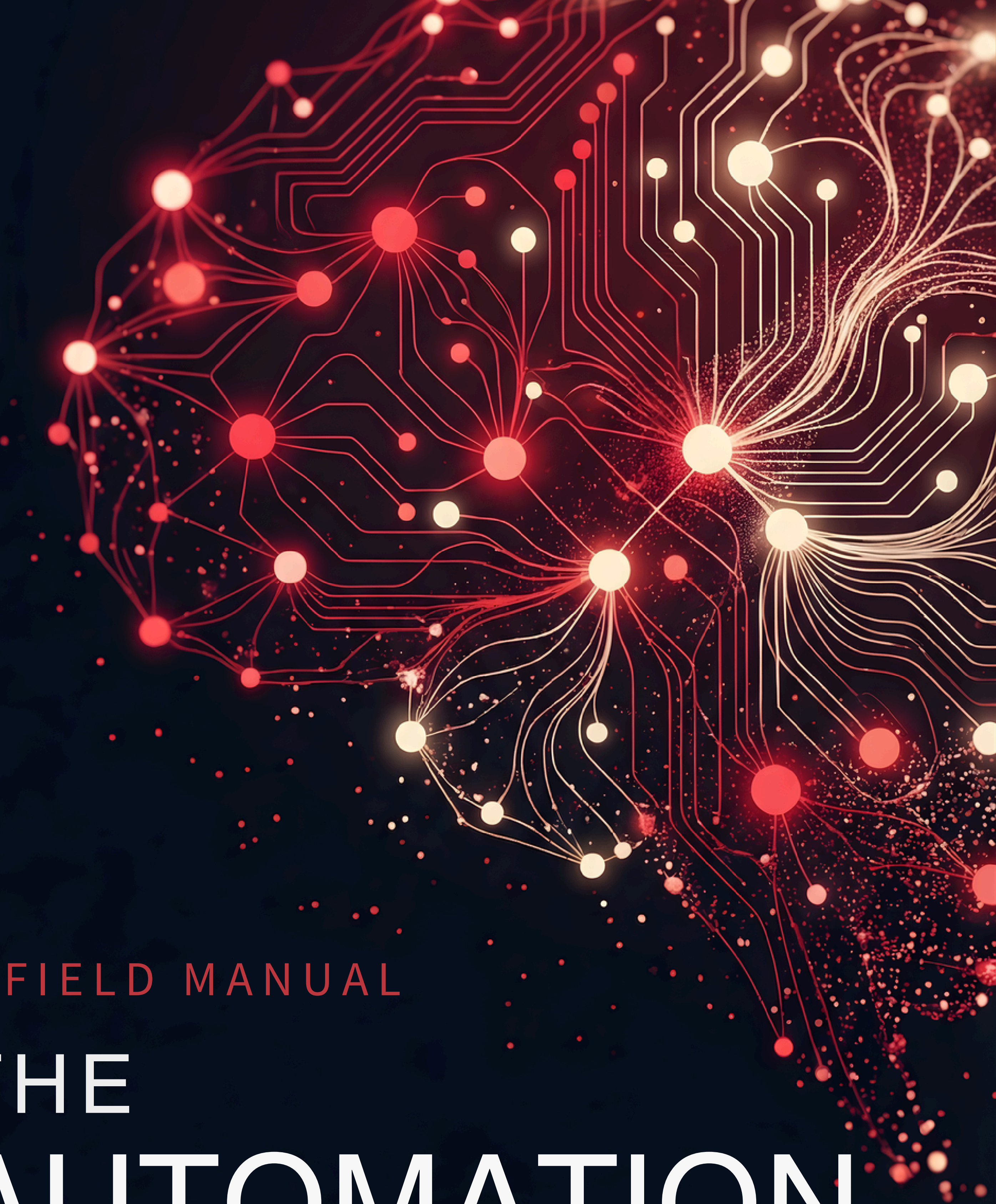

A FIELD MANUAL

# THE AUTOMATION PLAYBOOK

Building AI-Powered Systems That Work

# The Automation Playbook

*Building Systems That Actually Work*

**Published by Stackbuilt**

stackbuilt.co

Contact: nuno@stackbuilt.co

First Edition, 2026

# Contents

# Before You Start

I did not learn automation in software.

