

AI Workflows and Agents

*A Practical Foundation for Building
with Large Language Models*

Honest Earning

Copyright © 2026 Honest Earning

All rights reserved.

No part of this publication may be reproduced, distributed, or transmitted in any form or by any means, including photocopying, recording, or other electronic or mechanical methods, without the prior written permission of the publisher, except in the case of brief quotations embodied in critical reviews and certain other noncommercial uses permitted by copyright law.

Published by Honest Earning

ISBN: 978-93-5943-653-1

This book is intended for educational purposes only. The tools, application programming interfaces (APIs), pricing, and model names referenced throughout are accurate as of the time of writing and may change. Readers should consult the official documentation of any provider mentioned before relying on specifics such as pricing, model availability, or API behavior. The author and publisher make no warranties regarding the completeness or accuracy of third-party information and are not responsible for any losses or damages arising from the use of the concepts described in this book.

*To everyone curious enough to look behind the curtain
of “AI magic” and build something real with it.*

Contents

Contents.....	5
Preface.....	6
Introduction: Why This Book Exists.....	1
AI Workflows vs. AI Agents: Defining the Terms.....	2
No-Code and With-Code Approaches.....	7
Large Language Models, Applications, and APIs.....	8
Proprietary Models vs. Open-Weight Models.....	10
Setting Up Your Development Environment.....	13
Your First AI Workflow: Generating a Post.....	15
Prompt Engineering and Few-Shot Examples.....	18
Multi-Step Workflows and Model Selection.....	21
Running Models Locally with Ollama.....	24
Structured Outputs.....	26
Error Handling and Reliability Patterns.....	28
Advanced Workflows: Images, Messages, and File Organization.....	31
Managing Control Flow: Sequence, Parallel, Conditional, and Repeated Steps.....	33
Human in the Loop.....	35
Integrating Third-Party Services.....	37
An Introduction to Retrieval-Augmented Generation (RAG).....	39
From Workflows to Agents: How Tool Use Really Works	42
Function Calling with the OpenAI API.....	44
Case Study: A Customer Service Agent.....	46
Security Considerations for Agents.....	48
A Class-Based Approach to Agents and Tools.....	51
Multi-Agent Systems.....	53
Managing Memory: Short-Term and Long-Term.....	56
Testing and Evaluating AI Systems.....	58
Agent Frameworks: CrewAI and Beyond.....	61
Conclusion.....	64
Glossary.....	66

Preface

This book exists to give you a fast, practical, and honest foundation in building AI-powered workflows and agents.

The goal is not to overwhelm you with theory, nor to hand you a black-box framework you don't understand. Instead, this book walks through the core ideas step by step — what an AI workflow is, what an AI agent is, how large language models are actually used inside real applications, and how tools, memory, and control flow fit together — so that the knowledge transfers to any programming language or provider you choose to work with.

You will see concrete, worked examples throughout: sending your first request to a large language model, building a multi-step content pipeline, exposing tools to a model so it can act on your behalf, and finally combining multiple specialized agents into a working system. Along the way, you'll also see how third-party frameworks such as CrewAI can simplify parts of this work — and, just as importantly, when building things yourself is the better choice.

By the end, you should be able to look at any AI-powered workflow or agent you encounter and understand, at a mechanical level, what is actually happening behind the scenes.

CHAPTER 1

Introduction: Why This Book Exists

Welcome. This book will walk you through the fundamentals of building AI-powered workflows and agents — the core concepts you need in order to design and build your own, regardless of the programming language or provider you eventually choose.

The goal is to cover, in a compact and practical way, the difference between workflows and agents, the essential theory behind each, and plenty of concrete, working examples. The idea is that this knowledge should transfer directly to any project you take on afterward, whatever language or platform you use day to day.

Throughout the book, the emphasis is on showing you real code and real examples rather than abstractions. Some sections walk through complete, runnable programs; others focus on the concepts that make those programs work. Either way, the intent is the same: to get you comfortable enough with the fundamentals that you can start building on your own.

With that said, let's not waste any time. Let's start with the most important question of all: what exactly is an AI workflow, and what is an AI agent — and how do the two differ?

CHAPTER 2

AI Workflows vs. AI Agents: Defining the Terms

The distinction between an AI workflow and an AI agent ultimately depends on how you define each term. This book uses the following working definitions.

AI Workflows: Deterministic Sequences of Steps

An AI workflow is a deterministic sequence of steps — a program that automates a task by executing a defined series of operations. It might open a file, read its contents, transform that content, and save the result somewhere else. In that broad sense, almost any program could be called a workflow. If one or more of its steps uses a large language model, it becomes an AI workflow: a task automated with the help of AI.

