AI-Driven Reddit Story Video Pipeline

Complete Guide

AI Automated Reddit Story-to-Video Pipeline

1. Summary of Chapters

2. Introduction

After exploring AI automation and its vast potential, I discovered significant limitations across existing video generation workflows. Most solutions fall into these problematic categories:

1. **Free workflows with costly subscriptions** — While the workflow itself is free, essential nodes and API calls require monthly subscriptions.
2. **Free but broken workflows** — No error handling, documentation, or support when issues arise.
3. **Free workflows with poor quality** — Free APIs that produce low-quality, unengaging content.
4. **Limited free usage** — Severely restricted usage limits (often only 3 uses before requiring payment).
5. **Fully paid solutions** — Both workflow and APIs require payment.
6. **Local-only solutions** — Free workflows using local APIs that demand high-performance hardware.

This guide presents the result of two months of intensive research, testing every available alternative to create a **completely free, unlimited AI video generation automation** that produces quality content while remaining flexible and accessible to users with modest hardware specifications.

3. Popular Paid API Services Comparison

Service Category	Provider	Key Features	Starting Price	Use Case
Text Generation	OpenAI GPT-4	Advanced language models, coding assistance	~$0.03/1K tokens	Text generation, coding, analysis

Service Category	Provider	Key Features	Starting Price	Use Case
	Claude (Anthropic)	Long context, analytical reasoning	~$0.25/1K tokens	Document analysis, complex reasoning
	DeepSeek	Code-specialized models	~$0.14/1M tokens	Software development, debugging
	Qwen (Alibaba)	Multilingual capabilities	~$0.50/1M tokens	International content, translation
Image Generation	OpenAI GPT-Image-1	Professional-grade visual generation	~$0.04/image	Custom visual content
	Midjourney	Artistic image creation	$10/month	Creative artwork, marketing visuals
	DALL-E 3	Integrated with ChatGPT	$0.040-0.120/image	General purpose image generation
	Stability AI	FLUX.1 Pro, multiple models	$0.05/image	High-quality image generation
Text-to-Speech	ElevenLabs	Premium voice cloning, multilingual	$5/month	Voice overs, audiobooks
	OpenAI TTS	$0.015/minute (85% cheaper than ElevenLabs)	Cost-effective voice synthesis	
Video Processing	RunwayML Gen-3	AI video generation and editing	$12/month	Video creation, effects
	Pika Labs	Video generation from	$10/month	Short-form video content

Service Category	Provider	Key Features	Starting Price	Use Case
		text/images		
	FFmpeg Cloud APIs	Professional video processing	Variable pricing	Video encoding, manipulation

4. Popular Free / Self-Hosted Tools

Category	Tool / Model	Strengths
Text-to-Speech	**Kokoro-FastAPI / Kokoro Web (82M)**	High-quality natural TTS with low resource usage; fully free and private
Text-to-Text	Groq Console models, OpenRouter, Together AI, Gemini 2.5 pro from google	Free access to powerful LLMs; open, customizable, and no paywalls
Vision / Multimodal	Llama-4 Scout via Groq, Gemini 2.5 flash from google	Free, high-token, multimodal capabilities
Image Generation	Cloudflare - flux schnell, Freepik - flux	Free generation options, decent output quality; accessible and flexible
Media Processing	**NCA Toolkit (No-Code Architects)**	Self-hosted FFmpeg-based video/audio processing—truly unlimited and free

Note: Limitations & Usage:

- Cloudflare provides **10,000 free "neurons" daily** (reset at 00:00 UTC).
- Each generated image typically consumes **~173 neurons** (sometimes ~86 neurons), allowing for **60–100 free images per day** depending on output complexity.
- Usage can be monitored here: Cloudflare Usage Dashboard
    - Replace `$YOUR_CLOUDFLARE_ACCOUNT_ID` with your actual Cloudflare account ID.

Unit Pricing (beyond free quota):

- **$0.000053 per 512×512 tile**

- **$0.00011 per step**

5. Why These Open Tools Are Worth Choosing

- **Completely free and scalable**: No recurring costs or surprise fees.
- **Privacy and control**: All data stays local; no third-party tracking.
- **Customization freedom**: Prompts, templates, and pipelines fully adjustable.
- **Top-tier flexibility**: NCA Toolkit uses ffmpeg-style operations—the same engine companies behind services like HeyGen or Creatomate rely on.
- **Community-driven growth**: Models evolve through contributions and usage—not hidden behind a paid wall.

6. Final Note

By choosing this combination of **Kokoro TTS**, **Groq models or Gemini**, **Llama-4 Scout**, **Pollinations**, and **NCA Toolkit**, you gain a fullstack AI video generation solution that is:

- Free to use **without limitations**
- Operable on standard hardware
- Fully customizable and adaptable
- Privacy-respecting and self-contained

The tables and insights provided above clearly show why these **free and unlimited tools** are the smarter choice — you avoid burning cash on tightly restricted APIs and instead gain consistent, high-quality output without compromising on flexibility or control.

Note: This entire project was successfully tested on an HP ProBook 640 G2 equipped with 8 GB RAM and a 7th-generation Intel i5 processor. Performance may vary on different hardware configurations.

Prerequisites

Before setting up, here's an overview of **what we'll install**, **what tokens we'll need**, and why each service is important.
We will install everything locally using **Docker**, but also expose the services securely to the internet using **Ngrok**.

Services We Will Install

1. **Ngrok** – Exposes your local services to the internet with a secure tunnel, required for external API callbacks to work with n8n.
2. **Docker** – Runs all required tools (n8n, MinIO, Kokoro TTS, NCA Toolkit) in isolated containers.
3. **n8n** – Automation/orchestration tool for the entire video generation pipeline.
4. **MinIO** – Local object storage for files (images, audio, video segments, etc.).
5. **Kokoro TTS CPU** – Text-to-speech generator for narration.
6. **NCA Toolkit** – Video editing & compilation tool based on `ffmpeg`.

Disk Space Recommendation:
You should have at least **60+ GB free**. Docker images and containers will consume ~30 GB, plus Docker's own system files.

Required Tokens & API Keys

We'll be using several **free-to-use API services**. You will need to create accounts and generate API keys for each one.

Service	Sign-Up / Docs
Groq Console	Get Groq Token
OpenRouter (Optional)	Get OpenRouter Token
Freepik API (Optional)	Get Freepik API Token
Google Cloud Console (Tutorial - Youtube)	Get Google Cloud Token
Reddit API	Get Reddit API Keys
Telegram Bot Token (Tutorial - Youtube)	Get Telegram Bot Token
Google Ai Studio	Get Ai Studio Key

=> Keep all API keys in a secure file (`.txt`) so you can easily grab and use them later.

---

1. Ngrok

Purpose: Allows you to make your local services (like n8n) accessible from the internet via a temporary public HTTPS URL.
This is critical because n8n's webhooks must be reachable by external APIs.

Installation:

1. Download Ngrok: Ngrok link
2. Install it on your system.
3. follow this guide to get your public HTTP URL : Ngrok Tutorial - Youtube

---

2. Docker

Purpose:
Container engine that will run all our tools without messy installations.

Installation:

- Download from Docker Desktop
- On Windows:
- Install with **WSL2** (faster than Hyper-V)
- You can cap RAM/CPU usage via `.wslconfig` :

```
[wsl2]
memory=5GB          # Max RAM for WSL2
processors=3        # Number of CPU cores to use
swap=1GB            # Swap file size
localhostForwarding=true
```

---

3. n8n

Purpose:
Automation tool for orchestrating the video generation process.

Docker Command:

```
docker run -d                                  # Run container in detached mode
  --name n8n                                   # Name the container "n8n"
  -p 5678:5678                                 # Map port 5678 for n8n UI
  -e N8N_COMMUNITY_PACKAGES_ALLOW_TOOL_USAGE=true  # Allow extra packages
  -e N8N_EDITOR_BASE_URL=YOUR_OWN_LINK         # Public URL from Ngrok
  -e WEBHOOK_URL=YOUR_OWN_LINK                 # Public URL from Ngrok for webhoo
  -e N8N_DEFAULT_BINARY_DATA_MODE=filesystem # Store binary data on filesystem
```

```
  -v /c/Docker/n8n-data:/home/node/.n8n      # Persist n8n config/data
  docker.n8n.io/n8nio/n8n                    # Official n8n Docker image
```

Note: Replace `YOUR_OWN_LINK` with your Ngrok public link.

---

4. MinIO

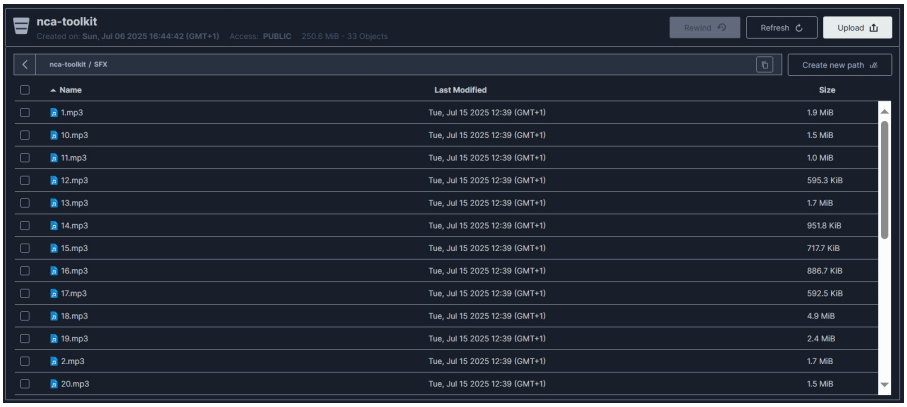
Purpose:
Local S3-compatible storage for assets.

