

ACE-Step XL 1.5 Premium v5.3 Written Tutorial

This README is the full Markdown tutorial generated from the public YouTube tutorial, local file ACESTEP_XL_15_Full_Tutorial.mp4, captions (2).srt, live local ACE-Step app screenshots, v5.3 Wildcards screenshots, one newly generated demo song, one Audio Processing run made from that song, and selected 4K frames extracted from the source video.

Generated demo song: G:\ACE_Step_v1\ACE-Step_Premium\outputs\0021\8642a822-c4d0-ff4a-531f-6d5cfb334381.mp3

Processed demo output: G:\ACE_Step_v1\ACE-Step_Premium\outputs\audio_processing_0004\8642a822-c4d0-ff4a-531f-6d5cfb334381_processed.wav

Source Video, Thumbnail, Links, And Chapters

The public video description presents this as a full ACE-Step XL 1.5 Premium guide for local AI music generation, remix, repaint, stem extraction, wildcard prompt variation, audio processing, SAM Audio segmentation, Windows installation, RunPod, Massed Compute, SimplePod, and Linux/cloud workflows.



Tutorial video thumbnail

- **Tutorial video:** [Forget Suno: Run the Ultimate AI Music Studio LOCALLY \(100% Free\)](#)
- **Channel:** [SECourses](#)
- **App/latest zip and Patreon post:** [Patreon ACESTEP XL Premium SAM Audio post](#)
- **Windows requirements guide:** [Windows requirements guide](#)
- **Discord/help/community:** [SECourses Discord server](#)
- **Duration:** 1:15:09
- **Published:** June 17, 2026

Video Chapters

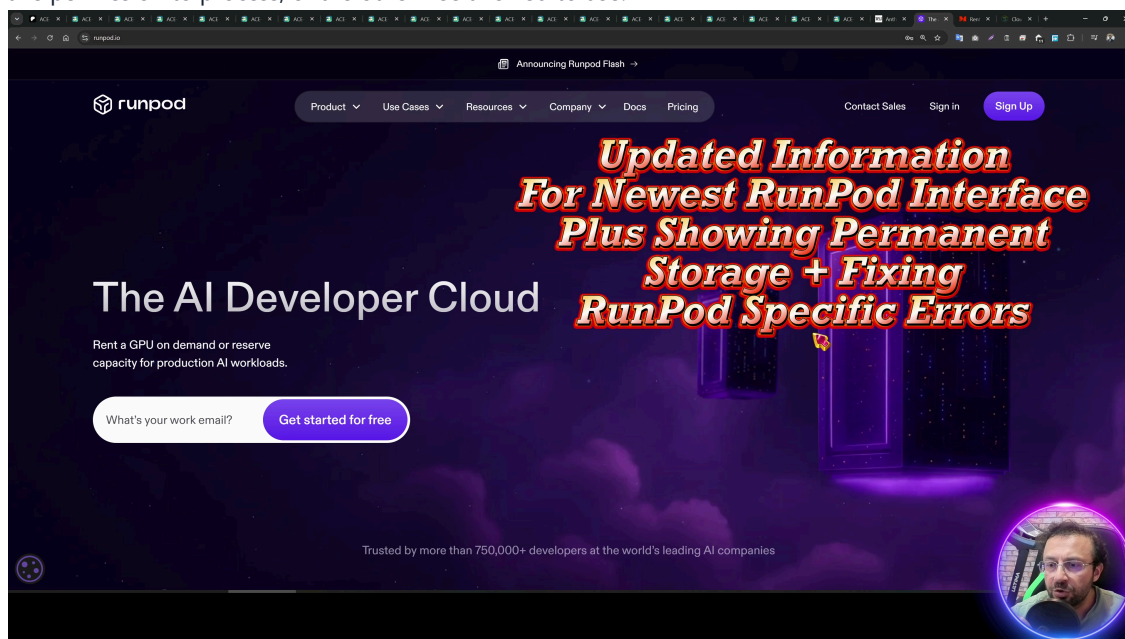
- [0:00](#) - Intro: ACESTEP XL 1.5 Premium local music, segmentation and processing tutorial
- [0:52](#) - Fast song generation examples across styles in under one minute
- [1:55](#) - Output manifest proof, 40-second generation time and supported models
- [2:29](#) - Turbo/SFT/Base models, LoRA support, GPU presets and Torch Compile boost
- [3:10](#) - Remix feature preview, same-lyrics requirement and responsible usage note
- [4:16](#) - Repaint mode: regenerate and merge only a selected song section
- [5:38](#) - Extract mode: stems, silence trimming, all-stems and batch folders
- [6:30](#) - Lego mode: add an instrument stem such as guitar into existing audio
- [7:25](#) - Audio Processing presets and manual enhancement controls for AI songs
- [8:35](#) - Auto-Editor silent trim for tutorials, videos, audio and workflow export
- [9:48](#) - DaVinci/Premiere/Final Cut/ShotCut/Kdenlive timeline export demo
- [11:01](#) - SAM Audio Segment: BF16 models, VRAM presets and advanced segmentation
- [11:47](#) - SAM outputs demo: vocals, drums, bass, remaining audio and saved files
- [12:47](#) - Custom SAM prompts, semicolon batch segmenting and speech cleanup example
- [14:19](#) - Batch processing, load metadata, manifests, saved settings and presets
- [15:09](#) - Why local open-source models matter and where to run ACESTEP
- [15:55](#) - Windows install begins: Patreon zip, changelog, attachments and download
- [16:53](#) - Windows requirements tutorial before Python/CUDA/C++/FFmpeg setup
- [17:29](#) - Extract zip safely, avoid bad paths and run Windows_Install_or_Update.bat
- [18:24](#) - Automatic VENV, FFmpeg, UV install, model downloads and hash verification
- [19:24](#) - Turbo default vs all-model download for SFT/Base and BF16 safetensors
- [20:32](#) - First Windows launch, default Generate Song test and CMD progress
- [21:44](#) - Model recommendations, VRAM tiers, languages, vocals and MP4 image output
- [23:29](#) - Torch Compile setup for faster repeated generations
- [24:05](#) - Outputs folder, model switching and full remix setup workflow

- [25:24](#) - Practical remix loop: adapted lyrics, strength, reference audio and seed lock
- [28:03](#) - Repaint workflow with source range preview, generated result and comparison
- [29:13](#) - Recap: extraction, Lego, audio processing and SAM text-prompt usage
- [30:20](#) - Windows wrap-up, LoRA training teaser and move to cloud installs
- [31:16](#) - RunPod setup: credits, template, CUDA filters, GPU choice and storage
- [34:53](#) - Upload zip in Jupyter Lab, extract, run instructions and handle installs
- [35:43](#) - RunPod errors, resume behavior, model downloads and hash verification
- [38:04](#) - Start ACESTEP on RunPod with Gradio Live, proxy ports and persistence
- [40:18](#) - Add 7860/7861 ports, verify storage reuse and rerun installer after resume
- [42:10](#) - RunPod connection troubleshooting and Gradio Live recommendation
- [44:12](#) - Fix corrupted VENV/stale handle errors, reinstall safely and retest
- [47:24](#) - Successful RunPod relaunch, default generation, nvitop and loading tips
- [49:26](#) - RunPod first load vs fast inference, 15-second second generation example
- [51:02](#) - Download outputs and delete RunPod pods/storage to stop spending
- [53:30](#) - Massed Compute setup: coupon, Creator image, GPU prices and ThinLinc
- [57:13](#) - Massed install from extracted folder, Linux notes and ultra-fast downloads
- [59:18](#) - Start app on Massed Compute via localhost or Gradio Live
- [1:00:23](#) - Default Massed generation, nvitop, faster loading and speed test
- [1:02:03](#) - Sync/download outputs and delete Massed Compute instance safely
- [1:03:25](#) - SimplePod setup: template, persistent volume, pricing and GPU choice
- [1:06:39](#) - Jupyter upload, direct file browser, install command and model downloads
- [1:08:21](#) - Start SimplePod, Gradio Live, default generation and one-time load errors
- [1:09:31](#) - nvitop monitoring, newer driver/CUDA details and generation completion
- [1:10:42](#) - Direct output/model downloads through SimplePod file browser
- [1:11:42](#) - Delete instance, keep storage, relaunch GPU and verify install
- [1:13:15](#) - Discord, subreddit, changelog, update guidance and support links
- [1:14:30](#) - Final cleanup: terminate servers, delete storage and LoRA training outro

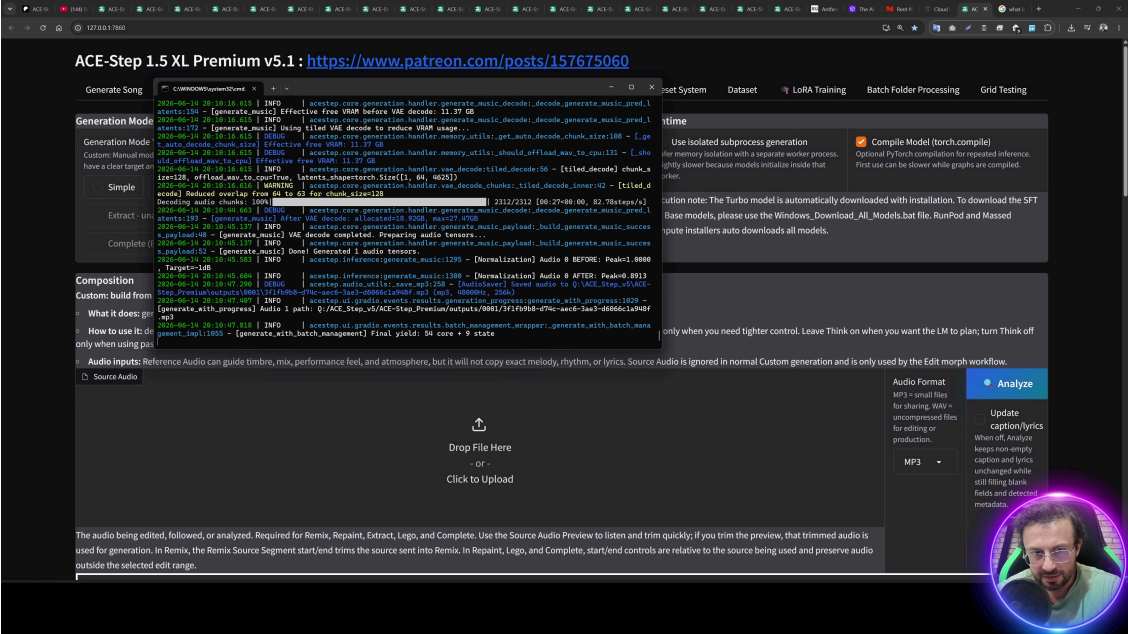
1. What ACE-Step XL 1.5 Premium Is

ACE-Step XL 1.5 Premium is a local-first music generation and audio utility suite. The video presents it as more than a song generator: it also includes Wildcards for prompt variation, advanced generation modes, remix/repaint workflows, stem extraction, LEGO-style stem addition, SAM Audio segmentation, Auto-Editor trimming, mastering-style audio processing, dataset tools, and LoRA/LoKr training pages.

Responsible-use note: the source tutorial says to use the application respectfully and for research/education. For remix, repaint, extraction, and pitch work, use material you own, have permission to process, or are otherwise allowed to use.



