OBSIDIAN NOTE SKILL · GENERIC OUTPUT SAMPLE

Build Your AI Operating System (Part 2)

Instructor: Vykintas
Description: An introduction to Claude Cowork built around three pillars, local files, long running autonomous tasks, and scheduled tasks, each with a plain trigger for when to reach for Cowork instead of Claude Chat.
Tags: aiMastery, claudeCowork, localFiles, agenticAI, subAgents, scheduledTasks, dispatch, threePillars, sandbox
Aliases: Build Your AI OS Part 2, Claude Cowork Three Pillars, When to Use Cowork Instead of Chat

---

Instructor: Vykintas

Lesson Link:

Duration: approximately 1h36m

🧠 EXECUTIVE SUMMARY

Vykintas introduces Claude Cowork, the desktop application that runs the same Claude model as Claude.ai but operates inside your computer rather than the cloud. He deliberately refuses the elaborate setup that most tutorials lead with, and spends the session on one question instead: how do you recognize the moment a task belongs in Cowork rather than Chat. The answer is three pillars, each with a single plain trigger. Live demonstrations carry almost all of the teaching, including one that fails on camera and one that takes over his screen. It suits anyone who works with files on their own machine, or who repeats a task often enough to want it running without them. #claudecowork #threepillars #agenticai #scheduledtasks #timestamped

---

🌟 KEY TAKEAWAYS

- 🧠 "Claude AI lives in the cloud. Claude Cowork lives in your computer." That single distinction drives every decision about which one to open [00:06].
- 🗂️ Three triggers, nothing more to memorize: files on your machine, a long multi step workflow, or a task you want run without you [00:09].
- 🏢 Claude.ai is a solo consultant. Cowork is a consulting firm, spinning up sub agents that work in parallel, which is what lets it finish work Chat cannot [00:57].
- 🔒 Folder selection is the guardrail. Tools install into a sandbox rather than your operating system, and Cowork still stops to ask before deleting a file even when told to act without asking [00:26], [00:38].
- ⚙️ "I hyped you up a little bit, but AI still is not ideal." One shot a large task only when speed beats quality, or when you repeat it enough to justify building the system underneath it [01:11], [01:13].

- 📆 Build a scheduled task inside a project, because project memory turns your feedback into permanent improvement without editing the instructions by hand [01:24].
- 🎵 The closing song carries the honest summary: "Usually use Chat, sometimes Cowork" [01:34].

🔄 MINDSET SHIFTS

These are the conceptual spine of the session. Each one reframes how to think about handing work to AI.

Shift: From Claude behind glass to Claude operating your machine

Old thinking: Claude sits in a browser tab. You carry files to it and carry the results back by hand.

New thinking: Cowork runs inside the computer. It reads folders, renames and moves files, creates new ones, installs what it needs, and edits in place, visibly, while you watch.

Why it matters: The shuttling of files between desktop and browser is the single largest friction for anyone whose work lives locally, and this removes it rather than easing it.

Shift: Simplicity ahead of setup

Old thinking: A powerful tool demands an elaborate configuration first, the kind demonstrated at length by experts online.

New thinking: Start plainly, get value, and invest in structure only once a use case has earned it.

Why it matters: Vykintas is unusually blunt about this. He calls the elaborate setup route "a dangerous zone" where the volume of tools and exceptions creates confusion, and where the time goes into building a system rather than getting anything from it [00:07].

Shift: From one worker to a firm

Old thinking: Claude proceeds linearly, one step after another, and halts when it needs you.

New thinking: Cowork acts as a supervisor that creates specialized sub agents and delegates to them in parallel.

Why it matters: This is the mechanism, not a metaphor. It is why a genuinely long autonomous workflow completes here and stalls in Chat.

Shift: Agentic AI is powerful and is not a panacea

Old thinking: Hand over everything at once and the result comes back as imagined.

New thinking: Split a long workflow into reviewed stages for quality. Reserve the single enormous prompt for the cases where speed outranks quality, or where repetition justifies building skills and memory underneath it.