Docker Command:

```
docker run -p 9000:9000                      # API endpoint port
  -p 9001:9001                               # Web console port
  --name minio                               # Name container "minio"
  -v C:\Docker\minio-data:/data              # Data persistence folder
  -e MINIO_ROOT_USER=admin                   # Admin username
  -e MINIO_ROOT_PASSWORD=password123         # Admin password
  quay.io/minio/minio:RELEASE.2025-04-22T22-12-26Z
  server /data --console-address ":9001"       # Start MinIO server
```

After starting MinIO:

1. Visit: `http://localhost:9001`
2. Login with the username & password set above.
3. Create a **bucket** named exactly: `nca-toolkit` .
4. Go to **Access Keys** → **Create Access Key** → Save both **Access Key** & **Secret Key**. These will be used in NCA Toolkit's configuration.
5. Inside the `nca-toolkit` directory, create two folders: - `final_result/` → For generated video outputs - `SFX/` → For sound effects (use uppercase `SFX` for compatibility) **Note**: Consistent casing (e.g., `SFX` vs `sfx`) is critical for the toolkit to function correctly. `SFX/` Folder content should be something like that :



All sound effects in this folder **must use numerical names** (e.g., `1.mp3`, `2.mp3`) to: Enable programmatic retrieval in workflows & Maintain consistent sorting

```
The SFX should be filled at Least with One audio file else the audio + vide
```

YouTube Guide:

- MinIO Tutorial - Youtube
- **MinIO Tutorial (Alt) - YouTube**

---

5. Kokoro TTS (CPU)

Purpose:
Generates realistic narration audio.

Docker Command:

```
docker run -p 8880:8880                          # Web/API port
  --name kokoro-tts-cpu                          # Name container
  ghcr.io/remsky/kokoro-fastapi-cpu:latest       # CPU-optimized Kokoro TTS image
```

After starting:
Visit: `http://localhost:8880/web/` to test voices and TTS output.

---

6. NCA Toolkit

Purpose:
Processes videos, applies effects, adds captions, compiles final outputs.

Docker Command:

```
docker run -d                                    # Run in background
  -p 8080:8080                                   # API port
  --name nca-toolkit                             # Name container
  -e API_KEY=thekey                              # API key for authentication (chan
  -e S3_ENDPOINT_URL=http://host.docker.internal:9000   # MinIO API URL
  -e S3_ACCESS_KEY=MINIO_ACCESS_KEY              # Access key from MinIO
  -e S3_SECRET_KEY=MINIO_SECRET_KEY              # Secret key from MinIO
  -e S3_BUCKET_NAME=nca-toolkit                  # Must match bucket created in Min
  -e S3_REGION=None                              # Region (None for local)
  stephengpope/no-code-architects-toolkit:latest # Toolkit image
```

Notes:

- Ensure MinIO bucket `nca-toolkit` exists **before** running this container.
- Use your MinIO Access Key & Secret Key here.
- Save the `API_KEY` (default `thekey`) — it will be needed for n8n API calls.

YouTube Tutorials:

- **NCA Toolkit Tutorial - YouTube** *By Stephen G. Pope (Creator of the Tool)*
- **NCA Toolkit Tutorial (Alt) - YouTube**

GitHub Repository: The official **NCA Toolkit - GitHub** is well-documented and maintained by the tool's creator. It also includes a **custom GPT** for additional assistance.

⚠️ IMPORTANT : NCA-Toolkit Integration with n8n

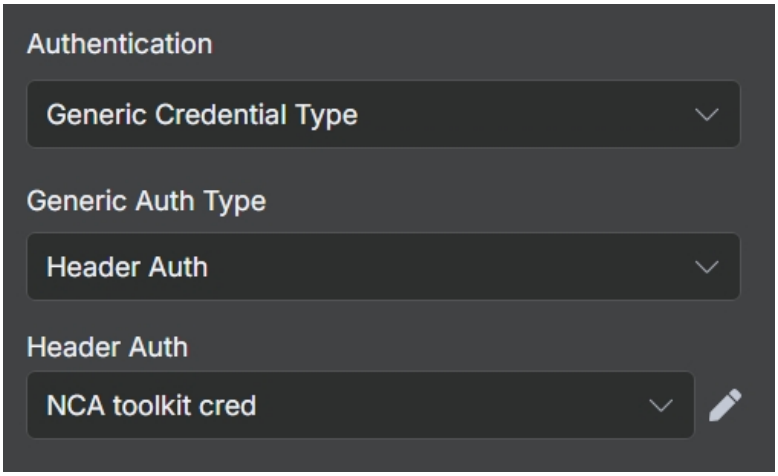
(Important: Without this setup, NCA-Toolkit nodes will not run)

When using **n8n** to connect with **NCA-Toolkit** via an HTTP Request node, you must configure **credentials** — otherwise, the request will fail.

Step 1 – Set Authentication

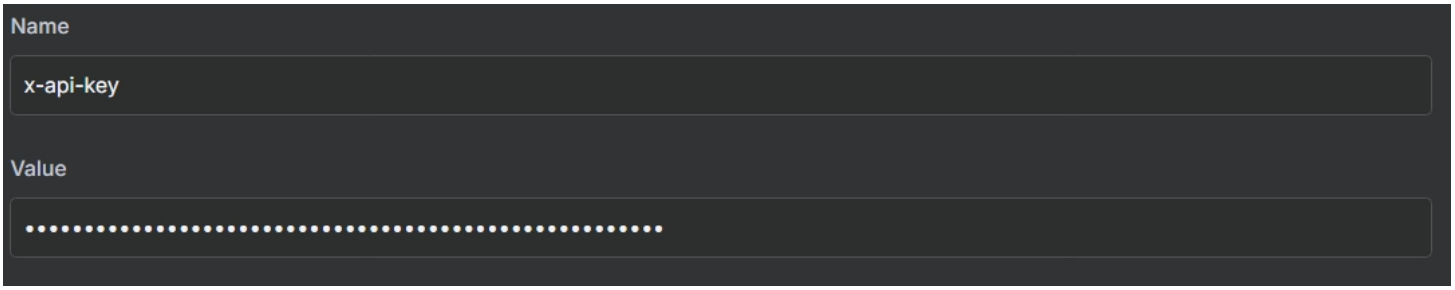
In the HTTP Request node:

1. Set **Authentication** to **Generic Credential Type**.
2. Select **Header Auth** as the method.



Step 2 – Add Credentials

1. For the **credential name**, enter it **exactly** as shown in the image below.
2. For the **value**, use the API key you configured in the Docker setup.



Important Note **:**

Do not modify the ports or advanced settings unless you are experienced with Docker and n8n configurations, and changing them incorrectly will cause the tool to malfunction.

Resource Limits (Prevent Crashes)

To avoid Docker crashes due to excessive resource usage, you can cap CPU and memory for your containers (especially Kokoro TTS & Nca-toolkit).

Recommended Limits:

```
docker update
  --cpus=2                # Limit to 2 CPU cores
  --memory=2G             # Limit to 2GB RAM
```

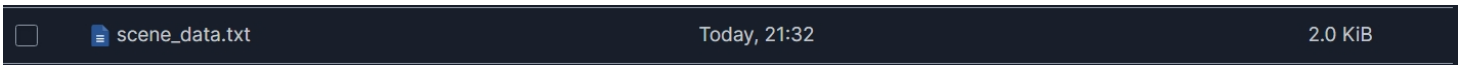
```
    --memory-swap=3G        # Allow 1GB swap (3G total incl. RAM)
    container-name
```

Important – MinIO Temporary Files

Purpose and Usage

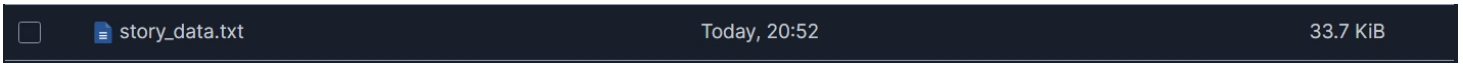
In this section, we'll go over the temporary files stored in **MinIO**, what they contain, and why they are needed.
These files are used to keep the workflow running smoothly, allow recovery after errors, and ensure we can resume or retry steps without starting from scratch.

1. `scene_data.txt`



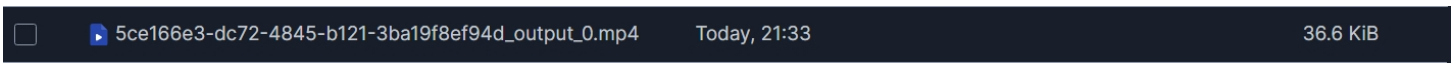
This file stores the **JSON data for the current scene** being generated.
When a new scene starts generating, the previous `scene_data.txt` is deleted to prevent conflicts between scenes.

2. `story_data.txt`



This is **the most important temporary file**.
It holds the current story's JSON data — including all scenes, prompts, scripts, and essential generation details.
If something goes wrong during the process (error, crash, interruption), this file ensures we can **resume exactly where we left off** without losing progress.
We can even regenerate individual scenes because of this file.
It is deleted after **Workflow IV**, once the story has been successfully turned into a complete video without errors.

