

# Turn the thing you just shipped into a twenty-second launch video.

`/brag` is an open-source Claude Code skill that reads your project's actual code, writes a storyboard, scores it, renders an MP4, and hands you the caption to post with it. Two commands to install, one word to run. Here's the setup, then five things worth knowing before your first run.

BY DAVE BROWN

---

The usual way to show somebody what you built is a screenshot, or a two-minute screen recording nobody watches to the end. `/brag` exists because the format that actually travels — a short, cut, scored video — costs an evening in an editor, so most people never make one.

What makes it work is that it never asks you for assets. It opens the repo, pulls the real colours from your stylesheet, the real headline from your hero, the real numbers from your data files, and builds the video out of those. You don't write a brief. You type `/brag` and read what it decided.

---

## 15–25s

The hard ceiling. A creative law in the skill, not a suggestion — long enough to land, short enough to autoplay.

## 7

Tone presets, from **polished** to **chaotic**, plus freeform direction that overrides any of them.

## 4

Files you get back: the plan, the composition, **brag.mp4**, and share copy written in the video's tone.

## 0

Screenshots, URLs or design assets you need to supply. It reads the source.

---

# Two commands in Claude Code, then restart.

`/brag` ships as a plugin with its own marketplace. Add the marketplace, install the plugin, and the skill is available in every project you open.

COPY THIS

```
/plugin marketplace add latent-spaces/brag
/plugin install brag@brag
```

Not in Claude Code? The same skill installs into Cursor, Codex, Gemini CLI, opencode and anything else that reads Agent Skills, through Vercel's skills CLI. Add `-g` to install it globally rather than into the current project.

ANY OTHER AGENT

```
npx skills add https://github.com/latent-spaces/brag --skill brag
```

| YOU NEED            | CHECK IT WITH                       | IF IT'S MISSING                                                                                                                                                   |
|---------------------|-------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Node.js 22+         | <code>node -v</code>                | Install from <a href="https://nodejs.org">nodejs.org</a> , or <code>brew install node</code> .                                                                    |
| FFmpeg on your PATH | <code>ffmpeg -version</code>        | <code>brew install ffmpeg</code> . Nothing renders without it.                                                                                                    |
| The Hyperframes CLI | <code>npx hyperframes doctor</code> | It downloads on first use. Run <code>doctor</code> once before your first brag so the failure, if there is one, happens in five seconds rather than five minutes. |

ONE THING THAT SURPRISES PEOPLE

There are two skills in the box. `/brag` hands the rendering to Hyperframes; `/brag-slim` builds the whole video itself with whatever is already on the machine, and needs no bundled assets. On Opus 5.5 `/brag` hands off to the slim version automatically and tells you it did. Type `/brag --full` if you want the classic Hyperframes path anyway.

## From the project directory, say `let's /brag`.

That's the whole invocation. Everything after it is optional steering. The run moves through four gates, and each one produces a file you can read.

---

---

### WHAT HAPPENS, IN ORDER

- 01 **Inspect.** It reads your main page, stylesheet, README, routes and data files, and answers nine questions about the product before it plans anything.
  - 02 **Plan.** It writes `brag-output/brag-plan.md` — the angle, the hook, the highlights, the punchline, and a scene-by-scene storyboard with durations. This is the file to read.
  - 03 **Compose.** The storyboard becomes a real composition, with music, sound effects on the moments that move, and a check pass that has to come back clean before anything renders.
  - 04 **Deliver.** `brag.mp4`, a poster frame picked from a settled moment and baked in as frame zero, and `share-copy.txt`.
- 

| OPTION                             | VALUES                                     | DEFAULT                   |
|------------------------------------|--------------------------------------------|---------------------------|
| <code>--tone</code>                | A preset, or any sentence you like         | Inferred from the project |
| <code>--format</code>              | <code>landscape · vertical · square</code> | landscape, 1920×1080      |
| <code>--duration</code>            | Seconds                                    | Auto, inside 15–25s       |
| <code>--voice</code>               | Flag — turns narration on                  | Off. Opt in per run.      |
| <code>--no-music · --no-sfx</code> | Flags                                      | Both on                   |
| <code>--title</code>               | String                                     | Inferred from the project |

---

## Five things that separate a postable brag from a generic one.

The defaults are good. These are the five places where thirty seconds of input changes the result more than anything else you could do.

