Claude Code vs PI Agent — Which Coding Agent for Local AI?

February 28, 2026 · by Mark Bartlett

Download this guide as PDF

> **Quick Answer:** Claude Code is the better coding agent if you're paying for cloud models and want everything to work out of the box. PI Agent is the better coding agent if you're running local models on your own hardware and want full control over the harness. For most local-first developers, PI + Qwen 3.5 35B-A3B on a 16GB GPU gets you 80% of the way there at zero ongoing cost. For the hardest 20% of tasks, Claude Code with Opus is still unmatched.

📚 **More on this topic:** PI Agent + Ollama Setup Guide · Local Alternatives to Claude Code · Best Local Coding Models 2026 · Best Local Models for OpenClaw

Two terminal-based coding agents. Two opposite philosophies about how to build one.

Claude Code ships with 18+ tools, a ~24,000-token system prompt, five permission modes, built-in sub-agents, and tight integration with Anthropic's model family. It works well out of the box because Anthropic made most of the decisions for you. The tradeoff: $20-200/month, your code goes to Anthropic's servers, and the system prompt assumes you're running a Claude model.

PI Agent ships with 4 tools, a system prompt under 1,000 tokens, no permission system, and no sub-agents. It works well because it gets out of your way and lets the model do what it's been trained to do. The tradeoff: you configure everything yourself, build your own guardrails, and you need TypeScript if you want deep customization.

Both run in a terminal. Both read files, write code, and execute commands. Which one you should use depends on whether you value batteries-included or total control, and whether you're paying for cloud models or running on your own GPU.

The comparison

Feature	Claude Code	PI Agent
License	Proprietary	MIT (open source)
Price	$20-200/month or API keys	Free
System prompt	~24,000 tokens (with all tools)	<1,000 tokens

Feature	Claude Code	PI Agent
Built-in tools	18+ (Read, Edit, Write, Bash, Glob, Grep, WebSearch, WebFetch, Notebook, Task…)	4 (read, write, edit, bash)
Model support	Claude family (Sonnet, Opus). Others via API proxy	15+ providers, hundreds of models. Ollama, llama.cpp, vLLM native
Local model support	Yes (Ollama v0.14.0+), but not optimized	First-class, model-agnostic
Permission system	5 modes (default, acceptEdits, plan, dontAsk, bypassPermissions)	None (YOLO). Build your own via extensions
Sub-agents	3 built-in types (Explore, Plan, Task), run in parallel	None. Spawn PI recursively via tmux, or build via extensions
MCP support	Yes, built-in	No. Uses CLI tools + READMEs instead
Extension system	CLAUDE.md, hooks (14 event types), skills, slash commands	TypeScript SDK, 50+ event hooks, custom UI widgets, themes, key bindings
Session management	Linear history	Tree-structured, forkable, shareable
IDE integration	VS Code, JetBrains	Terminal only
Git integration	Built-in	Via bash
Enterprise features	SSO, audit logs, team plans	None
GitHub stars	N/A (closed source)	18,187 (badlogic/pi-mono)

System prompt: why the size difference matters

Claude Code's system prompt runs ~24,000 tokens when all tools are loaded. Tool descriptions alone account for ~11,600 tokens. Add MCP servers and you can hit 34,000-44,000 tokens before you've typed a single message.

PI's system prompt is under 1,000 tokens total, including tool specifications.

Mario Zechner's argument: modern coding models have been RL-trained so extensively on agent tasks that a giant system prompt mostly tells them what they already know. Those 23,000 extra

tokens could hold actual code from your codebase instead. On a local model with 32K context, that difference is between fitting your project and not fitting it.

Anthropic's counter-argument (implicit in their design): detailed system prompts catch edge cases and prevent the agent from rm -rf'ing your project. The safety instructions alone justify thousands of tokens if you're running an agent with filesystem access.

The real question is context budget. If you're running Claude Opus with 200K context, 24K tokens of system prompt is 12% of your window. Not cheap, but manageable. If you're running Qwen 3.5 35B-A3B locally with an effective 32K context, burning 24K on system prompt would leave you 8K for actual work. PI's 1K prompt leaves 31K for code.

For local model users, this matters more than anything else in the comparison.

Tools: 18+ vs 4

Claude Code gives you specialized tools for file search (Glob), content search (Grep), web search, web fetch, notebook editing, task management (TodoRead/TodoWrite), and agent orchestration (Task, TeammateTool). Each tool has a detailed schema that the model can call directly.

PI gives you read, write, edit, and bash. Need to search files? Run `rg` via bash. Need web content? Run `curl` via bash. Need to manage tasks? Write them to a TODO.md file.

The PI philosophy: bash is universal. Every CLI tool becomes an agent tool automatically. No wrapper, no token-expensive tool description, no compatibility layer. Mario tested this against MCP servers and found that a 225-token CLI README outperformed a 13,700-token Playwright MCP server for browser automation tasks.

