

FROM A 3 HOUR PODCAST TO CLIPS READY TO POST

HOW TO MAKE MONEY CLIPPING WITH AI

The full workflow, the two ways to get paid and a free Master Prompt.

Most clipping campaigns pay **\$1 to \$2 per 1,000 views**, and Whop Content Rewards alone had **673 active campaigns** when we checked. Creators and brands pay for clips because every clip that travels is free reach for them. The hard part is the **hours of editing**. This is the full workflow to turn a 3 hour podcast into clips ready to post in a few minutes, with the exact settings we used.



Joe Rogan Experience #2404 with Elon Musk, 3 hours and 18 minutes. On the right, one of the 4 vertical clips Calliope cut from it.

01 · PASTE THE LINK

02 · SET IT UP

03 · POLISH THE CLIPS

04 · GET PAID

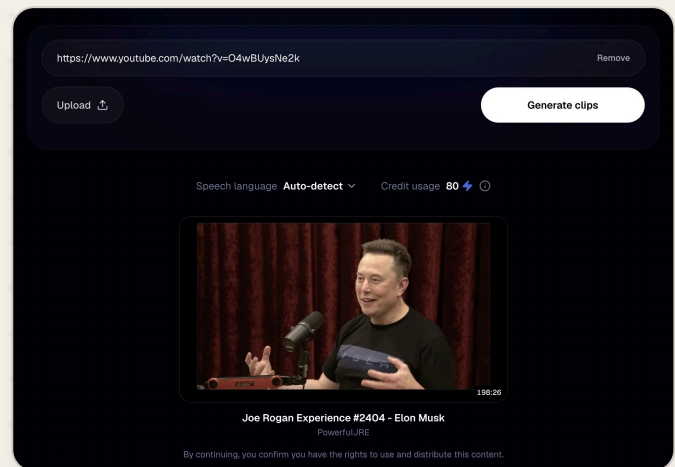
1 • PASTE THE LINK

One link, and the preview loads.

Open **Clipping** from the sidebar and paste a YouTube link, or click **Upload** to use your own file. Calliope loads a preview with the title, the channel and the length.

- A YouTube video can be up to **10 hours** long.
- Your own file: MP4, MOV or WEBM, up to **5 GB**.

Under the preview there's one line that matters later: *"By continuing, you confirm you have the rights to use and distribute this content."* Keep it in mind for step 4.



1.1 • One link and the preview loads. This episode runs 198 minutes.

2 • SET IT UP

Choose the model, the clips and the range you pay for.

Auto clip cuts the video into clips. **Captions only** adds editable captions to a video you upload, without cutting or reframing anything.

