introduction to **AutoGen:** Microsoft's framework for building multi-agent conversations.
- Designing agents that can communicate and negotiate with each other.
- Creating agent teams for coding, data analysis, and software development tasks.
- Hands-on Lab: Set up a coding team with AutoGen to write and debug a small Python script.

Week 7: Agent Memory & State Management

- The importance of persistent memory for long-running agent workflows.
- Using vector databases (e.g., ChromaDB, Pinecone) for long-term memory.
- Strategies for managing the state of an agentic workflow to ensure continuity and debugging.
- Hands-on Project: Upgrade an existing agent to use a vector database for long-term memory.

Part 3: Deployment, MLOps, and Expert Practice (Weeks 8-10)

This final section focuses on the practical aspects of deploying and maintaining agentic systems in production environments.

Week 8: MLOps for Agentic Workflows

- The unique challenges of MLOps for autonomous agents.
- Containerization with **Docker** for consistent and reproducible deployments.
- Monitoring agent performance, costs, and outputs in real-time.
- Hands-on Lab: Dockerize a multi-agent system and prepare it for deployment.

Week 9: Cloud Deployment & Safety Guardrails

- Deploying agentic workflows on major cloud platforms (e.g., AWS, GCP).
- Implementing human-in-the-loop protocols for critical decision points.
- Creating safety checks and guardrails to prevent unintended consequences.
- Hands-on Project: Deploy your multi-agent system to a cloud platform and implement a basic safety check.

Week 10: Final Project & Industry Trends

- **Capstone Project:** Design, build, and deploy a complete, professional-grade agentic workflow to solve a complex, real-world business problem.
- Review of the latest research, trends, and career paths in the agentic AI space.
- Case studies of successful agentic systems in various industries.

Assignments & Grading

- **Weekly Hands-on Labs & Exercises:** 20%
- **Intermediate Projects (Weeks 3 & 7):** 30%
- **Final Capstone Project:** 40%
- **Code Quality & Workflow Design:** 10%