Building Your First AI Agent

A Step-by-Step Guide from Use Case to Deployment

By Justin P. Barrett | justinbarrett.com | Alive LLC

What Is an AI Agent?

An AI agent is software that can perceive its environment, make decisions, and take actions autonomously. Unlike a chatbot that responds to prompts, an agent has goals, tools, memory, and the ability to complete multi-step tasks without constant human input.

Think of it as the difference between a calculator (you push buttons) and an employee (you give them a goal and they figure out the steps).

Step 1: Choose the Right First Use Case

Your first agent should be:

- Repetitive — something done daily or weekly
- Rule-based — clear criteria for success/failure
- Low-risk — internal, not customer-facing
- Measurable — you can quantify the time saved or quality improvement
- Self-contained — doesn't require access to 10 different systems

Good first agent ideas:

- Daily news/competitive intelligence briefing
- Email triage and drafting responses
- Meeting note summarization and action item extraction
- Lead research and enrichment
- Report generation from data sources

Step 2: Define the Agent Architecture

Every agent has four components:

1. Goal
   What is the agent trying to accomplish? Be specific: 'Research 10 leads per day and write personalized opening lines' not 'help with sales.'
2. Tools
   What can the agent use? Web search, email APIs, databases, file systems, calculators, other APIs.
3. Memory
   What does the agent need to remember? Past interactions, company context, user preferences, previous outputs.
4. Guardrails
   What should the agent never do? Spend limits, approval requirements, escalation triggers, data it shouldn't access.

Step 3: Choose Your Platform

For your first agent, pick one:

- No-code: Custom GPTs (OpenAI)
  Upload documents, define instructions, add actions. Best for: knowledge-based agents.
- Low-code: n8n or Make + AI nodes
  Visual workflow builder with AI steps. Best for: multi-step automation agents.
- Code: CrewAI or LangGraph (Python)
  Full control, multi-agent systems. Best for: complex agents with custom logic.
- Managed: Claude Code or Cursor
  AI writes and runs code for you. Best for: technical agents built fast.

Step 4: Build and Test

- & Start with the simplest possible version (MVP agent)
- & Test with real data, not hypothetical scenarios
- & Run the agent 10 times and review every output
- & Identify failure patterns — when does it get confused?
- & Refine the prompt/instructions based on failures
- & Add guardrails for each failure pattern you discover
- & Test edge cases — unusual inputs, missing data, ambiguous requests

Step 5: Deploy and Monitor

- & Set up logging — save every input, decision, and output
- & Start with human-in-the-loop: agent proposes, human approves
- & Gradually increase autonomy as trust builds
- & Set up alerts for anomalies (unusual outputs, errors, long run times)
- & Review agent performance weekly for the first month
- & Document what the agent does in an SOP so anyone can maintain it

Common Mistakes

- Building too complex of an agent first — start simple
- No guardrails — every agent needs boundaries
- No logging — if you can't see what it did, you can't fix it
- Skipping the human-in-the-loop phase — build trust gradually
- Not defining success metrics — how will you know it's working?

About the Author

Justin P. Barrett is an entrepreneur, artist, and author. As CEO of Eyesafe, he built the global standard for display health technology. As founder of Alive LLC, he helps lean teams implement AI systems that actually work — agent architectures, knowledge bases, and automation workflows.

justinbarrett.com | justin@justinbarrett.com