

Edit videos with *AI*.

The four free tools I use instead of paying an editor \$15 a video — installed for you by Claude. You don't type a single command. You drop this PDF in and say one sentence.

01

WhisperX

Listens to your video and timestamps every word — so silences and stumbles can be cut out automatically.

02

Remotion

Draws the graphics you describe — captions, titles, charts, animated text — as real video.

03

FFmpeg

Cuts, joins and exports — puts the picture and the sound back together into one file.

04

Python

The glue that runs WhisperX and the little helper scripts. It just does its thing.

HOW TO USE THIS PDF

1

Open Claude Code

The desktop app's *Code* tab, or `cLaude` in a terminal. Pick an empty folder for your editor.

2

Drag this PDF in

Drop the file straight into the chat box.

3

Say the sentence below

Claude explains each step, asks before installing, and checks every tool works.

TYPE THIS

“Read this PDF and set up my AI video editor. Explain each step simply and ask me before you install anything.”

— Windows 10/11 or a Mac

— Claude Code (needs a paid Claude plan — the free chat can't install software)

— An NVIDIA graphics card makes WhisperX fast — but it's optional

— About 10 GB of free space

— 20–40 minutes, mostly waiting for downloads

— Every tool here is free and open-source

Hello Claude — you are setting up an AI video editor.

The person who gave you this PDF is probably **not a coder**. Your job: install four free tools (Python, FFmpeg, Node.js + Remotion, WhisperX), prove each one works, and create a ready-to-use editing workspace. Talk in plain, short sentences.

Ground rules — follow all of them

- **Ask before each install.** Say what you are about to install, why, roughly how big it is, then wait for a yes.
- **Check first, install second.** If a tool is already installed and new enough, keep it — don't reinstall.
- Use only official sources: `winget` on Windows, `brew` on macOS, `pip`, `npm` / `npx`.
- **Never** delete or move the user's existing files, change system security settings, or run anything from a link you weren't given here.
- If a step needs an admin password or a terminal the user must type into, **tell them exactly what to do** and wait — never guess passwords.
- If something fails, read the error, try the obvious fix once, then explain the problem simply and ask how to proceed.
- Prefer the **latest stable** versions. If a command below is out of date, use the current official equivalent and say so.

Step 1 — Detect the computer

Find the OS (Windows / macOS / Linux) and the CPU type. On Windows run `nvidia-smi`; if it prints a GPU, note it — WhisperX can use it. On a Mac, Apple Silicon runs WhisperX on the CPU (fine, just slower). Tell the user what you found in one sentence.

Step 2 — Install the base tools

WINDOWS

```
# check first: python --version  ffmpeg -version  node --version
# install only what is missing:
winget install -e --id Python.Python.3.11
winget install -e --id Gyan.FFmpeg
winget install -e --id OpenJS.NodeJS.LTS
```

Open a new terminal after installing so the new commands are found. If `python` opens the Microsoft Store, use `py -3.11`.

MACOS

```
# Homebrew first, if missing. The USER must paste this into Terminal (it asks for their
password):
/bin/bash -c "$(curl -fsSL
https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
# then:
brew install python@3.11 ffmpeg node
```

VERIFY Python 3.10–3.12 · `ffmpeg -version` prints a version · Node.js 20 or newer. Report the three versions to the user.

Step 3 — Create the workspace

Ask the user where it should live (default: an `ai-video-editor` folder in their home folder, or the folder Claude Code is open in). Create:

```
ai-video-editor/
├── input/          ← the user drops raw clips here
├── work/           ← transcripts, cut lists, temporary files
├── output/         ← finished videos
├── remotion/       ← the graphics project (Step 5)
├── tools/          ← small Python helper scripts you write later
├── .venv/          ← private Python environment for WhisperX (Step 4)
└── CLAUDE.md       ← the editor's brain (page 4)
```

Step 4 — Install WhisperX (in its own environment)

```
# inside ai-video-editor/
py -3.11 -m venv .venv          # Windows   (macOS: python3.11 -m venv .venv)
.venv\Scripts\activate         # Windows   (macOS: source .venv/bin/activate)
python -m pip install --upgrade pip
pip install whisperx
```

- **NVIDIA GPU found in Step 1?** Install the CUDA build of PyTorch that matches the WhisperX version you just installed (check pytorch.org for the current command), so transcription runs on the GPU. Otherwise the CPU build is fine.
- The first transcription downloads the speech model (about 1.5–3 GB). Warn the user before it starts.

VERIFY `whisperx --help` runs. Then ask the user for any short clip of them talking (10–30 s), copy it into `input/`, and transcribe it with word timestamps into `work/`. Show them three lines of the result with times.

Step 5 — Create the Remotion graphics project

```
# inside ai-video-editor/
npx create-video@latest remotion # choose the "Blank" template if asked
cd remotion && npm install
```

- Add one composition called `TestTitle` (1080×1920, 30 fps, 3 s) that animates the user's name onto the screen.
- If the Remotion agent skills are offered, install them — they teach you Remotion's rules.

VERIFY Render it: `npx remotion render TestTitle ../output/test-title.mp4`. Then use FFmpeg to join that title in front of the user's test clip (`output/first-edit.mp4`) — that proves all four tools work together. Tell the user where the file is.

Step 6 — Write the editor's brain: CLAUDE.md

Create `ai-video-editor/CLAUDE.md` with the paths and versions you installed, plus this default workflow, so any future session already knows how to edit:

1. **Never touch originals.** Raw clips stay in `input/` ; everything new goes to `work/` and `output/` .
2. **Transcribe** with WhisperX (word timestamps, the user's language) → `work/<name>.json` . Re-transcribe a short range on its own if a word looks wrong — long takes can hide false starts.
3. **Cut** silences longer than ~0.35 s and repeated / stumbled lines. Show the user the cut list before cutting.
4. **Graphics** come from Remotion: captions, titles, lower-thirds, simple charts — whatever the user describes, in their colours.
5. **Assemble** with FFmpeg: vertical 1080×1920 for Reels/Shorts unless told otherwise, H.264, audio loudness about -14 LUFS.
6. **Always show a preview** (a few still frames or a low-res render) before the final export, and ask before re-doing anything slow.
7. Explain everything in plain words. The user is the director; you are the editor.

Finish by telling the user: what was installed (with versions), where the workspace is, where their test video is, and the three example requests from the box below. **Setup complete.**

FOR YOU • WHAT TO SAY NEXT

You're the director now.

1. *"Cut the silences and the mistakes out of the clip in my input folder."*
2. *"Add bold captions, one word at a time, in white with a yellow highlight."*
3. *"When I say 'four tools', pop up four boxes with their names above my head."*
4. *"Export it for Instagram Reels."*

Want the plug-and-play version — my full editing setup, templates and all the research behind it, so you skip the trial and error? That's the paid kit: you pay once, not per video.