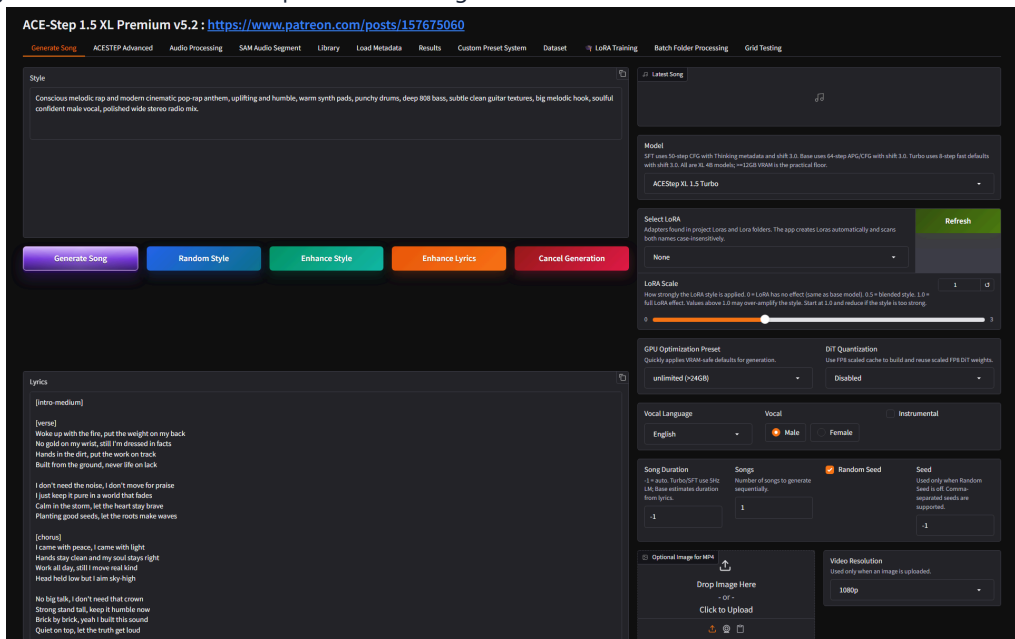
Video introduction



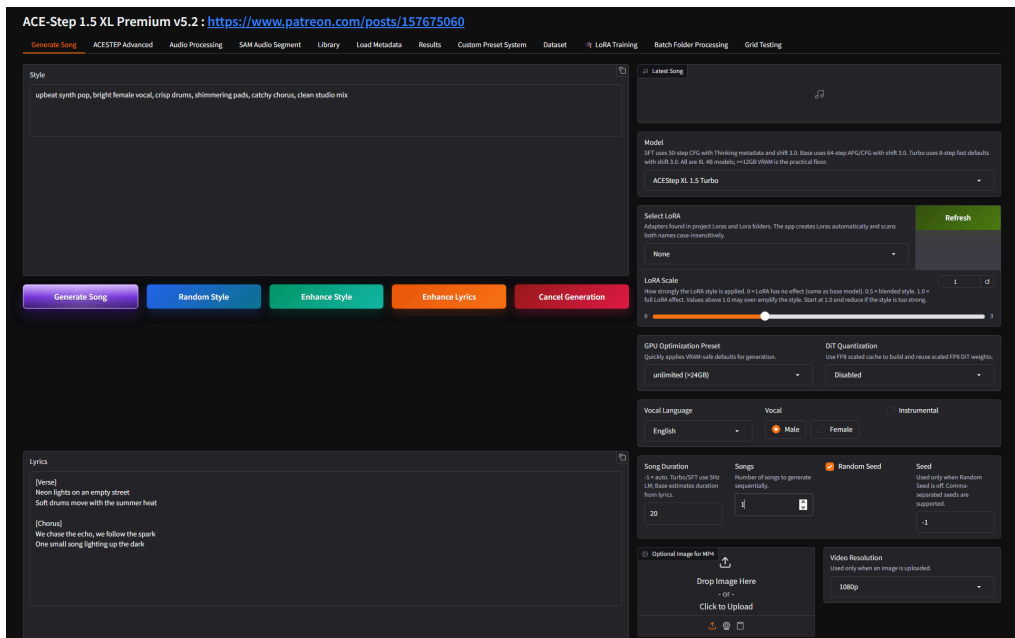
Windows first generation

3. Quick Song Generation

The **Generate Song** tab is the fast path. It exposes the controls most users need: style, lyrics, Wildcards, model, LoRA, GPU preset, quantization, language, vocal type, instrumental toggle, duration, count, seed, optional MP4 image, and video resolution.



Generate Song overview



Generate Song filled

Wildcards For Prompt Variation

ACE-Step XL 1.5 Premium v5.3 adds **Wildcards**. Write bracketed choices separated by pipes, such as [option A|option B|option C], and one option is picked when you generate. Wildcards can be used in the quick Generate Song Style field, the Advanced Music Caption field, and Lyrics.

ACE-Step 1.5 XL Premium v5.3 : <https://www.patreon.com/posts/157675060>

Generate Song

ACESTEP Advanced

Audio Processing

SAM Audio Segment

Library

Load Metadata

Results

Custom Preset

Style

cinematic [piano|guitar [clean|crunchy]], onscious melodic rap

Wildcards: use [option A|option B|option C]; one option is picked when you generate. Nested example:

cinematic [piano|guitar [clean|crunchy]] hook. Lyrics example: I feel [alive|ready|free] tonight. Lyric tags without |, like [Verse] and [Instrumental], stay unchanged.

Wildcards in Generate Song

Music Caption

Describe the target music: style, genre, instruments, mood, vocal delivery, and mix. In Repaint, describe what the selected replacement range should sound like.

Conscious melodic rap and modern cinematic pop-rap anthem, uplifting and humble, warm synth pads, punchy drums, deep 808 bass, subtle clean guitar textures, big melodic hook, soulful confident male vocal, polished wide stereo radio mix.

Wildcards: use [option A|option B|option C]; one option is picked when you generate. Nested example: cinematic [piano|guitar [clean|crunchy]] hook. Lyrics example: I feel [alive|ready|free] tonight. Lyric tags without |, like [Verse] and [Instrumental], stay unchanged.

Lyrics

Lyrics and section structure for the target result. In Repaint, provide the words or vocal direction for the selected replacement range.

[intro-medium]

[verse]

Woke up with the fire, put the weight on my back
No gold on my wrist, still I'm dressed in facts
Hands in the dirt, put the work on track
Built from the ground, never life on lack

I don't need the noise, I don't move for praise
I just keep it pure in a world that fades
Calm in the storm, let the heart stay brave
Planting good seeds, let the roots make waves

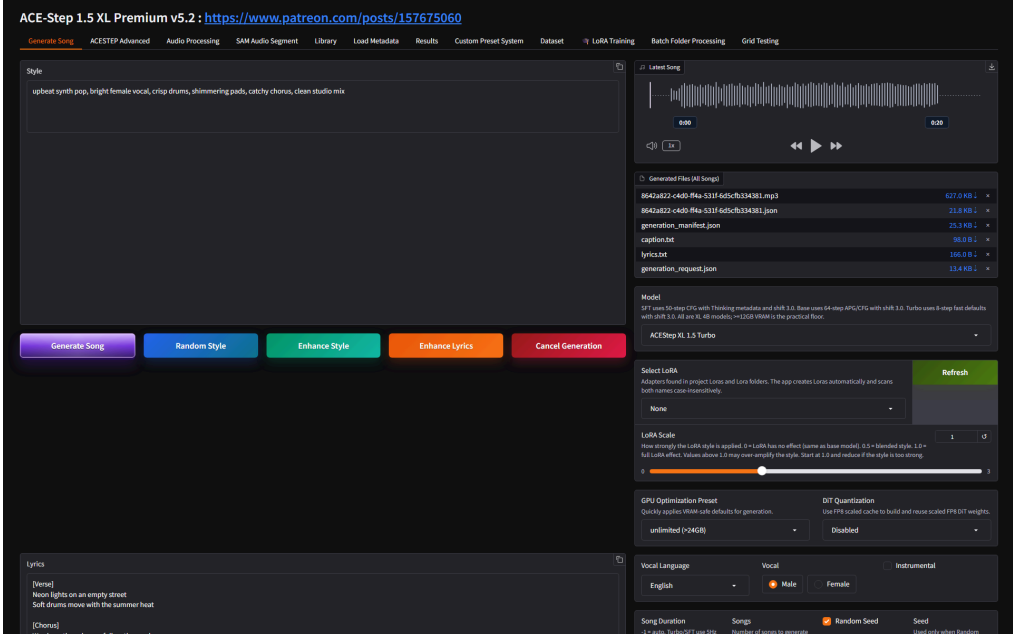
[chorus]

I came with peace, I came with light
Hands stay clean and my soul stays right
Work all day, still I move real kind
Head held low but I aim sky-high

Wildcards: use [option A|option B|option C]; one option is picked when you generate. Nested example: cinematic [piano|guitar [clean|crunchy]] hook. Lyrics example: I feel [alive|ready|free] tonight. Lyric tags without |, like [Verse] and [Instrumental], stay unchanged.

Wildcards in Advanced caption and lyrics

- Basic syntax: [piano|guitar|synth] picks one option when the job starts.
 - Nested syntax is supported. Example: cinematic [piano|guitar [clean|crunchy]] hook.
 - Lyrics can use the same pattern, for example: I feel [alive|ready|free] tonight.
 - Lyric tags without a pipe, such as [Verse], [Chorus], and [Instrumental], stay unchanged.
 - Do not enable Auto/Enhance Style or Auto/Enhance Lyrics when you need exact wildcard behavior; those improvement tools may rewrite the text and overwrite or remove wildcard expressions.
 - Batch folder processing uses the same Wildcards behavior, so batch jobs can vary instruments, moods, hooks, or lyric phrases across outputs without manually editing every run.
 - For repeatable comparisons, save the manifest/settings and lock other variables such as model, duration, and seed while testing wildcard choices.
- Write a concise Style prompt that describes genre, vocal character, instrumentation, production quality, tempo or mood, and mix target.
 - Write Lyrics with section tags such as [Verse] and [Chorus]. The included ACE_Step_Lyric_Generation_Instructions_For_LLMs.txt file can be given to an LLM to format lyrics or style prompts.
 - Optionally add Wildcards to Style or Lyrics when you want the app to choose between prompt variants automatically.
 - Select the Model. Start with **ACE-Step XL 1.5 Turbo** to verify the machine and workflow quickly.
 - Leave GPU Optimization Preset and DiT Quantization at safe defaults unless you are solving VRAM pressure or repeating a known workflow.
 - Set Song Duration and Songs. The demo run used 20 seconds and 1 song.
 - Use Random Seed while exploring. When a promising result appears, uncheck Random Seed and keep the seed so future edits stay comparable.
 - Click **Generate Song** and monitor the Status field plus the terminal window.



