# Agentic AI Builder™ (Exam AGB-110): Setting Up a Functional AI Agent

## Course Number:

CNX0026

## Course Length:

3 days

## Overview:

Your organization has assessed its need and ability to implement one or more AI agents with minimal risk. Now it's just a matter of building those agents—that's where you come in. In this course, you'll translate business requirements into a functional AI agent that can automate complex tasks and processes that would otherwise require significant human effort. Ultimately, this can lead to improved user productivity, a reduction in operational costs, and enhanced employee and customer satisfaction.

This is the second course in a series and is meant to build upon the foundation of the first course by giving technical practitioners the skills they need to successfully build agentic AI into their organizations.

## Course Objectives:

In this course, you will build an AI agent from scratch that can automate important tasks requiring some level of human-like judgment.

- Set up the agent development environment.
- Assess LLM behavior in agent contexts.
- Implement the agent reasoning loop.
- Implement tools for an agent to use.
- Add knowledge to an agent through retrieval-augmented generation (RAG).
- Enforce structure, safety, and reliability in an agent.
- Test the behavior and performance of an agent.
- Deploy a single-agent system to production.

## Target Student:

This course is designed for programmers, IT infrastructure personnel, DevOps/MLOps personnel, and any other technically minded professional who is responsible or may soon be responsible for implementing agentic solutions in their organization.

This course is also designed to assist students in preparing for the CertNexus® Agentic AI Builder™ (Exam AGB-110) credential.

## Prerequisites:

To ensure your success in this course, you should have foundational knowledge of agentic AI, including concepts like agentic frameworks, large language models (LLMs), tokens, context windows, agentic tools, retrieval-augmented generation (RAG), behavioral guardrails, and more. You can obtain this level of knowledge by taking the CertNexus AgenticAIBIZ™ (Exam AGZ-110): Foundations of Agentic AI course.

You must also have proficiency using at least one programming language. This course primarily uses Python® to illustrate agentic concepts, so experience with Python is recommended. Logical Operations provides the following courses that teach both Python and general programming skills:

- *Introduction to Programming with Python® (Second Edition)*
- *Advanced Programming Techniques with Python® (Second Edition)*

## Important Cost Information

This course requires the use of cloud-based large language models (LLMs) to power the agent that students build throughout the activities. Cloud LLMs usually have a cost associated with their usage. The rationale for using these services instead of local models is explained in the instructor manual, but the key point is that each student will need access to an application programming interface (API) provided by OpenAI.

Ultimately, you and your students will be charged by the provider for using the LLM. As long as you choose a low-cost LLM from the provider's offerings, the costs will be manageable. We developed and tested this course using OpenAI's GPT-5 mini, and the total costs ended up being less than $5. So, prepare for this to be the maximum price per student for each run of the course. However, we cannot guarantee that you or your students will not exceed this figure. Thankfully, there are ways to mitigate these costs, as explained in the instructor manual.

## Hardware

For this course, you will need one computer for each student and one for the instructor. Each computer will need the following minimum hardware configurations:

- A modern processor.
    - Hardware virtualization must be enabled in the computer's UEFI/BIOS if you're using Docker Desktop (see software requirements).
- 8 gigabytes (GB) of random-access memory (RAM).
- 10 GB of available storage space.
- Mouse, keyboard, and monitor.
- High-speed, stable Internet connection.
- For the instructor's computer, a method to project and/or share the screen as needed for local and remote class participants.

## Software

- Docker
    - There are two main options for installing Docker®: Docker Engine and Docker Desktop. Docker Engine is the core software layer that also provides a command-line interface (CLI) for interacting with containers. Docker Desktop is a graphical user interface (GUI) that is powered by Docker

Engine. The choice of either two comes down to the operating system (OS) you plan to use, as well as whether your system supports hardware virtualization. Support for hardware virtualization is required for Docker Desktop, but not Docker Engine.
    - **Windows®** or **macOS®** : You *must* install Docker Desktop, as Docker Engine by itself is not supported on either OS. Alternatively, you could create a Linux virtual machine (VM) and run Docker Engine on that.
    - **Linux®** : You can install either Docker Desktop or Docker Engine.
- Visual Studio® Code
    - VS Code is available for all three major operating systems.
- If necessary, software for viewing the course slides. (Instructor machine only.)

For reference, this course was developed and tested on both Windows 11 and Linux operating systems using x86-64 CPU architectures. The course was not tested on macOS or Arm-based architectures, but since both Docker and VS Code support macOS and Arm, we are confident that the technical environment will still work as intended.

NOTE: Docker Engine is open-source software. Docker Desktop and VS Code are free (as in cost), but not technically open source.

## Course Content

**Lesson 1: Setting Up the Agent Development Environment**

Topic A: Configure a Python-Based Agent Workspace

Topic B: Configure LLM Access

Topic C: Configure Runtime Constraints

**Lesson 2: Assessing LLM Behavior in Agent Contexts**

Topic A: Analyze LLM Capabilities and Limitations

Topic B: Design Prompts for Agent Reasoning

**Lesson 3: Implementing the Agent Reasoning Loop**

Topic A: Implement the ReAct Pattern

Topic B: Manage Agent State Across Iterations

Topic C: Manage Memory and Persistence

**Lesson 4: Using Agent Tools**

Topic A: Design Agent Tools and Interfaces

Topic B: Execute and Validate Tool Calls

**Lesson 5: Adding Knowledge with Retrieval-Augmented Generation**

Topic A: Implement Document Ingestion and Embeddings

Topic B: Retrieve and Use Context Effectively

**Lesson 6: Enforcing Structure, Safety, and Reliability in an Agent**

Topic A: Enforce Structured Outputs

Topic B: Handle Uncertainty and Failures

**Lesson 7: Testing an Agent**

Topic A: Monitor Agent Behavior

Topic B: Evaluate Agent Performance

Topic C: Analyze an Agent for Security Flaws

**Lesson 8: Deploying a Single-Agent System**

Topic A: Expose Agent Interfaces

Topic B: Review Readiness and Limitations