Member-only story

From Prompts to Claude Skills: 7 Steps How to Actually Ship Fully Customized AI For Your Needs

Easy Way To Create Claude Skills For AI Automation Without Losing Your Mind.

13 min read · 1 day ago

Reza Rezvani Following

Listen Share More

Claude Skills — Build Your Own and Fully Customizable Agent Skills and much more

Yesterday, I shared a deep dive into building your own AI Automation Workforce with the *Claude Skill Factory.* Today, I want to take it one step further — to show you how simple it can be to create your own AI-powered toolkit that actually works *for you*. No heavy setup, no steep learning curve — just a framework that adapts to your ideas, your workflow, and your creativity.

. . .

The Tuesday Morning Problem Nobody Admits. It's 9 AM on a Tuesday. You're in a meeting.

Your marketing manager says: *"Can Claude automate our campaign analysis?"*

> *You say yes (because of course you did).*

By Friday, you've spent hours writing, rewriting, and tweaking prompts. The results are okay. Mostly. Sometimes. On Tuesdays it works great. On Wednesday it's mediocre. By Friday, you've added so many *"clarifications"* to your prompt that it's 2,000 words long.

Your manager asks: *"Is this done?"*

You say: *"Almost. Let me just adjust one more thing…"*

> You never actually ship it.

This is the prompt engineering trap. You get *close* to solving the problem, but you're stuck in endless iteration mode. It works once, breaks the next time, works again with a different dataset. You're not building — you're debugging endlessly.

Claude Code v2.0.30: Full Guide of what is New? Production Readiness Edition

Claude Code v2.0.30: Full Guide of what is New? Production Readiness Edition How Anthropic's latest update for Claude…

alirezarezvani.medium.com

Here's the thing: ***that's not your fault.*** That's how prompt engineering works. It's like trying to build a house by arranging bricks differently every time instead of actually using concrete and blueprints.

> The bricks never stay in the same place.

I spent six months watching teams doing this. Marketing teams. Finance teams. Support operations. The pattern was always the same:

- Build a cool prompt
- Use it twice
- Realize it needs tweaking
- Update it
- Now three other people are using the old version

- Chaos

Then Anthropic released something that changed the game: **Claude Skills.**

Not a feature. An actual different way of thinking about AI automation.

Why Agent Skills Will Transform How We Build AI

How Anthropic's new Agent Skills framework turns general-purpose AI into specialized experts — and why it changes...

alirezarezvani.medium.com

Why Your Perfect Prompt Isn't Actually Production-Ready

Let me be honest about something: prompts are fantastic for exploration. They're terrible for production.

Here's why:

Your perfect prompt today is someone else's nightmare tomorrow. You spend two hours crafting the most brilliant financial analysis prompt. It's beautiful. Perfectly formatted instructions. Chain-of-thought reasoning. Everything.

Then our Clinical R&D Team lead from your team uses it. She gets different output. *"But I used the exact same prompt!"* she says.

You both did. Same prompt. Different results. Welcome to the inconsistency problem.

Maintaining prompts across your team is like herding cats. Someone copies your marketing prompt and modifies it slightly. Now you have two versions. Then someone else modifies that version. Now you have three versions. Six months later, you have nine versions and nobody knows which one is the *"right"* one.

Long prompts waste money. You add more and more instructions, context, and examples. Now you're sending 3,000 tokens just to describe what you want. Claude reads all of it, even for simple tasks. Your costs go up. Speed goes down.

They don't scale to teams. Prompts live in someone's head or in a shared doc. No version control. No testing. No approval process. When something breaks, nobody knows who to call.

Here's the research part ***(and I promise to keep it brief)*****:** A study from Wharton found that identical prompts produce wildly different results on individual tasks — sometimes by 60%. That's not user error. That's the fundamental nature of prompt-based systems.

The real insight? You're not building a system. You're crossing your fingers every time someone uses a prompt.