AI Agents: Autonomous Planners

An AI agent, by contrast, is autonomous. You build one by writing a program that uses a large language model to plan the execution of a task, and then to carry out that plan by calling tools you've made available to it.

A crucial point to understand — and one this book returns to repeatedly — is that the large language model itself never executes anything. It only generates tokens. What looks like the model “calling a tool” is really the model producing text that describes an intended tool call; it is your code that recognizes that intent and actually performs the action.

Control vs. Flexibility

This distinction produces a practical trade-off. With a workflow, you have a high degree of control: the sequence of steps is fixed, and you know exactly what will happen at each stage. With an agent, you have less control, because the model decides which tools to use and when, based on its own reasoning about the input it receives.

As a rule of thumb, workflows suit tasks with known inputs and outputs, where you already understand how the problem should be solved. Agents suit tasks with unknown or highly variable inputs and outputs, where the right approach can't be fully specified in advance — a customer service conversation is a good example, since you can't know in advance what a customer will ask.

In practice, most problems are solved well by controlled workflows, simply because you know exactly what is going to happen. Agents earn their added complexity mainly when the task genuinely requires that flexibility.

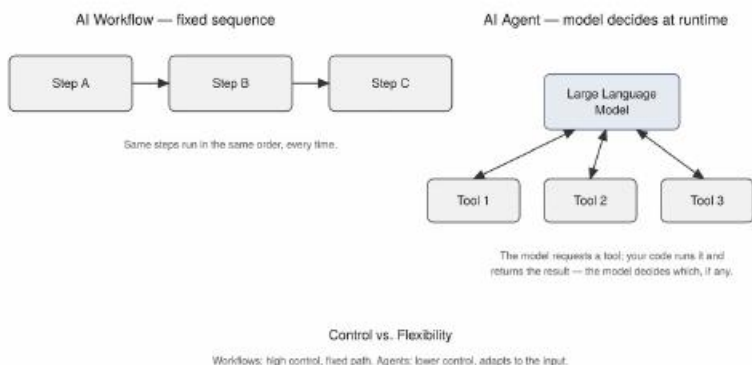


Figure 2.1 — A fixed workflow versus a model that decides which tools to call at runtime.

Aspect	AI Workflow	AI Agent
Decision-making	Fixed by the developer	Made by the model at runtime
Tool usage	Predefined, always the same	Model chooses which, if any
Predictability	High — same path every run	Lower — varies with input
Best suited for	Known inputs and outputs	Open-ended or variable requests
Typical cost/latency	Lower — fewer, targeted calls	Higher — planning and retries add up

A Blurry Line in Practice

It's worth being upfront that this is only one way of defining the terms. In everyday use, "AI agent" is often applied to any program or any single step that uses AI at all — even a plain, sequential workflow. Later in this book, a multi-step workflow generates a blog post, evaluates it, and then produces a thumbnail and a LinkedIn post from it. By the strict definition above, that is a workflow. Many people, however, would call it an agentic system, with a "blog post agent," an "evaluation agent," a "thumbnail agent," and so on. Both descriptions are reasonable — the answer really does depend on your definition.

Chatbots as a Familiar Example of Agents

Before writing any code, it's worth recognizing that you've likely already interacted with an AI agent, knowingly or not: ChatGPT, Google Gemini, xAI's Grok, or Anthropic's Claude. These chat assistants are agentic in the sense used here, because they have access to tools

— sometimes requiring you to enable them — and they choose when to use those tools based on your query.

If you ask ChatGPT which new AI models were released in the last two weeks, it will typically perform a web search using a tool OpenAI has exposed to it. Behind the scenes, the model requested that tool, the application executed it, and the results were folded back into the response you see. That is exactly what an AI agent does: it receives instructions (set by the provider, not shown to you), has access to a set of tools, receives an input (your message), and produces an output — without the provider knowing in advance exactly what you'll ask.

That unpredictability is precisely why a flexible, tool-using system makes more sense here than a rigid, fixed sequence of steps. It would make little sense for ChatGPT to always perform a web search regardless of the question; instead, it needs the flexibility to decide, case by case, which tool (if any) is relevant.

What This Book Covers

The chapters ahead are organized in two broad parts. Part One covers AI workflows: interacting with large language models programmatically, working with structured input and output, managing control flow, adding a human into the loop, and integrating third-party services. This section is the foundation for everything that follows, because the same building blocks reappear when constructing agents. Part Two covers AI agents: how tool use actually works under the hood, how to build both general-purpose and specialized agents, how to connect multiple agents into a system, and how to manage short-term and long-term memory. A final chapter looks at third-party frameworks, such as CrewAI, that can simplify parts of this work.