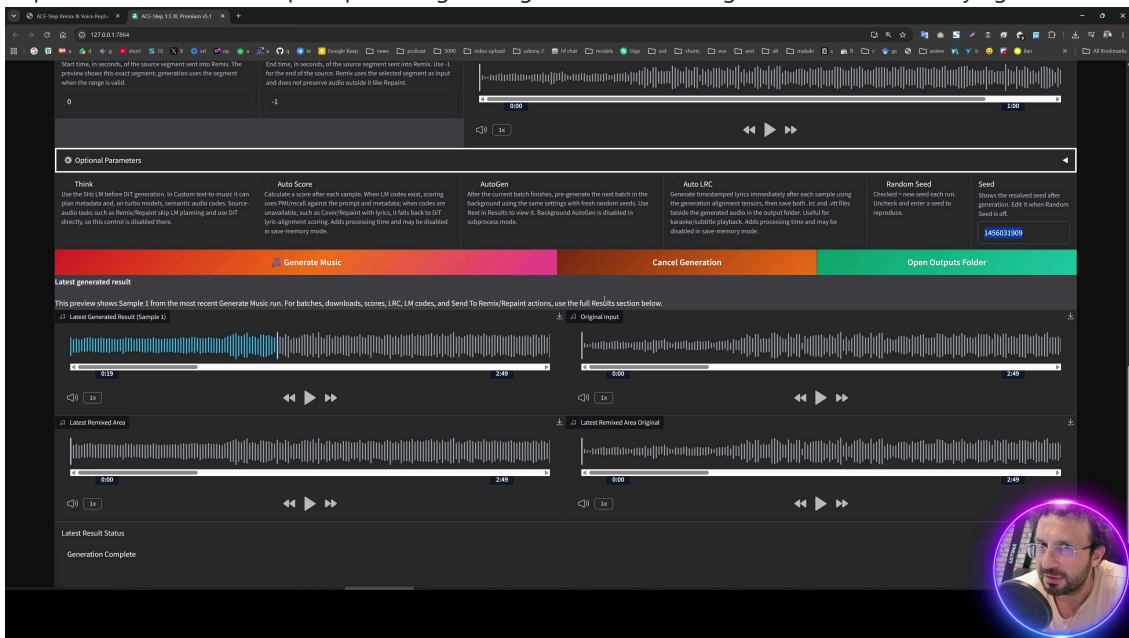
Demo generation result

Useful quick-tab buttons:

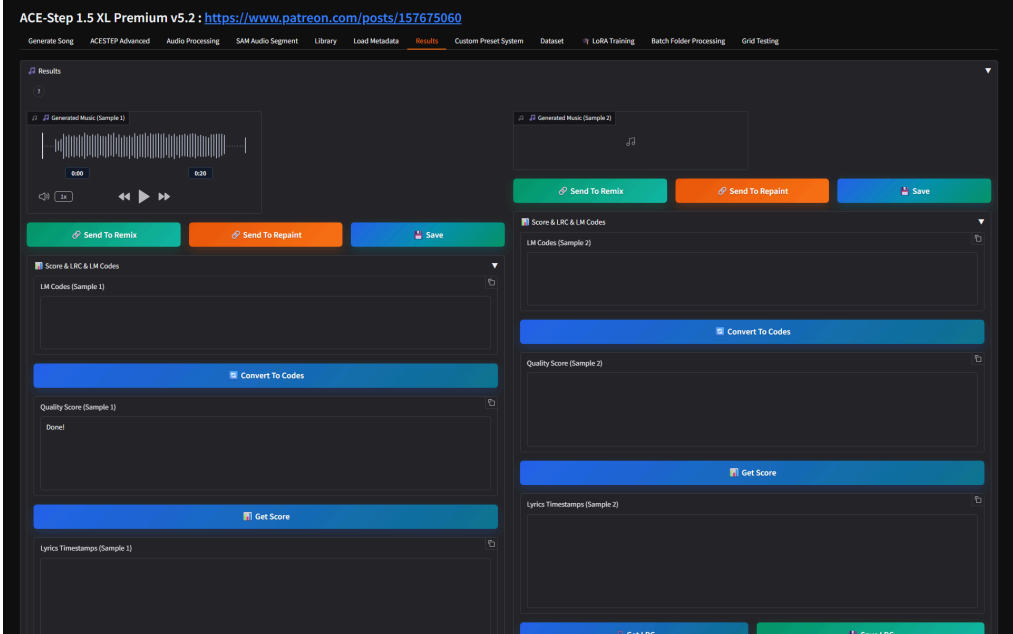
- **Random Style** creates a starting style prompt.
- **Enhance Style** improves the style prompt.
- **Enhance Lyrics** improves lyric structure or phrasing before generation.
- **Cancel Generation** stops a run from the UI.
- **Open Outputs Folder** opens the app's outputs directory where audio, manifests, lyrics, captions, sessions, and metadata are saved.

4. Results, Seeds, And Reuse

The tutorial stresses generating repeatedly until you have a good base result, then locking the seed and making controlled edits. This is especially important for remix and repaint work, where small prompt or range changes can be tested against the same underlying random state.



Seed and remix discussion



Results after generation

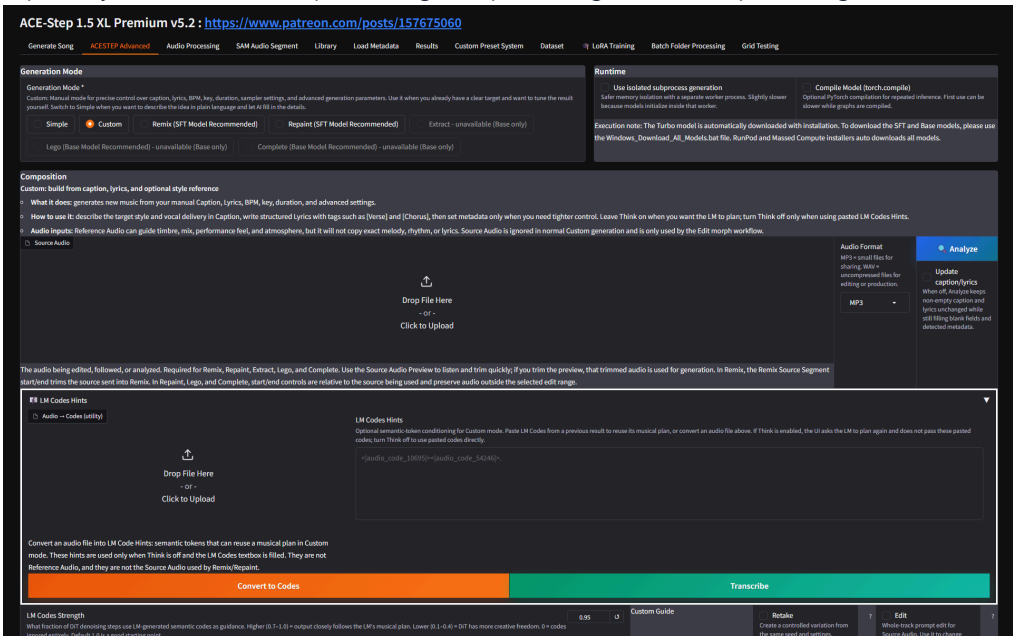
- **Send To Remix** loads the generated song as Source Audio and prepares the advanced Remix workflow.
- **Send To Repaint** loads the generated song and prepares a repaint range workflow.
- **Convert To Codes** reuses the musical plan as LM Codes Hints in compatible Custom workflows.
- **Get Score** and **Get LRC** create quality-score and lyric-timestamp artifacts.
- **Save** and **All Generated Files** expose generated files for download or reuse.

Seed workflow:

1. Keep Random Seed on while searching for a usable base result.
2. When the result is close, copy or keep the seed shown by the UI.
3. Turn Random Seed off.
4. Change one word, one range, or one strength setting at a time.
5. Compare outputs against the locked seed.

5. Advanced Generation Modes

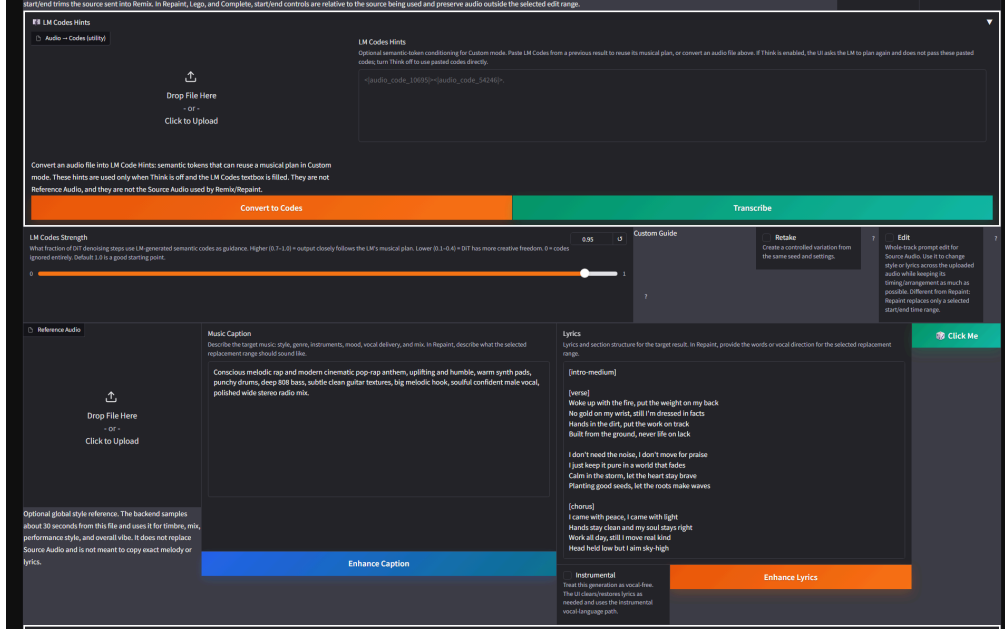
The **ACESTEP Advanced** tab is the full workstation. It exposes generation mode, runtime settings, source/reference audio, LM code utilities, advanced prompts, Wildcards in Music Caption/Lyrics, metadata, sampler settings, output settings, and batch processing.



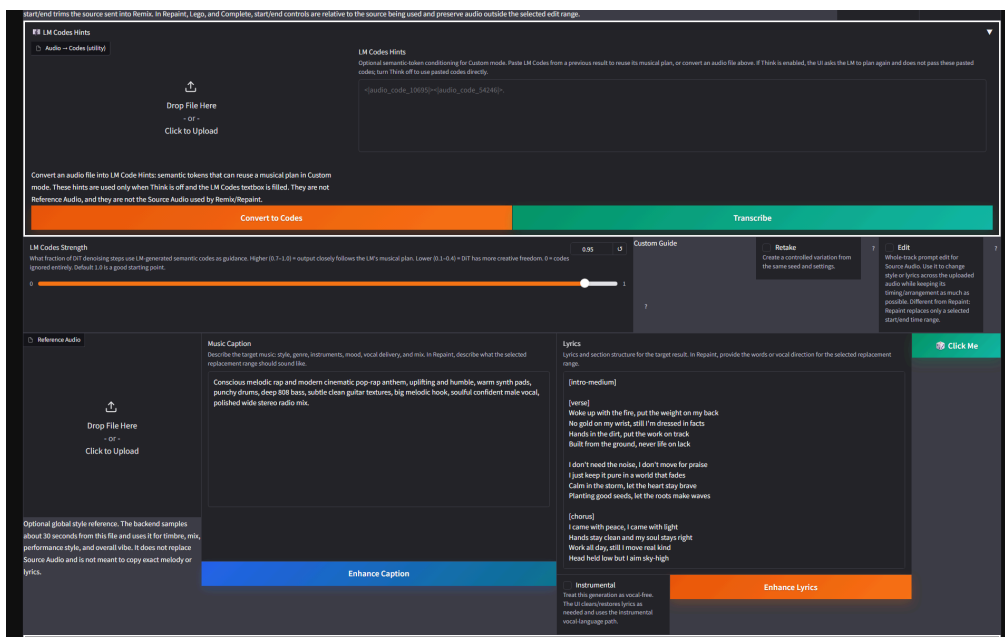
Advanced overview

Generation modes:

- **Simple**: plain-language generation when you want the app to fill many details.
- **Custom**: precise manual control over caption, lyrics, BPM, key, time signature, language, duration, and advanced settings.
- **Remix**: create a new version from source audio. The video recommends SFT for Remix and keeping the same lyrics for best results.
- **Repaint**: replace or modify a selected time range while preserving the rest of the source audio.
- **Extract**: use ACE-Step extraction or stem workflows where the selected model supports it.
- **LEGO**: add a new stem to existing audio, such as adding a guitar stem to a track.
- **Complete**: continue or complete source audio when the selected model supports it.



