Agentic AI Developer

6-Week Course Syllabus

Offered by **Vipracube AI Technologies**

LLMs | LangChain | RAG | Tool Calling & MCP | LangGraph | Agent Harness

Duration	6 weeks
Who is it for	Developers and learners who already know Python and want to specialize in building Agentic AI applications.
Prerequisites	Working knowledge of Python (functions, OOP, file handling, JSON) and basic understanding of REST APIs. Familiarity with FastAPI is helpful.
Outcome	Build, evaluate, and deploy LLM applications, RAG systems, tool-calling and MCP-connected agents, multi-agent workflows, and production-ready agent harnesses.

Course Roadmap

Week	Focus
Week 1	Generative AI & LLM Fundamentals
Week 2	LangChain Core
Week 3	RAG & Advanced RAG
Week 4	Tool Calling, MCP & Agents
Week 5	LangGraph & Agentic Workflows
Week 6	Agent Harness, Evaluation, Deployment & Capstone

Weekly Syllabus: Agentic AI with LangChain & LangGraph

Weeks 1 - 6

Week 1: Generative AI & LLM Fundamentals

- How LLMs work: tokens, context windows, temperature, sampling
- Overview of model providers (OpenAI, Anthropic, Google, open-source via Hugging Face/Ollama)
- Working with LLM APIs directly
- Prompt engineering: zero-shot, few-shot, chain-of-thought, system prompts
- Structured outputs (JSON mode, Pydantic schemas)
- API key management, cost, and rate limits

> **Lab / Project:** Prompt-driven text summarizer/classifier served through FastAPI

Week 2: LangChain Core

- LangChain architecture and components
- Chat models, prompt templates, and output parsers
- LCEL (LangChain Expression Language): chains, runnables, parallelism, streaming
- Conversation memory and message history
- Document loaders (PDF, web, CSV) and text splitters

Lab / Project: Multi-step chains, including a chatbot with memory

Week 3: RAG & Advanced RAG

- Why RAG: hallucinations, knowledge cutoffs, private data
- Embeddings and semantic search
- Vector databases: FAISS, Chroma, Pinecone/Qdrant
- Chunking strategies and retrievers
- Building an end-to-end RAG pipeline
- Advanced RAG: hybrid search, re-ranking, query rewriting, multi-query retrieval, metadata filtering

Lab / Project: Mid-course Project - "Chat with your PDFs/documents" application with advanced retrieval

Week 4: Tool Calling, MCP & Agents

- Tool/function calling: how LLMs select tools, JSON schemas, parallel tool calls, error handling
- Building custom tools in LangChain; integrating external APIs, web search, databases, and code execution
- Agent concepts: ReAct pattern, reasoning and acting loops
- Model Context Protocol (MCP): architecture (host, client, server), tools, resources, and prompts
- Building an MCP server in Python and choosing transports (stdio, streamable HTTP)
- Connecting MCP servers to LangChain/LangGraph agents; using existing community MCP servers
- MCP security basics: authentication, permissions, and trusting third-party servers

Lab / Project: Build a custom MCP server (e.g., database or business-API tools) and connect it to a tool-using agent

Week 5: LangGraph & Agentic Workflows

- Why LangGraph: limitations of linear chains, graph-based orchestration
- State, nodes, edges, and conditional routing
- Loops, cycles, and self-correction (reflection patterns)
- Persistence and checkpointing, short- and long-term memory
- Human-in-the-loop: interrupts, approvals, resuming workflows
- Multi-agent patterns: supervisor, hierarchical, and collaborative agents

Lab / Project: Multi-agent workflow (e.g., research, write, and review pipeline)

Week 6: Agent Harness, Evaluation, Deployment & Capstone

- What is an agent harness: the runtime around the LLM that runs the agent loop, tools, context, and memory
- Harness components: tool execution layer, context management (compaction, summarization), planning/task lists, sub-agents, workspace/file access
- Safety and control: permissions, sandboxed code execution, guardrails, hooks, turn/cost limits, prompt-injection basics
- Building a custom harness with LangGraph; overview of existing harness frameworks (e.g., LangChain Deep Agents, Claude Agent SDK)
- Observability and tracing with LangSmith; evaluating agents and RAG: test datasets, trajectory evaluation
- Deploying: Docker, environment configuration, cloud deployment (Render/AWS/Azure)
- Simple frontend integration (Streamlit or basic React/HTML); capstone demos and feedback

> **Lab / Project: Capstone Project** - Build, deploy, and present an agentic AI application using tool calling/MCP and a custom agent harness

Capstone ideas: AI customer support agent with RAG; resume/job-matching assistant; document Q&A platform with authentication; multi-agent research assistant; SQL-powered business analytics chatbot; MCP-powered DevOps or productivity assistant.

Assessment Structure (Suggested)

Component	Weightage
Weekly assignments / labs	35%
Mid-course project: RAG application (end of Week 3)	20%
Capstone project	45%

Tools & Tech Stack

Python, FastAPI, LangChain, LangGraph, LangSmith, MCP (Python SDK/FastMCP), ChromaDB/FAISS, Pinecone/Qdrant, OpenAI/Anthropic/Google APIs, Ollama, Docker, Git/GitHub, Streamlit.

Enroll / Contact

Vipracube AI Technologies

For more details, please contact: **9960923965**