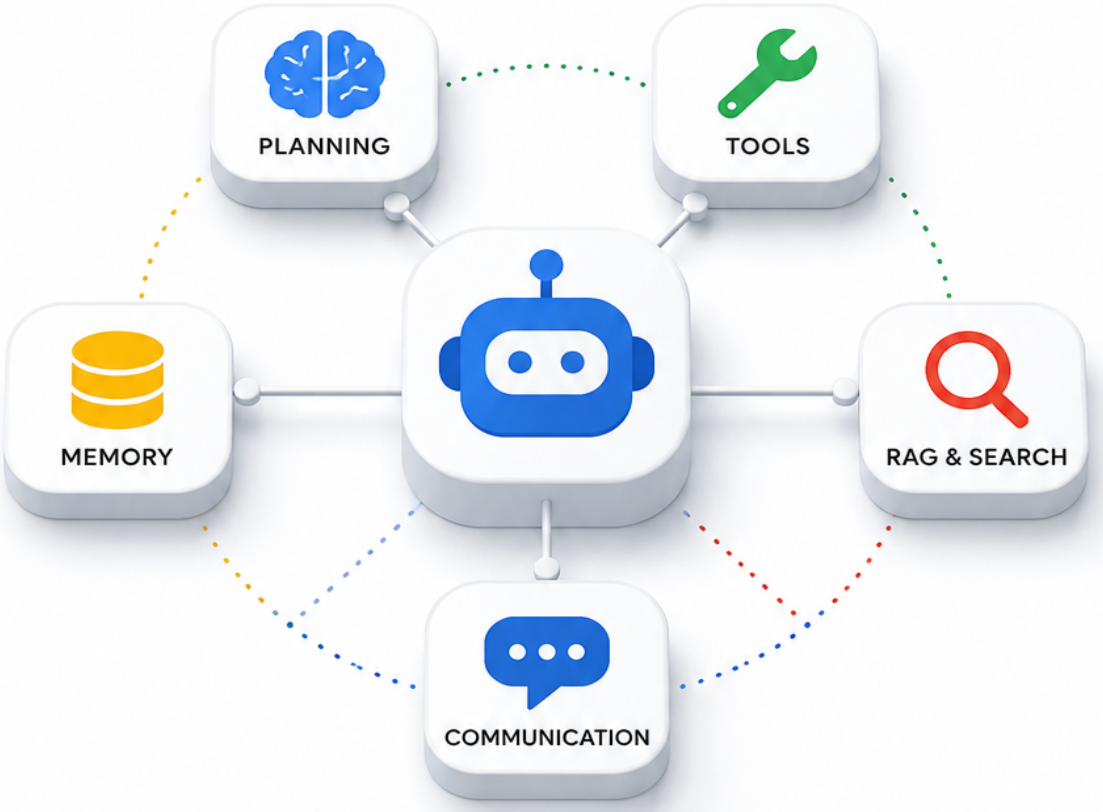
Hands-On AI Agents with Google ADK

Build, orchestrate, and deploy production-ready multi-agent systems using Python and Google Cloud



DIEGO MEDIERO

Hands-On AI Agents with Google ADK

Build, orchestrate, and deploy production-ready multi-agent systems using Python and Google Cloud

Steve T. Publications

This book is available at
https://leanpub.com/hands-onaiagentswithgoogleadk

This version was published on 2026-07-13

Contents

Build, orchestrate, and deploy production-ready multi-agent systems using Python and Google Cloud

This book takes you from your very first AI agent to production-grade multi-agent systems using the Google Agent Development Kit (ADK). Every chapter is a hands-on walkthrough with complete, runnable Python code. You will learn the core abstractions that power ADK, how to wire tools into agents, manage memory and context, orchestrate multi-agent workflows, stream real-time responses, integrate retrieval-augmented generation, and deploy your agents securely on Google Cloud. No filler, no quizzes, no theory without implementation. Just practical knowledge you can use from day one.