3. Random Video File Names



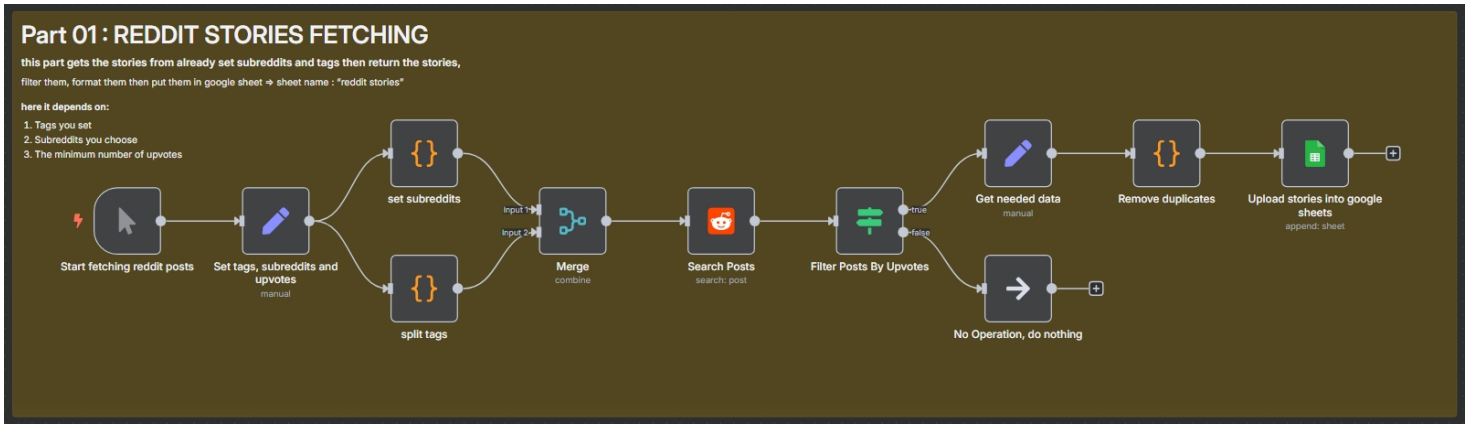
These are temporary files generated by **nca-toolkit** while creating scene videos.
They are purely intermediate outputs and are automatically deleted immediately after the final scene video is produced.

4. Story Folder

This folder contains all **generated images, speech files, and videos** for the current story.
It is deleted at the end of **Workflow IV** to free up storage once the final video is complete.

Workflow I: Reddit Story Fetching and Processing

This is the foundation workflow where we extract Reddit stories to serve as base scripts for video content creation. Run this workflow **only** when there are no unused stories available or when the sheet is empty. If stories are already present, running it again will have no effect.



Process Overview

The workflow imports two essential CSV files located in the main project folder:

- **reddit stories.csv** - Raw repository for fetched Reddit posts
- **formatted reddit stories.csv** - Processed stories optimized for video narration

Configuration Steps

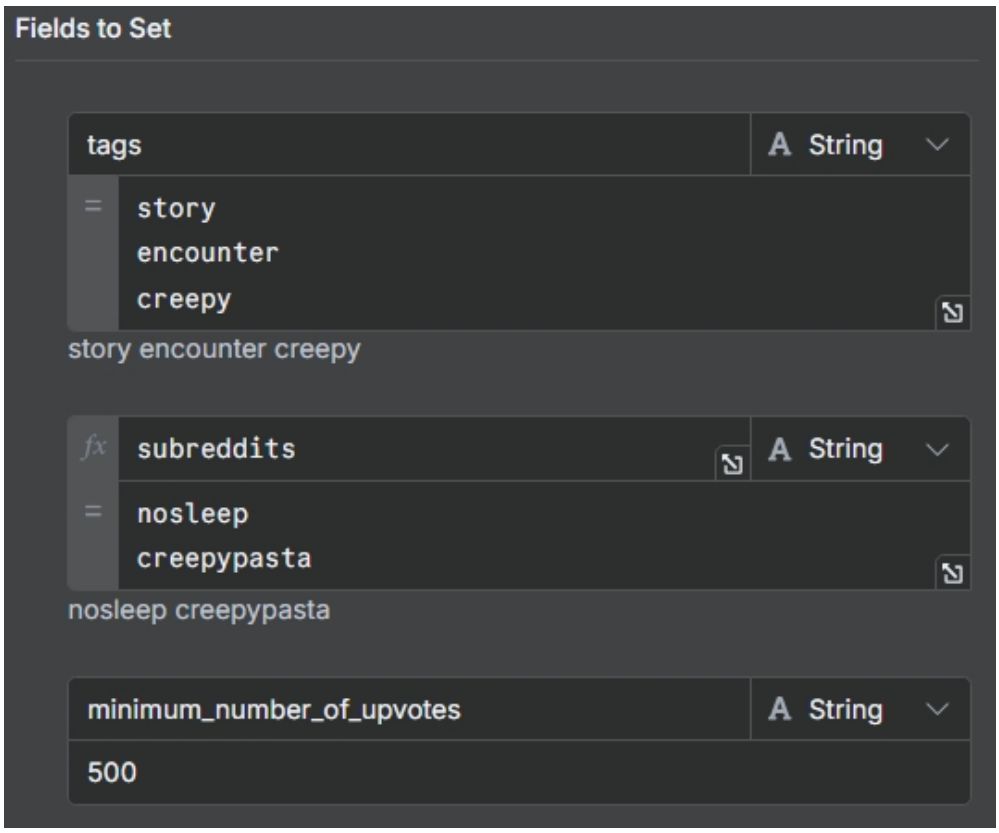
Step 1: Import CSV Files

Navigate to the main folder and ensure both CSV files are properly linked:

- **reddit_stories.csv** - Stores raw fetched content from Reddit
- **formatted*reddit*stories.csv** - Contains processed stories ready for video production

Step 2: Configure Search Parameters

The **"Set tags and subreddits to fetch"** node controls all filtering parameters:



Default Configuration:

- **Subreddits**: By default : r/creepypasta, r/nosleep
- **Search Tags**: By default : Story, encounter, creepy

- **Minimum Upvotes**: By default : 500 (filters quality content through the IF NODE)

All parameters are fully customizable based on your content niche and audience preferences.

Step 3: Credential Setup

Remaining nodes require only Reddit API credentials configuration - no additional modifications needed.

Recommended Subreddits by Content Category

Horror & Suspense

Subreddit	Content Focus	Recommended Tags
r/nosleep	Horror stories, creepypasta	horror, scary, paranormal, ghost, nightmare
r/creepypasta	Internet horror legends	creepy, urban legend, mysterious, dark
r/LetsNotMeet	Real-life scary encounters	stalker, creepy encounter, dangerous, survival
r/Paranormal	Supernatural experiences	ghost, paranormal, unexplained, supernatural
r/TwoSentenceHorror	Micro horror stories	horror, twist, dark, scary

Relationship Drama

Subreddit	Content Focus	Recommended Tags
r/AmItheAsshole	Moral judgment scenarios	AITA, conflict, relationship, drama, judgment
r/relationships	Relationship advice and stories	breakup, cheating, toxic, advice, love
r/relationship_advice	Relationship problem solving	advice, dating, marriage, conflict, help
r/offmychest	Personal confessions	confession, secret, guilt, regret, truth

Subreddit	Content Focus	Recommended Tags
r/TrueOffMyChest	Unfiltered personal stories	confession, controversial, honest, raw

Workplace & Life Drama

Subreddit	Content Focus	Recommended Tags
r/MaliciousCompliance	Revenge through rule-following	revenge, compliance, workplace, petty, justice
r/ProRevenge	Elaborate revenge stories	revenge, justice, karma, satisfying, payback
r/pettyrevenge	Small-scale revenge tales	petty, revenge, annoying, neighbors, minor
r/antiwork	Workplace horror stories	toxic boss, quit, workplace, bad job, labor
r/entitledparents	Dealing with difficult parents	entitled, Karen, parent, public freakout, crazy

Comedy & Wholesome Content

Subreddit	Content Focus	Recommended Tags
r/tifu	"Today I Messed up" stories	mistake, embarrassing, funny, fail, awkward
r/wholesomememes	Positive, uplifting content	wholesome, positive, heartwarming, good news
r/MadeMeSmile	Feel-good stories	happy, smile, uplifting, positive, heartwarming
r/HumansBeingBros	Acts of kindness	kindness, helping, good deed, humanity, bros

True Crime & Mystery

Subreddit	Content Focus	Recommended Tags
r/UnresolvedMysteries	Unsolved cases and mysteries	mystery, unsolved, cold case, disappearance

Subreddit	Content Focus	Recommended Tags
r/RBI	Reddit Bureau of Investigation	mystery, investigation, help, solve, clues
r/mystery	General mystery content	puzzle, mystery, unsolved, investigation

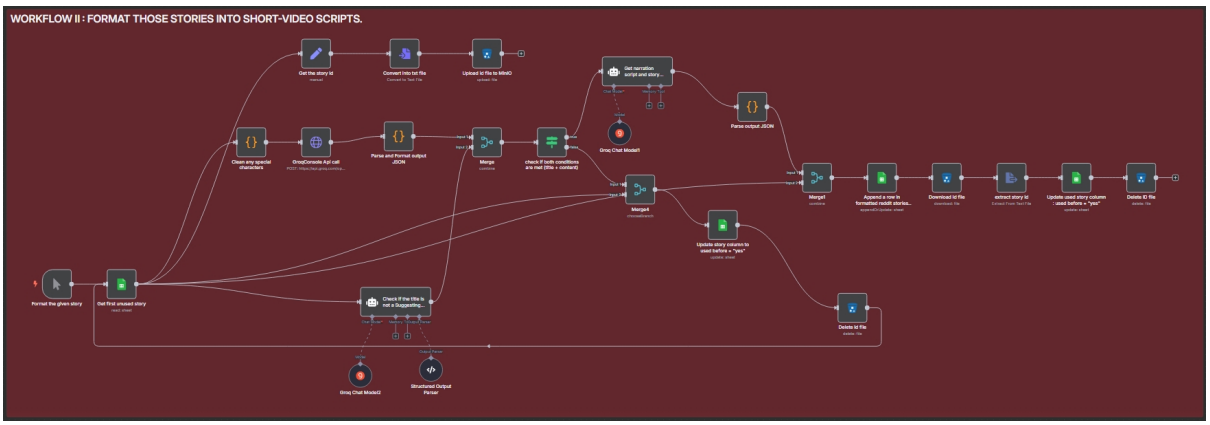
Confession & Personal Stories

Subreddit	Content Focus	Recommended Tags
r/confession	Personal admissions	confession, secret, guilt, mistake, truth
r/unpopularopinion	Controversial viewpoints	unpopular, controversial, opinion, debate
r/changemyview	Intellectual debates	debate, argument, perspective, change, view

This configuration ensures a steady stream of high-quality, engaging content suitable for video production while maintaining flexibility for different content niches and audience preferences.