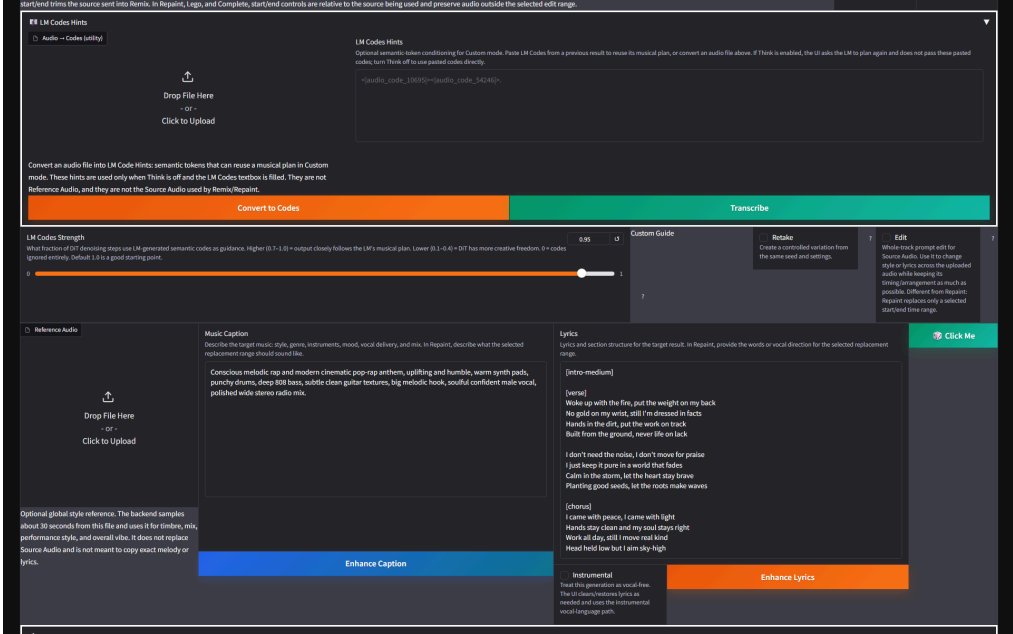
Advanced source audio



Advanced generation controls

Important advanced controls:

- **Source Audio** is required for Remix, Repaint, Extract, LEGO, and Complete.
- **Reference Audio** guides timbre, mix, performance feel, and atmosphere; it is not meant to copy exact melody, rhythm, or lyrics.
- **Analyze** can fill or update caption/lyrics/metadata from source audio.
- **BPM Auto, Key Auto, TimeSig Auto, Language Auto, and Duration Auto** let the model infer metadata.
- **Think** enables LM planning. Turn it off only when deliberately using pasted LM Codes Hints.
- **Wildcards** in Music Caption and Lyrics are expanded at generation time and also work when the same prompt fields are used for batch folder processing.
- **Retake** creates controlled variation from the same seed/settings.
- **Edit** changes the whole uploaded source using source and target prompts.
- **Auto Score, AutoGen, and Auto LRC** can create score, metadata, and lyric timing outputs during generation.

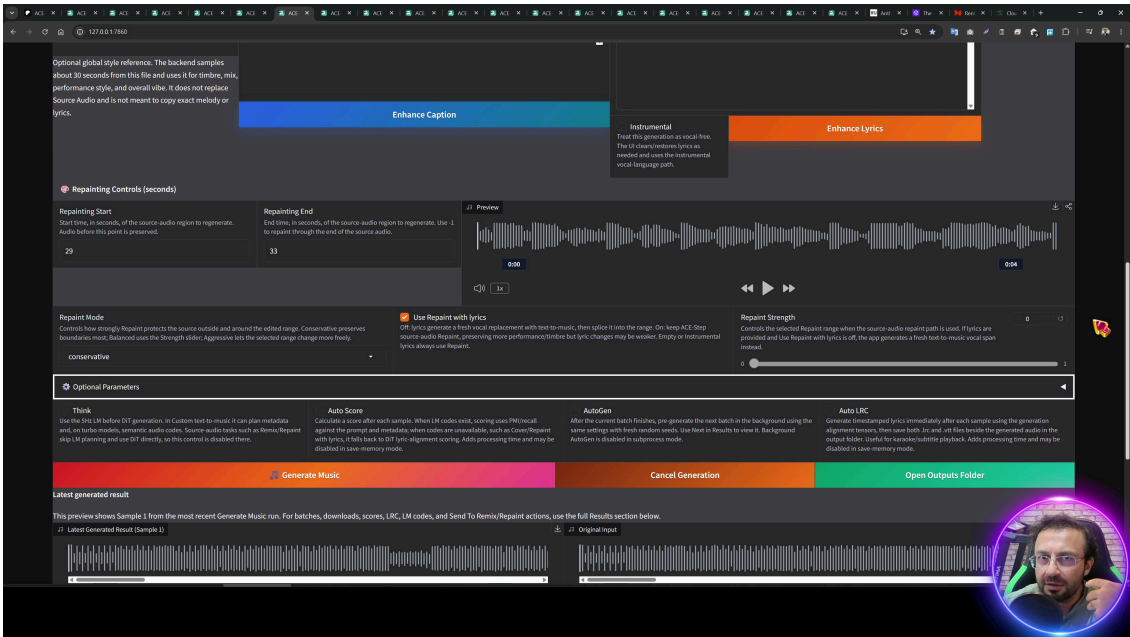


Engine settings

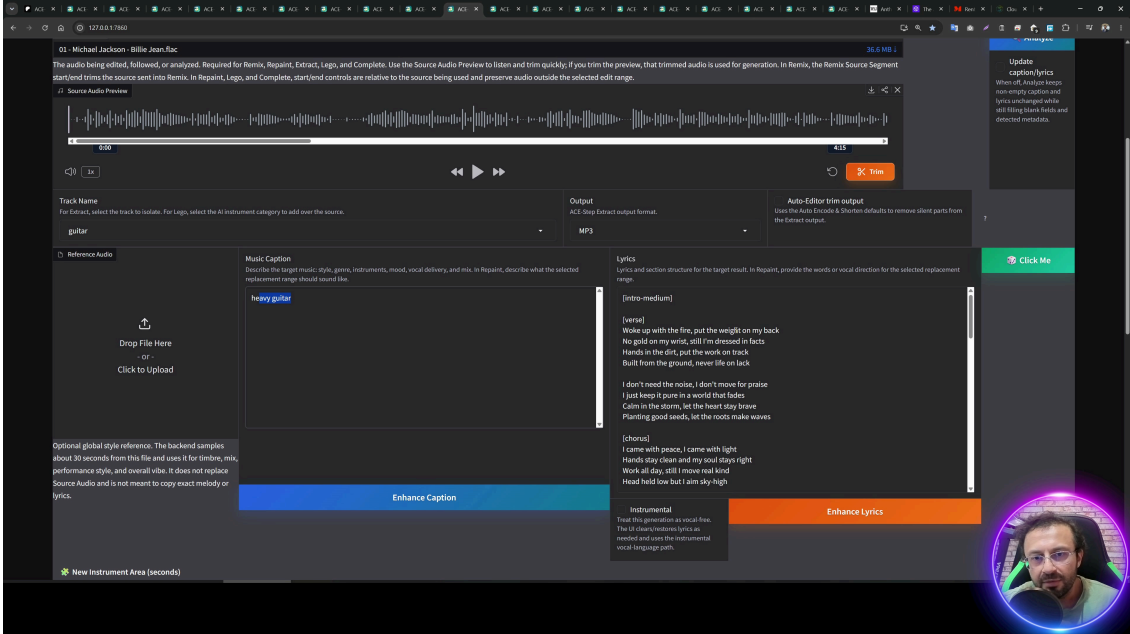
Engine settings include GPU tier, checkpoint file, main model path, device, VAE, 5Hz LM model/backend, Flash Attention, CPU offload, compile, DiT quantization, LoRA path/folder, LoRA scale, inference steps, sampler, DCW, ADG, MP3 bitrate/sample rate, normalization, fades, LM temperature, top-k/top-p, negative prompt, and LM code settings. Leave these at defaults until you have verified a basic generation.

6. Remix, Repaint, Extract, LEGO, And Auto-Editor Features

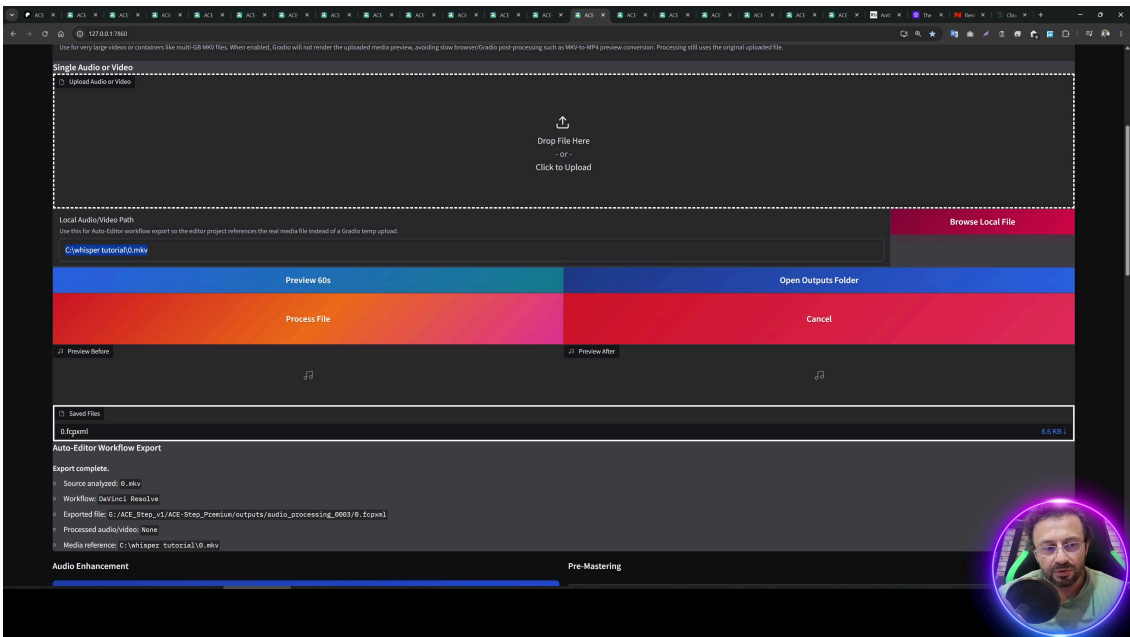
The first part of the video demonstrates feature outcomes before the installation section. These are not separate apps; they are modes and panels inside the same ACE-Step interface.



