Tekk.coach cheat sheet

What to do, step by step, to get the most out of Tekk from your coding agent.

Loops find → Proposal → You decide → Spec → Agent builds → Merge closes it ↻

1 SET UP ONCE — 2 min

Claude Code: one plugin = MCP server + 11 skills.

```
> /plugin marketplace add guccikudo92615/tekk-skills
> /plugin install tekk@tekk
```

Codex, Cursor & others

```
$ npx skills add guccikudo92615/tekk-skills -a codex
```

plus the MCP server `https://app.tekk.coach/api/mcp`. Check with `/mcp`. Nothing installed? Every skill is also `/mcp__tekk__<name>`.

2 CONNECT YOUR TOOLS — Settings → Integrations

CODE · WAKES LOOPS ON EVERY PUSH

GitHub · GitLab · Bitbucket

PRODUCTION · CONNECT FIRST

Sentry · Datadog · PostHog · Stripe · Supabase · PagerDuty · Mixpanel · incident.io

CUSTOMER FEEDBACK · ONE SWITCHES THE LOOP ON

Intercom · Zendesk · Gmail · X · Jam · Canny · Freshdesk · Plain

CONTEXT & STATUS

Notion · Confluence · Slack

Read-only by default. Production signals wake a loop; the loop reads the code behind them before it proposes anything.

★ FIVE HABITS

- **Decline with a reason.** It's how the loops learn your noise.
- **Revise, don't reject,** a right finding with the wrong fix.
- **Write `Closes TEK-n`** in every PR body, and tick the boxes.
- **Grill before you spec.** Open decisions never become checkboxes.
- **Retro every round.** Silence teaches the loops nothing.

3 THE PLAYBOOK

what you do, in order. Green: build the work · Blue: aim the loops

Day 1
30 minutes

1. `setup` — Install the plugin, connect your repo, then Sentry and one feedback tool.
2. `/ideate-loops` — See which loops fit what you connected. Switch those on.
3. `/grill-to-steer` — Tell Tekk what you're building toward and what you'd hate to see break. The first runs aim there.
4. `greenlight` — Every loop's first run asks first, with its cost on the card. Say yes to the ones you want tonight.

Day 2
your first round

1. `/sweep` — Decide the first proposals in one pass. Reasons matter most now: **this is the round that teaches the loops what you care about.**
2. `/blitz` — Build what you accepted: one coding session per spec, in parallel.
3. `merge` — Review and merge the PRs. Each says `Closes TEK-n`, so the spec closes itself.
4. `/retro` — Say what was worth it, too small or wrong. The next round looks somewhere better.

Every day
15 minutes

1. `/triage` — Make the board true before planning from it. *Skip it and you rebuild what shipped.*
2. `/sweep` — Clear what the loops found overnight. Say **"go"**.
3. `/blitz` — Fan the ready specs out. Merge what came back.
4. `/grill-to-spec` — Whenever an idea of your own shows up. It becomes a spec ready to build, and joins the next blitz.

Every week
20 minutes

1. `/retro` — Which loops earned their keep? Too noisy, too quiet, wrong place?
2. `/grill-to-steer` — Re-aim at this week: a launch, a migration, a customer complaint.
3. `/steer-loops` — Check spend and cadence. Turn down what's noisy, up what's landing.

Every month
prune

1. `/retro` — Look across the month: keep the loops whose proposals you accept, quiet the ones you keep declining.
2. `/triage` — Archive specs nobody will build. Completed means shipped; archived means not now.

Once it runs
scale it up

1. `/night-shift` — Set it once: a time, a cap, the lineup. Wake up to PRs and a morning report.
2. `/grill-to-loop` — Something keeps biting you that no loop catches? Build your own (page 2).

The skills, and how to get the most out of each

GREEN: PLAN AND BUILD · BLUE: AIM AND BUILD LOOPS

4 · PLAN THE WORK

`/grill-to-spec` — idea → ready spec

WHEN	You have an idea, not yet a plan.
DO	Answer each round in one line: *"yes, yes, no, it's the admin view"*.
PRO	Can't decide something yet? It becomes its own spec, never a checkbox.