Introduction

In 2025, Google announced the Agent Development Kit (ADK) at Google Cloud NEXT, an open-source framework designed to make building reliable AI agents accessible to developers who already know how to write code. Within months, the Python repository accumulated over 20,000 GitHub stars, and by mid-2026 it had become one of the most actively developed agent frameworks in the industry. The framework ships on a roughly bi-weekly release cadence, supports Python, TypeScript, Go, Java, and Kotlin, and is licensed under Apache 2.0.

This book exists because building AI agents well is harder than calling an LLM API. Agents need to remember things across conversations. They need to call external tools reliably. They need to handle errors gracefully when a tool fails or an API returns unexpected data. They need to stay within their instructions and not hallucinate capabilities they do not possess. And in production, they need observability, security, cost controls, and the ability to scale.

ADK addresses these challenges through a code-first design philosophy. Instead of configuring agents through YAML files or visual builders, you write Python code. You define agents as classes. You compose workflows with functions and decorators. You add tools by wrapping regular Python functions. This approach gives you the same flexibility you already use in your daily development work, while ADK handles the plumbing of state management, event loops, session persistence, and deployment.

What This Book Covers

The chapters follow a deliberate progression. You start with fundamentals: what AI agents are, how they differ from basic chatbots, and why frameworks matter. Then you install ADK, configure your environment, and write your first agent. From there, each chapter builds on the previous one.

You will learn to build agents with tools that connect to real APIs and databases. You will implement memory systems so agents remember user

preferences across sessions. You will orchestrate multi-agent teams where specialized agents collaborate through routing, delegation, and graph-based workflows. You will integrate retrieval-augmented generation (RAG) to give agents access to proprietary knowledge bases. You will stream responses in real-time using ADK's event system. And you will deploy your agents to production on Cloud Run or GKE with proper observability, security, and cost controls.

Who Should Read This Book

This book assumes you are comfortable writing Python, including async/await patterns. You should understand how APIs work, have used pip to install packages, and be familiar with basic command-line operations. No prior experience with LLMs or agent frameworks is required. The first chapter establishes the foundational concepts from scratch.

How to Use This Book

Read the chapters in order. Each one depends on concepts introduced earlier. The code examples are complete and runnable, not fragments. You can copy them directly into your project, modify them, and extend them as you learn. When a chapter introduces a new concept, it provides enough context for you to understand why that concept matters before showing you how to implement it.

The book focuses exclusively on the Python implementation of ADK. The same concepts apply to TypeScript, Go, Java, and Kotlin, but the syntax and some API details differ. If you work in another language, the architectural patterns and design decisions described here will still translate directly.

What You Will Be Able to Do After Reading

By the end of this book, you will be able to:

- Design and implement AI agents with clear instructions, tools, and behavior

- Build multi-agent systems where specialized agents collaborate through structured workflows
- Integrate external APIs, databases, and knowledge sources into your agents
- Stream real-time responses to users with proper event handling
- Deploy agents to Google Cloud with production-grade observability and security
- Debug agent behavior using tracing, logging, and evaluation frameworks
- Optimize agent performance and control costs in production environments

Let us begin.

Chapter 1: Understanding AI Agents

What Are AI Agents

An AI agent is a system that uses a large language model (LLM) as its reasoning engine to perceive information, make decisions, and take actions in pursuit of a goal. This definition matters because it distinguishes agents from simpler systems. A chatbot receives a message and produces a reply. An API wrapper sends structured input to an LLM and returns the output. Both are useful, but neither acts autonomously.

An agent closes the loop between thinking and doing. It reads user input, decides what to do next, executes actions through tools or APIs, observes the results, and iterates until the task is complete. This perception-reasoning-action cycle is the fundamental architecture pattern that separates agents from everything else in the LLM ecosystem.

Consider a customer support agent. A basic chatbot might answer frequently asked questions from a knowledge base. An agent goes further: it looks up order status in a database, checks shipping tracking through an external API, drafts a refund request, and sends a confirmation email – all within a single interaction, making decisions at each step about what action to take next.

The Agent Architecture Pattern

Every AI agent, regardless of framework, follows the same core architecture:

Perception: The agent receives input from its environment. This typically means user messages, but it can also include sensor data, API events, file changes, or scheduled triggers.