Remix demo



Extract and LEGO demo

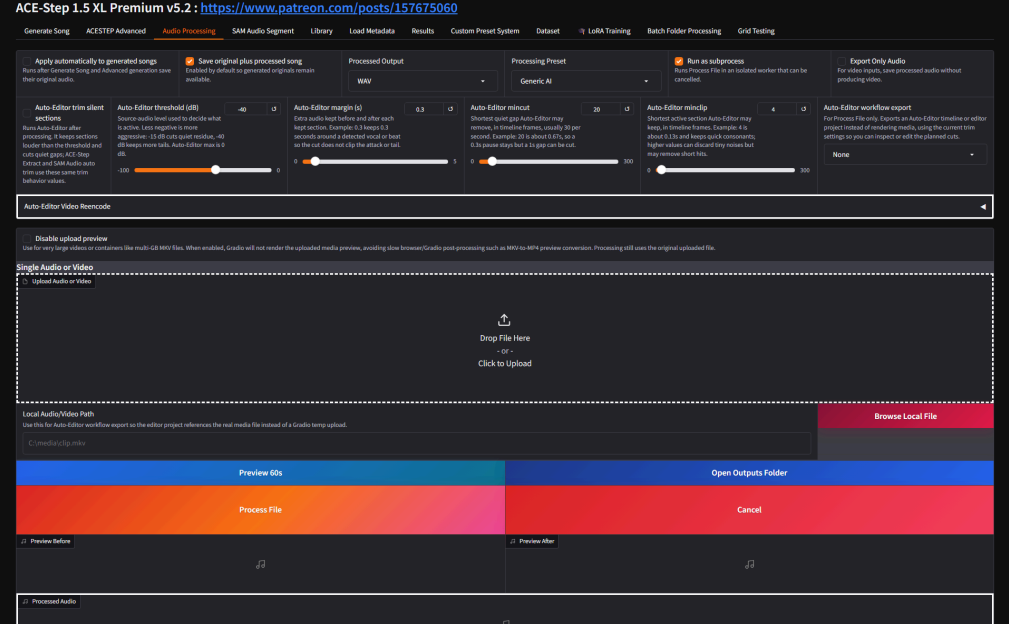


Auto-Editor demo

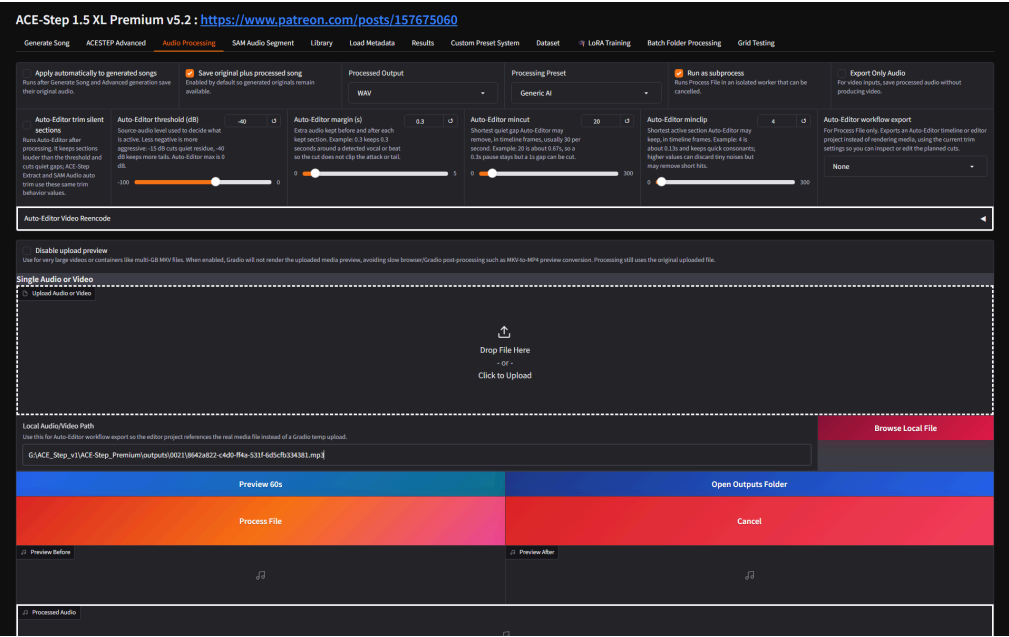
- For **Remix**, upload the source song, keep the same lyrics, use SFT when available, and start with the default remix strength before changing one variable at a time.
- For **Repaint**, set Repainting Start and End carefully, preview the selected range, then choose repaint mode and strength.
- For **Extract**, choose Track Name or Extract All Stems. Batch folder processing can extract from multiple files.
- For **LEGO**, choose or describe the new stem to add and use source audio as the foundation.
- For **Auto-Editor**, set threshold, margin, mincut, and minclip. Workflow export can produce an editor timeline instead of rendering media.

7. Audio Processing

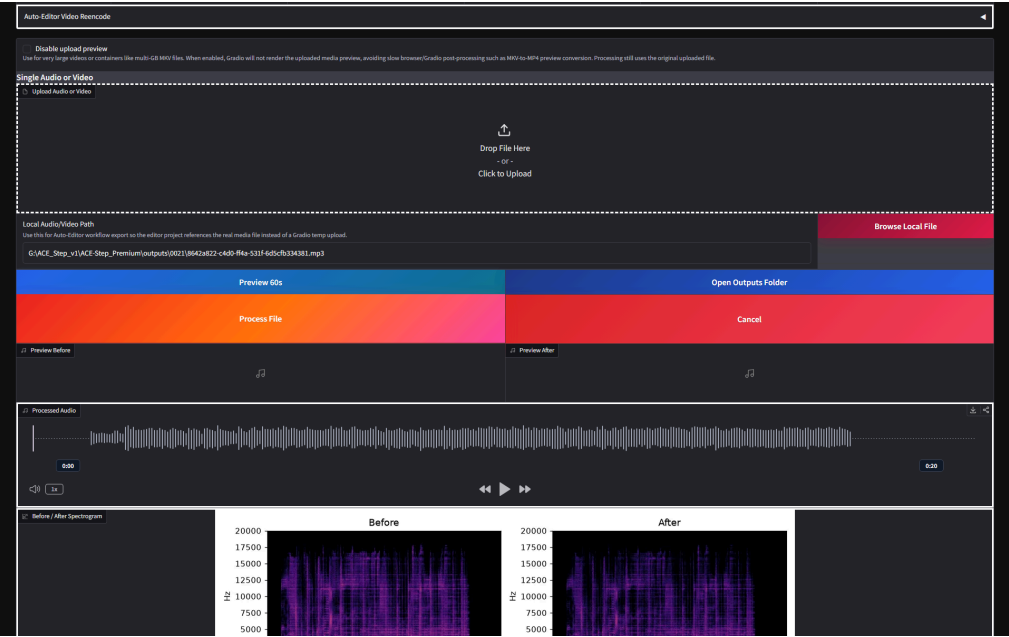
Audio Processing is used on uploaded or local audio/video and can also be applied automatically to generated songs. It includes format output, Auto-Editor trimming, video re-encode controls, audio enhancement stages, pre-mastering stages, DiffPitcher, and batch folder processing.



Audio Processing overview



Generated song loaded for processing

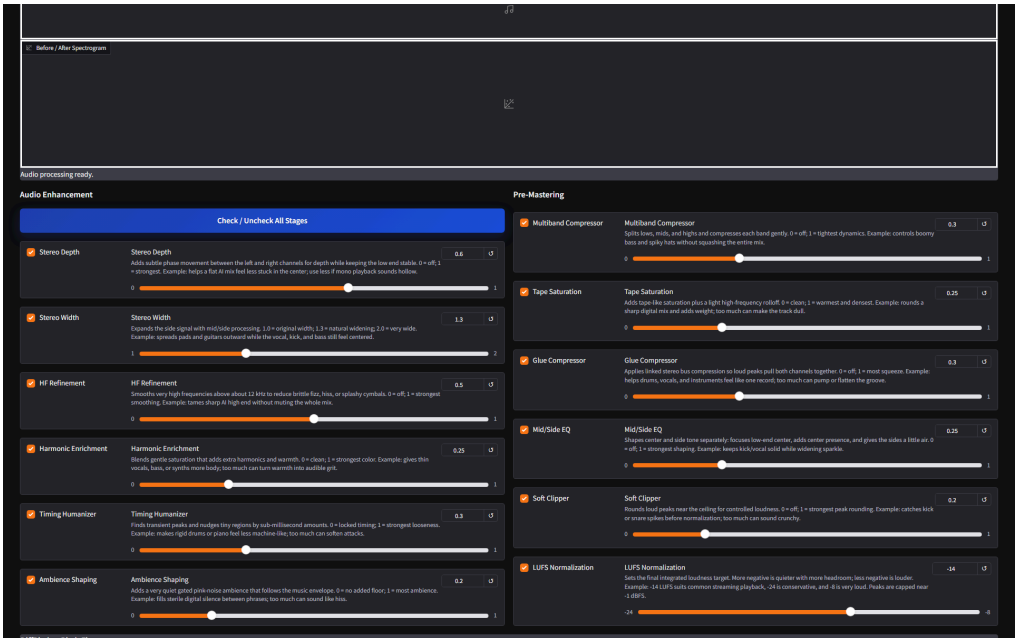


Audio Processing result

Core Audio Processing controls:

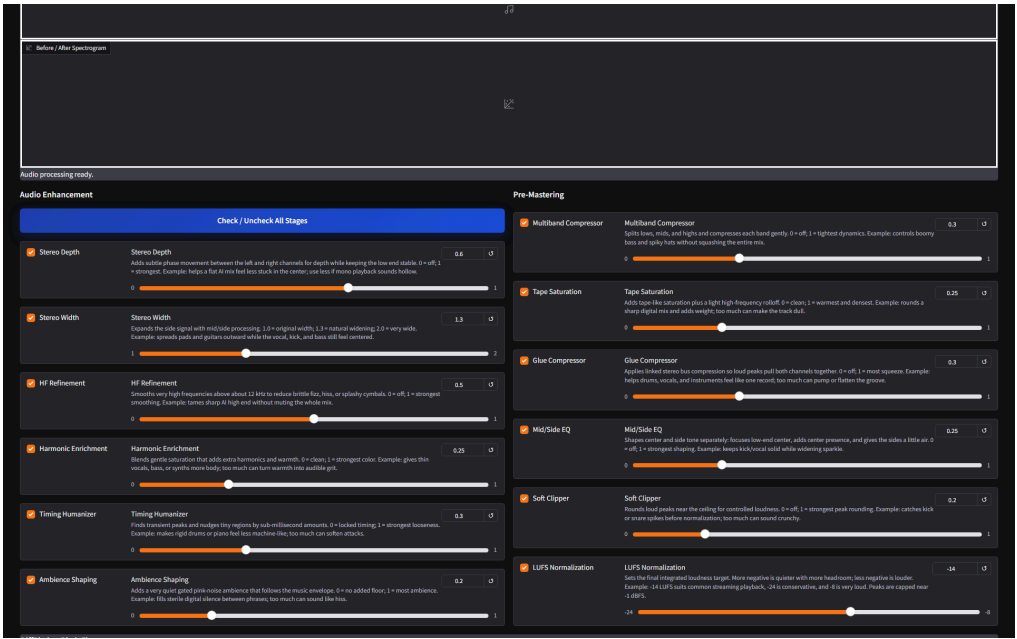
- **Apply automatically to generated songs** runs the processing chain after generation.
- **Save original plus processed song** keeps an untouched copy beside the processed copy.

- **Processed Output** selects WAV/MP3 or another output format.
- **Processing Preset** sets a preset chain before you tune individual stages.
- **Run as subprocess** isolates processing so cancellation and memory cleanup are safer.
- **Export Only Audio** extracts processed audio from video inputs.
- **Auto-Editor trim silent sections** removes quiet/silent segments using threshold, margin, mincut, and minclip.
- **Auto-Editor workflow export** exports an editing timeline/workflow instead of only rendering media.
- **Disable upload preview** helps with very large MKV or multi-GB media.



Audio Enhancement and Pre-Mastering

Audio Enhancement includes Stereo Depth, Stereo Width, HF Refinement, Harmonic Enrichment, Timing Humanizer, and Ambience Shaping. Pre-Mastering includes Multiband Compressor, Tape Saturation, Glue Compressor, Mid/Side EQ, Soft Clipper, and LUFS Normalization.



DiffPitcher controls

DiffPitcher is for isolated vocals that sing the wrong notes. Use a guide vocal or MIDI score for the same phrase. The tutorial text in the UI warns that this is not for copying another singer or another song.

8. SAM Audio Segment

SAM Audio Segment is a heavier but more flexible segmentation system. It can extract target audio from a prompt, save the residual/remaining audio, process video inputs, use explicit span anchors, and run batch prompt lists separated by semicolons.

5. Use explicit span anchor to provide positive/negative time anchors as JSON.
6. Choose a VRAM preset and candidate count that match the GPU. Higher candidate counts can improve quality but cost runtime and VRAM.
7. Enable Save remaining audio when you need both the extracted target and the residual track.

9. Library, Metadata, Presets, Dataset, And Training Pages

The remaining app tabs are operational pages. They help you find previous generations, restore metadata, manage presets, inspect datasets, and train adapters.

