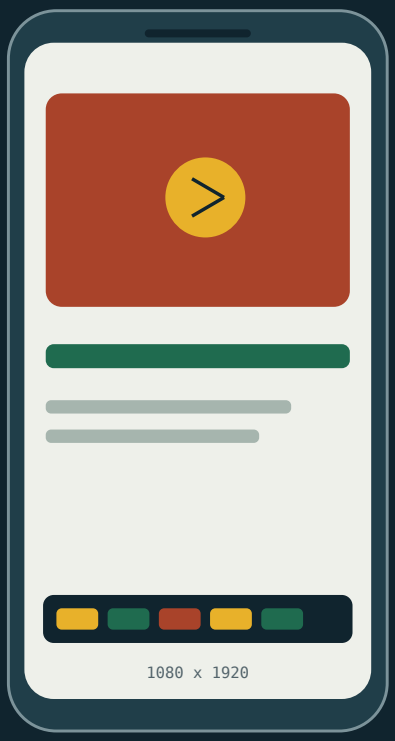
[A PRACTICAL GUIDE]

Edit your first video with Claude Code

A beginner-friendly walkthrough for setting up a local video-editing workflow on Windows - with an interactive AI-guided route if you would rather not do the setup alone.



THE LOOP

CUT · CAPTIONS · CHECK · EXPORT

Two routes: ask an AI assistant to guide you one step at a time, or follow the installation checklist yourself.

[ARHAM FAHMI]

Tech for local business, in plain words.

INDEPENDENT GUIDE / REPOSITORY VERIFIED / OCT 2026

What you are setting up

Think of this as a workflow, not a magic video editor. Claude Code coordinates the work; local tools process the footage.

01

CLAUDE CODE

Reads your request, inspects files, helps plan the edit and calls tools.

02

FFMPEG + PYTHON

Processes video/audio, applies cuts and effects, and assembles the output.

03

VIDEO SKILL

Reusable instructions and scripts triggered by the /videoediting command.

The safe workflow

01 **Inspect** → 02 **Plan** → 03 **Preview** → 04 **Review** → 05 **Export**

What it can help with

- Cut pauses and filler words based on a transcript.
- Add word-level burned-in captions.
- Apply basic colour and audio finishing.
- Create graphic cards and simple infographics.
- Prepare vertical exports for Reels and Shorts.

What it is not

- Not a one-click installer.
- Not a full replacement for Premiere or DaVinci Resolve.
- Object tracking and masked grading are not supported by this core pipeline.
- Complex frame-accurate timing and music-beat edits may need a timeline editor.

Review matters. Speech recognition can mishear names and words, including Malayalam. Proofread the transcript and captions. Always preview the video before a final export.

Choose your setup route

This guide focuses on a local Windows workflow. macOS and Linux users should follow the repository installation guide for their platform.

RECOMMENDED IF YOU ARE NEW

Route A - guided
Paste the prompt on the next page into ChatGPT, Gemini, or Claude. Ask it to inspect first, explain each action, and wait after every step.

FOR CONFIDENT USERS

Route B - do it yourself
Follow the Windows steps on pages 5-8. You will need to use PowerShell for a few commands.

Before installing anything

- Windows 10/11 PC with internet access and permission to install software.
- Claude Code installed and signed in. Use a local session that can access files on this PC, not a browser-only chat or cloud workspace.
- Free disk space: reserve at least 4 GB for the speech model; more is safer for source files, cache and exports.
- A short video you own or have permission to edit. Work on a copy. Keep the original safe.
- Time to review: the setup has several dependencies and the first model download can take a while.

Important distinction

The blazerior/videoediting-claude-skill repository is a third-party open-source project, not an official Anthropic tool and not a skill created by Arham Fahmi. This guide is an independent walkthrough. Setup screens and repository steps can change.

Repository and official docs

GitHub repository

Full INSTALL.md

Official Claude Code setup

Scan the QR code to open the skill repository.

Let an assistant guide you

Use this if you do not want to manage every technical detail yourself. Paste the entire prompt into an AI assistant you trust.

Before you paste: tell the assistant you are on Windows and want the local Claude Code workflow. Never share passwords, API keys, or private footage just to troubleshoot setup.

I am a beginner with no programming experience. Help me set up the third-party video-editing skill from this repository on my Windows PC, then edit one short video for an Instagram Reel. Repository: https://github.com/blazerior/videoediting-claude-skill . Read its current README.md and INSTALL.md before giving steps. Do not pretend this is an official Anthropic tool.

Work with me one small step at a time. Start by asking which interface I use and confirming that it is a local Claude Code session that can access files on this computer. Do not assume Claude Desktop, a cloud session, or a browser chat has access to local tools.

First inspect what is already installed. Check Claude Code, Python 3.12, FFmpeg and its required filters (subtitles/libass and lut3d), Chrome, available disk space, and whether the videoediting skill already exists. Do not install, download, overwrite, delete, change settings, or run an unfamiliar script without explaining what it does and asking me first.