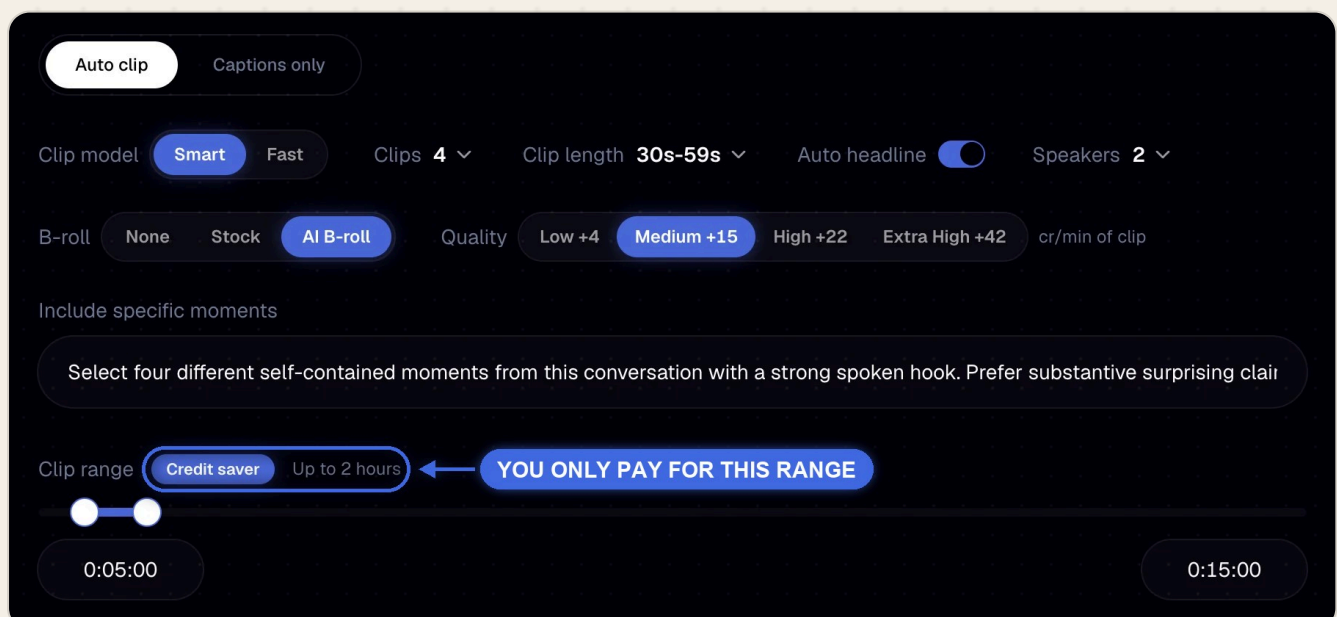
Smart reads everything that's said in your range, picks the strongest moments and frames the speaker for vertical. **Fast** splits the video into equal parts.

Ask for up to **10 clips**, and turn on **Auto headline** for a hook on screen in the first 5 seconds.

WHAT IT COSTS

Smart uses **4 credits per minute of range**, or **2 on the Creator plan**, the same whether you get 1 clip or 10. A range covers **up to 2 hours**, and our 10 minute range cost 20 credits on Creator. AI B-roll costs extra.

Fast and **Captions only** are **unlimited on every paid plan**.



2.1 • The Rogan batch: Smart, 4 clips of 30 to 59 seconds, 2 speakers, AI B-roll at Medium and a 10 minute range.

CALLIOPE VS OPUSCLIP

Same 1h45m podcast, two prices.

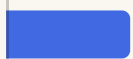
CALLIOPE ON THE CREATOR PLAN

4.9x cheaper

And still 2.4x cheaper on our Starter plan.

Calliope

CREATOR PLAN

 **≈ \$2.10**
212 credits · ~\$0.02 per minute

OpusClip

PRO PLAN

 **≈ \$10.20**
~\$0.10 per minute

GOOD TO KNOW

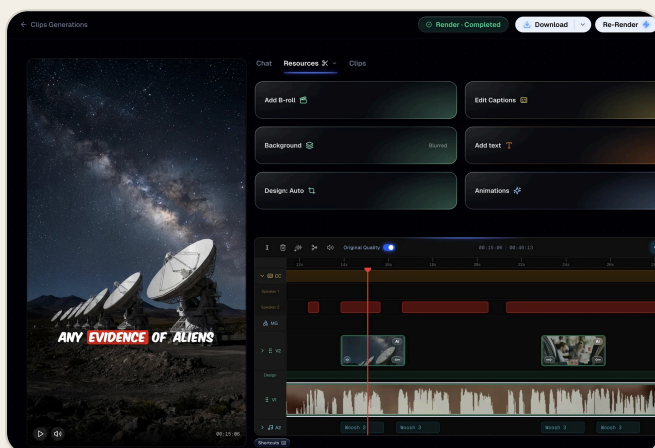
- You pay for the minutes you ask it to analyze, not per clip.
- **Fast clips** and **Captions only** are **unlimited on every paid plan**.
- In fairness: Calliope currently makes up to **10 clips at once**.

