Project: Documentary Creator v2.0

This repository contains a comprehensive Python toolchain for generating documentary-style videos from text scripts. It automates everything from script writing and image prompt creation to video assembly with voice-over, background music, and effects.

Table of Contents

Overview

This tool—**Documentary Creator v2.0**—turns textual scripts into fully edited videos. It can:

- Draft a structured script from a topic outline.
- Chunk text into sections for image generation.
- Build rich image prompts (mood, setting, action, symbols).
- Automate generation via Sora (mouse + clipboard).
- Optionally validate images with BLIP captioning.
- Produce Ken Burns effect clips from generated or existing images.
- Stream TTS voice-over (Google Chirp3/Orus) in real time.
- Mix background music with side-chain ducking.
- Concatenate video segments, overlays, and audio into a final master.

Everything runs from a single `main.py` CLI with menu-driven modes.

Features

- **End-to-end pipeline** from topic → outline → script → images → voice → video.
- **Modular modes** for full generation, images-only, existing images, and script-only.
- **Advanced prompt engineering**: symbolic elements, mood detection, texture, dynamics, focal points.
- **Multiple sources**: Pexels, Pixabay, Videvo, Coverr, Mixkit, or your own images.
- **Real-time TTS streaming** using Google Cloud streaming API.
- **Video effects**: crossfades, fade-ins/outs, Ken Burns zoom.
- **Audio mixing**: background music ducking.

Prerequisites

- Python ≥3.9
- ffmpeg & ffprobe installed and on PATH
- Chrome installed (for Sora automation)
- Google Cloud credentials for Text-to-Speech

Installation

1. Clone this repo:

```
git clone https://github.com/yourusername/documentary-creator.git
cd documentary-creator
```

2. Create & activate a virtual environment:

```
python -m venv venv
source venv/bin/activate   # macOS/Linux
venv\Scripts\activate    # Windows
```

3. Install Python dependencies:

```
pip install -r requirements.txt
```

Configuration

- Copy `.env.example` to `.env` and fill in API keys:

```
PEXELS_API_KEY=...
PIXABAY_API_KEY=...
VIDEVO_API_KEY=...
DEEPSEEK_API_KEY=...
GOOGLE_APPLICATION_CREDENTIALS=/path/to/your-gcp-key.json
OPENROUTER_API_KEY=...
```

- Ensure `ffmpeg` and `ffprobe` are accessible in your PATH.
- Adjust any hardcoded paths (e.g. `INTRO_FONTFILE`) in `main.py` if needed.

Usage

Run the main script:

```
python main.py
```

Follow the interactive menu:

Interactive Menu Modes

1. **Full generation**: Script → images → voice → final video.
2. **Images-only**: Generate & validate images, then assemble video.
3. **Compile existing images**: Use pre-saved `image_<n>.png` files to build video.
4. **Script only**: Generate script with DeepSeek outline & sections.
5. **Exit**

Each mode prompts for the necessary inputs (paths, topic, tone, lengths, etc.) and prints progress indicators.

Core Components

1. Script Generation

- **Outline**: Uses DeepSeek (`deepseek-reasoner`) to draft 10–20 bullets.
- **Section Expansion**: Follows strict word counts, style rules, cliché avoidance, and self-critique.
- **Output**: Saves `script_YYYYMMDD_HHMMSS.txt` with `OUTLINE` and `Section X:` headers.

2. Image Prompt Pipeline

- **Text chunking**: Splits script into `total_images` semantic chunks.
- **Prompt builder**: Assembles `[SUBJECT]`, `[SETTING]`, `[ACTION]`, `[MOOD]`, `[TEXTURE]`, etc.
- **Review**: Optional OpenRouter step to refine prompt into cinematic masterpiece.

3. Image Generation with Sora

- **Browser automation**: Opens Chrome to Sora, pastes prompts, downloads screenshots as `.mp4` clips.
- **Validation**: Captions each clip's first frame via BLIP and computes semantic similarity to prompt.

4. Ken Burns & Title Cards

- **Ken Burns**: `generate_kenburns_clips()` applies zoompan + fade filters to `image_<n>.png` → `clip_<n>.mp4`.
- **Title card**: `create_title_card()` uses ffmpeg drawtext on a black background with fade-in/out.

5. Voice-Over Streaming

- **Chunking**: `text_generator()` splits script into ≤500-char sentences.
- **Streaming**: `synthesize_streaming()` yields audio chunks from Google Cloud TTS.
- **Assembly**: Writes per-chunk `.wav`, then concatenates or uses ffmpeg concat.

6. Final Assembly

- **Concatenation**: `concat_with_crossfade()` or simple concat demuxer to merge `clip_*.mp4`, `intro.mp4`, `outro.mp4`.
- **Audio mux**: `mux_audio_video()` to add voice/music.
- **Color grading**: Optional 3D LUT via `apply_color_grading()`.
- **Cleanup**: Removes temporary files on completion.

API Keys & Environment Variables

- `PEXELS_API_KEY`, `PIXABAY_API_KEY`, `VIDEVO_API_KEY`: Stock video providers.
- `DEEPSEEK_API_KEY`: DeepSeek script generation.
- `GOOGLE_APPLICATION_CREDENTIALS`: Path to GCP service account JSON.
- `OPENROUTER_API_KEY`: For prompt refinement (optional).

Dependencies

See `requirements.txt`. Key libraries:

- `transformers`, `sentence-transformers`, `torch`
- `google-cloud-texttospeech`, `openai`, `google-generativeai`
- `spacy`, `nltk`, `language-tool-python`, `textstat`
- `opencv-python`, `mss`, `pyautogui`, `pyperclip`
- `bs4`, `requests`, `PyPDF2`

Contributing

1. Fork & create a feature branch.
2. Write clear code & docstrings.
3. Add tests if modifying core logic.
4. Submit a pull request.

License

This project is released under the MIT License. See LICENSE for details.