That's fine for *"write me a funny poem."* That's a disaster for *"generate my company's financial reports for the last Q."*

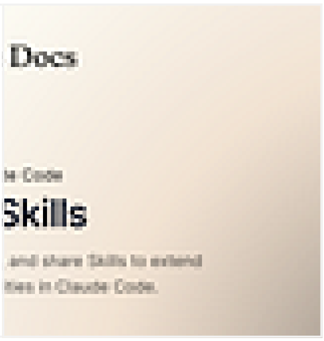
Skills: What Happens When You Actually Think Like an Engineer

Two weeks ago, Anthropic released Claude Skills. Think of it as: ***"What if instead of a long, fragile prompt, you packaged your expertise like real software?"***

Agent Skills - Claude Docs

Create, manage, and share Skills to extend Claude's capabilities in Claude Code.

docs.claude.com


Docs

e Code

Skills

and share Skills to extend

ties in Claude Code.

Here's the mental shift:

Instead of: *"Here are my instructions. Good luck."*

You get: *"Here's how my business works, here are the rules, here's what good output looks like, here are the templates, and here's how to validate the results."*

Skills is a folder. In that folder, you put:

- Instructions *(plain English, not some weird prompt format)*
- Code *(optional, for the tricky logic bits)*
- Templates *(the exact format you want)*
- Examples *(here's good output, here's bad output)*
- Rules *(business logic that never changes)*

Claude loads all of this when it needs to, applies it consistently, and produces results that actually work.

The secret sauce? Something called **progressive disclosure**. Fancy term for a simple idea: Claude doesn't read instructions it doesn't need.

When you ask Claude to analyze customer support tickets using your Support Analyzer Skill, it doesn't waste tokens reading the financial reporting instructions. It loads the one skill that matters, uses it, and gives you the answer.

Real-world impact? A famous corporate, a massive e-commerce company, used Skills to automate their finance operations.

Before Skills: Finance team spends 1 full day doing accounting reports. **After Skills:** Same reports in 1 hour.

That's not a 10% improvement. That's 75% of an entire day freed up. Per person. Every month.

Three Business Problems Skills Actually Solves

Let me show you exactly where this matters in your business, because theory is nice but results matter.

Problem #1: Your Marketing Team Is Drowning in Repetitive Analysis

Right now, here's what your marketing team probably does:

1. Get competitor campaign data *(manual download)*
2. Paste it into a doc
3. Write out all the analysis criteria they want *(takes 20 minutes)*
4. Paste output somewhere
5. Reformat because Claude's output doesn't match the template
6. Send it to leadership
7. Repeat for next competitor

Multiply this by 10 competitors per month. That's 10+ hours of work that's 90% repetitive.

With a Marketing Campaign Analyzer Skill:

Your skill knows:

- Your company's brand voice and messaging framework
- The specific KPIs you care about (engagement rate, conversion, cost per lead)
- The exact template you always use
- The format leadership expects (usually a specific PowerPoint layout)
- What "good analysis" looks like for your company

Now your team just uploads competitor data. The skill runs. Output appears formatted, consistent, ready to present.

Same work gets done in 15 minutes instead of 2 hours.

Problem #2: Your Finance Team Manually Builds Monthly Reports

This is painful to watch. Finance team:

1. Pulls data from three different systems
2. Loads into Excel

3. Calculates variances manually *(or with formulas they don't quite trust)*
4. Writes narrative explanations
5. Formats into PowerPoint
6. Sends to CFO for review
7. CFO sends back: *"Can you add the Q2 comparison?"*
8. Back to Excel
9. Update formulas
10. Update narrative
11. Format again
12. Resubmit

This process takes 6 hours. Every month. That's 72 hours a year of finance team time.