`/write-spec` — decisions already made

WHEN	The plan exists: a chat, a doc, a bug you understand.
DO	Point it at the source. It checks claims against your code first.
PRO	Big feature? Ask for a parent with sub-tasks, not sibling specs.

`/triage` — board vs. git

WHEN	Before planning, after merges, when you no longer trust the board.
DO	Let it fix plain status errors. Closures wait for your yes.
PRO	Run it before `/blitz`, or you'll build what already shipped.

BUILD IT

`/sweep` — the proposal inbox

WHEN	Proposals pile up. Don't open them one by one.
DO	Read the verdicts, adjust, then say **"go"**: one call decides all.
PRO	**"Moved" isn't "fixed".** Code that moved gets a revise.

`/blitz` — parallel sessions

WHEN	Accepted specs sit in To do.
DO	Say nothing for the whole board, or name one epic or wave.
PRO	Specs that declare `touches` make bigger safe batches.

`/night-shift` — while you sleep

WHEN	Once to set up. Again only to change the lineup.
DO	Pick a time, a cap (default 4, max 10) and what goes first.
PRO	~$1–2 of agent usage per small spec. **Never merges, never decides.**

AIM THE LOOPS

`/retro` — was it worth it?

WHEN	After a round merged, and weekly.
DO	Answer: worth it, too small, wrong, never looked at.
PRO	"Too small" and "wrong" teach different things.

`/grill-to-steer` — point them

WHEN	Weekly, and before any launch.
DO	Check Tekk's read of where you're heading first. Wrong? Correct it.
PRO	A steer runs once. A wrong direction is fixed in the spec it read.

`/steer-loops` — run, cadence, cost

WHEN	A loop hasn't run, runs too often, or costs too much.
DO	Fire one now, schedule a group tonight, work the pitch queue.
PRO	**Autonomy ≠ cadence.** Autonomy decides a pitch; cadence makes it run.

5 IF THIS HAPPENS, DO THAT

The inbox keeps growing	**`/sweep`** daily, then **`/retro`** on the noisy loop, or lower its cadence in **`/steer-loops`**.
The same kind of finding keeps coming back	Decline it with the reason *why* it's noise, not just "no".
A spec merged but stays open	The PR needs **`Closes TEK-n`** in its body and every box ticked. **`/triage`** finds them all.
Loops look at the wrong things	**`/grill-to-steer`**, and fix the spec or goal that sent them there.
A loop never seems to run	**`/steer-loops`**: autonomy doesn't schedule anything, cadence does. A full inbox also holds it back.
It costs more than you expected	**`/steer-loops`** shows spend per loop. Lower cadence on what you rarely accept.
A night-shift spec is too big for one session	Split it with **`/grill-to-spec`** into a parent with sub-tasks.

6 CLOSE THE LOOP IN THE PR

```
Closes TEK-123   finishes the spec on merge
Part of TEK-123  contributes, keeps it open
```

- **Tick the build-order boxes in the same PR.** One open box keeps it open.
- **Only the line in the body closes.** A branch name never does.
- **Red `tekk/spec-checklist`?** Tick, trim, or switch to `Part of`.

7 BUILD YOUR OWN LOOP

`/ideate-loops` → `/grill-to-loop` → greenlight run 1

`/ideate-loops` writes nothing: it ranks 3–4 ideas from what's connected. Then bring `/grill-to-loop` answers to:

1. **What exactly keeps happening?** The event, not the area: "webhooks stop after a deploy", not "billing".
2. **How do you find out today?** That's the evidence it reads.
3. **Where does it live in the code?**
4. **What should a run hand you?** A proposal, plus a Slack post or alert on accept.
5. **What would annoy you to be told?** That becomes its "not a finding" rule.

- **Steer before you build.** Most good ideas end up as a steer for a loop you already run.
- **One failure class per loop.** "Make the app better" is refused.
- **Start it on "ask", every 48h.** Run 1 always asks first, with its cost.