I learned it in the chemical and materials industries, where processes run every day whether you are ready or not. Where delays cost real money. Where mistakes can affect safety, quality, or reputation. Where "almost working" is the same as failure.

In those environments, you do not build systems because they are interesting. You build them because they must deliver measurable value, reliably, under real constraints. I spent years in the field—always with a foot in innovation, always looking for ways to make operations more intelligent without making them more fragile.

That mindset shaped how I approached automation. When I began exploring modern AI tools, I found two worlds. One full of demos, experiments, and impressive one-time results. Another where systems had to run every day and justify their existence economically.

This book belongs to the second world.

The most important lesson I carry came from a process optimization project early in my career. We built a system to automate quality control decisions on a production line—classifying material samples, flagging deviations, recommending adjustments. The model was accurate. The dashboard was clean. On paper, it reduced waste by double digits. In practice, the operators ignored it.

It failed because we had automated the measurement without understanding the decision. The operators did not need a faster classification. They needed confidence that the recommendation accounted for the batch variability they could see but the model could not. We had solved the data problem. They were solving the trust problem.

This book is the result of failures like that. Every framework in these pages exists because I got it wrong first. The decision framework exists because I used to fall in love with technical solutions. The testing chapter exists because I learned that a system that works 99% of the time is a system that fails every day in a continuous process. The section on agents exists because I learned that a clever prompt is not a system.

I wrote *The Innovator's Playbook* to share what I learned about turning ideas into real outcomes in industrial environments. I write the *Innovative Impulse* newsletter to keep that conversation going. This book applies the same discipline to a specific domain: building AI-powered automation that works under real constraints.

If you take one thing from this book, let it be this: **the goal of automation is not to build the most impressive system. It is to solve a real problem, measurably, under real constraints.** The most important work happens before you write a single line of code. It happens when you stand on the plant floor, or in the control room, or at the operator’s station, and ask the person who does the job: "What is the worst part of your day?"

Start there. Everything else follows.

---

*Nuno Formiga*

February 2026

# Welcome

You are here because you have repetitive work that consumes time without creating value. You have probably tried to automate parts of it before. It either broke, cost more than it saved, or you abandoned it after two weeks.

This book does not promise to change that by itself. What it offers is a system: a set of frameworks, patterns, and tools that have been tested in environments where automation must work reliably or not exist at all.

I am not going to tell you automation is "the future" or that you need to "work smarter." You do not need motivation. You need a method that survives contact with reality.

I come from the chemical and materials industries, where I spent years building and evaluating systems that had to run every day under real constraints. When I began applying those same principles to AI-powered automation, I found that the discipline transferred directly: define the problem, justify the cost, build the minimum viable system, measure it, and kill it if it does not deliver. Some of the systems I built worked. Some did not. This book is what I learned from both.

## WHAT YOU WILL FIND INSIDE

- **Complete workflows** that show you the finished product before you learn the components
- **A decision framework** that tells you what to automate and what to ignore
- **A build process** that scales from small experiments to production systems
- **Seven patterns** that cover 95% of real business problems
- **Practical integration guidance** that actually connects tools
- **Fifty prompts** that work in production
- **Agent architectures** for when prompts are no longer enough
- **A production playbook** for testing, logging, versioning, and failure handling

Most of the systems I've built landed between **$50 and $200 per month** to run, and the ones that worked paid for themselves within **two to four months**. But not all of them worked. Some never paid back. The difference was almost always in the decision layer, not the technology. If you can't articulate the ROI before you build, that's a signal. **Kill it early.** The frameworks in this book exist to help you make that call before you've invested forty hours in something that doesn't matter.

This is not a comprehensive course. It is a tactical handbook. Read it. Use the frameworks. Build one system. Measure it. Then decide whether to build the next.

*One reliable system at a time.*

## How This Book Works

Each section follows a similar rhythm because that's how I learn best and that's how most practical people I know work best.

**The Problem**—what we're solving and why it matters. This gives you context before diving into solutions. You'll understand the business problem, not just the technical implementation.