Workflow II : Format those stories into video scripts

In this workflow, we take the **first story** where its `Used before` column is equal to `"no"` (unused story).
We then filter it to determine if it is a **good and compelling story** suitable for video content.



1. "GroqConsole API Call" NODE

Model used: `deepseek-r1-distill-llama-70b`

- Purpose: Check if the story is good enough to be turned into a video.
- Criteria: Must have all the essential components of a good story.
- Output: Returns `"yes"` or `"no"`.
- Alternatives:
- `tngtech/deepseek-r1t2-chimera:free` (via OpenRouter)
- Any other model with **high token input/output limits**.

2. "check if both conditions are met (title + content)" NODE

- **Conditions checked:**

1. **Story quality** — Response from **GroqConsole API Call Node** must be `"yes"`.
2. **Series check** — The **"Check if the title is not suggesting it is a sequel"** Node verifies that the story is **not** part of a series.

- **Outcome:**
- If **both** return `"yes"`, the story will be used.
- If **either** condition fails, the `Used before` column will be set to `"yes"` (marking it as used).

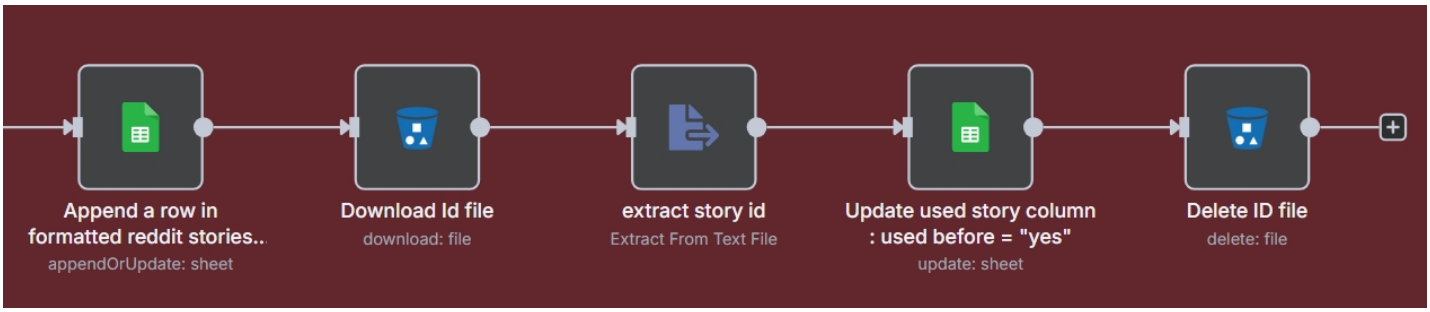
3. "Get narration script and story title" NODE

- **Input:** Raw Reddit story post.
- **Process:**
- Formats the story into a **narration script** optimized for TTS models.
- Adds a **captivating hook** in the first few seconds.
- Generates a **story title**.
- Detects our story genre so we choose which style to use in our story
- **Output:** Narration script + story title + genre, ready for further processing.

4. Post-Processing Steps

- Set the `Used before` column of the used story to `"yes"`.

- Append the processed story to the `formatted reddit stories` sheet.
- Perform **temp file cleanup** to free up space.



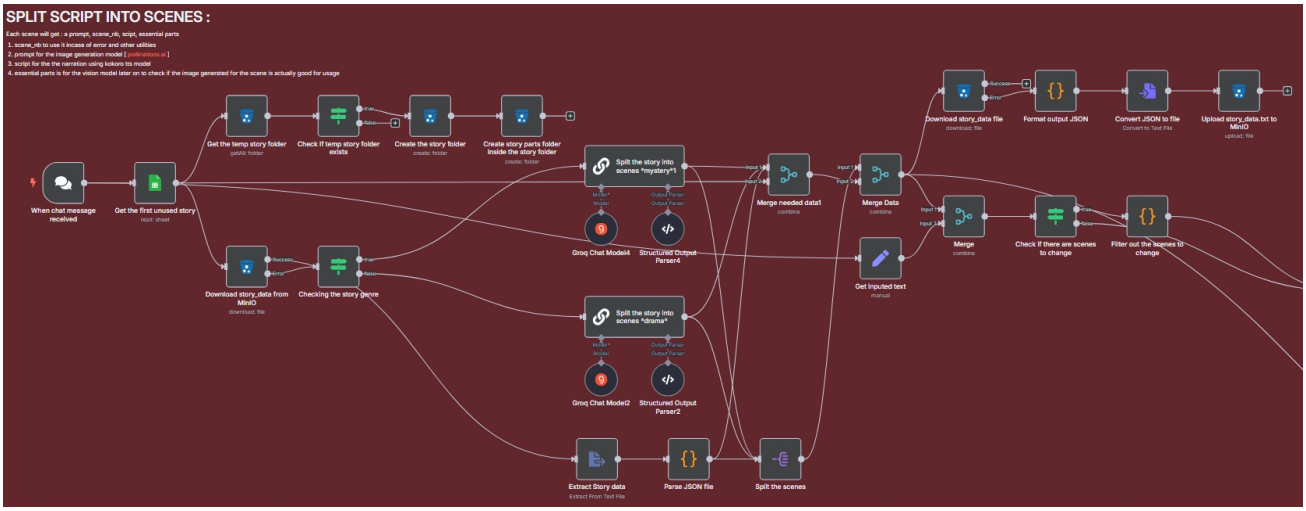
Workflow III : Split content into scenes and generate video per scene (Hand-drawn/Realistic)

Note:

The following steps apply to both art styles. Whether you choose the realistic or hand-drawn approach, these instructions remain essential for both workflows.

This workflow transforms a complete story into individual scene videos by splitting the script, generating narration and images, validating results, and assembling each scene with video effects and captions.

PART I — Split Script into Scenes



⚠️ IMPORTANT — HOW THE WORKFLOW TRIGGERS WORK

This workflow can be started in **two ways**: **Manual Trigger** or **Chat Trigger**.
Each works slightly differently:

=> 1. Manual Trigger

If you start the workflow using the **Manual Trigger** in n8n, it will **always** generate **all scenes in order**, starting from scene 1 and ending with the last scene.
(No need to type anything — it ignores inputs.)

=> 2. Chat Trigger

When starting the workflow through the **Chat Trigger** in n8n, your input text determines **which scenes are generated**.
There are **two possible modes**:

2.1 — Full Generation

If your input has **no numbers**, the workflow will generate **all scenes in sequence** — starting from scene 1 and ending with the last scene.

Example inputs (generate all scenes):

- `no`
- `hello`
- `randomtext`
- *(any text without numbers)*

2.2 — Specific Scene Re-Generation

If your input contains **one or more numbers**, the workflow will detect them and generate **only the matching scenes**.
Numbers can be separated by spaces, commas, symbols, or even be mixed inside words — they will still be detected.

Example inputs (generate specific scenes):

- `1 5 7` → scenes 1, 5, and 7
- `2,4,6` → scenes 2, 4, and 6
- `a1_test3...12` → scenes 1, 3, and 12

After receiving the full story script from the previous workflow, we split it into **17–25 logical scenes** by default. For each scene, we generate:

- **Speech script** — text for the Kokoro TTS model to narrate.
- **Image prompt** — for generating the scene's background image.
- **Essential parts** — required characters, that must appear in the generated image.

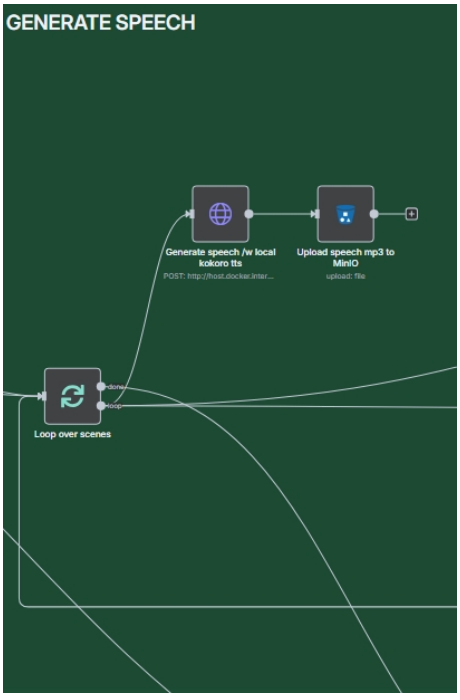
Routing depends on the story's genre:

- **Mystery** → *Split the Story into Scenes (Mystery) Node*
- **Drama** → *Split the Story into Scenes (Drama) Node*

`Story_data.txt`

A temporary file stored in **MinIO** containing the generated script, scenes, and essential parts. This allows resuming the workflow without regenerating content if a failure occurs. S3, extract, and convert nodes are used for this storage process.

