

# **Automating Specialized Prompts**

MGMT 675: Generative AI for Finance

---

Kerry Back

# What is a Skill?

A **skill** is a folder containing instructions and optional code that transforms Claude Code into a specialized agent for a specific domain or task.

## A Skill Provides

- System prompt (domain knowledge)
- Workflow instructions
- Best practices and constraints
- Python/JavaScript helper scripts
- Reference documentation

## Claude Code Provides

- The LLM (Opus or Sonnet)
- Agent control loop
- File read/write tools
- Code execution sandbox
- Web search capability

# Anatomy of a Skill

## Skill Folder Structure

```
skills/  
  xlsx/  
    SKILL.md      <- Main file  
    scripts/  
      recalc.py  
    references/  
      schema.md
```

## SKILL.md Structure

```
---  
name: xlsx  
description: "Excel file..."  
---  
  
# Requirements for Outputs  
- Zero formula errors  
- Use formulas, not hardcodes  
  
# Workflows  
1. Choose pandas or openpyxl  
2. Create/modify file  
3. Recalculate formulas
```

# SKILL.md: The System Prompt

- **Frontmatter:** Name and description (YAML header)
- **Requirements:** Quality standards and constraints
- **Workflows:** Step-by-step procedures
- **Code Examples:** Patterns for the LLM to follow
- **Error Handling:** How to diagnose and fix problems

The SKILL.md file is automatically injected into Claude's context when working in a project that contains the skill.

# Scripts: The Custom Tools

- Python or JavaScript files in `scripts/` folder
- Claude Code can execute them
- Extend Claude's capabilities
- Handle tasks LLM can't do directly

## Example Scripts

- `recalc.py` – Recalculate Excel formulas via LibreOffice
- `validate.py` – Check file structure
- `unpack.py` – Extract XML from Office files
- `html2pptx.js` – Convert HTML to PowerPoint

## Example: Rice Database Skill

### Stand-Alone App

- Custom web application
- Hard-coded SQL generation prompt
- Fixed agent logic in Python
- Single-purpose: query database
- Requires developer to update

### Claude Code + Skill

- SKILL.md with database schema
- Connection code examples
- Table descriptions
- **Plus:** Can also make charts, Excel files, Word reports
- User can extend easily

**Same database access, but infinitely more flexible**

## Skill vs Stand-Alone Agent

Feature	Stand-Alone	Skill
Agent logic	Custom code	Claude Code
System prompt	Hard-coded	SKILL.md file
LLM	Your choice	Claude Opus/Sonnet
Tools	Custom built	Scripts + built-in
Maintenance	Developer	Edit markdown
Combine tasks	No	Yes

Skills let you create **specialized agents** without writing agent logic. Claude Code handles the control flow; you just provide the domain knowledge.

## Where Skills Live

- **Project skills:** `.claude/skills/` in your project folder
- **User skills:** `~/.claude/skills/` (shared across projects)
- Claude Code automatically loads skills from both locations
- Skills can reference each other (e.g., `xlsx` skill uses `ooxml` validation)
- Skills also work in **Claude.ai**, **Chat**, and **Cowork** (details later)

**Community Skills Repository:** [github.com/VoltAgent/awesome-agent-skills](https://github.com/VoltAgent/awesome-agent-skills)  
200+ skills from official teams and the community



# Creating Your Own Skill

1. Create folder: `.claude/skills/my-skill/`
2. Create `SKILL.md` with:
  - YAML frontmatter (name, description)
  - Domain knowledge and instructions
  - Code examples and workflows
3. Optionally add `scripts/` folder with helper code
4. Start using Claude Code – skill is automatically loaded

# Example: Investment Memo Skill

## SKILL.md

```
---
name: investment-memo
description: "Create investment
  memos with DCF, comps,
  and risk analysis"
---

# Output Format
- Executive summary (1 paragraph)
- Business overview
- Financial analysis (DCF + comps)
- Risk factors (bull/bear case)
- Recommendation with price target

# Workflows
1. Gather financials from SEC filings
  or uploaded files
2. Build DCF in Excel with formulas
3. Pull peer group multiples
4. Write memo in Word format
```