Can Boluk tested this further with oh-my-pi (a PI fork). By changing only the edit tool format, he improved 15 models at coding in a single afternoon. Grok Code Fast 1 went from 6.7% to 68.3%. Gemini 3 Flash gained 5 percentage points over Google's own str_replace implementation. The harness design mattered more than the model.

The practical difference: Claude Code's specialized tools work reliably across a wide range of models and tasks because the model was trained specifically to use them. PI's bash-everything approach works well with capable models but fails more often with smaller ones that struggle to compose complex shell commands.

Local model support: the real story

Claude Code + local models

Since Ollama v0.14.0 (January 2026), Claude Code can connect to any Ollama-served model via the Anthropic Messages API:

```
ANTHROPIC_BASE_URL=http://localhost:11434/api
ANTHROPIC_MODEL=qwen3-coder
```

It works. But there are problems.

Community benchmarking shows ~68x slower performance compared to cloud. Each tool call takes about 2 minutes on a MacBook Pro M1 Max with a 14B model, compared to 2-3 seconds with cloud Claude. The system prompt assumes Claude-level capabilities. 24,000 tokens of instructions written for Opus don't help a 7B local model. They hurt it by consuming context it needs for your code.

Claude Code's specialized tool descriptions (Glob, Grep, WebSearch, etc.) add token overhead that a local model running everything through bash doesn't benefit from. You're paying context tax for tools the model won't use as effectively as Claude would.

Claude Code + Ollama works. It's just not what either tool was designed for.

PI Agent + local models

PI was built for this. The minimal system prompt leaves maximum context for your code. The model-agnostic design means no assumptions about what the model can or can't do. Configuration is explicit:

```
{
  "providers": {
    "ollama": {
      "baseUrl": "http://localhost:11434/v1",
      "api": "openai-completions",
      "apiKey": "ollama",
      "models": [
        { "id": "qwen3.5:35b-a3b", "name": "Qwen 3.5 35B-A3B", "contextWindow": 131072 }
      ]
    }
```

```
    }
  }
```

Switch models mid-session with `/model`. Use a small model for fast lookups, a large one for complex reasoning. PI tracks token usage across models so you can see where context goes.

The minimal tool set means fewer chances for a local model to misfire on a complex tool schema. Read, write, edit, bash. If the model knows how to use a terminal, it knows how to use PI.

This is what PI was designed for, and the experience reflects it.

Customization: medium vs extreme

Claude Code gives you CLAUDE.md for project instructions, shell-command hooks on 14 event types, skills (reusable prompt templates), and custom slash commands. You can configure permission rules, add to the system prompt, and build workflows around hooks. It's a reasonable set of customization points that covers most needs.

PI gives you the source code. Beyond that, the extension system exposes 50+ event hooks with full access to the agent loop, tool execution, validation, and TUI rendering. You can build custom UI widgets, inject context dynamically, intercept and modify tool calls, implement RAG pipelines, add custom compaction strategies, and create entirely new tools. PI packages bundle extensions, skills, prompts, and themes into installable units.

This is where PI's "build it yourself" philosophy either thrills or exhausts you. Want sub-agents? Write an extension that spawns PI instances via tmux. Want permission gates? Write an extension that intercepts tool calls. Want plan mode? Write an extension. Tobi Lutke (Shopify CEO) described PI as "the most interesting agent harness" because it "RLs itself into the agent you want." He told it to interrogate Claude Code's task system via tmux and implement something similar for itself.

That's powerful. It's also a weekend project for every feature Claude Code ships for free.

Cost: what you actually pay

Claude Code

Plan	Monthly	What you get
Pro	$20	~45 Sonnet messages per 5-hour window. Enough for light use
Max 5x	$100	~225 messages per 5-hour window. The “sweet spot” for daily use
Max 20x	$200	~900 messages per 5-hour window. Full Opus access
API (pay-as-you-go)	Variable	Anthropic says average $6/day, 90th percentile $12/day

Rate limits are demand-dependent. During US business hours, the Pro plan might drop to 35-40 messages. Off-peak, you might get 50-60. Heavy users on the $200 plan still report hitting weekly caps.

PI Agent + local models

Component	Cost	One-time or recurring
PI Agent	$0	Free forever (MIT license)
GPU (RTX 5060 Ti 16GB)	$430-500	One-time
Electricity (~180W, 8 hrs/day)	~$6/month	Recurring
Total first year	**~$500-570**	Mostly one-time

Or if you already have a GPU: $0.

No rate limits, no throttling, no weekly caps. The only limit is how fast your GPU generates tokens.

The hybrid math

If you use Claude Code Max 5x ($100/month) and switch to PI + local models for 80% of your work:

- Before: $1,200/year (Claude Code Max 5x)
- After: $240/year (Claude Pro for hard tasks) + $500 (one-time GPU, if needed)
- First-year savings: $460+
- Second-year savings: $960+

The hybrid approach is the consensus on r/LocalLLaMA and Hacker News. Use local for routine coding, fall back to Claude for complex multi-file reasoning.