### TIP 01

#### Give it an angle, not a style.

Presets set pacing. What makes a video feel made is a premise, and freeform direction beats a preset every time — the skill maps your sentence to the nearest preset for timing but keeps your direction all the way into the composition. One line is enough: *a fake Series A launch from 2016, or a museum exhibit, or the one I used*, which was that the site's own brand guide forbids noticeable animation, so the video should be the most restrained launch video possible and pay the joke off in a footnote.

### TIP 02

#### Point it at one thing, and prefer the app over the landing page.

If you just shipped a feature, say so in the invocation — the skill will make that the subject instead of averaging the whole project into twenty seconds. And when there's a choice between the marketing page and the product doing its job, the product wins. One of the skill's own laws is to show the real thing: entry, key action, result. A video of software working is more convincing than a video of software being described.

### TIP 03

#### Argue with the plan, not with the render.

brag-plan.md is written before a single frame exists, and it is specific enough to judge — scene by scene, with the exact on-screen text and the timings. Read it and push back there. Fixing a hook in the plan costs a sentence; fixing it after the render costs the render. A second run writes to a timestamped folder rather than overwriting the first, so you can keep both and choose.

#### WORTH ASKING FOR BY NAME

"Show me the plan before you compose." Then check one thing: does the first two seconds work as a still, with no context and no sound? That is the only part of the video most people will see.

#### TIP 04

### Pick the format for the feed, then look at the poster.

vertical for stories, Reels and TikTok; landscape for X, LinkedIn and Discord; square when you genuinely don't know. Then open `brag.jpg`. The skill picks the strongest *settled* frame — text fully arrived, nothing mid-transition — and bakes it in as frame zero so every platform's thumbnail shows it. If that frame isn't the one you would have chosen, say which moment you want instead. It's a cheap fix and it's the image your post actually leads with.

#### TIP 05

### Assume muted, then check the loudness anyway.

Every feed autoplays silent, so the video has to read with no sound — which is why the skill holds every line long enough to be read rather than flashing it. But when somebody does unmute, level matters. My first render came back at  $-31.4$  LUFS, which on a phone reads as “this video has no audio.” Platforms normalise to around  $-14$ . Ask for a loudness measurement, and if it's quiet, scale the whole mix at once — music and effects together — so the balance somebody designed survives the fix.

| FIRST RENDER                                | AFTER SCALING $\times 2.3$        | PLATFORM TARGET                                |
|---------------------------------------------|-----------------------------------|------------------------------------------------|
| $-31.4$ LUFS — inaudible on a phone speaker | $-24.2$ LUFS — quiet, but present | Roughly $-14$ LUFS is where normalisation sits |

#### WHAT A GOOD PLAN SOUNDS LIKE

### One real example, so you know what you're reading.

I ran it on [davecto.com](https://davecto.com). The site's motion guide opens with a rule — *if an animation is noticeable as an animation, it's wrong* — so the plan committed to a launch video for a site whose design philosophy forbids launch videos.

Motion is a whisper.

Four words, quoted from the brand guide, on empty paper, with a single hairline rule drawing underneath. From there: the real hero rebuilt, three résumé rows arriving one per beat, the site's own theme toggle firing on screen to flip the video into dark, three real outcome figures counting up, and the closing line the site already uses. Then, small and teal, at the very bottom: *no animations were noticed in the making of this video*.

Nothing in it was invented. Every number came out of `data/projects.ts`, every line of type was already published somewhere on the site. That's the thing to look for when you read your own plan: not whether it sounds impressive, but whether any of it could have been written about somebody else's project.

---

#### ONE MORE THING

## If your first brag comes out generic, send it to me.

Nine times out of ten the fix is in the angle rather than the render, and it takes one sentence to find.

---

#### GET IN TOUCH

Email [hi@davecto.com](mailto:hi@davecto.com) with the subject line **"/brag."** Paste the plan, not the video — the plan is where the problem is.

---

## More guides like this one, for people trying to build software without fooling themselves.

Weekly, plain-language breakdowns on Instagram.

[@davecto](#)

---

A note on sourcing. Everything here about how `/brag` behaves — the install commands, the options table, the four steps, the 15–25 second ceiling, the tone presets and the automatic handoff to `/brag-slim` on Opus 5.5 — comes from the skill and its README as published at [github.com/latent-spaces/brag](https://github.com/latent-spaces/brag). It's an actively developed open-source project, so check the README before you copy a flag out of here. The loudness figures in Tip 05 are measurements from one run on one project, not a benchmark; treat them as an illustration of why to check, not as numbers to aim at. `/brag` is not mine — I have no relationship with the people who make it, or with Hyperframes. I just use it.