Why it matters: The instructor breaks his own demonstration to say he oversold it. That candour is the teaching, because it tells you which of the two approaches your situation actually calls for.

> **FOR YOU**
>
> The simplicity shift is the one most likely to be resisted by anyone who enjoys building systems. As a general example, a consultant who has spent a weekend designing folder structures before running a single task has already spent the time the tool was supposed to save. Let the plain version run until a workflow repeats often enough to earn its structure.

#mindset #threepillars

Pillar 1: Claude works inside your computer

The keyword is local. Claude.ai cannot touch your machine. It works only with files in the cloud, so anything on your disk has to be carried in by hand, and carried back out again [00:17].

> **VYKINTAS**
>
> Claude AI lives in the cloud. Claude Cowork lives in your computer.

Cowork runs directly inside the machine. It reads folders, moves files, installs programs when it needs them, implements changes, and processes anything on the disk [00:18].

The trigger: "I have files and folders on my computer that I want Claude to work on" [00:18]. When a task matches that sentence, the answer is Cowork.

Vykintas notes he is personally a cloud person, with everything in the cloud, and so uses Cowork less than daily. His point is that the value here is uneven and honest about it: if your work lives locally, this pillar is transformative, and if it does not, it is occasional [00:11].

The three local demonstrations

File reorganization [00:12]. A single prompt: rename files to begin with the lesson number, then a keyword for the file type, then the lesson name and cohort year, and sort them into folders. Cowork renamed everything, created Chats, Transcripts and Workbooks folders that had not existed, and moved each file into place, all visible live on the desktop [00:14], [00:15]. This is the demonstration nearly every teacher opens with, and Vykintas says so plainly. Its purpose is orientation, not ambition.

Batch image editing [00:19]. A folder of speaker photographs at inconsistent sizes, one vertical among horizontals, one zoomed too far out. The prompt named Vishen's photo as the size standard, asked that every other match it exactly, and that each speaker be centered. Settings: act without asking, since

nothing here is destructive, and model Sonnet 4.6 rather than the most powerful option, to conserve credits [00:21].

What makes this demonstration matter is what was never configured. No image editing capability was installed in advance. Cowork evaluated the photographs, decided which tools the task required, installed them, wrote code, and ran it in a sandbox [00:24].

The result was imperfect. Four photographs came back correct. Vykintas's own was cropped badly [00:28]. Asked what to do about it, the room offered two answers and both were right: give feedback for a one off, or build a skill if the task will recur [00:28].

Audio format conversion [00:30]. A WhatsApp voice file downloaded as .opus, which would not open on the desktop and which NotebookLM rejected outright with an upload error. Cowork converted it to a standard playable format [00:34].

An accident during this demonstration taught something the prepared material did not. Vykintas selected the wrong folder, noticed the interface now read one folder "plus one", and used it: **a task can be scoped to more than one folder at a time** [00:33].

> **FOLDER SELECTION IS THE SECURITY MODEL**
>
> Asked why Cowork insists on choosing a folder, a student answered "security" immediately, and that is exactly right. Because Cowork can act on the machine the way you can, it could damage things. Scoping it to a folder is the boundary. Tools it installs go into a sandbox, a virtual environment on your computer rather than into the operating system itself [00:26]. Vykintas is careful not to overclaim: "It doesn't mean it's 100% safe. Definitely not 100%." The claim is that risk is minimized with a trusted service [00:28].

He contrasts this with platforms in the same space, naming the wave of excitement a few months earlier around more autonomous alternatives. Those carry more significant security issues. Cowork is less autonomous by design, and the guardrails are the reason [00:27].

It creates local files, not only edits them

Returning briefly to the long task running in the background, Vykintas points at two of its deliverables to make a distinction easy to miss: the Excel financial model and the PowerPoint deck did not exist before [00:39]. Cowork writes new files to your disk in whatever format the task calls for, spreadsheets, documents, presentations, data files [00:40].