What each tool does better

Claude Code wins at

Complex multi-file refactoring. Claude Opus with 200K context can coordinate changes across dozens of files. Specialized tools like Glob and Grep let it search precisely without burning tokens on bash output formatting. Built-in sub-agents can explore the codebase in parallel while the main agent plans changes.

Out-of-box reliability. Install it, give it an API key, and it works. No config files, no model selection, no debugging tool-calling failures with a local model. The permission system prevents catastrophic mistakes. The system prompt handles edge cases.

Team and enterprise use. SSO, audit logs, managed plans, consistent behavior across developers. PI has none of this.

Hard coding problems. When you need the absolute best model intelligence, Claude Opus 4.5/4.6 is still the highest-scoring agent on SWE-bench Verified (80.9%) and Terminal-Bench (65.4%). The gap between Opus and the best local model (Qwen3-Coder-Next at 70.6%) is 10+ points. On the hardest 20% of tasks, that gap is larger.

PI Agent wins at

Local model performance. A 1,000-token system prompt vs 24,000 means 23K more tokens for your actual code. On a 32K context model, that's the difference between fitting your project and not.

Cost at scale. Zero marginal cost per message, per day, per month. Run 1,000 tool calls a day and it costs you electricity.

Privacy. Nothing leaves your machine. No code sent to any server, no conversation logs on anyone's cloud, no data retention policy to read. The only option for air-gapped environments and classified work.

Customization depth. 50+ extension hooks, full harness control, custom UI, custom tools, custom compaction. If Claude Code doesn't let you do something, you're stuck. PI lets you do anything if you're willing to write TypeScript.

Context efficiency. The harness-vs-model argument is real. Can Boluk proved that changing just the edit tool improved 15 models by 5-68 percentage points. PI's minimal harness gives the model more room to think.

Offline operation. No internet required. Pull your model, configure PI, and work from an airplane, a cabin, or a classified facility.

The verdict

If you run a home lab with Ollama: PI Agent. It was built for exactly this. Pair it with Qwen 3.5 35B-A3B on a 16GB GPU and you have a capable coding agent at zero ongoing cost. The 1,000-token system prompt means your local model spends context on your code, not on instructions it doesn't need.

If you're a daily driver using cloud models: Claude Code. The out-of-box experience with Sonnet/Opus is hard to beat. Sub-agents, specialized search tools, and the permission system save you from building infrastructure that's already built. The $100/month Max plan is the sweet spot.

If you want maximum control and enjoy building tools: PI Agent regardless of whether you use local or cloud models. The extension system is the deepest in any coding agent. Armin Ronacher (creator of Flask) uses PI "almost exclusively." Tobi Lutke calls it "the most interesting agent harness."

If you work on a team or in an enterprise: Claude Code. PI has no team features, no audit trail, no managed deployment.

If you want the hybrid (and you probably should): Use PI + local models for the 80% of routine coding tasks that don't need frontier intelligence. Keep a Claude Code Pro plan ($20/month) for the 20% that does. That's $240/year in cloud costs plus a one-time GPU investment, instead of $1,200-2,400/year for full Claude Code.

User Type	Pick	Why
Budget hardware, local-only	PI Agent	Zero cost, maximum context for local models
Privacy/security-critical	PI Agent	Nothing leaves your machine
Power user, likes customizing tools	PI Agent	50+ extension hooks, full harness control
Daily driver, cloud models	Claude Code	

User Type	Pick	Why
		Best out-of-box experience, best model quality
Team/enterprise	Claude Code	SSO, audit, managed plans
Hybrid (80/20 local/cloud)	PI for routine + Claude for hard problems	Best of both, lowest total cost

Getting started

PI Agent + Ollama:

```
npm install -g @mariozechner/pi-coding-agent
ollama pull qwen3.5:35b-a3b
```

See the full PI + Ollama setup guide for config files and model recommendations.

Claude Code:

```
npm install -g @anthropic-ai/claude-code
claude
```

Requires an Anthropic API key or Claude Pro/Max subscription.

The hybrid setup:

1. Install both. Use PI for daily coding with your local model
2. Keep a Claude Pro plan ($20/month) for problems that stump the local model
3. Switch between them based on task difficulty

They're different tools for different situations. Pick based on your hardware, your budget, and how much setup you're willing to do.

📚 **Related guides:** PI Agent + Ollama Setup · Best Local Coding Models 2026 · Local Alternatives to Claude Code · Best Local Models for OpenClaw · RTX 5060 Ti Local AI Benchmarks

Get notified when we publish new guides.

Subscribe — free, no spam

Source: https://insiderllm.com/guides/claude-code-vs-pi-agent-local-ai/

Free guides for running AI locally