# **📘 Documentation: Text-to-Video Music Video Workflow (Wan Humo / ComfyUI + Custom Nodes)**

## **1. Overview**

This workflow takes a **reference image** and a **song** and automatically generates a full **music video**.  
 It uses the **Wan 2.1 Humo model** for text-to-video generation, along with custom **vrgamedevgirl custom nodes** that handle audio splitting, lyric transcription, timing, indexing, and run management.

Runs are repeated in batches until all audio chunks are processed, then combined into a final video.

## **2. Workflow Steps**

1. **Upload a reference image** – Your main character that will be used throughout the video.
2. **Upload a song (audio file)** – this will be split into timed segments automatically
3. **Set index = 0** before the first run.
4. **update output folder path** – where intermediate video clips will be stored.
   1. This is inside your comfy UI outputs folder.
   2. If the folder does not exist, it will be created.
   3. The folder must be empty before the first run.
5. **Enable groups** (make sure all the groups are enabled under the Fast Groups Muter by rgthree ).
6. **Click Run Once.**
   1. The **README node** will display how many more runs are required.
   2. This only updates the one time and won’t update on next runs.
   3. It will tell you how many more times you need to hit “Run” then what groups you need to disable and hit “Run” once more.
7. **Repeat runs** until all chunks are generated.
8. On the final run, the ReadMe Note will tell you how many groups to mute (if fewer than 16 are needed on the last run).
   1. Note, you must have at least two enabled so if it says 0 or 1 just enabled 2.
9. The **last run assembles all clips** into one finished video.
   1. This video is saved in the normal output folder unless you update the video combine node.

## **3. Node Responsibilities**

### **🎵 Audio + Lyrics**

* **VRGDG\_LoadAudioSplit\_HUMO\_TranscribeV2**
  + Splits the uploaded song into **3.88s clips** (limit of Wan Humo for now).
  + Internally runs Whisper transcription (no separate node).
  + Outputs: audio segments, transcription text, metadata.

### **🤖 Prompt Generation with Gemini LLM**

* **Native comfy UI API node.** 
  + This node connects to **Google’s Gemini LLM**.
  + It takes as input:
    - The **lyrics** (from VRGDG\_LoadAudioSplit\_HUMO\_TranscribeV2),
    - The **reference image** (to guide visual style/consistency).
  + The LLM analyzes the lyrics and image, then generates **scene-level text prompts** that describe what should be visualized in each 3.88 second video segment.
  + These prompts are passed into the **VRGDG\_PromptSplitter**, which breaks them into smaller parts that flow into the video generation chain.

**Purpose of this node:**  
 Gemini acts as the **creative director** — it interprets the meaning/emotion of the lyrics and adapts them into visually descriptive prompts that keep consistency with the chosen reference image.

This node can be replaced with any LLM node that takes an image, string and instructions and outputs a string. Keep in mind, the instructions are very intensive and most open source LLM’s have a hard time following them.

You also do not need to use this node but then the workflow is no longer automated.

### **📝 Prompt Handling**

* **VRGDG\_PromptSplitter**
  + Breaks generated prompts into multiple sub-prompts for video generation.

### **🛠 Index + Timing Utilities**

* **VRGDG\_TimecodeFromIndex**
  + Converts index into timecodes (keeps audio and video aligned).

### **📂 File + Folder Helpers**

* **VRGDG\_GetFilenamePrefix**
  + Generates file prefixes for naming outputs so the video’s saved into the correct folder.
* **Folder Management**
  + Ensures output folder exists and is empty before the first run.

### **📖 Run Instructions**

* **README Node**
  + Displays what to do after each run:
    - How many runs remain,
    - When to mute groups for the last pass.

### **🎛 Run Control**

* **Fast Groups Muter (rgthree)**
  + Only needed on the last run.
  + Used to mute groups if the final batch has fewer than 16.

### **🎬 Final Assembly**

* **Intermediate Clip Storage**
  + Each run outputs 1m 2s clips to the working folder.
* **Final Combine Node**
  + Joins all generated clips into a complete synchronized video.

## **4. How It Flows**

1. Song is split into 3.88s segments by **VRGDG\_LoadAudioSplit\_HUMO\_TranscribeV2**.
2. Lyrics are transcribed internally and fed into the **LLM + PromptSplitter**.
3. Prompts + reference image → processed by **Wan 2.1 Humo** → generates short video clips.
4. **Index/ timing nodes** keep runs aligned.
5. Clips are saved to the working folder.
6. Runs are repeated until all audio is processed.
7. The **README node** tells you when to adjust **Fast Groups Muter**.
8. Final assembly merges clips into the complete video.