Use the repository's current instructions. Explain that the Windows install script copies the skill but does not install every dependency. Prefer the ZIP/manual route if Git is missing. If a skill already exists, stop and help me back it up before replacing anything. Use an isolated Python 3.12 virtual environment if appropriate, and make sure Claude Code is launched with the correct Python on PATH.

Give me ONE step only, including the exact button or command, what it changes, what output to expect, and what to do if it fails. Wait for my result before continuing. Verify each step instead of assuming it worked. Keep the first-time Whisper model download and its disk-space requirement separate from the skill installation.

Once setup is verified, help me create a project folder and copy my own short video into it. Inspect the footage and transcript first, including proof-reading the captions. Propose an edit plan and wait for my approval before rendering. Never overwrite the original video or an existing export. Render a preview first; check captions, audio sync, cuts, framing, glitches and playback. Export only after I approve the preview, using a new filename. If anything fails, diagnose the exact error and fix one issue at a time rather than starting over.

Begin by asking the single most important question needed to choose my setup path. Do not give me the whole checklist at once.

Do not skip verification. The assistant should stop after each step and wait for your output. If it suggests a command that downloads, installs, deletes or changes settings, ask what it does before approving.

Step 1 - install the foundations

Follow one install at a time. If the tool is already installed, do not install it again just to match this guide.

Use official sources. Claude Code and the media tools are different components. Installing the skill alone will not make the editing pipeline work.

A

Claude Code

Install Claude Code using Anthropic's current official Windows instructions. The official setup page is linked below. Sign in when prompted.

B

FFmpeg (full build)

Required to process video/audio. The full build matters because this skill depends on the subtitles/libass and lut3d filters.

```
winget install --id Gyan.FFmpeg -e
```

C

Python 3.12

Required by the skill scripts. Python 3.10+ is stated by the repository; this guide uses 3.12 to keep the workflow consistent.

```
winget install --id Python.Python.3.12 -e
```

D

Google Chrome

Used for rendering graphic overlays. If you do not need graphics, ask the assistant whether the current workflow can run without it.

```
winget install --id Google.Chrome -e
```

After installation: close and reopen PowerShell or the terminal so PATH refreshes. Verify each tool before moving on. If `winget` is not available, use the official download route linked in the repository INSTALL.md - do not use random download sites.

Official Claude Code setup

FFmpeg full builds

Step 2 - add the video skill

The skill is the reusable instruction folder. You can download it without Git; Git is optional.

1

Open the repository

Go to the GitHub repository. Select the green Code button, then Download ZIP. Save it somewhere you can find, such as Downloads.

2

Extract the ZIP

In File Explorer, right-click the ZIP and choose Extract All. Open the extracted folder. It should contain items such as README.md, INSTALL.md, scripts, and skill.

3

Install the skill folder

The folder that must be copied is skill\videoediting. The destination for a user-level install on Windows is %USERPROFILE%\.claude\skills\videoediting.

Option 1: use the repository installer script

Open PowerShell in the extracted repository folder (the folder that contains the scripts folder), then run this one command:

```
powershell -ExecutionPolicy Bypass -File scripts/install.ps1
```

Know what this command does. According to the repository INSTALL.md, the script copies the skill to your user-level Claude skills folder and reports missing dependencies. It does not install FFmpeg, Python, Chrome, or faster-whisper. Review the script first if you are unsure about running third-party code.

Option 2: copy the folder manually

Copy only the `videoediting` folder from `skill`, then paste it into `%USERPROFILE%\.claude\skills\`. If the parent folder is missing, create it first. If a videoediting folder already exists, stop and back it up before replacing anything.

Read the repository installation steps

Step 3 - set up Python and verify

These commands are for Windows PowerShell. Copy one command at a time. They download Python packages; do not run them if you do not understand or approve the change.

A. Create an isolated Python environment

This keeps the packages for the video workflow separate from other Python projects.

```
py -3.12 -m venv "$env:USERPROFILE\.venvs\videoediting"
```

B. Install the Python packages

```
& "$env:USERPROFILE\.venvs\videoediting\Scripts\python.exe" -m pip install --upgrade pip
```

```
& "$env:USERPROFILE\.venvs\videoediting\Scripts\python.exe" -m pip install faster-whisper pillow
```

This downloads Python packages (roughly a few hundred MB in total depending on versions). The first transcription later downloads a separate speech-recognition model. yt-dlp is only needed if you use a source video URL; an API key is only needed for cloud transcription.

C. Help the model download complete

The repository documents this setting as a fix for model downloads that stall on some networks. The first command saves it for future sessions; the second applies it to this one:

```
[Environment]::SetEnvironmentVariable("HF_HUB_DISABLE_XET","1","User")
$env:HF_HUB_DISABLE_XET = "1"
```

D. Launch Claude Code with the correct Python

Open the project folder first (see page 8). In that folder's PowerShell window, add the virtual environment to PATH, verify it, then start Claude Code:

```
$env:Path = "$env:USERPROFILE\.venvs\videoediting\Scripts;$env:Path"
python --version
claude
```

Expected: `python --version` reports Python 3.12.x. If Claude Code cannot be launched from this window, do not guess at paths; stop and ask the assistant to inspect the exact error.