PART II — Generate Speech

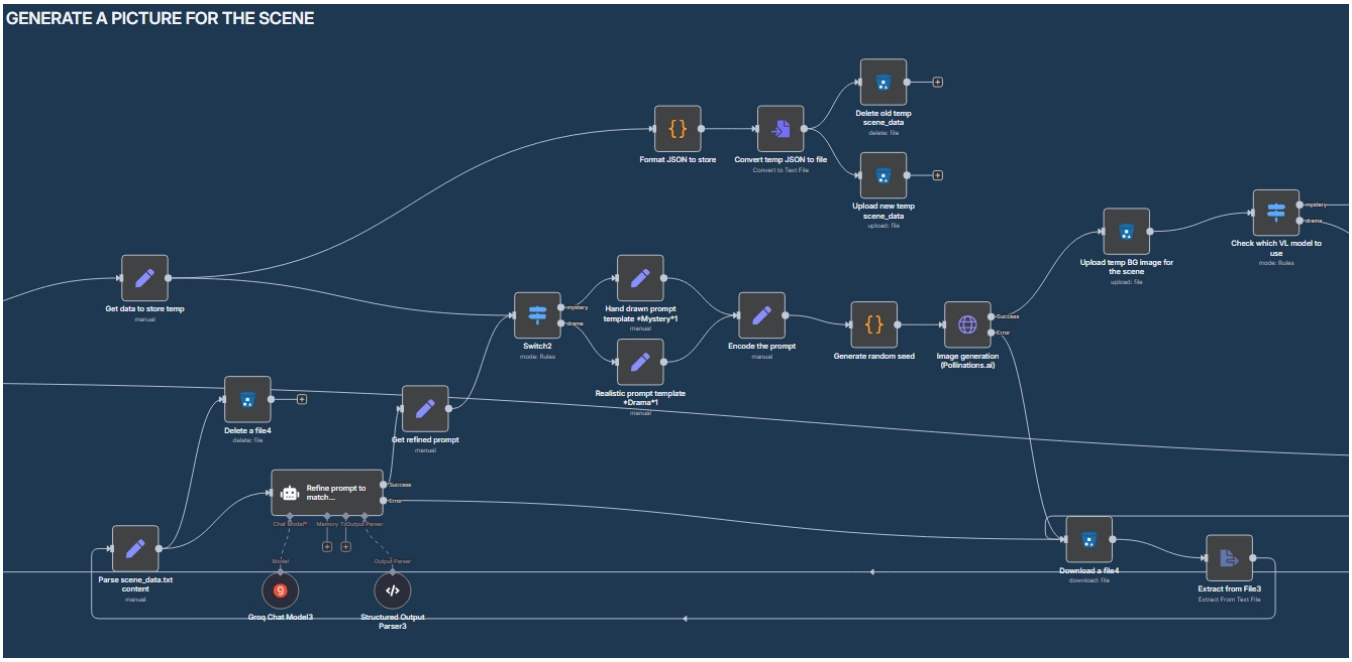


Using **Kokoro TTS**, we generate narration with one of two voices:

- **af_nicole** — a slower, deeper tone for horror and mystery stories.
- **af_bella** — a warmer, friendly tone for relationship and everyday stories.

Speech is saved in **MP3** format and uploaded to **MinIO**.

PART III — Generate Scene Images



We create an image for each scene that matches the **story vibe**, **genre**, and **art style**, while maintaining visual consistency across scenes.

"Prompt Template" Nodes

- *Hand-Drawn Prompt Template (Mystery)*
- *Hand-Drawn Prompt Template (Drama)*
- *Realistic Prompt Template (Mystery)*
- *Realistic Prompt Template (Drama)*

These templates ensure a consistent art style per genre. The hand-drawn templates produce a stylized illustrated look. The realistic templates, on the other hand, generate cinematic, high-detail visuals, ideal for outputs intended as realistic video. If you want the stylized look, use the ***Workflow III : Split content into scenes and generate video per scene (Hand-Drawn)*** and if you want the realistic look use the ***Workflow III : Split content into scenes and generate video per scene (Realistic)***.

"Image Generation" Node

- **Cloudflare - Flux Schnell Model — Main Image Generation**

  Limitations & Usage:

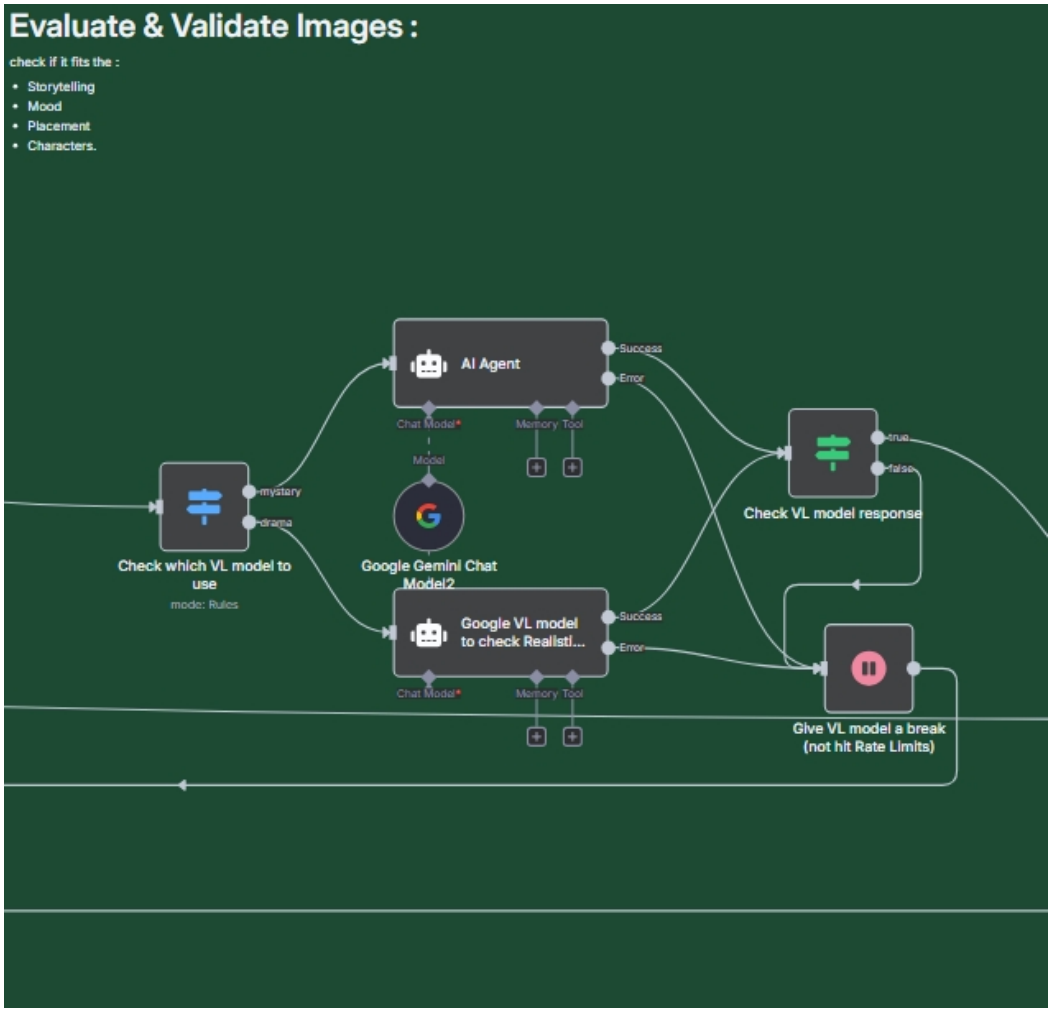
- Cloudflare provides **10,000 free "neurons" daily** (reset at 00:00 UTC).
- Each generated image typically consumes **~173 neurons** (sometimes ~86 neurons), allowing for **60–100 free images per day** depending on output complexity.
- Usage can be monitored here: Cloudflare Usage Dashboard

- Replace `$YOUR_CLOUDFLARE_ACCOUNT_ID` with your actual Cloudflare account ID.

Unit Pricing (beyond free quota):

- **$0.000053 per 512×512 tile**
- **$0.00011 per step**
- **Alternatives:**
- Freepik Flux → API Docs

PART IV — Evaluate and Validate Images



We verify that each generated image meets all requirements — genre style, essential parts, and overall visual quality.

Primary model:

- `google/gemini-2.5-flash` **(Google AI Studio):** Offers generous token limits, it also delivers high throughput with rate limits of **10 requests per minute**, **250,000 tokens per minute**, and **500 requests per day**—making it a reliable choice for efficient, high-volume testing