Reasoning: The LLM analyzes the input in context. It considers the conversation history, any available tools, system instructions, and the current state of the world. Based on this analysis, it decides what to do next.

Action: The agent executes a decision. This could mean calling a tool function, making an API request, generating code for execution, or simply producing a text response for the user.

Observation: The agent receives the result of its action. If it called a weather API, it gets back temperature data. If it executed code, it gets the output or an error message.

Iteration: With new information in hand, the agent returns to the reasoning step. It might decide another tool call is needed, or that it has enough information to formulate a final response.

This cycle repeats until the task is complete or a termination condition is met. The number of iterations varies: a simple question-answer exchange might be one cycle, while a complex research task could involve dozens of tool calls and reasoning steps.

The key insight is that the LLM does not just generate text. It generates decisions about what to do, and the framework translates those decisions into actual actions in the real world. This is why agents are fundamentally different from chatbots.

Why Frameworks Matter for Agents

You could build an agent by calling the Gemini API directly, writing your own loop to handle tool calls, managing conversation history manually, and implementing error handling yourself. This works for a proof of concept. It does not work for production.

Production agents need session management so conversations persist across server restarts. They need state management so agents can store intermediate results and user preferences. They need observability so you can trace what an agent did when it misbehaved. They need structured testing so you can evaluate agent behavior before deploying changes.

Frameworks like ADK provide these capabilities as built-in features rather than things you build from scratch. The trade-off is learning the framework's abstractions, but the payoff is dramatically reduced implementation complexity and a standardized approach to common patterns.

The agent framework landscape includes several notable options. LangChain and LangGraph focus on chains and graphs of LLM calls with

extensive tool integrations. CrewAI emphasizes role-based multi-agent orchestration with a declarative style. AutoGen from Microsoft supports conversational agents that communicate with each other. ADK takes a different approach: it is code-first, model-agnostic (while optimized for Gemini), and designed specifically for Google Cloud deployment.

Introducing Google ADK

Google announced the Agent Development Kit at Google Cloud NEXT in April 2025 as an open-source framework for building AI agents. The Python version reached v1.0.0 stability later that year, signaling production readiness. Version 2.0 introduced a graph-based workflow runtime that fundamentally changed how complex agent orchestration works.

ADK's design philosophy centers on several principles:

Code-first development: You write Python code to define agents, not YAML configurations or visual diagrams. Agents are Python objects. Workflows are Python functions and decorators. This means you get full IDE support, type checking, testing frameworks, and the same development tools you already use.

Model flexibility: While optimized for Google's Gemini models, ADK supports other models through its `BaseLlm` interface. You can connect to Claude, open-source models via Ollama or vLLM, or any provider through LiteLLM. The agent code stays the same regardless of which model you use.

Progressive complexity: ADK lets you start simple and grow into sophistication. A single-agent application with one tool requires only a few lines of code. Adding memory, multi-agent orchestration, streaming responses, and RAG integration layers on additional capabilities without restructuring your existing code.

Deployment flexibility: Agents written with ADK can run locally for development, deploy to Google Cloud Run for serverless scaling, run on GKE for full Kubernetes control, or use the managed Agent Runtime on Google Cloud for hands-off operations.

ADK at a Glance

Before diving into implementation, here is a preview of the core abstractions you will work with throughout this book:

Agent: The primary building block. An `Agent` (or `LlmAgent`) encapsulates a model, system instructions, tools, and behavior configuration. It is the unit of intelligence in your application.

Runner: The execution engine. A `Runner` coordinates agent workflows, manages the event loop, handles session services, and orchestrates the interaction between agents, tools, and external systems.

Tool: A capability that extends what an agent can do. Tools are Python functions with type hints and docstrings that ADK exposes to the LLM for function calling. They let agents interact with APIs, databases, file systems, and other services.

Session: A conversation thread between a user and your agent. Sessions track message history (as `Event` objects) and maintain working state (a dictionary of key-value pairs).

Workflow: A structured sequence of operations. ADK supports template workflows (sequential, parallel, loop) and graph-based workflows (ADK 2.0+) that define deterministic execution paths combining AI reasoning with programmatic control.

