

Loop to Cloud Routine Conversion

Move the same scheduled work onto Anthropic's servers so it runs with your laptop shut and your team can actually use the output.

01 When to use this

Use this when a local loop has proven itself and the limits start to bite: you close your laptop and nothing runs, or a teammate needs the output and cannot get it. It is rung four, the last one. Do not start here.

02 The prompt

Turn the `<LOOP_OR_SKILL_NAME>` loop into a cloud routine so it runs without my machine on.

Mirror the structure of `<EXISTING_ROUTINE_NAME>`. Query that routine first so this one inherits a shape I already know works.

Wire these three things:

1. Connectors. Attach `<CONNECTOR_LIST>`, e.g. Google Drive to the routine itself. My local connectors do not travel, so do not assume any of them exist.
2. Credentials. Re-add `<KEY_NAMES>`, e.g. OPENAI_API_KEY under the routine's cloud default environment. Local environment variables do not travel either.
3. Output. Write to a dated subfolder inside the shared Drive folder `<DRIVE_FOLDER>` instead of a local path, so my team can open it.

Code lives in the private GitHub repo `<REPO_URL>`. Pull from there rather than from anything on my disk.

Trigger: `<schedule | API call | webhook>`. If schedule, use `<TIME_AND_DAYS>`.

Tell me anything you cannot do yourself so I can do it by hand.

03 What it is actually doing

A routine is the same job running on Anthropic's infrastructure instead of your computer. Your laptop can be closed, dead, or on a plane. The work still happens.

Three things have to be wired, and all three fail in the same way: something that exists on your machine simply is not there in the cloud.

Connectors are the first. The connectors you set up locally belong to your machine, not to the routine. You attach them to the routine separately.

Credentials are the second. An API key living in your local environment does not travel. You re-add it under the routine's cloud default environment or the routine runs blind.

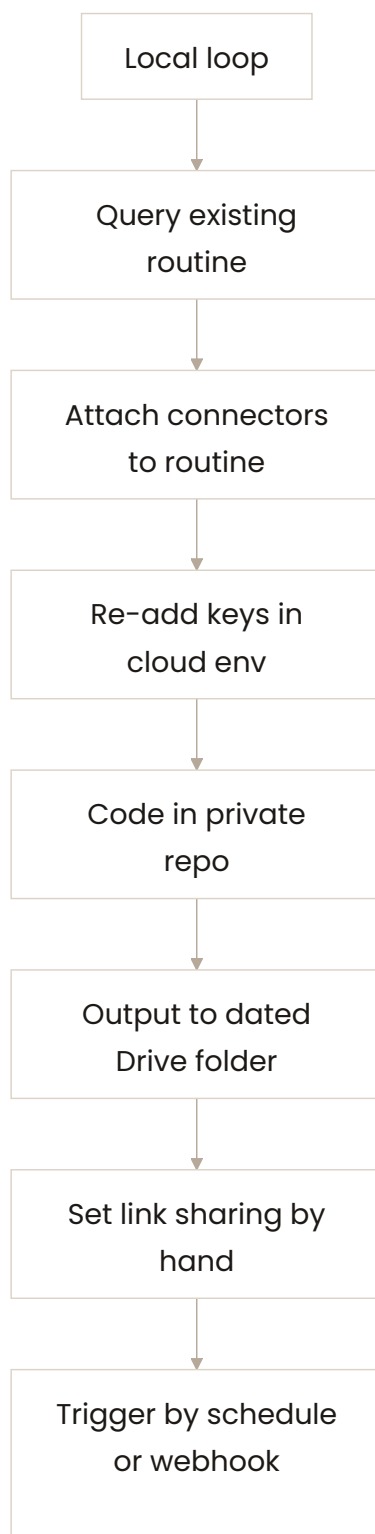
Output is the third and the one people forget. A folder on your desktop is useless to a team. Point it at a dated subfolder in a shared Google Drive instead. If code needs to travel too, put it in a repo. Nick used a private GitHub repo.

One gotcha he hit live: the Drive connector cannot change sharing settings. It can write files, but it cannot make them viewable. You set anyone with the link can view by hand, once.

Routines can be triggered three ways: on a schedule, by an API call, or by a webhook. A webhook is strictly better than a schedule when there is a real event to hang off.

And once you have done this once, later conversions collapse into a single sentence. Tell Claude to mirror the structure of an existing routine and it inherits a shape that already works.

04 How it flows



05 Step by step

01 Confirm the loop earned it

Only promote a loop that already runs clean locally. A routine multiplies whatever the loop does, including its bugs, and it does it where you are not watching.

02 Point at a routine you already trust

Tell Claude to query an existing routine and mirror its structure. Nick's later conversions compressed into one prompt because of this.

03 Attach connectors to the routine

Local connectors do not travel. Attach Drive, Gmail, or whatever else the job touches to the routine itself.

04 Re-add credentials in the cloud environment

Open the routine's default environment and put the API keys back. Your local environment panel is a different place entirely.

05 Move the output somewhere shared

Swap the local folder for a dated subfolder in a shared Drive. Dated means today's run never overwrites yesterday's.

06 Put the code in a repo

If the routine runs scripts, they need to exist somewhere the cloud can reach. A private GitHub repo is enough.

07 Fix sharing by hand

The Drive connector cannot change sharing settings. Set anyone with the link can view yourself, one time, on the parent folder.

08

Pick the right trigger

Schedule, API call, or webhook. If a real event exists to hang off, use the webhook instead of polling on a timer.

06 **Words explained****Routine**

A scheduled job that runs on Anthropic's servers rather than your computer. It does not care whether your machine is on.

Connector

A saved link between Claude and an outside service like Google Drive or Gmail. It carries the permission to read and write there.

Cloud default environment

The routine's own settings drawer for secrets and variables. Separate from your local one, and it starts empty.

Webhook

An address another service calls the instant something happens. Better than checking on a timer, because it fires on the real event.

Dated subfolder

A folder named for the day it was made. Every run drops into its own folder instead of overwriting the last one.

Repo

A hosted folder of code, usually on GitHub. It is how code gets from your machine to somewhere the cloud can read it.

07 Watch out for

-
- Local connectors do not travel. The routine sees none of them until you attach them to the routine itself.
-
- Local environment variables do not travel either. An API key left only in your local panel means the routine runs without it.
-
- The Google Drive connector cannot change sharing settings. Set anyone with the link can view by hand or your team sees nothing.
-
- Writing to a local path from a cloud routine silently produces output nobody can reach.
-
- Skipping the repo step leaves the routine pointing at scripts that only exist on your machine.
-

08 How it connects

Skill to Loop Conversion

Every limit that made the loop hurt, dead laptop, brittle local setup, no team access, is exactly what this rung removes.

Automation Ladder – Prompt Skill Loop Routine

This is rung four, the top of the ladder, and the ladder rule is that you climb it in order rather than jumping here first.

Ad Creative Generation Pipeline

Nick took the creative pipeline all the way to this rung: private repo for the code, dated Drive subfolder for the ads.

Batch Generation and Contact Sheet

The contact sheet is what your team opens in the shared Drive folder, which is why the output destination has to move off your disk.

Credit. Framework and approach by Nick Saraev, from the loop to routine rung of the automation ladder in his Zero to One course on Claude Code for marketing. Written up for the Kape vault by BrewedOps.