3 • POLISH THE CLIPS

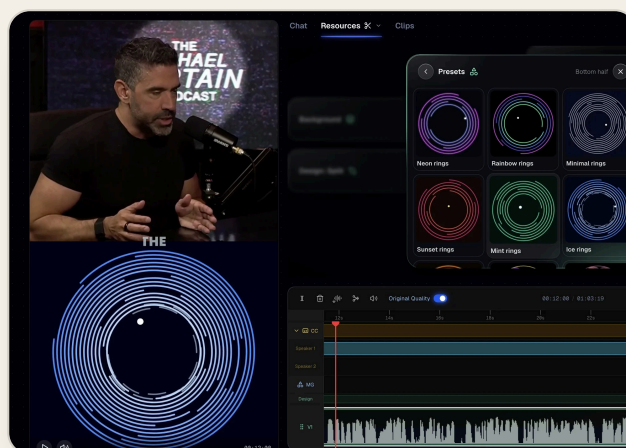
Plenty are ready to post as they are. The editor is for the rest.

Every clip comes out vertical, with the speaker framed, captions, a hook headline and the B-roll you asked for. In the editor it's a **timeline**, not a flat video: each piece sits on its own track, so you fix a clip instead of throwing it away.

- **Design:** Auto, Fill, Fill Tracking, Fit or Split.
- **Captions:** 20 styles that never cost credits.
- **Split:** speaker on top, one of **13 presets** below.
- **B-roll:** stock is **free**, AI images **5 credits** at Medium.



3.1 • AI B-roll on V2, captions on top.



3.2 • Split with the Ice rings preset.

4 • GET PAID

Two ways to make money with clips. Only one of them actually pays.

WAY 1 • FRAGILE

Post on your own accounts

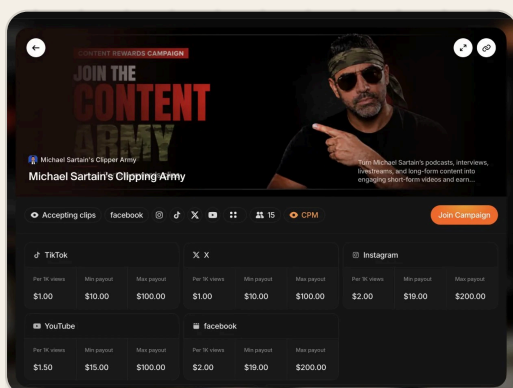
Build a page around a creator or a niche and monetize the views. YouTube, TikTok and Facebook keep reused content out of their payout programs, and one copyright claim can take down the whole account. Only worth it as a **funnel to something you own**: an info product, a newsletter or an affiliate link.

WAY 2 • THE CLEAN WAY

Join clipping campaigns

Creators and brands set a budget and **pay you per 1,000 views** on the clips you post. They ask for the clips, so you have permission.

Whop Content Rewards (contentrewards.com) had **673 active campaigns** when we checked, with a median of about **\$1.10 per 1,000 views**. Speak Spanish? clipeamos.com has creators and brands like Willyrex or the Kings League.



4.1 • Sartain's campaign on Content Rewards.

EXAMPLE • MICHAEL SARTAIN'S CLIPPING ARMY

\$1 per 1K views on TikTok and X, **\$1.50** on YouTube and **\$2** on Instagram and Facebook. On TikTok a clip has to earn \$10 (about 10,000 views) before it pays, and it stops at \$100. Only approved clips get paid, about 10 days after approval.

The workflow: open the campaign, copy an episode's link with **Share**, paste it in Calliope and post the best clips. We did it with Sartain's episode 164: one link, four clips.

THE MASTER PROMPT

Do it yourself, for free: paste it into an AI agent like Claude Code, Codex or Cursor.

It works: we ran it end to end. But you need **yt-dlp**, **Whisper** and **ffmpeg** and a **powerful computer** (on an M5 Pro Mac, one hour of audio took about 16 minutes to transcribe). It's **slow and fragile**: a 10 minute range took over 5 minutes to download. And the clips come out bare: one caption style, a fixed crop, no B-roll and **no editor**.

COPY AS IS

You are a professional short-form clipper and a careful terminal operator. You turn long videos (podcasts, streams, interviews) into vertical clips that hold attention from the first second. You work locally on my computer, one step at a time, and before each step you tell me in one line what you are about to do.

