Text-to-Video Generation with AI

A walkthrough of creating videos from prompts using Gemini and Stable Diffusion.

The Goal: What are we building?

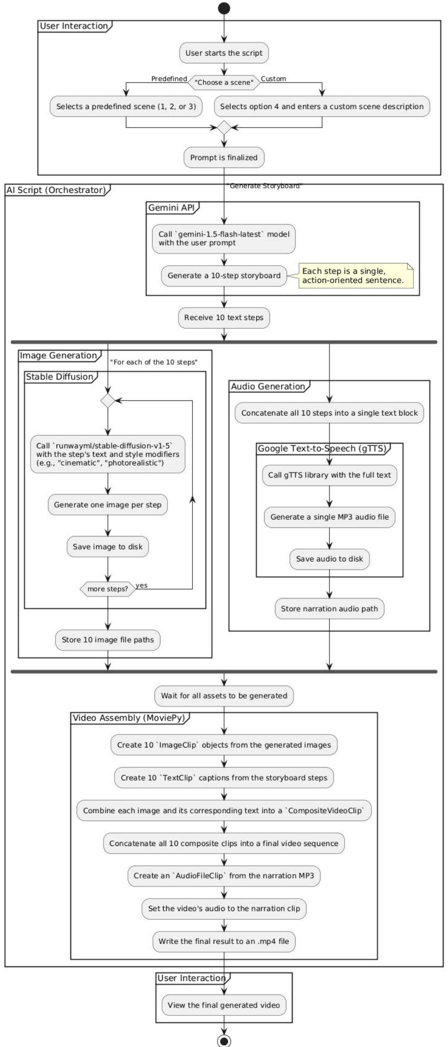
We're creating a Python script that automatically converts a simple text description (a "prompt") into a short, narrated video. The idea is to go from a single sentence to a complete video with visuals and a voiceover, all generated by AI.

The Overall Process: The AI Assembly Line

Our script follows a clear, multi-step pipeline:

1. **AI Storyboard (Gemini):** Takes the user's prompt and breaks the scene into 10 steps.
2. **AI Image Generation (Stable Diffusion):** Creates a picture for each of the 10 steps.
3. **AI Narration (Google TTS):** Creates a voiceover from the text of the steps.
4. **Video Assembly (MoviePy):** Stitches the images, text captions, and audio together into a final video.

Text-to-Video Generation Workflow


User Interaction
User starts the script
"Choose a scene"
Predefined
Custom
Selects a predefined scene (1, 2, or 3)
Selects option 4 and enters a custom scene description
Prompt is finalized
"Generate Storyboard"
AI Script (Orchestrator)
Gemini API
Call `gemini-1.5-flash-latest` model with the user prompt
Generate a 10-step storyboard
Each step is a single, action-oriented sentence.
Receive 10 text steps
Image Generation
"For each of the 10 steps"
Stable Diffusion
Call `runwayml/stable-diffusion-v1-5` with the step's text and style modifiers (e.g., "cinematic", "photorealistic")
Generate one image per step
Save image to disk
more steps?
yes
Store 10 image file paths
Audio Generation
Concatenate all 10 steps into a single text block
Google Text-to-Speech (gTTS)
Call gTTS library with the full text
Generate a single MP3 audio file
Save audio to disk
Store narration audio path
Wait for all assets to be generated
Video Assembly (MoviePy)
Create 10 `ImageClip` objects from the generated images
Create 10 `TextClip` captions from the storyboard steps
Combine each image and its corresponding text into a `CompositeVideoClip`
Concatenate all 10 composite clips into a final video sequence
Create an `AudioFileClip` from the narration MP3
Set the video's audio to the narration clip
Write the final result to an .mp4 file
User Interaction
View the final generated video

Generating the Narrative

First, we need a story. We use the Gemini language model (`gemini-1.5-flash-latest`) to act as a director.

- **Input:** The user's simple scene description.
- **Action:** We instruct Gemini to break the scene down into exactly 10 distinct, action-oriented steps.
- **Output:** A list of sentences that will form the video's narrative and captions.

Visualizing the Story

With the storyboard ready, we create the visuals.

- **Input:** Each individual step from the storyboard.
- **Action:** We feed each step into **Stable Diffusion** (`runwayml/stable-diffusion-v1-5`), adding keywords like "cinematic style" and "photorealistic" to improve the result.
- **Output:** A unique image for each of the 10 steps.

Adding a Voice

Next, we generate the voiceover.

- **Input:** The complete list of 10 storyboard steps.
- **Action:** The steps are joined into a single block of text and processed by **Google's Text-to-Speech (gTTS)** library.
- **Output:** A single MP3 audio file containing the spoken narration.

The Final Edit

This is where all the AI-generated assets come together using the **MoviePy** library.

- **Process:** MoviePy turns each image into a video clip, overlays the text caption, concatenates all the clips, and adds the final audio track.
- **Output:** A finished `.mp4` video file.

Key Technologies Used

- **Gemini 1.5 Flash:** For natural language processing and storyboard generation.
- **Stable Diffusion v1.5:** For high-quality text-to-image synthesis.
- **gTTS (Google Text-to-Speech):** For creating the audio narration.
- **MoviePy:** For all programmatic video editing and assembly.
- **Python & Google Colab:** The environment that brings it all together.

Conclusion

This project provides a framework for turning an idea into a multimedia presentation by orchestrating language, image, and speech models.

Future Enhancements

- Use a true text-to-video model for dynamic motion.
- Integrate another AI to generate background music.
- Add more user controls for style, pacing, and voice.
- Include scene transitions for a more polished look.