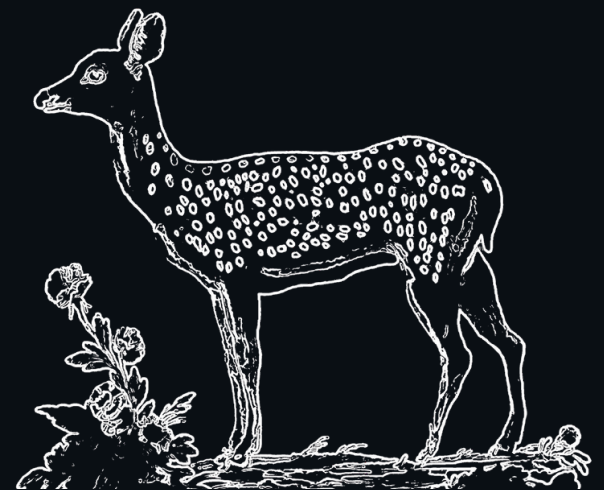
PRACTICAL TECH GUIDE

Building AI Agents

A Hands-On Guide to Goal-Driven Agents



python -m exif_privacy_agent.agent

Building AI Agents

A Hands-On Guide to Goal-Driven Agents with Python

Blue J. Lion
Quiet Line Press (quietlinepress.com)

First Edition: May 2026

Published by Quiet Line Press (quietlinepress.com)

Table of Contents

About This Book

Book Positioning

This book is for developers, technical creators, AI builders, and curious learners who want to understand how practical AI agents work and how to build one without getting lost in hype or abstraction.

It is a practical, project-based guide rather than a book about vague autonomy.

The companion project for this book is `exif_privacy_agent`, a small Python agent that helps review image privacy risk, decide when cleanup is needed, ask for approval before file mutations, and summarize the results in plain language.

It currently depends on `exif_mcp_server` for the EXIF capability layer.

`exif_mcp_server` is the companion capability project behind this agent. It provides the EXIF inspection, privacy summarization, and cleanup operations that the agent uses through either direct shared-core imports or a real MCP stdio server.

The clearest open-source setup is to clone both repositories as siblings:

```
your-workspace/
  exif_mcp_server/
  exif_privacy_agent/
```

In this book, sample-image paths assume that layout. If you want to use your own local images instead, substitute those paths throughout.

If you are starting from GitHub, a simple first setup looks like this:

git clone https://github.com/nextframedev/exif_privacy_agent.git
git clone https://github.com/nextframedev/exif_mcp_server.git
cd exif_privacy_agent

The companion project grows in stages:

1. first, a direct local adapter keeps the agent loop simple

2. then, a real stdio MCP client lets the same agent talk through the protocol boundary

3. and optionally, an HTTP MCP client can point at a separately running `streamable-http` server

This book uses one useful distinction from the start:

1. an MCP server defines what the system can do

2. an agent decides what the system should do next

That distinction matters because many real systems need both.

The MCP server exposes capabilities. The agent uses those capabilities to pursue a goal.

What You Will Learn

By the end of this guide, you should understand:

1. what an AI agent is and what it is not

2. how an agent differs from a tool, a workflow, and an MCP server

3. how to structure a small agent loop in Python

4. how to read plain-language intent and map it to actions

5. how to keep approvals and safety boundaries explicit

6. how an agent can use structured tool results instead of raw prose

7. how to test agent behavior in a narrow, practical domain

8. how to extend a simple agent without turning it into a fragile platform

9. how to add a model-backed planner without blurring the safety boundary

Who This Book Is For

This book is a good fit if you are:

1. comfortable with basic Python

2. interested in agent behavior beyond simple chat prompting

3. looking for a practical companion to an MCP server project

4. trying to understand how planning, approvals, and tool use fit together

This is not a full agent survey or a research-heavy treatment. It is about building one good agent clearly.

The Companion Project

The book centers on `exif_privacy_agent`.

That project is intentionally small.

It can:

1. review one image before sharing

2. review a whole folder before sharing

3. decide whether a file contains privacy-sensitive metadata

4. request approval before cleanup

5. clean one file or one folder through a clear workflow

6. remove only selected metadata such as GPS fields when full cleanup is unnecessary

7. summarize results in plain language

8. keep the planning boundary clear enough for an optional model-backed planner

It is a strong agent example because:

1. the user goal is easy to understand

2. the tool results are structured

3. the safety boundary is real

4. the difference between read-only and mutating actions matters

The Core Idea

A practical agent loop can be surprisingly small:

```
Observe request
  |
  v
Plan next step
  |
  v
Check safety or approval
  |
  v
Call tool if needed
  |
  v
Evaluate result
  |
  v
Respond or continue
```

This book builds that loop in small steps.

Quick Start in 5 Minutes

If you want the fastest possible first success, start with a tiny agent that uses only plain Python.

Fastest First Success: A Tiny Agent

Create a file named `tiny_agent.py` with:

```
from dataclasses import dataclass


@dataclass
class AgentResponse:
    status: str
    message: str


def choose_action(request: str) -> str:
    text = request.lower()
    if "hello" in text or "hi" in text:
        return "greet"
    if "help" in text:
        return "help"
    return "unsupported"


def run_agent(request: str) -> AgentResponse:
```

```
    action = choose_action(request)

    if action == "greet":
        return AgentResponse(status="ok", message="Hello! I am
a tiny agent.")
    if action == "help":
        return AgentResponse(
            status="ok",
            message="Try asking me to say hello.",
        )
    return AgentResponse(
        status="unsupported",
        message="I only know how to greet right now.",
    )


if __name__ == "__main__":
    request = input("Request: ")
    response = run_agent(request)
    print(response)
```

Run it with:

python tiny_agent.py

Then try:

hello

You should get a small structured response back.

That tiny example already shows the basic shape:

1. read a request

2. choose an action

3. run the action

4. return a result

The companion project in this book uses the same basic loop, but adds:

1. a real capability layer

2. structured EXIF tools

3. approval gates

4. richer workflow decisions

About the Author

Blue J. Lion has over 20+ years of experience in software development, with a focus on programming, data security, and privacy. He has worked across engineering and product environments, building practical solutions and tools.

Beyond software, he enjoys creating simple, thoughtful products-ranging from books and visual tools to creative projects that explore the intersection of technology and everyday life.

In his free time, he enjoys running, swimming, and working on new ideas.

Quiet Line Press

Author Portfolio