## What This Gives You

- Type: "Write an investment memo for Apple"
- Claude follows the structure automatically
- Consistent format across every memo
- Produces Excel model + Word report

## Other Finance Skill Ideas

- **Earnings call:** Summarize transcripts, extract guidance, flag surprises
- **Portfolio report:** Monthly performance, attribution, charts
- **Loan analysis:** Parse credit agreements,

## Skills Across Claude Formats

Skills are not limited to Claude Code. The same skill can be deployed and used across Claude.ai, Claude Desktop, and Claude Code.

Format	How to Install	How to Invoke
Claude.ai (web)	Upload ZIP in Settings → Capabilities	Automatic or /name
Chat (Desktop)	Same as Claude.ai (shared account)	Automatic or /name
Cowork (Desktop)	Install as a plugin via sidebar	Automatic or /name
Code tab (Desktop)	Place in <code>.claude/skills/</code>	Automatic or /name
Claude Code CLI	Place in <code>.claude/skills/</code>	Automatic or /name
VS Code extension	Place in <code>.claude/skills/</code>	Automatic or /name

- **Automatic:** Claude loads the skill when it detects a relevant task
- `/name`: You invoke it explicitly (e.g., `/investment-memo`)
- Skills in `~/.claude/skills/` are shared across all your projects

# Deploying Skills: Three Methods

## Claude.ai / Chat

1. ZIP your skill folder
2. Go to **Settings** → **Capabilities**
3. Click **Upload skill**
4. Toggle it **ON**
5. Requires code execution enabled

Pro, Max, Team, Enterprise

## Cowork

1. Click **Plugins** in sidebar
2. Browse official plugins or click **Upload plugin**
3. Plugins bundle skills + connectors + slash commands
4. 11 official plugins available

Pro, Max, Team, Enterprise

## Code / CLI / VS Code

1. Create folder:  
`.claude/skills/name/`
2. Add `SKILL.md`
3. Optionally add `scripts/`
4. Done—auto-detected

Pro, Max (via Claude Code)

# Using Skills: Examples Across Formats

## Claude.ai or Chat

- “Write an investment memo for Tesla”
- Claude sees the skill is relevant and loads it automatically
- Uses code execution to build Excel model
- Produces downloadable Word doc

## Cowork

- Drag a folder of 10-K filings into Cowork
- “Analyze these filings and produce an investment memo for each company”
- Claude works through them autonomously
- Outputs organized files to your folder

## Claude Code / VS Code

- Type `/investment-memo` to invoke explicitly
- Or just describe the task—Claude loads the skill automatically

## Key Difference

- **Claude.ai / Chat:** best for one-off tasks; download results manually
- **Cowork:** best for batch tasks on local files

## Skills vs. Subagents

Skills and subagents both customize Claude, but they serve different purposes. A **skill** is a reference manual Claude follows in your conversation. A **subagent** is a separate worker you dispatch to handle a task independently.

	Skill	Subagent
What it is	Instructions + scripts	A separate agent
Analogy	Giving Claude a playbook	Handing a task to a colleague
Runs where	Same conversation	Its own context window
Works in	Claude.ai, Chat, Cowork, Code	Claude Code only
Defined in	SKILL.md	AGENT.md
Created with	Manual or /agents	/agents
Best for	Standards, templates, workflows	Delegating well-defined tasks

- **Use a skill** when you want Claude to follow specific standards *while you work with it interactively* (e.g., “always format memos this way”)
- **Use a subagent** when you want to *hand off* a self-contained task (e.g., “go

# Summary

## Claude Code

- General-purpose agent
- LLM + Agent Logic + Tools
- Works out of the box

## Skills

- Specialize Claude Code
- System prompt + scripts
- Easy to create and modify

### **Skills replace stand-alone agents**

Same capabilities, less code, more flexibility