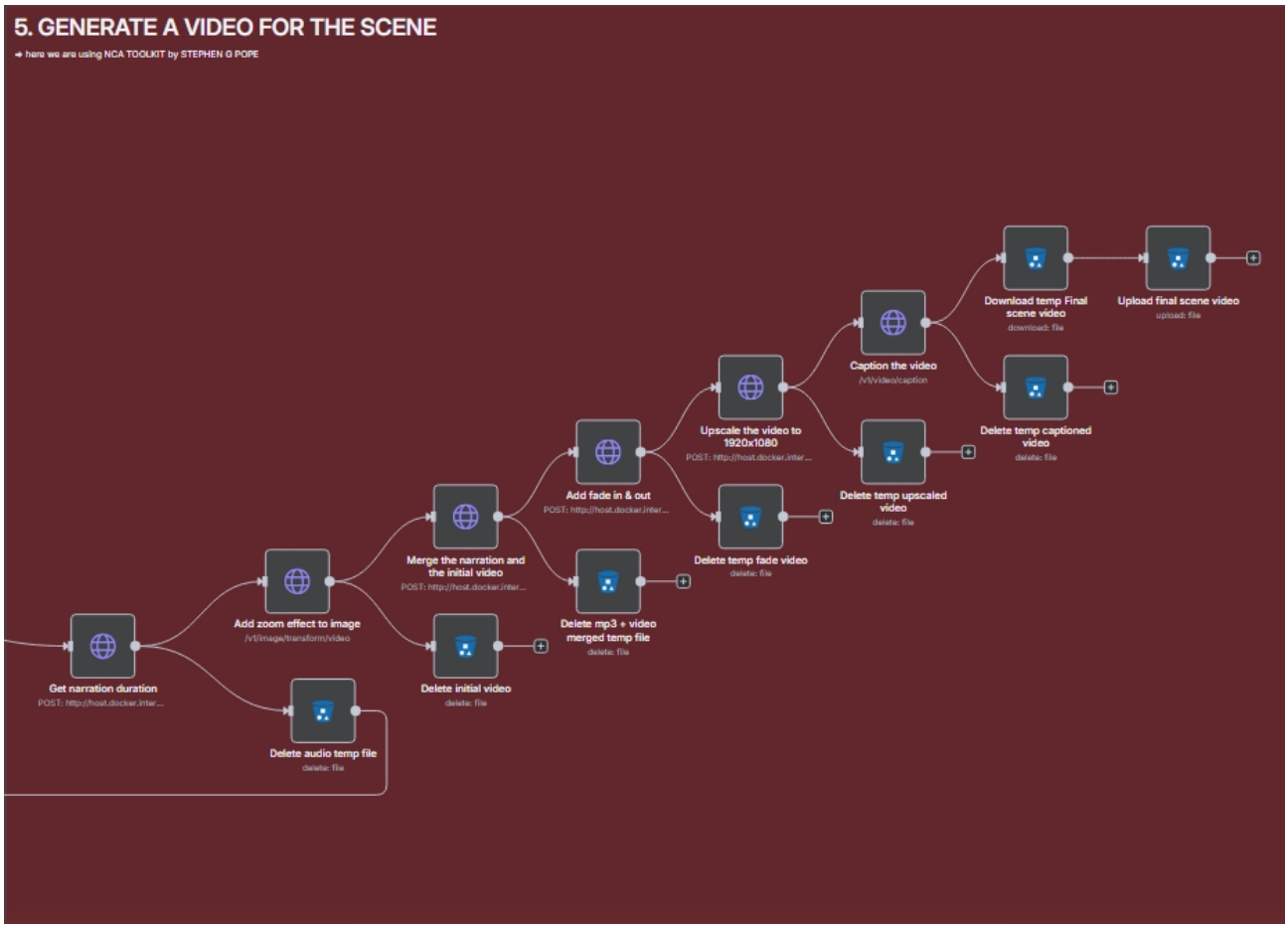
Alternative model:

- `google/gemma-3-27b-it:free` (OpenRouter)
- `meta-llama/llama-4-scout-17b-16e-instruct` (GroqConsole)

If validation fails, a new image is generated.

Note: These models have rate limits — too many API calls may result in errors. If that happens, pause the workflow and resume from the failed scene.

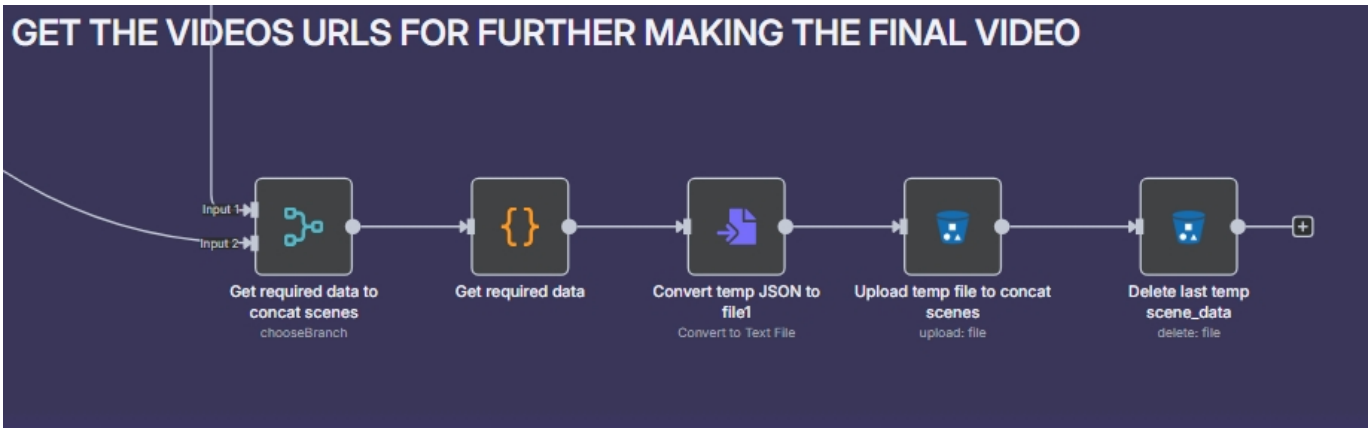
PART V — Assemble Scene Video



Using **nca-toolkit**, we process media and create the scene video:

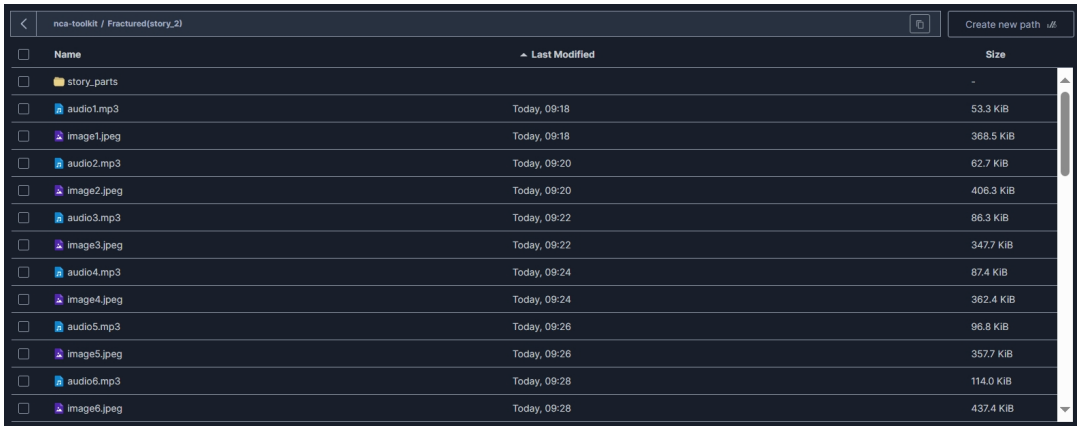
1. Measure narration duration to match video length.
2. Add a zoom effect to the generated background image.
3. Merge narration audio with the zoomed video.
4. Apply fade-in and fade-out effects.
5. Resize to **1080x1920** for proper caption placement.
6. Add captions.
7. Remove temporary files from **MinIO**.

PART VI — Prepare for Final Video Assembly



We clean up remaining temporary files and gather essential outputs for the next workflow.
If this step is skipped, the fourth workflow will not function correctly.

After the story is processed, **MinIO storage** will contain the following:



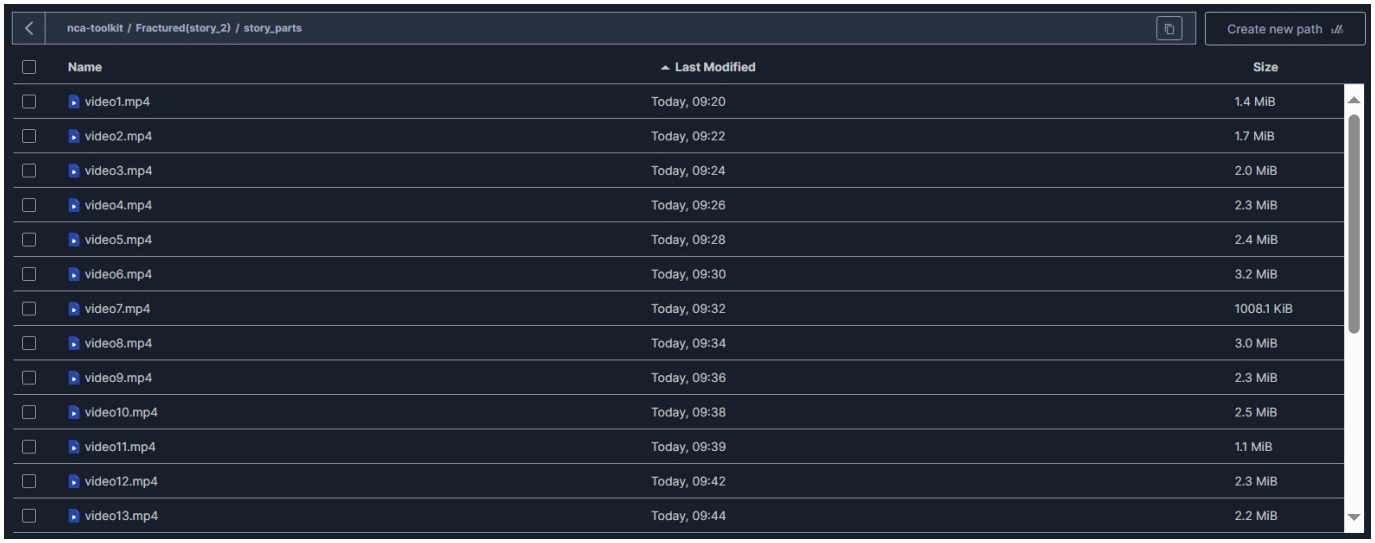
Each scene has a corresponding audio and image file, with filenames following this format:

- `audio{scene number}.mp3`
- `image{scene number}.jpeg`

This structure allows you to review each scene's components individually if needed.

The `story parts` folder contains the processed videos created by **NCA Toolkit**, where each audio and image pair has been merged and processed.

Example folder content:



Here, you can check the generated scene videos. If you want to replace a specific scene, simply reference its number when prompted in **WORKFLOW N.03**.