ACE-Step 1.5 XL Premium v5.2 : <https://www.patreon.com/posts/157675060>

Generate Song ACESTEP Advanced Audio Processing SAM Audio Segment **Library** Load Metadata Results Custom Preset System Dataset LoRA Training Batch Folder Processing Grid Testing

[Refresh Library](#) Generated Songs by Day 18 June 2026

Recent Generations

Created	Title	Duration	Model	Format	Score
16 June 2026, 11:24 PM	epic	-1	ACEStep_1.5_XL_SFT_BF16	mp3	Done!

epic

- Created: 16 June 2026, 11:24 PM
- Model: ACEStep_1.5_XL_SFT_BF16
- Duration: -1
- Format: mp3
- Score: Done!
- Audio: G:/ACE_Step_v1/ACE_Step_Premium/outputs/8628/7864462-7814-2484-7722-E77c3d884e23.mp3
- Metadata: G:/ACE_Step_v1/ACE_Step_Premium/outputs/8628/7864462-7814-2484-7722-E77c3d884e23.json
- Manifest: G:/ACE_Step_v1/ACE_Step_Premium/outputs/8628/generation_manifest.json

epic

Selected Song

0:00 2:40

Lyrics

[intro-medium]

[verse]
Wake up with the fire, put the weight on my back
No gold on my wrist, still I'm dressed in facts
Hands in the dirt, put the work on track
Buck from the ground, never let me back

I don't need the noise, don't move for praise
I just keep it pure in a world that fades
Calm in the storm, let the heart stay brave
Planting good seeds, let the roots make waves

[chorus]
I came with peace, I came with light
Hands stay clean and my soul stays right
Work all day, still I move real kind
Head held low but I aim sky high

No big talk, I don't need that crown
Strong stand tall, keep it humble now
Brick by brick, yeah I built this sound

Full Metadata

```
1 {  
2   "task_type": "cover",  
3   "instruction": "Generate audio semantic tokens based on the given conditions:",  
4   "reference_audio": null,  
5   "src_audio":  
6     "G:/Users/Vathan/AppData/Local/Temp/igradio/36d37876d7a12b46cfc7beaa0f6e338e1876da37  
7     1241-d589-f16e-ca6d8ba16a51.mp3"  
8 },  
9   "audio_codes": "",  
10  "caption": "epic",  
11  "prompt": "epic"
```

Library

ACE-Step 1.5 XL Premium v5.2 : <https://www.patreon.com/posts/157675060>

Generate Song ACESTEP Advanced Audio Processing SAM Audio Segment **Library** **Load Metadata** Results Custom Preset System Dataset LoRA Training Batch Folder Processing Grid Testing

generation_manifest.json

Drop File Here
or
Click to Upload

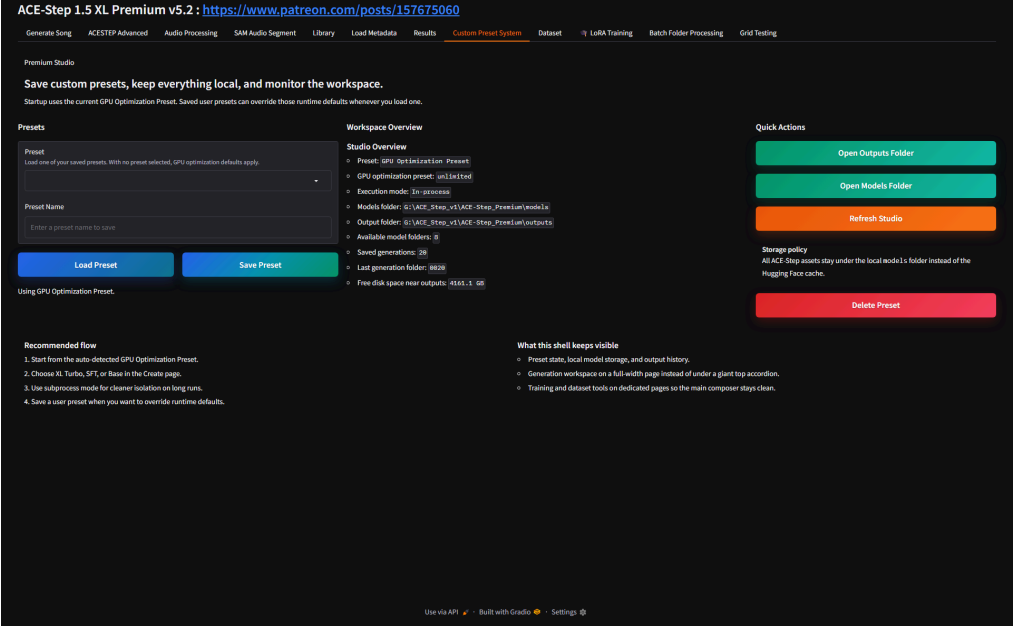
File Path
G:/ACE_Step_v1/ACE_Step_Premium/outputs/.../generation_manifest.json

[Browse](#) [Load Metadata](#)