Memory: Long-term knowledge storage separate from session state. Memory services persist information across sessions using keyword matching, semantic search, or vector similarity retrieval.

These abstractions compose together to create everything from simple single-purpose agents to complex multi-agent systems with dozens of specialized components working in concert.

The next chapter gets your development environment set up and running your first agent within minutes. By the end of it, you will have a working agent that responds to user input, calls tools, and maintains conversation state.

Chapter 2: Getting Started with Google ADK

Prerequisites and Environment Setup

Before installing ADK, ensure your development environment meets the requirements. ADK Python requires Python 3.10 or later and pip for package management. If you are starting from scratch, install Python from python.org or use a version manager like pyenv.

Create a dedicated virtual environment for your project. This isolates your ADK dependencies from other Python projects and prevents version conflicts:

```
# Create a project directory
mkdir my_adk_project
cd my_adk_project

# Create a virtual environment
python -m venv .venv

# Activate it (Linux/macOS)
source .venv/bin/activate

# Activate it (Windows)
.venv\Scripts\activate
```

Verify your Python version:

```
python --version
# Output should be Python 3.10.x or later
```

Installing ADK and Dependencies

Install the core ADK package using pip. The base installation includes everything needed for local development with Gemini models via the Google AI API:

```
pip install google-adk
```

For optional integrations like Vertex AI session services, memory banks, and additional model providers, install the extensions bundle:

```
pip install "google-adk[extensions]"
```

Verify the installation by checking the installed version:

```
pip show google-adk
```

The ADK Python package follows a roughly bi-weekly release cadence. As of mid-2026, the stable 2.x line includes graph-based workflows and the Task API, while the 1.x line continues to receive maintenance updates. Check PyPI for the latest version and review the CHANGELOG.md in the GitHub repository when upgrading between major versions.

Getting a Google API Key

ADK agents need access to an LLM to function. The simplest path uses Google's Gemini models through the Google AI API, which requires a free API key.

1. Visit aistudio.google.com and sign in with your Google account
2. Navigate to the API keys section
3. Create a new API key
4. Copy the key value for use in your project

Store the API key in an environment variable rather than hardcoding it in your source files. ADK looks for the `GOOGLE_API_KEY` environment variable automatically:

```
# Linux/macOS
export GOOGLE_API_KEY="your-api-key-here"

# Windows (Command Prompt)
set GOOGLE_API_KEY=your-api-key-here

# Windows (PowerShell)
$env:GOOGLE_API_KEY="your-api-key-here"
```

For a more organized approach, use a `.env` file in your project root:

```
echo 'GOOGLE_API_KEY="your-api-key-here"' > .env
```

Add `.env` to your `.gitignore` to prevent accidentally committing credentials. ADK does not automatically load `.env` files, but you can use the `python-dotenv` package if you prefer this pattern:

```
pip install python-dotenv
```

Then at the top of your agent file:

```
from dotenv import load_dotenv
load_dotenv()
```

Your First Agent

Create a new Python file called `agent.py`. This is the entry point that ADK uses to load your agent. The framework requires exactly one variable named `root_agent` in this file, which defines the top-level agent for your application.

Here is the minimal working agent:

```python
from google.adk import Agent

root_agent = Agent(
    name="greeting_agent",
    model="gemini-2.5-flash",
    instruction="You are a helpful assistant. Greet the user warmly and ask how you can help.",
)
```

That is all the code needed for a functioning agent. The `Agent` class (which is an alias for `LlmAgent`) takes three essential parameters:

- **name**: A unique identifier for the agent. Avoid reserved names like "user". This name appears in logs, traces, and the developer UI.
- **model**: The LLM identifier. Use `"gemini-2.5-flash"` for fast, cost-effective responses, or `"gemini-2.5-pro"` for more complex reasoning tasks.
- **instruction**: The system prompt that defines the agent's behavior, persona, and capabilities. This is the most important parameter and deserves careful attention.