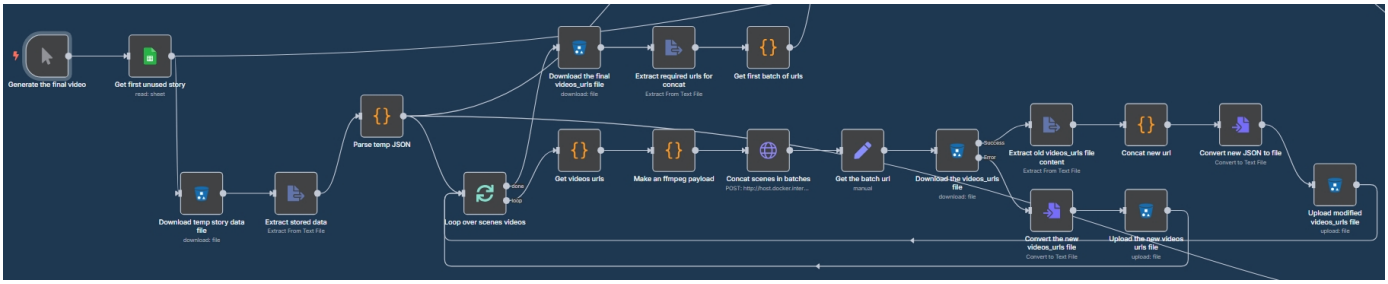
All other files in storage are temporary assets needed for **WORKFLOW N.04**. These will be cleaned up once processing is complete.

Workflow IV : Concatenate scenes and add audio effects into final video

In this workflow, we combine the generated videos from the previous step and merge them with audio using **NCA Toolkit**, producing a final video between **30 seconds up to 10+ minutes** in length.

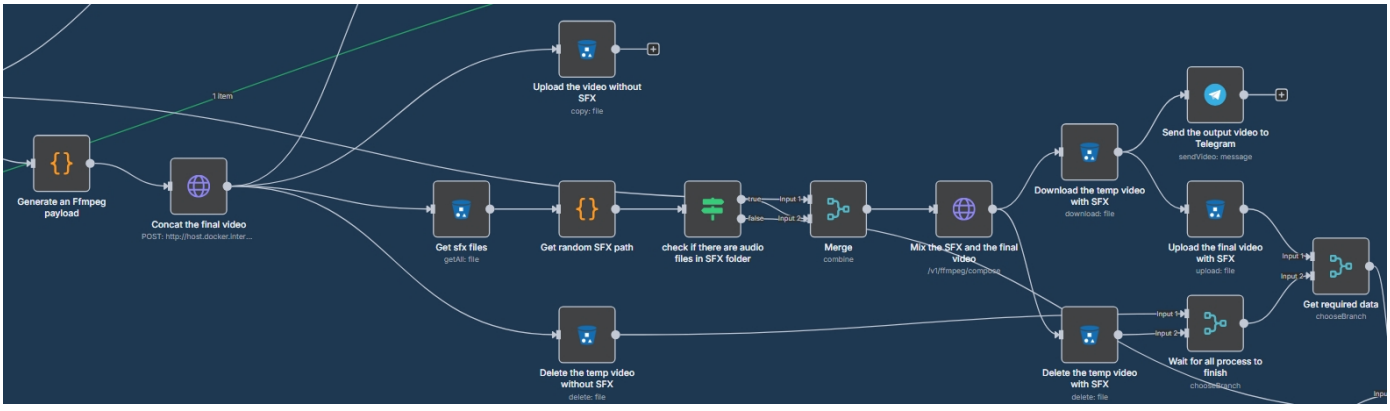
PART I: Batch Video Concatenation

To minimize errors or service crashes, we use a safe, staged process. First, the videos are concatenated in **batches of five**. Then, these batches are combined into a single full-length video.
During this stage, **NCA Toolkit** handles the concatenation process, and the intermediate video URLs are stored in **MinIO**. This allows us to retrieve and merge them into the final output later.



PART II: Final Assembly & Audio Integration

Next, the final video is concatenated and merged with sound effects. Details on preparing and using the **SFX folder** are covered in the ***Prerequisites Section***.



Finally, the workflow performs cleanup by removing temporary files from storage and marking the used story's **"used before"** column as **"yes"**, ensuring it will not be reused.

Workflow V : Create video metadata (title, thumbnail, description, etc.)

In this workflow, we will generate **SEO-optimized metadata** necessary to publish videos across multiple platforms:
YouTube, Rumble, Instagram, TikTok, Snapchat, and Facebook.

This metadata will be tailored to help you grow faster and perform better on each platform.
We will also create a **custom thumbnail image** for each story — featuring **bold, clear text** and an **atmosphere that matches the story and its genre**.

Important Notice

This project **does not** claim to automatically upload videos to these platforms.
Why? Because our goal is to use **free tools and APIs**, and all well-known auto-upload tools are paid — with free tiers offering fewer than **10 uploads/month** (far too low, even if counted per day).

The only viable method is to use the **official platform APIs**, which come with the following limitations:

=> YouTube

- **Quota limits**: Default daily quota is **10,000 units per project**
- Video upload (`videos.insert`) consumes **~1,600 units**
- This means ~**6 uploads/day** by default

But still it can be done via the official api which will included in the workflow

=> TikTok

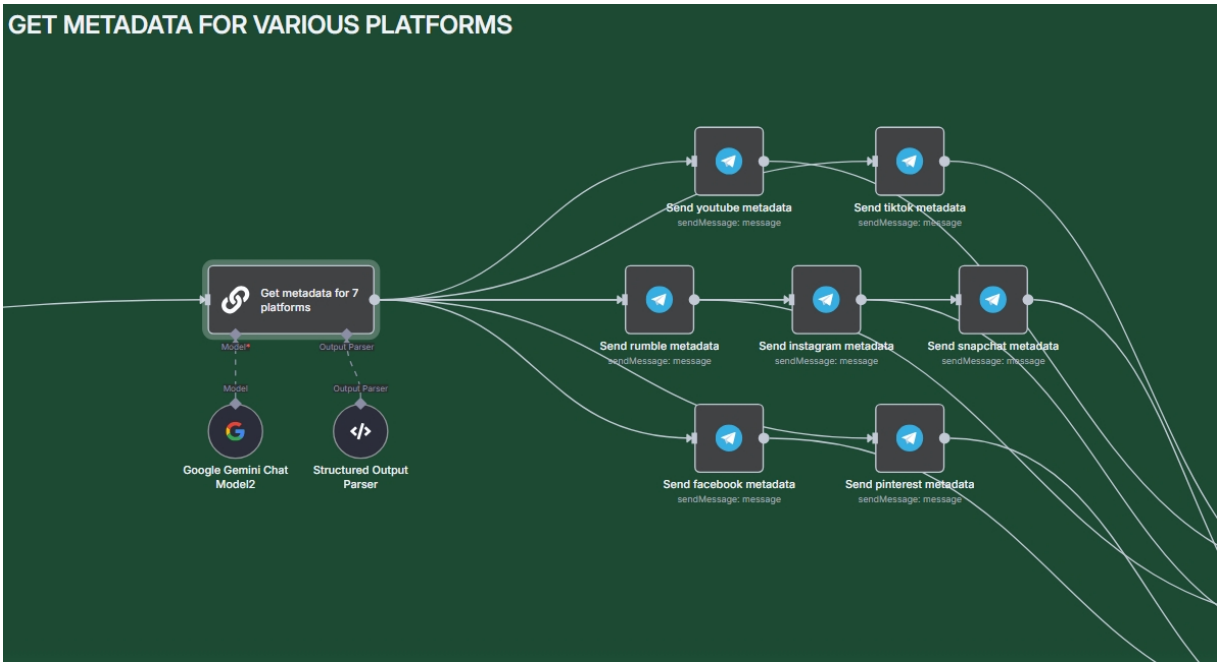
- Requires a **Business Account** to access the TikTok for Business API
- **Limitations:** Business accounts cannot access most creator monetization programs (e.g., Creator Rewards Program, Creator Marketplace, live gifts)
- These programs are reserved for **Creator (personal) accounts**

=> Rumble & Snapchat

- **No official API available**
- Publishing must be done **manually**

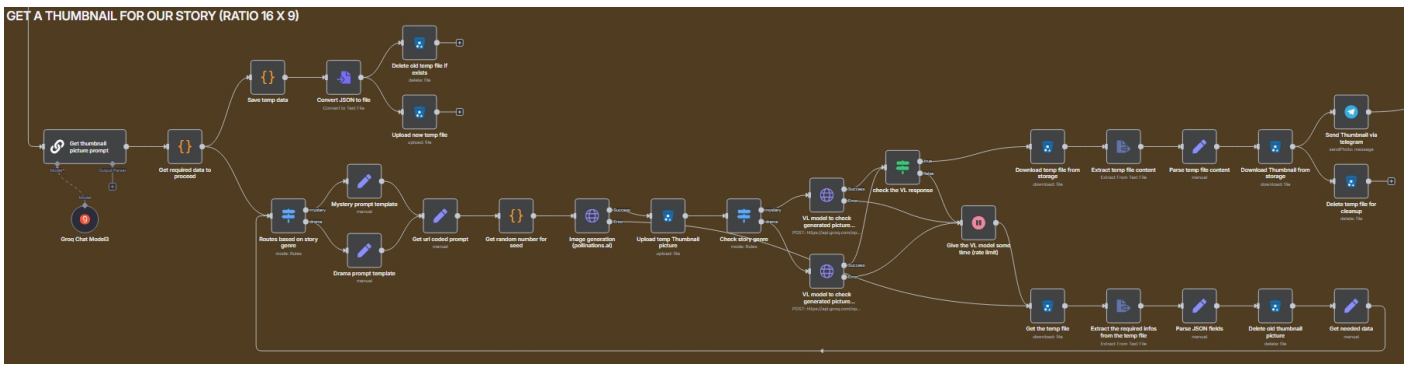
=> Facebook & Instagram

- To obtain access keys from the **Meta Graph API**, follow this tutorial: Meta Graph API - Tutorial



In this part of the workflow, we focus on preparing all the necessary publishing assets for our content.