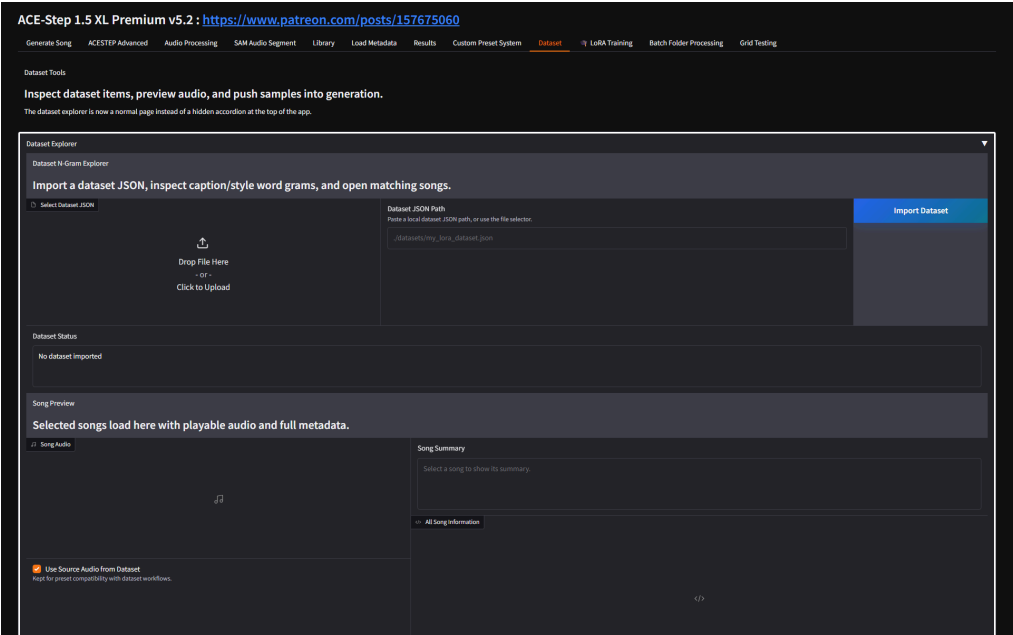
Status

Use via API Built with Gradio Settings

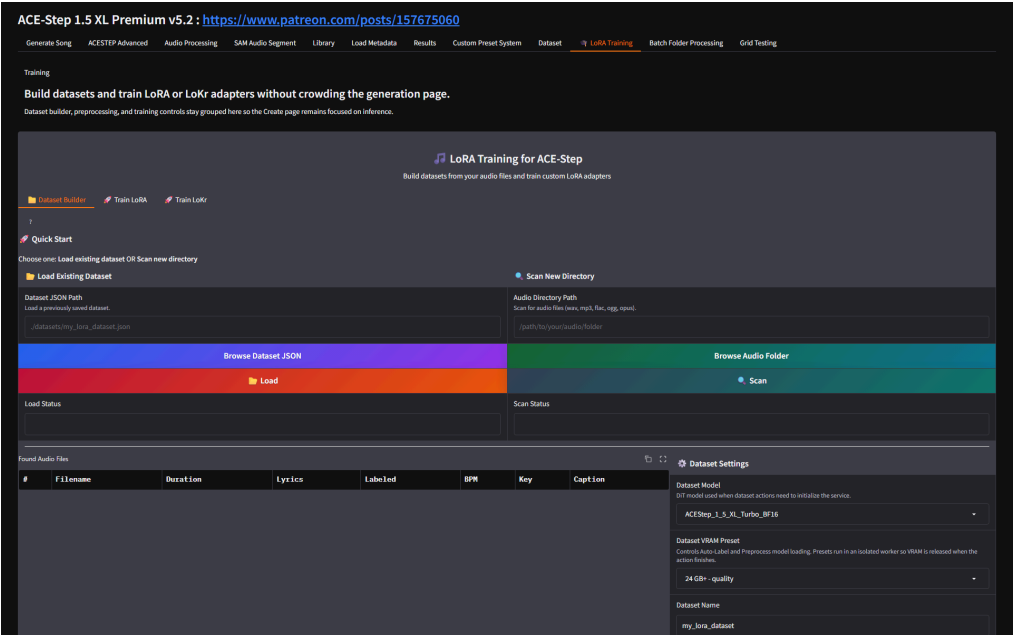
Load Metadata



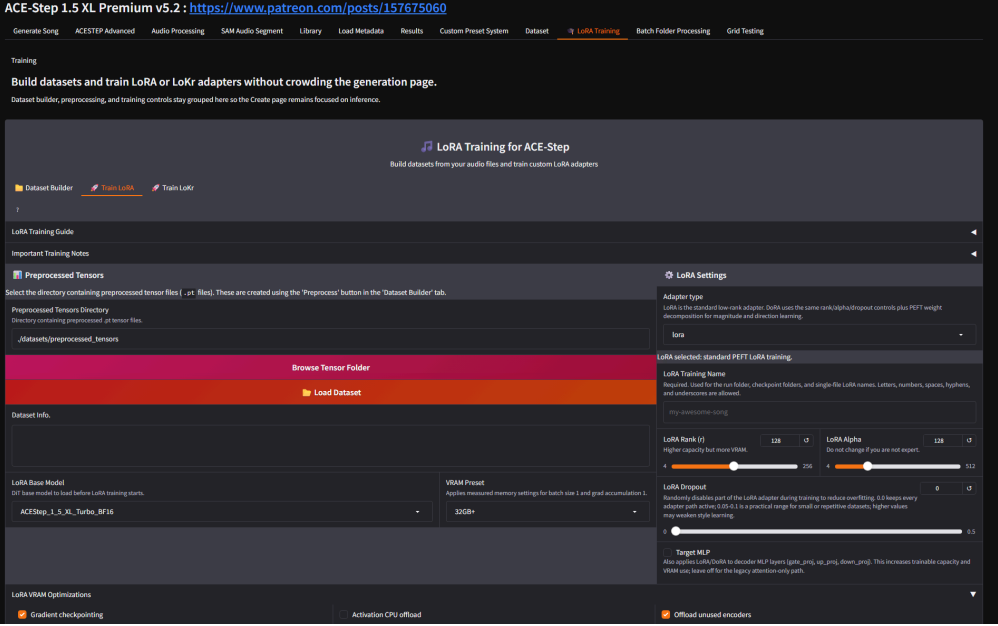
Custom Preset System



Dataset browser



LoRA Dataset Builder

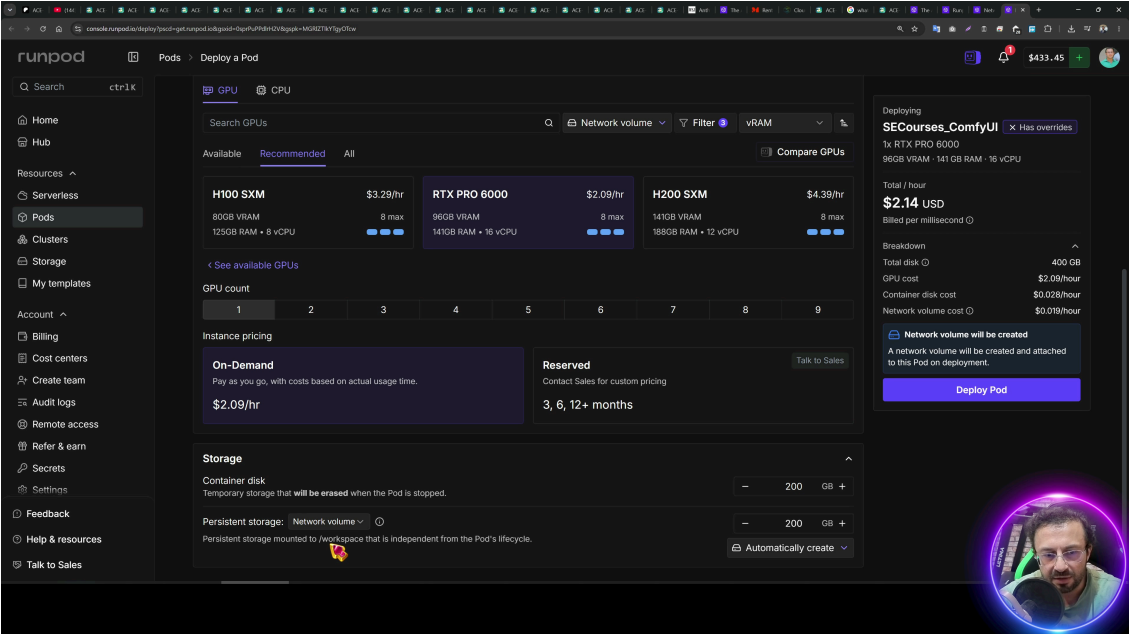


Train LoRA

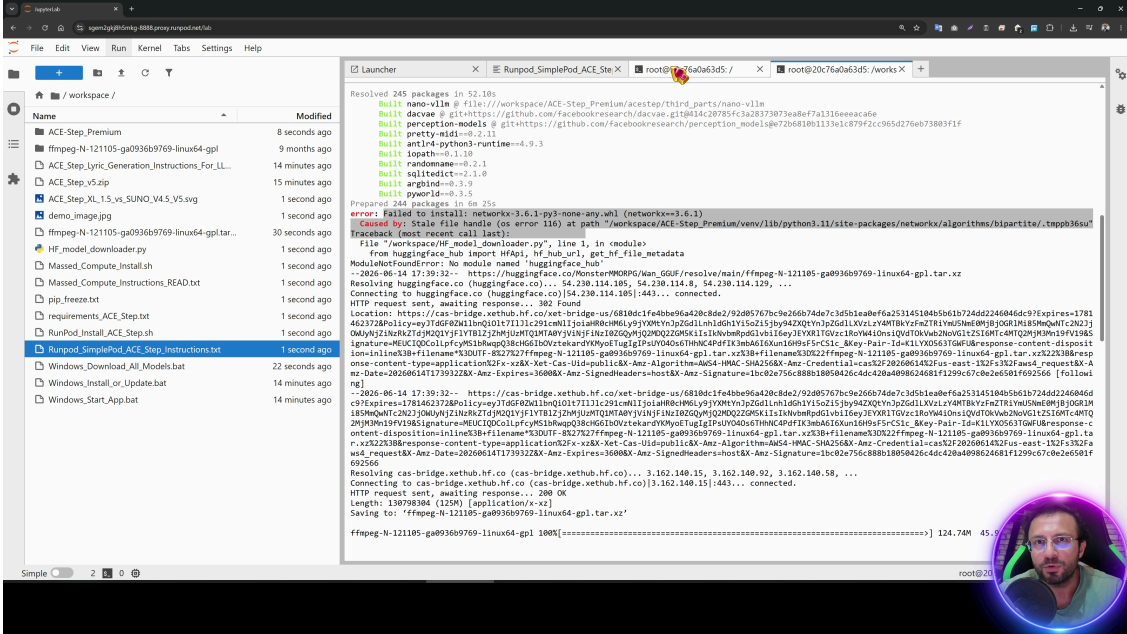
- Use **Library** when you want to find a past song by day and inspect its lyrics/metadata.
- Use **Load Metadata** when you have a `generation_manifest.json` and want to restore a generation into the UI.
- Use **Custom Preset System** to persist frequently used model, GPU, LoRA, audio, and generation defaults across sessions.
- Use **Dataset Builder** to scan audio, auto-label captions/lyrics/BPM/key/time signature, review samples, and save a dataset JSON.
- Use **Preprocess** before training; it creates tensor files for faster LoRA or LoKr training.
- Use **Train LoRA** or **Train LoKr** only after preparing a clean dataset. The video states LoRA training is intended for a separate deeper tutorial.

10. RunPod Deployment

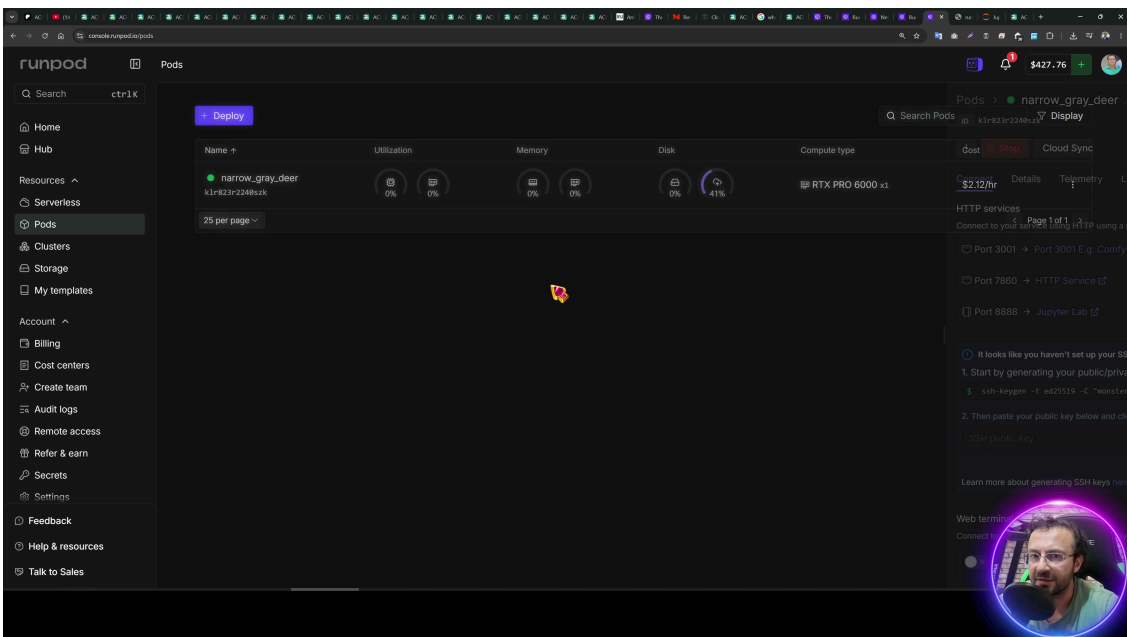
The RunPod chapter focuses on persistent network storage, GPU/region selection, unreliable installs, Gradio live URLs, nvitop monitoring, output downloads, and safe termination.



