PROFESSIONAL COURSE SYLLABUS

AI Agent for Developers

המכללה לניו-מדיה

From “I know Python” to “I can design, build & ship real agent systems”

Software development is shifting from writing lines of code to building systems that think. This course teaches you to design, build and operate AI agents — autonomous software entities that plan tasks, use external tools and correct themselves as they go. We work with the leading tools in the market — LangChain, LangGraph, LangSmith and Claude Code — and learn to make large language models (LLMs) the core component of stable, policy-driven software systems that deliver direct business value.

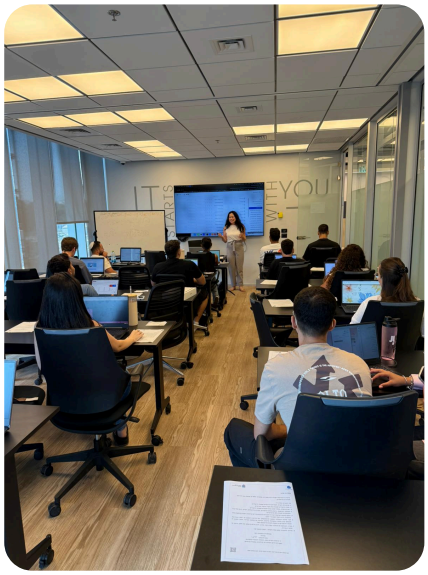
170	32	9	Capstone
Academic hours	Sessions	Chapters	Hands-on final project

Who it's for

Software engineers and developers with strong Python — libraries, OOP, async work, APIs and Git. No prior Machine Learning background required.

Technical requirements

An Anthropic Console API key with a small prepaid credit (~$5–10) and a spending cap; a Claude Pro/Max subscription is recommended for continuous work with Claude Code and the capstone.



Tools & models in the course:

Python · Claude · Claude Code · LangChain · LangGraph · LangSmith · MCP · Docker · Hugging Face · Ollama · Git

CURRICULUM Chapter outline

By the end of the program you will master:

- Context engineering, updated for 2026
- ReAct • Plan-and-Execute • Reflection
- Tool calls • MCP • Skills
- RAG over private org knowledge
- Production: state, cost, observability
- Hands-on projects throughout

01 LLMs and the Model Landscape — Chapter 1

LLMs from a developer's point of view — tokens, context windows and token economics; the model landscape (text, image, multimodal), Hugging Face as the practical hub, and hosted APIs vs. local inference.

Hugging Face · Ollama · Anthropic API

02 Claude Code as a Developer Tool — Chapter 2

The workbench for every lab in the course: a daily workflow of generation, iteration, review and refactoring on real code; CLAUDE.md and project configuration, permissioning, review discipline and sub-agents.

Claude Code

03 Prompt and Context Engineering — Chapter 3

From “write a clever prompt” to deliberately engineering the context an agent reasons over: prompt patterns, structured output, what goes in the window and what stays out — and why context and retrieval usually beat fine-tuning.

04

LangChain and LangGraph

Chapter 4

Where “getting into agents” happens: agents theory — the agent loop, ReAct, tools, state and memory, failure modes — and the frameworks that turn a model call into an agent, ending with your first single-agent graph.

LangChain LangGraph

05

Building Single Agents

Chapter 5

A deep, practical pass at one well-built agent before adding complexity: API integration, tool definition, structured outputs, error handling and retries, guardrails — building an agent that solves a genuine problem end to end.

06

Retrieval-Augmented Generation (RAG)

Chapter 6

Teaching agents to work over custom, private knowledge: embedding models in practice, vector stores, chunking and retrieval patterns — wiring retrieval into an agent that reasons over your team's own documents and data.

07

Model Context Protocol (MCP) and Skills

Chapter 7

Extending what an agent can reach and do: connecting to external tools, services and enterprise systems, defining custom skills and connectors — testing your first MCP server against Claude Code before wiring it into an agent.

MCP Claude Code

08 Multi-Agent Systems

Chapter 8

Multiple agents with distinct roles collaborating: state sharing, hand-offs and debate/consensus patterns — plus just enough deployment, containerization and monitoring literacy to ship.

09 Running Agents in Production

Chapter 9

Why agents break the standard deployment model: long-running stateful runs, checkpointing and durable execution, observability and runtime cost, and security — from prompt injection to tool permissioning.

LangSmith · Docker

10 Capstone Projects

Capstone

Each capstone solves a real problem from the student's own team or workflow: a multi-agent system, a RAG-powered agent, or an MCP/Skills integration — built with LangGraph or the Claude Agent SDK.

Claude Agent SDK · LangGraph

WHERE GRADUATES LAND

Key roles in the industry

AI Agent Engineer

Building and extending AI agents: state-management graphs, tool definition and autonomous loops like ReAct.

AI Software Engineer

Integrating LLMs into enterprise software architecture, connecting systems via MCP and optimizing token economics.

RAG Specialist

Designing advanced RAG systems: vector stores, smart chunking and secure access to private organizational knowledge.

“The best way to predict the future is to create it.” — Peter Drucker

Graduates leave with a complete toolbox for building autonomous agent systems — from the first prompt to production — and hands-on experience from real projects along the way.

המכללה **לניו-מדיה**