With a Finance Report Generator Skill:

Your skill packages:

- Your reporting template (specific Excel schemas, exact PowerPoint layouts)
- Variance analysis rules (what counts as "reportable," how to flag it)
- Narrative guidelines (how your CFO expects the story told)
- Approval workflows (compliance requirements, sign-off process)
- Historical context (last year's numbers for comparison)

Now: Finance team uploads raw data. Skill generates complete report with:

- Populated spreadsheets
- Calculated variances
- Written narratives
- Formatted presentation slides

- Compliance checks built in

Review time: 20 minutes instead of 6 hours.

Problem #3: Your Content Team Can't Keep Up With Market Demands

Your content strategist needs to:

1. Research market trends *(browsing, reading, taking notes)*
2. Identify content gaps *(comparing to competitors)*
3. Generate article outlines *(based on your audience)*
4. Optimize for SEO *(keyword research, structure)*
5. Ensure brand alignment *(voice, tone, values)*

This takes all day. For one article strategy.

With a Content Strategy Skill:

Your skill knows:

- Your target audience *(who they are, what they care about)*
- Your SEO framework *(keywords you care about, structure you prefer)*
- Your brand voice *(examples of good content)*
- Your publication standards *(length, format, calls-to-action)*
- Your competitive positioning

Now: Drop your topic. The skill:

- Researches trends
- Identifies gaps
- Generates outlines
- Optimizes for SEO
- Aligns with brand

- Produces research-backed recommendations

Time: 2 hours instead of 8 hours.

GitHub - alirezarezvani/claude-code-skill-factory: A comprehensive toolkit for generating…

A comprehensive toolkit for generating production-ready Claude Skills and Claude Code Agents at scale. …

github.com

Actually Building Your First Skill (Stop Being Scared, It's Easier Than You Think)

This is the part where people get intimidated. *"I'm not technical enough." "I don't know YAML." "This seems complicated."*

Stop. You don't need to be a developer. Anthropic built this specifically so regular people could do it.

I'm going to walk you through building one from start to finish. Real steps. Real examples. No bullshit.

The Setup (Before You Start)

You need:

- Claude Code (free trial works)
- One specific workflow from your team (something they do weekly)
- 15 minutes of your time

That's it. No IDE. No terminal *(unless you want to)*. No special knowledge.

Pick your first skill wisely. Not something grand. Something your team does literally every week that takes 1–2 hours.

Examples that work great:

- *"We analyze support tickets and categorize them"*
- *"We review weekly social media performance"*

- *"We extract key metrics from sales data"*
- *"We summarize meeting notes into action items"*

Avoid picking:

- Something you've never done *(you won't know the rules)*
- Something vague *("help with business analysis")*
- Something that changes format every time

Step 1: Tell Claude What You Want (5 minutes)

Open Claude Code. Type this:

```
I want to build a Skill that [describes your specific workflow].

Here's what my team currently does:
1. [Step one]
2. [Step two]
3. [Step three]
Here's what I want automated:
[Be specific]
Current pain points:
[What's annoying about doing this manually?]
```

Real example:

```
I want to build a Skill that automates our weekly support ticket triage.

Here's what my team currently does:
1. Export support tickets from our system
2. Read through each ticket
3. Categorize by urgency (critical, high, medium, low)
4. Categorize by type (billing, technical, feature request, bug report)
5. Flag tickets that need immediate escalation
6. Generate a summary report
Here's what I want automated:
Automatically categorize incoming tickets and generate a daily summary
showing critical/high priority items at the top.
Current pain points:
- Takes 30 minutes daily
- Sometimes tickets are miscategorized
```

```
   - Easy to miss critical issues when you're tired
   - Report format keeps changing
```

Step 2: Claude Asks Clarifying Questions (2 minutes)

Claude will ask:

“What data will this skill work with?”
Answer: “JSON export from our support system with fields: ticket_id, customer_name, description, created_at”

“How do you currently decide priority?”
Answer: *“Critical = customer is blocked from using product. High = feature not working. Medium = minor bug. Low = feature requests.”*

“What should the output look like?”
Answer: *“A markdown report with critical tickets listed first, then organized by category, with action recommendations.”*

“Do you have templates or examples?”
Answer: *“Here’s our current template…” (paste example)*

Don’t overthink these answers. Just be specific about what your team actually does.