> **USE CASE**
>
> As a general example, a coach with three years of client session notes sitting in one unsorted folder could have them renamed by a convention and filed by theme in a single pass, then ask for a spreadsheet summarizing the themes across all of them. Both halves, the reorganizing and the new file, are the same pillar.

#local #pillar1 #localfiles #security #sandbox

Mobile Dispatch, and why not to use it at your desk

Dispatch answers Cowork's structural limitation. Cowork lives on one machine and is not on your phone. In the Claude mobile app's side menu there is a function called Dispatch that sends a task to Cowork running on your desktop [00:43].

The demonstration began with a limit worth knowing independently. Vykintas tried to drop a folder of PDFs into Claude.ai and hit a wall: **this chat has reached the 100 image limit, including PDF pages** [00:42]. Working the same files locally in Cowork sidesteps it entirely.

From his phone he sent a request: go to a named folder, read all the PDFs, and report back the most important tools and frameworks covered, briefly, to prepare for a client call while away from the laptop [00:44]. The task appeared in the Dispatch section of desktop Cowork and began running on its own [00:45]. Results can come back as text or as any file Cowork creates, so a deck built on the desktop can arrive on the phone [00:46].

Then it went wrong, usefully. Cowork chose to use computer use, visually navigating the screen and clicking through folders rather than working in the back end [00:46]. It took over the machine Vykintas was teaching from. He could not move windows, and had to message Dispatch to stop the task before regaining control [00:54], [00:55].

> **TWO LESSONS FROM A DEMONSTRATION THAT MISFIRED**
>
> Do not use Dispatch while sitting at the computer it controls. There is no reason to, and it will interrupt you. Second, add an instruction such as "don't use computer use unless it's really necessary" to the prompt, because computer use is markedly less efficient than letting Cowork navigate in the back end [00:56].

On whether a separate machine is wise: Vykintas recommends against it for Cowork specifically. Your main computer is where your files already are, where your applications are installed and logged in, and where your connectors and skills already exist. Moving to a dedicated machine means moving all of that first [00:48]. The requirement is only that it stays on, logged in, and not sleeping [00:49].

Cowork also reaches cloud storage through the same connectors as Claude.ai, so Dropbox and Google Drive are available alongside local files. Vykintas calls this combination the real strength: both at once [00:50].

#dispatch #pillar1 #mobile #computeruse #connectors

Pillar 2: Hand it off and walk away

The keyword is long running tasks, or agentic autonomous work. Claude.ai cannot run a long workflow to completion. It works linearly and stops when it needs you. Cowork commits to finishing and runs agentic execution through to the end [00:56].

The trigger: "I have a long multi step workflow for Claude to execute" [01:01].

> VYKINTAS
>
> See Claude AI as a solo consultant. See Claude Cowork as a consulting firm.

The mechanism underneath the analogy

Vykintas stops screen sharing and builds it in Lego, which is worth noting because the physical version makes the structure obvious [00:58].

Claude.ai is one figure working in sequence until it stops. Cowork is a supervisor. It creates sub agents and assigns each a piece: one responsible for the presentation, another for research, more as the task demands, all working at once and reporting back [00:59].

Cowork creates these sub agents on its own judgment. You can also create your own, defining instructions, tools and connectors, in the same way you create skills [01:00]. He declines to go deeper, explicitly to avoid overcomplicating the session.

The business launch package

Triggered in the first minutes of class and left running throughout [00:02], this was one prompt asking for a complete launch package for an AI coaching program: market research report, ideal client profile, product definition, business plan, Excel financial model, HTML landing page, and PowerPoint pitch deck [00:02].

The structure around it matters as much as the prompt. An Inputs folder held a branding document, an offer and packages document, a CSV of feedback, and a photograph. Outputs was empty at the start [01:02].