GOAL

One YouTube video in, up to 10 vertical clips out: 1080x1920 MP4 files with burned-in captions, cut so each one opens on a hook and ends on a payoff.

STEP 0. ASK ME, THEN WAIT

Ask me all of this in one message and wait for my answers:

1. The YouTube URL.
2. How many clips (default 5, max 10) and how long (default 30 to 60 seconds).
3. Which part to scan, as start and end time. Default: the whole video. If it runs longer than 90 minutes, suggest scanning one 60 to 90 minute range at a time.
4. Anything specific to look for (optional).
5. The campaign rules, if I'm clipping for a campaign: required tags, disclosure, minimum length, banned topics.
6. Confirmation that I own the video or have the creator's permission to clip it, for example through an active clipping campaign. If I don't, stop here.

STEP 1. CHECK THE TOOLS

Create a folder named clips-<video id> with two empty subfolders, clips and previews, and work only inside it. Check each tool and print its version:

- ffmpeg and ffprobe, with the "subtitles" filter (libass). Check with: `ffmpeg -hide_banner -filters | grep subtitles` (Windows: findstr instead of grep). On macOS the default Homebrew ffmpeg no longer includes it: install `ffmpeg-full` (brew install `ffmpeg-full`) and call it by its full path, `$(brew --prefix ffmpeg-full)/bin/ffmpeg` (same for ffprobe). On Windows use a full build (winget install Gyan.FFMpeg). On Linux the distro package works (`apt install ffmpeg`).
 - yt-dlp, updated to the latest version (brew upgrade yt-dlp or winget upgrade yt-dlp.yt-dlp if it's already installed; an old one fails with "403 Forbidden"), plus Deno, the JavaScript runtime yt-dlp needs for YouTube (brew install yt-dlp deno, or winget install yt-dlp.yt-dlp DenoLand.Deno). If ffmpeg is not on the PATH (ffmpeg-full on macOS), always give yt-dlp its folder: `--ffmpeg-location "$(brew --prefix ffmpeg-full)/bin"`.
 - Python 3.9 or newer, with faster-whisper in a virtual environment inside the folder: `python3 -m venv .venv`, then `.venv/bin/pip install -U faster-whisper "av<19"` (PyAV 19 breaks faster-whisper 1.2.1; on Windows the pip is `.venv\Scripts\pip`).
- If something is missing, show me the exact install command for my system and wait for my OK. Never install anything without asking.

STEP 2. DOWNLOAD

1. Read the basics first and show them to me: `yt-dlp --print "%(title)s | %(channel)s | %(duration_string)s" URL`
2. Download the whole video at up to 1080p, then cut the part to scan into source.mp4 without re-encoding:


```
yt-dlp -f "bv*[height<=1080]+ba/b[height<=1080]" --merge-output-format mp4 -o full.mp4 URL
```

`ffmpeg -ss START -to END -i full.mp4 -c copy source.mp4`
 Exception: if the part to scan is less than a fifth of the video (say 10 minutes of a 1 hour video), download only that part instead. It's slower per minute but moves far less data:


```
yt-dlp -f "bv*[height<=1080]+ba/b[height<=1080]" --merge-output-format mp4 --download-sections "%START-END" -o source.mp4 URL
```
3. Extract the audio: `ffmpeg -i source.mp4 -vn -ac 1 -ar 16000 audio.wav`
 From here on, every timestamp is relative to source.mp4, not to the original video.
 If YouTube answers "Sign in to confirm you're not a bot", ask me before retrying with `--cookies-from-browser` and my browser's name.

STEP 3. TRANSCRIBE

Use faster-whisper with `word_timestamps=True` and `vad_filter=True`. Default model: "large-v3-turbo", on device "cuda" with compute_type "float16" if I have an NVIDIA GPU, otherwise on the CPU with compute_type "int8". It downloads about 1.6 GB the first time. If my

computer is slow or I'm in a hurry, use "small" instead: about twice as fast, but it may skip punctuation. Detect the language unless I gave you one. Print progress: this is the slow step.
 Save transcript.json (segments with their words and start and end times) and transcript.txt (one line per segment, as [mm:ss.s] text).