[FIRST SUCCESS]

Step 4 - test before editing

Do not start with a long video. First confirm the tools and skill are visible, then use one short copied clip.

Quick verification

- FFmpeg responds
  Run ffmpeg -version.
- Filters exist
  Run ffmpeg -hide_banner -filters | Select-String "subtitles|lut3d". Both filters should appear.
- Python package imports
  Run python -c "import faster_whisper; print('faster-whisper OK')" in the same terminal.
- Skill appears
  In Claude Code, type / and confirm videoediting appears.

First transcription

The first transcription downloads a Whisper speech model. The repository currently lists options from about 0.5 GB to 3.1 GB; its default recommendation is large-v3-turbo at about 1.6 GB. Choose a model based on your free space and accuracy needs.

Privacy: local transcription can keep audio on your computer. Do not use cloud transcription unless you understand that audio is sent to the selected provider and approve that choice.

Open the right folder and start Claude Code

In File Explorer, create Videos\my-first-reel and COPY your video into it. Open that folder, click the address bar, type powershell, and press Enter. In this PowerShell window, run the PATH line on page 7, then type claude. When Claude Code opens, use `/videoediting` or the exact command shown by the skill menu.

```
Use /videoediting to help edit the copied video in this project for an Instagram Reel. Inspect the source and create a transcript/contact sheet first. Show me the proposed cuts and edit plan. Do not render or change the source until I approve the plan. Use local transcription; proofread the transcript before captions. Preserve the original audio and create preview files with new filenames.
```

After the plan is approved

- Review the transcript, including the opening and final line.
- Review the preview full-size for cut points, caption spelling, face-safe space, sync and black frames.
- Ask for one specific correction at a time; save each iteration as a new version.
- Approve a final export only after the preview is acceptable.

[WHEN SOMETHING BREAKS]

Troubleshooting without panic

Capture the exact error and fix the earliest failed step. Reinstalling everything is rarely the best first move.

The /videoediting command is missing

Confirm this is a local Claude Code session. Check that the file exists at %USERPROFILE%\.claude\skills\videoediting\SKILL.md. Restart the session and type / again.

FFmpeg is not found

Close and reopen the terminal. Run ffmpeg -version. If it still fails, check the install and PATH before reinstalling. Use the full FFmpeg build.

subtitles or lut3d filter is missing

The FFmpeg build is too limited. Install the full build referenced by the repository and re-check the filter list.

faster-whisper import fails

Check that the virtual environment is on PATH in the same terminal that launched Claude Code. Re-run the import test with the venv Python executable.

Whisper model download appears stuck

Check free disk space and your connection. The repository recommends setting HF_HUB_DISABLE_XET=1 for networks where downloads stall. Do not keep restarting blindly.

Graphics / overlay PNG does not appear

Chrome is used for the overlay renderer. Confirm Chrome is installed and follow the Acceptance test in INSTALL.md. The test should happen before a full video render.

Captions are wrong or invented

Speech recognition is not perfect. Proofread the transcript and check the first/last lines, especially in Malayalam or noisy audio. Correct the transcript before burning captions in.

The preview fails or looks broken

Ask Claude to report the last successful step and exact error. Request a new version with a new filename; never overwrite the source or a known-good export.

When asking for help, send: the exact error text, which step you were on, the command you ran, the output, your Python version, and the output of `ffmpeg -version`. Remove private paths or details if needed. Never send passwords, API keys, or private footage.

[FINISH WELL]

A repeatable workflow you can trust

The goal is not to automate blindly. It is to get a reviewable first version, then improve it safely.

INSPECT -> TRANSCRIBE -> PROPOSE PLAN -> APPROVE -> PREVIEW -> REVIEW -> EXPORT

If a step fails, pause and diagnose it before moving forward.

Before editing

- Original footage is backed up.
- Working from a copy in a dedicated project folder.
- Correct video, language and target platform selected.
- Transcript read and corrected.
- Edit plan approved before rendering.
- No unexplained downloads or installations.

Before final export

- Preview checked at full size.
- Captions spelling and safe placement checked.
- Audio, lip-sync, cut points and playback checked.
- No black frames or glitches.
- Output saved under a new filename.
- Final file plays from beginning to end.

Resources to bookmark

1. Skill repository
Code, files, README, current requirements.

2. Repository installation guide
Platform-specific setup, dependencies and acceptance test.

3. Official Claude Code setup
Use for the current Claude Code installation/sign-in path.

4. FFmpeg full builds for Windows
Use the full build if required filters are missing.

OPEN THE REPOSITORY

This guide is an independent walkthrough of a third-party project maintained by blazerior. Repository files and dependencies can change; check the current INSTALL.md when the steps differ.

Keep it simple. Start with one short video, check every step, and change one thing at a time.