What came back, in roughly half an hour [01:08]:

- A pitch deck that pulled the real offers from the input document and a photograph from the same folder [01:03].
- An Excel financial model with conservative, base and optimistic scenarios, profit and loss for each, monthly cash flow calculated automatically, and an executive summary [01:07].
- A business plan covering products and services, phases, marketing and sales channels, an operations plan, and a financial projection summary [01:08].
- An HTML landing page.

Vykintas reviews it honestly on camera. The market opportunity slide is inflated, "a bit too big" [01:03]. A satisfaction figure reads ten out of ten because the feedback CSV he supplied contained only the submissions where students had praised him, which he notes and says he would correct before using it [01:04]. This is a first draft that needs review, and he treats it as one.

One guardrail appeared mid run. Despite act without asking being set, Cowork stopped and requested permission to permanently delete files in the folder [00:37]. Destructive actions are held back regardless of the permission setting.

> **EDITABLE BEATS BEAUTIFUL**
>
> NotebookLM produces visually nicer slides, but each slide is an image. Double clicking the text does nothing; improvement has to go through prompting [01:05]. Cowork's slides are plainer and are real files, editable element by element [01:06]. When you need final control over what you deliver, the plainer editable file wins.

For video editing specifically, Vykintas redirects to Claude Code, since reliable video work needs code defined workflows and libraries [01:09].

The honest rule about one shot delegation

Having demonstrated it, Vykintas immediately asks whether anyone should actually work this way, and answers no for most cases [01:10].

> **VYKINTAS**
>
> I hyped you up a little bit, but AI still is not ideal.

Handing over many deliverables at once removes exactly the input and feedback that produce quality. The better sequence is to split: research first, reviewed; then the financial model, reviewed; then the business plan, with feedback; and only then the presentation [01:11].

Two exceptions justify the single enormous prompt [01:13]:

1. **Speed matters more than quality.** You accept roughly ninety percent, often because you are testing an idea rather than shipping it.
2. **The task repeats often enough to justify a system underneath it.** Skills, a brain, feedback and memory. With that investment, Vykintas says the same one shot delegation does become reliable and high quality. The setup is the price [01:15].

> **USE CASE**
>
> As a general example, a founder testing five business concepts in a week wants the fast version, because the point is comparison rather than polish. The same founder, having chosen one, should split the real build into reviewed stages. The decision is not about the tool. It is about whether this run is disposable.

#pillar2 #agenticai #subagents #longrunning #quality

Pillar 3: Claude on autopilot

The keyword is scheduled. On Claude.ai you are always the trigger. You have to arrive and send a message before anything happens [01:17].

The trigger: "I want Claude to do this task automatically without me triggering it every time" [01:18].

Build it inside a project, not on its own

Vykintas begins creating a scheduled task directly, then cancels [01:20]. The direct route works, but for anything you intend to keep, there is a better one.

The reason is memory. Project memory on Cowork is a local text file in the project's folder. When a scheduled task runs inside a project, feedback given after a run is saved there, and the task improves over time without your editing its instructions by hand [01:24].

> **COWORK PROJECTS AND CLAUDE.AI PROJECTS ARE UNRELATED**
>
> They share a name and share elements such as memory, files and instructions, but they are separate features with separate lists [01:21]. Each Cowork project is backed by a real folder on your computer, which is why the interface calls a folder a project when you select one [01:20]. To move one across, use New Project then Import a Project, which recreates the setup as a local folder [01:23].

The build sequence

Work through this in order:

- [ ] Create the project: Projects, New Project, Start from scratch. Name it. Leave instructions and files empty. Memory is on by default [01:21].
- [ ] Inside the project, click the plus beside Scheduled, then New Task, then Set up manually. The alternative is Create with Claude, which works the same way skill creation does [01:24].
- [ ] Name the task, add a short description, and paste the instructions.
- [ ] Confirm the folder. It is preselected when the task is created inside a project [01:25].