**The Framework**—how to think through the decision. Most automation books skip to the how. But the why matters more. If you understand the framework, you can adapt it to your specific situation.

**The System**—step-by-step implementation. I'll show you exactly what to build, with real tool names, real configurations, and real code examples. No abstract pseudocode that you have to translate to your stack.

**The Numbers**—real ROI with examples. Every system includes cost breakdowns, execution times, and payback periods. You'll know whether it makes sense before you build it.

**IMPORTANT**

Start with the "Start Here" chapter to see three complete workflows in action. Then read Part 1 and Part 2 for the decision framework and build process. Pick one of the patterns in Part 3. Build it. Measure it. Improve it. When you outgrow prompts, Part 6 shows you agents. When you're ready for production, Part 7 gives you the discipline.

**Don't try to build everything at once. You'll fail.**

**THE DEFAULT PATH: IF YOU DO ONLY ONE THING**

This book presents many options. If you are starting out and need a single, reliable path to follow, use this one:

- **Workflow Orchestrator:** Use **Make.com**. It has a gentler learning curve than n8n and a more visual interface, which is ideal for your first 1-5 automations.
- **Core Workflow Pattern:** Use **Pattern 1: Capture, Enrich, Store**. It is the most fundamental automation pattern and the foundation for many others.
- **LLM Provider:** Use the **OpenAI API** with the latest cost-efficient model. It provides the best balance of performance, cost, and reasoning capability for the majority of tasks.
- **Agent Architecture:** When you are ready, start with a **Declarative Agent** using the **OpenAI Assistants API**. It is the fastest way to build a tool-using agent without managing the ReAct loop yourself.

This stack is a proven starting point. Master it first. Then explore the other tools and architectures as your needs become more complex.

## The Mental Model

This is the mental model I use for every automation project. It is not original. It comes from watching what works and what fails in production environments. Its purpose is to prevent investment in systems that will not deliver.

> **Layer 1: The Decision Layer.** Before you write a single line of code or connect a single API, you need to decide whether the task is worth automating. Most people skip this. They see an interesting tool, build something, and then wonder why it does not matter.

> **Layer 2: The Design Layer.** Once you've decided to build, you need to design the system. What's the scope? What's the ROI? What happens when things break? What's the happy path? What are the edge cases? This layer prevents scope creep and ensures you're solving the right problem.

> **Layer 3: The Build Layer.** Pick the tools, connect them, write the code, configure the workflows. This is where most people start, but if you've done the work in layers 1 and 2, the build layer becomes straightforward. You're not figuring out what to build—you're just building what you designed.

> **Layer 4: The Measure Layer.** You cannot improve what you do not measure. Track execution times, success rates, costs, and whether the system is actually delivering value. This layer ensures you are not deceiving yourself about ROI.

Most people start at Layer 3. They see an interesting tool, build something, and then wonder why it does not matter. The decision framework prevents investment in low-value work. The design layer prevents scope creep. The build layer becomes straightforward when you know what you are building. The measurement layer ensures you are not deceiving yourself about whether the system delivers.

**USE THIS LOOP FOR EVERY AUTOMATION PROJECT**

- Spend **15 minutes** on the decision framework
- Spend **30 minutes** on the design template
- Spend **2-4 hours** building the initial system
- Spend **10 minutes measuring** for two weeks
- Spend **5 minutes** deciding whether to keep, tweak, or kill it

**Total upfront investment: 3-5 hours.** If the ROI isn't clear in two weeks, kill it.

START HERE

# Three Workflows in Action

Before looking at prompts, tools, or architectures, it is essential to see what a working system actually looks like. Not in theory, but end to end:

- a trigger that starts the process
- decisions made automatically
- information transformed into something useful
- value delivered without manual effort

The three workflows in this chapter are simple on purpose. Not because the problems are trivial, but because clarity matters more than complexity.

In the chemical and materials industries where I come from, the systems that survive are rarely the most sophisticated. They are the ones that are:

- understandable
- measurable
- maintainable
- economically justified

The same is true for automation with AI.

Read these workflows to understand the destination. Then the rest of the book will show you how to decide, design, build, and operate systems that deserve to exist.

**One reliable system is worth more than a hundred clever experiments.**