Adding a Tool

Agents become useful when they can do more than generate text. Add a tool by defining a Python function with type hints and a docstring, then registering it with the agent:

```python
from google.adk import Agent
from google.adk.tools import FunctionTool


def get_current_time(city: str) -> dict:
    """Get the current time in a specified city.

    Args:
        city: The name of the city to get the time for.

    Returns:
        A dictionary with 'status' ('success' or 'error') and
        'time_info' containing the location and current time.
    """
```

```
    from datetime import datetime
    import pytz

    try:
        tz = pytz.timezone(city.replace(" ", "_"))
        now = datetime.now(tz)
        return {
            "status": "success",
            "time_info": f"The current time in {city} is
            ↪  {now.strftime('%H:%M:%S')} ({now.strftime('%Z')})."
        }
    except pytz.exceptions.UnknownTimeZoneError:
        return {
            "status": "error",
            "error_message": f"Time zone for '{city}' is not recognized."
        }


root_agent = Agent(
    name="time_agent",
    model="gemini-2.5-flash",
    instruction=(
        "You are a helpful assistant that can tell users the current time "
        "in any city. Use the 'get_current_time' tool when a user asks about "
        "the time in a specific location. If the tool returns an error, "
        "politely inform the user and ask them to try a different city name."
    ),
    tools=[FunctionTool(func=get_current_time)],
)
```

Several implementation decisions are worth noting:

The function signature uses type hints (`city: str` and `-> dict`). ADK extracts these types to build a JSON schema that the LLM sees when deciding whether to call the tool. Without proper type hints, the framework cannot generate a valid schema, and the tool will not be available to the agent.

The docstring describes what the tool does, what parameters it accepts, and what it returns. This text is sent to the LLM as part of the tool definition. A clear, specific docstring dramatically improves the model's ability to use the tool correctly.

The return value is a dictionary with a `status` key. This convention helps the agent understand whether the tool succeeded or failed. The agent's instruction then tells it how to handle each case. Without this pattern, the agent might misinterpret error conditions as normal output.

Running Agents Locally

ADK provides two ways to test your agent locally: the command-line interface and a web-based developer UI.

CLI mode runs an interactive chat session in your terminal:

```
adk run path/to/agent.py
```

This starts a REPL where you can type messages and see the agent's responses. The CLI is useful for quick testing and debugging.

Web UI mode launches a browser-based interface with richer features including conversation history, state inspection, and multi-agent visualization:

```
adk web path/to/agents_dir
```

The `adk web` command serves a local development server at `http://localhost:8000`. It supports both single-agent directories and multi-agent project structures. The web interface is strictly for development and debugging, not production deployment.

Project Structure and Organization

As your agent grows in complexity, organize your code into a clear project structure. A typical ADK project looks like this:

```
my_adk_project/
├── .env                    # Environment variables (API keys)
├── .gitignore              # Ignore .env, __pycache__, .venv
├── pyproject.toml          # Project metadata and dependencies
├── agent.py                # Entry point with root_agent definition
├── tools/
│   ├── __init__.py
│   ├── weather_tool.py     # Weather-related tool functions
│   └── database_tool.py    # Database query tools
├── agents/
│   ├── __init__.py
│   ├── researcher.py       # Researcher agent definition
│   └── writer.py           # Writer agent definition
└── tests/
    ├── __init__.py
    └── test_tools.py       # Unit tests for tool functions
```

The `agent.py` file is the entry point. It imports and composes agents, tools, and configuration from your modules, then exports the `root_agent` variable. This separation keeps your code organized and testable.

Common Pitfalls in Setup

Wrong Python version: ADK requires Python 3.10+. If you see import errors about missing syntax features, check your Python version.

Missing API key: Without a valid `GOOGLE_API_KEY`, your agent will fail with an authentication error. Double-check that the environment variable is set and contains the correct value.

Forgetting type hints on tool functions: Tools without type hints will not be properly registered. The LLM will not see them as available capabilities.

Using reserved agent names: Names like "user" are reserved by the framework. Use descriptive, unique names for your agents.

Not activating the virtual environment: If you installed ADK in a virtual environment but forgot to activate it before running `adk run`, the command will fail because it cannot find the package.