- [ ] Set permissions to act without asking. A scheduled task that stops to ask defeats its own purpose [01:25].
- [ ] Choose the model. Sonnet 4.6 is sufficient for routine work and conserves credits [01:26].
- [ ] Set frequency to Manual for now. The options are manual, hourly, daily, weekdays and weekly, and you can change it once it works [01:26].
- [ ] Save, then Run now, because anything built should be tested before it is trusted [01:27].

The email triage instructions asked Cowork to check Gmail for the last twenty four hours and return a briefing in priority tiers: critical, legal and financial, action required, awaiting reply, follow up, FYI worth knowing, and skipped [01:28]. The run appeared under History, and the briefing came back grouped correctly, including a skipped section explaining what it judged irrelevant [01:29].

No reconnection to Gmail was needed. Connectors and skills are shared between Claude.ai and Cowork, so anything established once is available in both [01:28].

> **USE CASE**
>
> As a general example, an operations lead who reviews the same three dashboards every Monday morning is describing a scheduled task, not a habit. Built inside a project, the first month of corrections to what counts as urgent stays with it permanently.

Vykintas flags that this version is deliberately simple. A summary of your inbox is useful; a task that labels emails and drafts replies is more useful, and he confirms both are possible before deferring the advanced build to the implementation lab [01:30].

#pillar3 #scheduledtasks #automation #projectmemory

Limitations worth knowing

Gathered here because several surfaced in passing rather than in one section.

Limitation	Detail
The machine must be awake	Cowork runs only on the computer it is installed on. That computer must be on, logged in, connected, and not sleeping [00:36], [00:49].
Not portable	It cannot easily be used from a phone or another computer. Dispatch is a partial workaround, not a solution [00:37].
The 100 page ceiling	Claude.ai caps a chat at roughly 100 images or PDF pages. Working the files locally avoids it [00:42].
Computer use is inefficient	When Cowork navigates visually rather than in the back end it is slower and demands repeated permissions [00:56].

Limitation	Detail
Video editing	Possible, but Claude Code is the right tool where workflows need to be defined as code [01:09].

Two limits raised by students were answered as not being Cowork limits at all: model intelligence belongs to the underlying models and improves as they do, and cloud file access is available through connectors [00:35].

Deferred to the implementation lab: connecting Obsidian to Cowork for co editing long text files, together with building a knowledge graph [00:51]; the advanced email triage that labels and drafts [01:30]; and the dashboard to sheet demonstration [01:31].

#limitations #connectors

The closing frame

The session ends on a deliberate correction to its own enthusiasm. In the closing song, written for the class, one line reads "Usually use Chat, sometimes Cowork" [01:34]. Vykintas points at it directly: for the majority of ordinary daily tasks you will still reach for Claude.ai, and Cowork is for the moments the three triggers describe [01:34].

That is the whole lesson compressed. Not a replacement, a second instrument, with three clear signals telling you when to pick it up.

Resources mentioned

- The prompt document shared at the start of class, containing every demonstration prompt used in the session [00:01].
- obsidian.md, for the deferred co editing and knowledge graph demonstration [00:51].
- NotebookLM, used both as the destination for the converted audio file and as the comparison for slide generation [00:31], [01:05].

TAGS

#aimastery #claudecowork #threepillars #localfiles #agenticai #subagents #scheduledtasks #dispatch #automation #projectmemory #security #sandbox #limitations #connectors #computeruse #quality #mindset #timestamped

---

Claude Cowork - Claude.ai - Claude Code - Vykintas - Three Pillars - Local Files Pillar - Long Running Tasks - Scheduled Tasks - Agentic AI - Sub-Agents - Consulting Firm Analogy - Custom Sub-Agents -

Dispatch - Computer Use - Sandbox - Folder Guardrail - File Reorganization - Batch Image Edit - Opus Audio Conversion - Multiple Folder Scope - Business Launch Package - Excel Financial Model - HTML Landing Page - Daily Email Triage - Priority Tiers - Project Memory - Cowork Projects - Import Project - NotebookLM - Editable Slides - Skills First - One Shot Versus Split - 100 Page Limit - Connectors

---