Step 3: Claude Generates Your Skill Structure (3 minutes)

Claude creates a folder structure. You don’t need to understand everything. Here’s what gets created:

```
support-ticket-analyzer/
├── SKILL.md (your instruction file)
├── categorize_tickets.py (the logic)
├── priority_rules.json (business rules)
├── sample_input.json (example data)
└── sample_output.md (what good output looks like)
```

The important part: SKILL.md

This looks like:

~~~
---
name: support-ticket-analyzer
description: Analyzes support tickets, categorizes by urgency and type,
             flags items needing immediate escalation
model: sonnet
version: 1.0
---
## What This Skill Does
Takes raw support tickets and generates a prioritized report showing:
- Critical issues needing immediate attention
- Categorized by type (billing, technical, feature request, bug)
- Recommended actions
## When to Use This
Every morning to review overnight tickets. Any time you need ticket analysis.
## How It Works
1. Reads ticket descriptions
2. Applies priority rules (critical = product blocked)
3. Categorizes by ticket type
4. Flags for escalation if needed
5. Generates markdown report
## Your Priority Rules
- **CRITICAL**: Customer cannot use core product feature
- **HIGH**: Important feature not working, workaround exists
- **MEDIUM**: Minor bugs, cosmetic issues
- **LOW**: Feature requests, suggestions, questions
## Example Input
```json
{
  "tickets": [
    {
      "id": "T-1001",
      "customer": "Acme Corp",
      "description": "Payment processing completely down",
      "created": "2025-10-30T08:00:00Z"
    }
  ]
}
~~~

Example Output

```
## CRITICAL ALERTS (1)
**T-1001** | Acme Corp
Payment processing completely down
**Recommended Action**: Immediate escalation to engineering
**Category**: Technical/Critical
**Created**: Today 8:00 AM
```

```
---
## HIGH PRIORITY (0)
[No high priority items]
```

Step 4: Test With Real Data (3 minutes)

This is crucial. Don't skip this.

Export some real tickets from your system. Test the skill:

```
Here are 5 real support tickets from this morning.
Analyze them using the support-ticket-analyzer skill.
```

Does the output look right? Are priorities correct? Is the format what you wanted?

If yes: move to step 5.

If no: *"Claude, the output should [what it should do]. Can you adjust?"*

Claude adjusts. Test again.

Step 5: Install in Claude Code (2 minutes)

Claude will give you the folder. Copy it:

```
cp -r support-ticket-analyzer ~/.claude/skills/
```

That's it. Done.

Now, whenever you're working in Claude Code, you can just say:

"Analyze these support tickets using my support-ticket-analyzer skill."

Claude loads it. Applies it. Gives you results.

Step 6: Share With Your Team (1 minute)

Upload to GitHub or shared folder:

```
Your skill is ready. Team can install it with:
cp -r support-ticket-analyzer ~/.claude/skills/
```

Now everyone on your team uses the exact same categorization logic. Consistency guaranteed.

The "Oh Crap, I Need to Change Something" Moment

You're going to build your skill. You'll use it for a week. Then you'll realize: *"Wait, I need it to also flag VIP customers."*

Here's the good news: ***You don't rebuild everything. You just update.***

Open your skill. Find `priority_rules.json`. Add:

```
{
  "vip_customers": ["Acme Corp", "TechCorp Inc"],
  "vip_escalation": true
}
```

Update the SKILL.md to mention VIP handling. Test with one ticket. Done.

No rewriting prompts. No copy-pasting. No *"which version am I using?"* chaos.

You version control it. You update it. Everyone gets the new version.

Why This Matters for Your Business (The Real Conversation)

Let's be real: Claude Skills could be just another tool. It's not.

Here's why:

> Your team's knowledge finally compounds. Right now, you build a workflow. Someone learns it. They leave. Knowledge is gone. You rebuild it worse.

With Skills, the workflow is captured. It stays. Your team gets better, not reset.

You can actually measure consistency.
"Last month, tickets were categorized wrong 15% of the time. This month, 0%."

You can track this. You can see the improvement. You can tie it to dollars saved.

Your business becomes less dependent on individual people.

Our Senior Finance Manager knows how to do the financial analysis. When he goes on vacation, chaos. With a skill? Anyone can run the analysis. Same quality every time.

Cost goes down. Speed goes up. Reliability increases.

This isn't hype. It's math.

Open Source, Community-Driven, Shaped by Your Feedback

Here's something important: **This ecosystem isn't locked up.**

The Claude Code Skill Factory is open source. Alireza isn't deciding what skills the world needs. You are.

GitHub - alirezarezvani/claude-code-skill-factory: A comprehensive toolkit for generating...

A comprehensive toolkit for generating production-ready Claude Skills and Claude Code Agents at scale. ...

github.com

Every skill you build. Every workflow you automate. Every time you go *"Hey, wouldn't it be cool if..."* and then actually build it — that shapes what comes next.

Right now, the community is still small. Skills just launched. This is the moment where the direction gets set.

You could build:

- A marketing skill that goes viral with 5,000 downloads
- A finance template that becomes the industry standard
- A support automation that saves your industry millions

Or you could just solve your team's one specific problem and help everyone else trying the same thing.

Either way, you're not just building for yourself. You're building infrastructure.

Your Next 30 Minutes

Stop thinking. Stop planning. Go build something.

Pick one workflow. The one that's annoying you right now. The one you said *"someday we should automate this."*

Today is someday.

Open Claude Code. Follow the steps above. Build your skill. Test it. Share it.

Then come back and tell me what you built. File an issue on the GitHub repo. Leave a comment. Tell Alireza what worked, what didn't, what you wish was different.

alirezarezvani/claude-code-skill-factory

Claude Code Skill Factory - A powerful open-source toolkit for building and deploying production-ready Claude Skills...

github.com

zvani/claude-
ill-factory

That feedback matters. It's not noise. It's directly shaping the toolkit that the next person will use.

The future of AI isn't clever prompts people argue about on Twitter. It's production-grade tools that your team actually uses. Tools that work consistently. Tools that compound your knowledge over time.

You're not just automating one workflow.

You're building the infrastructure for how your team works in 2026.

. . .

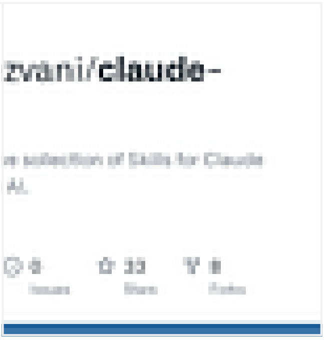
Additional Claude Skill Free Resources:

Ready to Use Claude Skills for Your Teams:
Production-Ready Claude Skill packages for Claude AI & Claude Code — Reusable expertise bundles combining best practices, analysis tools, and strategic frameworks for marketing teams, executive leadership, product development, your web and mobile engineering, and Regulatory Affairs and QA teams. Many other teams will be included soon and regularly. It is Free and Open Source!

GitHub - alirezarezvani/claude-skills: A comprehensive sollection of Skills for Claude Code or...

A comprehensive sollection of Skills for Claude Code or Claude AI. - GitHub - alirezarezvani/claude-skills: A...

github.com



Check out also My Claude Code Tresor Project:
Claude Code Tresor is the ultimate collection of professional-grade utilities for Claude Code users such as Cutom Slash Commands, Claude Code Subagents or recently added Workflow Hooks and Agent Skills. Whether you're a solo developer or part of a team, this repository provides battle-tested tools that integrate seamlessly into your development workflow. It is Free and Open Source!

GitHub - alirezarezvani/claude-code-tresor: A world-class collection of Claude Code utilities...

A world-class collection of Claude Code utilities: autonomous skills, expert agents, slash commands, and prompts that...

github.com

I'm continuously improving these toolkits and frameworks in my free time and would really appreciate it if you shared your ideas, feedback, and thoughts with me on GitHub.

. . .

✨ *Thanks for reading! If you'd like more practical insights on AI and tech, hit* ***subscribe*** *to stay updated.*

I'd also love to hear your thoughts — drop a comment with your ideas, questions, or even the kind of topics you'd enjoy seeing here next. Your input really helps shape the direction of *this channel.*

. . .

About the Author

Me, Alireza Rezvani work as a CTO @ an HealthTech startup in Berlin and architect AI development systems for my engineering and product teams. I write about turning individual expertise into collective infrastructure through practical automation.

Connect: Website | LinkedIn
Read more: Medium Reza Rezvani

Explore my other open source projects: GitHub

Ai Automation
Ai Agent
Marketing Automation
Artificial Intelligence
Productivity

Following

Written by Reza Rezvani

972 followers · 73 following

As CTO of a Berlin AI MedTech startup, I tackle daily challenges in healthcare tech. With 2 decades in tech, I drive innovations in human motion analysis.

No responses yet

Bgerby

What are your thoughts?

More from Reza Rezvani

In nginity by Reza Rezvani

I Let Claude Sonnet 4.5

IMAGINE this: It’s 6 a.m., the kind of quiet dawn where the world’s still wrapped in that soft, hazy light filtering through your blinds…

Sep 29 101 1

Reza Rezvani

“7 Steps” How to Stop Claude Code from Building the Wrong Thing (Part 1): The Foundation of...

Learn how to stop Claude Code from rewriting your architecture with vague prompts. This guide introduces Spec-Driven Development...

Sep 17 44 2

In nginity by Reza Rezvani

How Cursor and Claude Code Plugins Turned Me Into a 20x Developer – The Agentic Coding Setup That...

My Background Agent just submitted a PR that made our senior architect ask, “Who wrote this?”

Oct 14 73

Reza Rezvani

Gemini CLI: What Happened When I Replaced My IDE With a Free AI Terminal Agent for 30 Days

I gave Google’s new Gemini CLI full access to my development workflow and tested it on real production code. Here’s what actually worked...

Oct 10 20

See all from Reza Rezvani

Recommended from Medium

In Coding Nexus by Code Coup

Claude Desktop Might Be the Most Useful Free Tool You'll Install This Year

I didn't expect much when I first saw the announcement for Claude Desktop. Another AI wrapper, I thought. Maybe with a shiny UI.

Oct 23 264 8

In Towards AI by Teja Kusireddy

We Spent $47,000 Running AI Agents in Production. Here's What Nobody Tells You About A2A and MCP.

Multi-agent systems are the future. Agent-to-Agent (A2A) communication and Anthropic's Model Context Protocol (MCP) are revolutionary. But...

Oct 16 1.3K 24

Agen.cy

20+ Genius Ways Power Users Are Using Claude Code Right Now

Here are 10+ ways power users are using Claude Code🧵

Oct 23 31 1

In Data Science Collective by Simon Greenman

The AI Vibe Coding Paradox: Why Experience Matters More Than Ever

AI can now code faster than any developer. But without experienced leadership, it breaks down in record time

4d ago 179 16

In Artificial Intelligence in Plain English by aakash

Perplexity's New AI Agents Are INSANE

Build complex apps, games, and tools in a single click, deploy them instantly, and transform your creative potential—all without writing...

Aug 24 146 3

In Dare To Be Better by Max Petrusenko

Unleash Your Inner Wizard: Claude Skills Are Automating Enterprise for Pennies

How a simple folder is replacing $50K consultants and saving companies literal days of work

Oct 17 399 6

See more recommendations