With your environment set up and your first agent running, you are ready to dive into the core concepts that power every ADK application. The next chapter examines the Agent class, Runner, Sessions, State, and Events in detail.

Chapter 3: Core ADK Concepts

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

The Agent Class

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

The Runner

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Sessions and Session Services

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

State Management

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Events and the Event Loop

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Chapter 4: Building with Tools

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

What Are Tools and How Agents Use Them

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Function Tools

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Tool Context

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Built-in Tools

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Custom Tool Design Patterns

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Toolsets: Grouping and Dynamic Provisioning

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Common Pitfalls with Tools

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Chapter 5: Memory and Context Management

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Short-Term vs Long-Term Memory

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Memory Services

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Using Memory in Agents

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Context Caching

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Context Compression and Compaction

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Artifacts

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Common Pitfalls with Memory and Context

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Chapter 6: Prompt Engineering for Agents

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

The Instruction Field

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Referencing Tools in Instructions

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Handling Tool Return Values

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Multi-Turn Conversation Design

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Model Selection and Configuration

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Common Pitfalls with Instructions

Chapter 7: Workflows and Orchestration

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Template Workflows

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Agent Routing

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Graph-Based Workflows (ADK 2.0)

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Human-in-the-Loop Patterns

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Dynamic Workflows

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Common Pitfalls with Workflows

Chapter 8: Multi-Agent Systems

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Agent Team Architecture

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Agents as Tools

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Collaborative Workflows with Shared State

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

A2A Protocol (Agent-to-Agent)

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Real-World Multi-Agent Patterns

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Common Pitfalls with Multi-Agent Systems

Chapter 9: Streaming and Asynchronous Execution

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Understanding Events

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Streaming Responses

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

The Event Loop Deep Dive

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Gemini Live API Toolkit

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Streaming Tools

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Common Pitfalls with Streaming

Chapter 10: Retrieval-Augmented Generation (RAG)

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

RAG Fundamentals for Agents

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Vertex AI Search Integration

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Building a RAG Tool from Scratch

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Agentic RAG Patterns

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Common Pitfalls with RAG

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Chapter 11: MCP and External API Integration

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Model Context Protocol (MCP) Overview

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Connecting to Existing MCP Servers

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Building an MCP Server with ADK

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

OpenAPI Tools

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Authentication for External APIs

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Common Pitfalls with External Integration

Chapter 12: Structured Outputs and Validation

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Output Schemas with Pydantic

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

The output_key Parameter

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Validating LLM Outputs

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

JSON Mode and Structured Function Returns

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Error Handling Patterns for Structured Data

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Common Pitfalls with Structured Outputs

Chapter 13: Callbacks and Plugins

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Callback Types

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Callback Patterns

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Plugin Architecture

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Callbacks vs Plugins: When to Use Each

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Common Pitfalls with Callbacks and Plugins

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Chapter 14: Testing and Evaluation

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Unit Testing Agent Components

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Integration Testing with Runners

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Golden Datasets and Evaluation Frameworks

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Custom Evaluation Metrics

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

User Simulation Testing

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Common Pitfalls with Testing

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Chapter 15: Observability and Debugging

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Built-in Logging

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Distributed Tracing with OpenTelemetry

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Metrics Collection

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Third-Party Observability Platforms

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Debugging Common Issues

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Common Pitfalls with Observability

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Chapter 16: Deployment

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Deployment Options Compared

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Deploying to Cloud Run

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Deploying to GKE

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Agent Runtime on Google Cloud

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Environment Configuration

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Common Pitfalls with Deployment

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Chapter 17: Security and Safety

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Content Safety Filters

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Guardrails with Callbacks

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Authentication Patterns

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Prompt Injection Defense

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Common Pitfalls with Security

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Chapter 18: Production Best Practices

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Performance Optimization

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Cost Management

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Error Handling and Resilience

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Scaling Patterns

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Monitoring and Alerting in Production

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Migration from 1.x to 2.0

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Common Pitfalls in Production

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

Conclusion

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.

References

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/hands-onaiagentswithgoogleadk.