STEP 4. FIND THE MOMENTS

Read the whole transcript.txt. Shortlist about twice as many candidates as the clips I asked for, then keep the best. A good clip:

- opens with a strong line in its first 3 seconds (a claim, a question, a number, a conflict), never with "so", "yeah" or "and";
- makes sense without the rest of the episode;
- has a payoff: an answer, a punchline, a twist or a clear conclusion;
- fits the length I asked for and doesn't overlap any other clip.

 Start each clip on the first word of a sentence and end it on the last word of a sentence, using the word timestamps: start exactly at the first word's start and end 0.15 s after the last word's end. Whisper leaves no gap between words, so any padding before the first word catches the end of the previous one.
 Save clips.json with: index, start, end, duration, title (max 8 words), the opening line and one sentence on why it works. Show me a table and wait for my OK before cutting.

STEP 5. CUT, REFRAME AND CAPTION

For each approved clip, NN being its number:

1. Captions: write clip_NN.ass from the word timestamps, with times relative to the clip start. `PlayResX 1080 and PlayResY 1920`. One to three words per line, 16 characters max, each line shown from its first word's start to its last word's end. Style: Arial Black (or any bold sans serif you find), size 80, uppercase, white text, black outline 6, no shadow, bottom centered (Alignment 2) with MarginV 560 and MarginL and MarginR 60.
2. Reframe to 9:16. The default works for any video: the full frame, scaled to 1080 wide, over a blurred and darkened copy of itself. If I ask for a tight crop instead, extract a frame from the middle of the clip, look at it, and pick the horizontal crop that centers the speaker. That crop is fixed: it won't follow anyone who moves or a camera that changes.
3. Encode in one pass, seeking before the input so cuts are frame accurate:


```
ffmpeg -y -ss START -i source.mp4 -t DURATION -filter_complex "[0:v]split=2[bg][fg]; [bg]scale=1080:1020:force_original_aspect_ratio=increase,crop=1080:1920,gblur=sigma=40,eq=brightness=-0.15[bgb]; [fg]scale=1080:-2[fgs]; [bgb][fgs]overlay=(W-w)/2:(H-h)/2,subtitles=clip_NN.ass[v]" -map "[v]" -map 0:a -c:v libx264 -preset medium -crf 20 -pix_fmt yuv420p -r 30 -af loudnorm=I=-14:TP=-1.5:LRA=11 -ar 48000 -c:a aac -b:a 160k -movflags +faststart clips/NN-title-slug.mp4
```

 Run ffmpeg from the project folder and pass the .ass file by its plain relative name: the subtitles filter breaks on paths with colons, commas or backslashes.

STEP 6. CHECK EVERYTHING

For every clip, ffprobe must report 1080x1920, an audio stream and a duration within 0.5 s of the plan. Transcribe the first 3 seconds of each clip again: if they don't start with the planned opening line, move the start 0.1 s later and re-render. Extract one frame at 40% of each clip into previews/ and, if you can view images, look at it: captions readable, nothing important cut off. Fix and re-render any clip that fails, up to 3 tries, then tell me what's wrong.

STEP 7. DELIVER

Give me a table: file, start and end in source.mp4, duration, title and opening line. Under it, one ready-to-post caption per clip (150 characters max, plus the campaign's required tags or disclosure if I gave you rules).

RULES

- Never upload, post or send anything. Everything stays in the project folder.
- Don't download or transcribe again if the file already exists.
- If a command fails, show me the error, fix it and retry. After 3 failed tries on the same step, stop and explain.
- Remember my answers from Step 0 for the whole session. Don't ask again.

START CLIPPING

Paste a link, pick the range, generate.

Open Clipping: calliope-labs.co/tools/clipping

START AT [CALLIOPELABS.CO](https://calliope-labs.co) →