First, we generate platform-ready metadata—**tags**, **titles**, and **descriptions**—using the `Google's Gemini 2.5-pro` model. The generated metadata is then sent to our Telegram bot in **7 separate messages**.



Genre-Based Routing and Thumbnail Creation

We start by taking the **story content**, **title**, and **genre**, then route them according to the story's genre. This determines the creative direction for our thumbnail and prompt templates.

"Get Thumbnail Picture Prompt" **NODE**

Uses the story's content, title, and genre to generate a thumbnail image prompt. The style and mood of the prompt adapt to the story's genre to ensure it matches the intended vibe.

"Mystery Prompt Template" **NODE &** ***"Drama Prompt Template"*** **NODE**

Provide genre-specific prompt templates to maintain a **consistent art style** that complements the story.

"VL Model Check" **NODES**

- **VL Model to Check Generated Picture (Mystery)**
- **VL Model to Check Generated Picture (Drama)**

These nodes validate the generated thumbnail against defined requirements.
The key requirement: **The story title must appear on the thumbnail** to ensure it works effectively as a visual preview.

Post-Processing and Storage

After validation:

- Temporary files are cleaned up from **MinIO**.
- The final thumbnail is sent to **Telegram**.
- The thumbnail is saved in **MinIO** for future use.

General Advice & Resources

1. Free AI Models & Tools Overview

1. Free Text Models

- **LM Studio** – Run local LLMs if you have a strong PC (preferably with a high-VRAM GPU).
- **Groq Console** – Free models with relatively loose rate limits.
- **OpenRouter** – Free access to some hosted models.
- **Google AI Studio** –
- Free: Gemini 2.5 Pro, Gemini 2.5 Flash, Gemini 2.5 Flash-Lite, Gemma models (good rate limits).
- Paid: The other Pro models & Imagen image generation.
- **Together AI** – 10+ free models, plus a $1 free credit for paid models.

2. Free Embedding Models

- **Jina AI** – Free embeddings API.

3. Free Vision-Language (VL) Models

(all included in workflow n.03)

- **Pollinations.ai** – VL API endpoint (OpenAI-compatible).
- **Google AI Studio** – Gemini 2.5 Flash & Gemini 2.5 Flash-Lite.
- **Groq Console** – `meta-llama/llama-4-scout-17b-16e-instruct` .
- **Together AI** – `meta-llama/Llama-Vision-Free` .
- **OpenRouter** – `google/gemini-2.0-flash-exp:free` .

4. Free Text-to-Speech (TTS)

- **ElevenLabs** – Free tier: 10,000 characters (~10 minutes audio).
- **Kokoro TTS** – CPU-based TTS.
- **Open-source TTS options** *(for strong CPU/GPU users, preferably NVIDIA)*:
- Coqui TTS
- Bark
- Piper

5. Free Image Generation

- **Freepik API** –
- 500 images with **Flux** model.
- 1250 images with **Classic Fast** model (lower quality, less prompt adherence).

- **Pollinations API** – Unlimited and free but the quality isn't its strongest point and also it can crash at any point
- **Local Generation** – Run models like **Waifu Diffusion**, **Flux (Flux.1 Dev)**, or **Stable Diffusion v1.5** using:
- Automatic1111
- ComfyUI
- Civitai
- **Other APIs**:
- Cloudflare Workers AI (the completely free models) – *Not recommended: poor quality & low prompt adherence.*
- AI Horde – Community-powered, but can have >10 min queue times and error risk.

6. Free (or Trial) Video Generation

- **Google Cloud Console** – Activate free trial ($300 credits, requires credit card).
- Use for paid APIs like **Imagen 4** and **Veo 3**.
- Pricing:
  - Imagen 4 Ultra: **$0.06/image**
  - Imagen 4 Standard: **$0.04/image**
  - Veo 3: **$0.075/video**

---

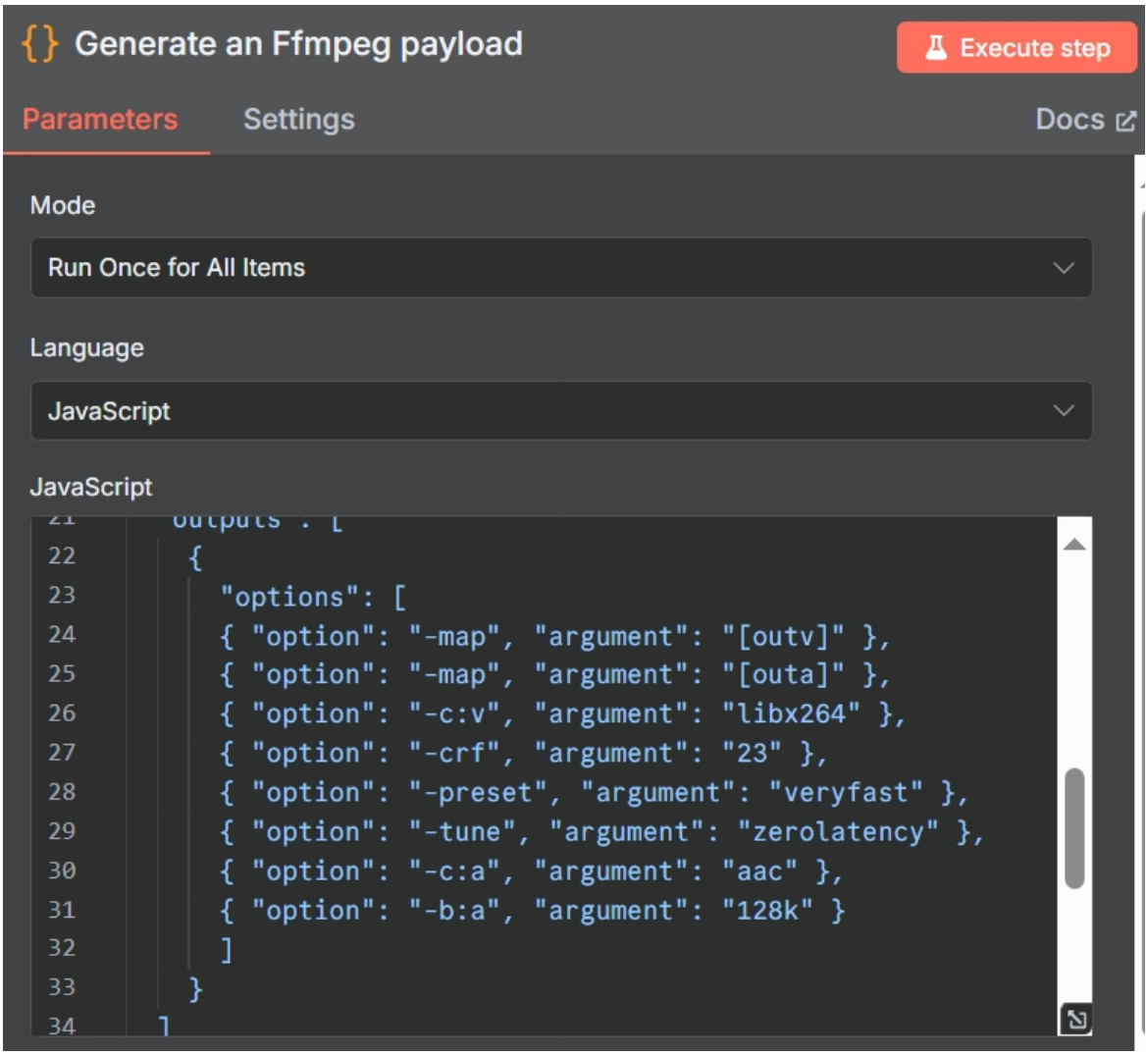
2. Improving Final Video Bitrate & Quality

The final video is rendered using the **nca-toolkit** nodes with `libx264` encoding.
To increase quality and bitrate, you can adjust the **CRF value** (lower CRF = better quality, but larger file size) and tweak the encoder settings. You can change it in the **Code Node** in Workflow N.04: ***Generate an Ffmpeg payload NODE*** : You can use this payload :

```
[
  { "option": "-map", "argument": "[outv]" },
  { "option": "-map", "argument": "[outa]" },
  { "option": "-c:v", "argument": "libx264" },
  { "option": "-crf", "argument": "18" },
  { "option": "-preset", "argument": "slow" },
  { "option": "-tune", "argument": "film" },
  { "option": "-c:a", "argument": "aac" },
  { "option": "-b:a", "argument": "320k" }
]
```

=> If you want **maximum possible quality regardless of file size**, you could even set:

```
{ "option": "-crf", "argument": "16" },
{ "option": "-preset", "argument": "veryslow" }
```



By default the CRF is 23. For most cases, a CRF value of **18–23** is a good balance between quality and file size. Use **preset=slow** or **preset=slower** for better compression efficiency if speed and PC Specs are not an issue

---

Final Word

This AI automation workflow and the entire project are built from a place of pure enthusiasm, passion for technology, and love for the fields of automation and content

creation. While it provides powerful tools to streamline your creative process, it does **not** guarantee high view counts, viral success, or financial gain at any scale. The core intention behind this work is to inspire and enable creators, giving them the means to express themselves on social media more easily and efficiently, and to explore the fascinating possibilities of AI-driven content production. I genuinely hope that this project empowers you to share your ideas, stories, and creativity with the world, and that it becomes a stepping stone toward achieving your personal or professional goals. Whether it's reaching a wider audience or simply enjoying the process of making content, I wish you all the best on your journey. Best Of Luck My Friend ;-)