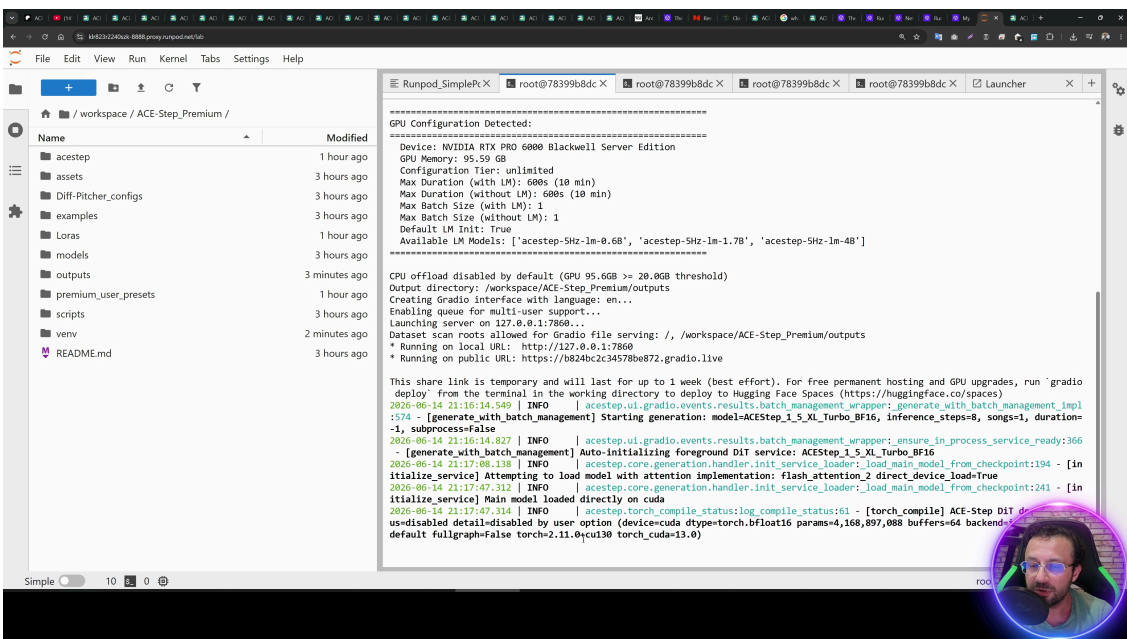
RunPod storage



RunPod install



RunPod Gradio services

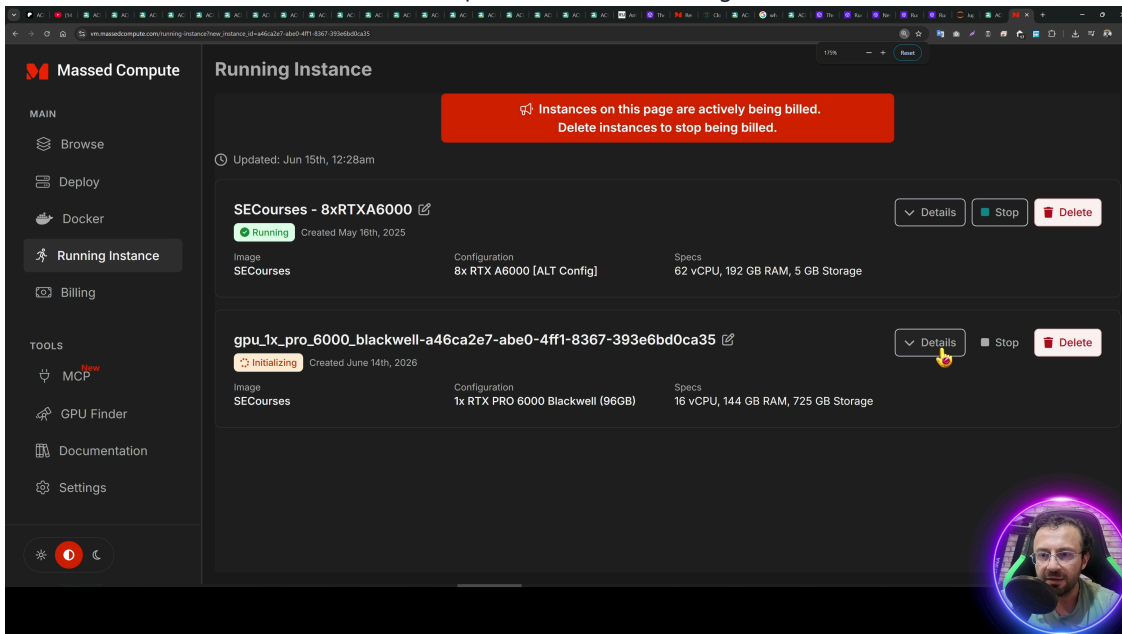


RunPod monitoring

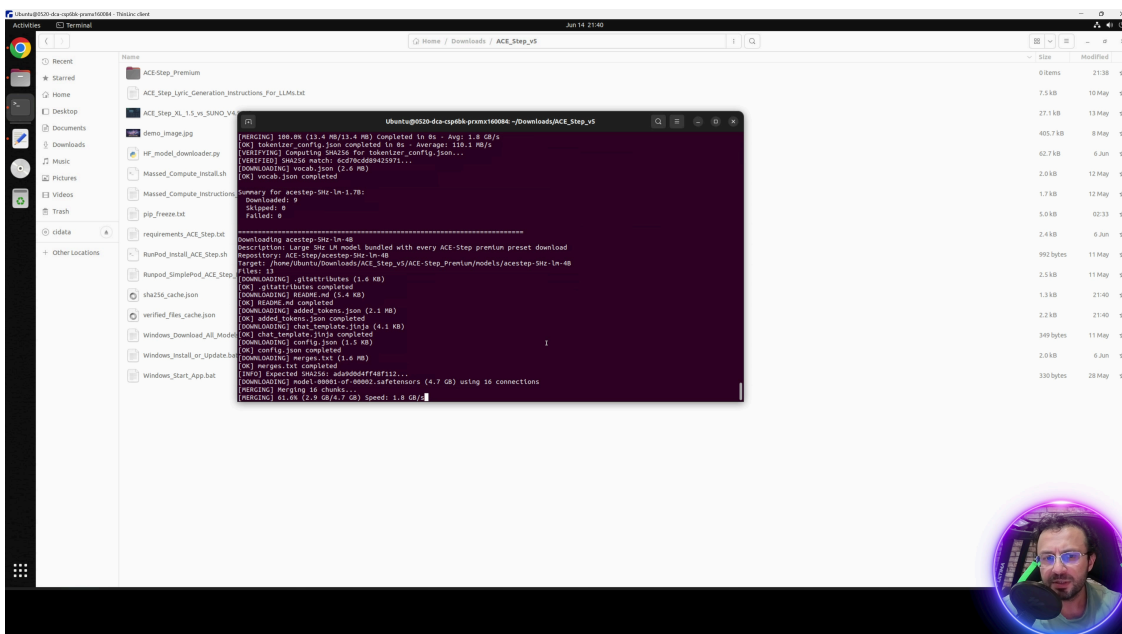
1. Create persistent network storage in the same region as the GPU you intend to rent.
2. Deploy the pod/template with the storage mounted. Choose a GPU with enough VRAM for the selected model and quality target.
3. Run the installer. If RunPod throws an OS/server error, run the installer again; it should resume from completed work.
4. If installation stalls from excessive parallelism, delete the virtual environment, lower installer thread count as shown in the video, and rerun.

11. Massed Compute Deployment

The Massed Compute chapter is similar to the Linux/cloud workflow, but the tutorial emphasizes faster disk performance and lower friction compared with RunPod. The tradeoff called out in the video is the lack of the same persistent network storage flow.



Massed Compute GPU selection

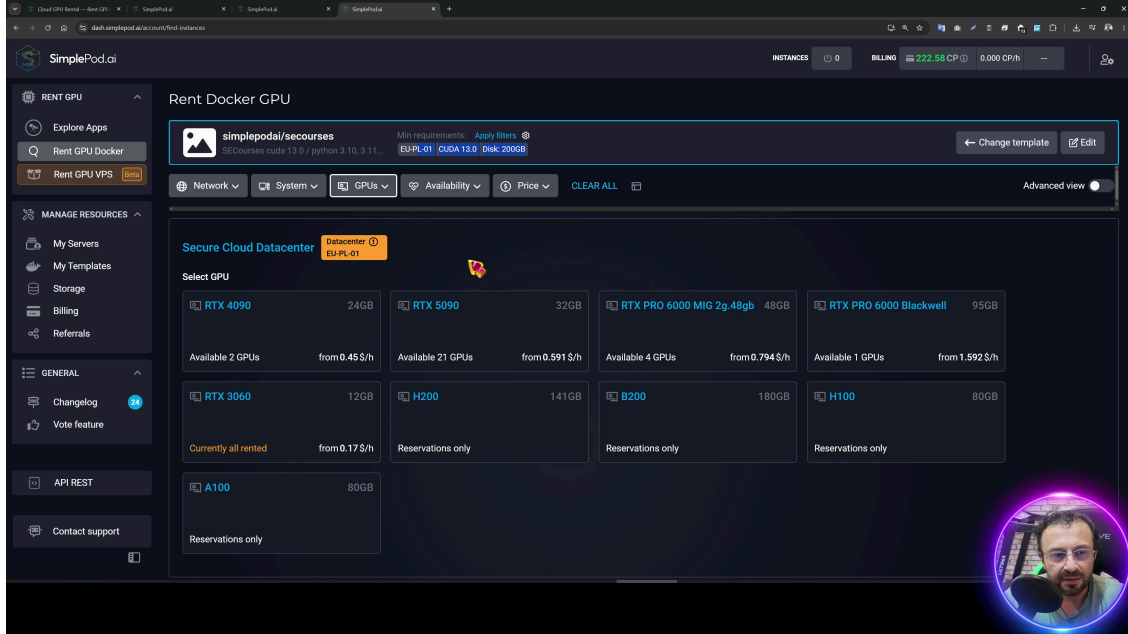


Massed Compute install

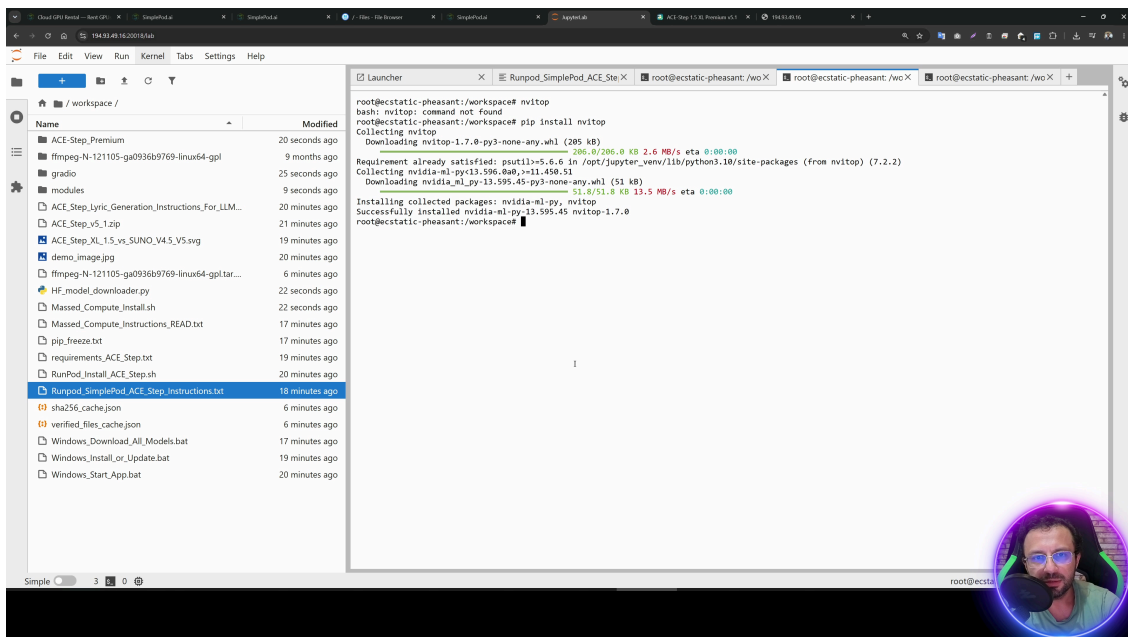
1. Choose the creator category and the SECourses image when following the video workflow.
2. Select a GPU appropriate for ACE-Step XL 1.5. The tutorial mentions RTX Pro 6000 and RTX 5090 class GPUs.
3. Upload the ACE-Step ZIP to Downloads, extract it, open `Massed_Compute_Instructions_READ.txt`, and copy the install command.
4. Open a terminal inside the extracted ACE-Step folder and run the command from that location.
5. Start ACE-Step and use the Gradio live URL. If Gradio live shows a transient error, refresh the page.
6. Back up large outputs or model/data folders to Hugging Face, Google Drive, OneDrive, or another storage service if you need to recreate the machine later.

12. SimplePod Deployment

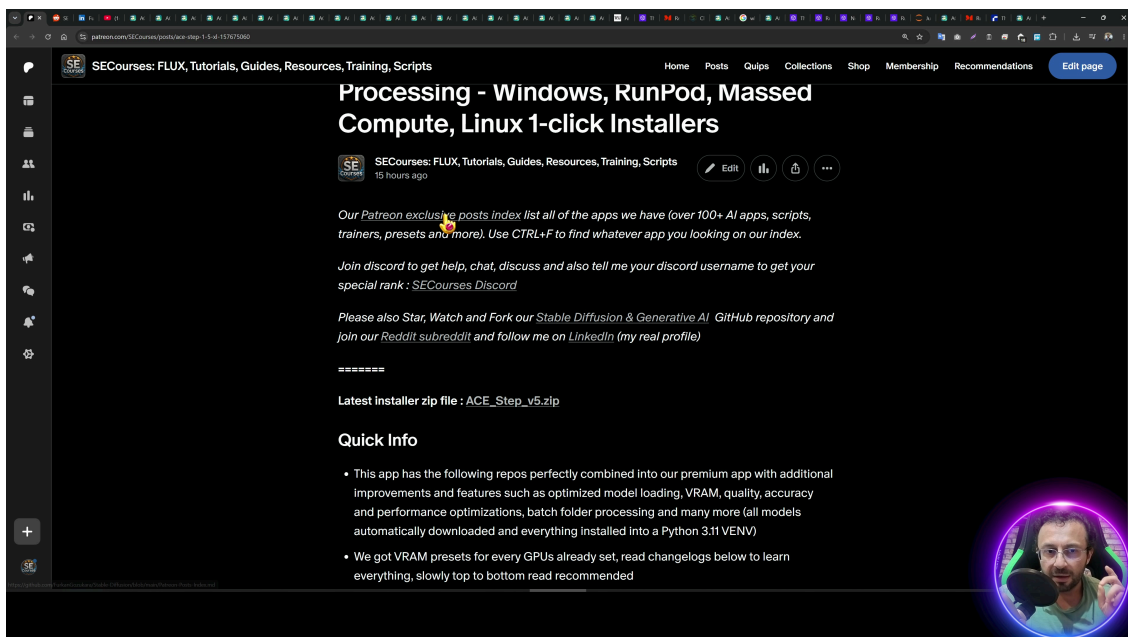
The SimplePod chapter uses the RunPod/SimplePod instruction file and shows a persistent-storage flow that resembles RunPod. The tutorial demonstrates starting, generating, monitoring, stopping, and resuming from the same storage volume.



SimplePod setup



SimplePod generation



SimplePod resume

1. Register, add credits, and create/use persistent storage as shown in the instruction file.
2. Open the template link, attach the storage volume, choose a GPU, and run the machine.
3. Use the JupyterLab or console link to run the installer/start commands from the workspace.
4. If the Gradio live page throws a first-click error, refresh or click again after the page is fully loaded.

5. Install `nvitop` when you want GPU/VRAM visibility: `pip install nvitop`, then run `nvitop`.
6. To resume, reuse the template link, attach the same volume, select a GPU, start the machine, and run the app start command again.
7. Stop or terminate compute and remove storage when finished to avoid unwanted billing.

13. Troubleshooting And Best Practices

- If Gradio errors after opening, refresh the browser or click again after the page finishes loading.
- If RunPod installer errors, rerun the installer. If the virtual environment is corrupt, delete only the virtual environment and run the installer again.
- If first generation is slow, wait for model load and monitor the terminal or `nvitop`.
- For Remix quality, use SFT when available, keep lyrics aligned with the source, generate until you get a good base, then lock the seed.
- For very large videos, use Disable upload preview in Audio Processing or supply a local path when available.
- For prompt variation in many runs, use Wildcards in Style, Music Caption, or Lyrics. Batch folder processing can use the same wildcard syntax to vary each batch output. Keep Auto/Enhance Style and Auto/Enhance Lyrics disabled if you want exact wildcard expressions preserved.
- Save presets and keep `generation_manifest.json` with outputs.
- For VRAM pressure, use GPU Optimization Preset, quantization, offload controls, smaller duration/count, lower SAM candidates, or a larger GPU.
- Stop/terminate cloud compute and delete storage volumes when the tutorial work is finished.