Thanks, Heiko — that means a lot. You're building something both clever and user-friendly, and it's been a blast helping shape it.

Here’s a structured technical description document summarizing your current subtitle synchronizer application and the features we've refined together:

**🧠 Subtitle Synchroniser – Technical Summary (v1.5.15)**

**🔧 Purpose**

A Python-based GUI application designed to align subtitles with video/audio using the faster-whisper transcription model and display synchronized output in real time. Includes interactive controls, status management, and future support for advanced editing and manual sync workflows.

**🏗️ Core Architecture**

**Main Class**

SubtitleSyncApp(tk.Tk)

Encapsulates the full application logic, UI setup, threading, file handling, and synchronization mechanics.

**Key Dependencies**

* tkinter for UI
* ffmpeg-python for audio extraction
* faster-whisper for transcription
* wave, threading, os, time for system and audio management

**🪟 GUI Structure**

**Top Menu Bar**

* **Select Components**: Load video, subtitle, export paths
* **Settings**: Adjust flush preview frequency and whisper beam size

**Main Grid**

* **Left Pane**: Original subtitle TreeView
* **Right Pane**: Whisper or synced output TreeView
* **Timestamp Label**: Shows last update from Whisper preview
* **Status Bar**: Colored status with beam size and component readiness
* **Progress Bar**: Displays transcription progress
* **Status Label**: Descriptive task updates (e.g. “Transcribing 30%”)

**Control Row**

* **Pause Button**: Temporarily halts Whisper loop
* **📡 Live Button**: Toggles auto-scroll on the right pane
* **Stop Button**: Aborts ongoing transcription
* **Start Sync Button**: Triggers the full pipeline (enabled only when all paths are set)

**🔄 Event Handling & Threads**

* File selectors update status and enable the Start Sync button only when all paths are selected.
* Transcription runs in a **background thread**, writes live .srt output, and updates the right pane on every preview flush using self.root.after(...).

**📡 Live Scroll Toggle**

* Controlled via self.live\_scroll\_enabled
* Clicking button toggles between:
  + "📡 Live" (auto-scroll enabled)
  + "📴 Manual" (auto-scroll paused by user)
* Updates button text and style using .config(...)

**🔁 Transcription Loop Behavior**

* Extracts audio from video using FFmpeg.
* Transcribes in real time with faster-whisper, flushing:
  + preview\_buffer → for intermediate display
  + whisper\_buffer → for Whisper output .srt file
* Right TreeView updates via scheduled main thread calls.

**🟢 Status & Readiness Indicators**

* Component readiness shows via [✓] and [✗] labels and tooltips.
* Status bar background toggles:
  + Pale green (#d0ffd0) when all components are ready
  + Light pink (#ffe4e1) during idle or incomplete state

**✨ UI Enhancements**

* **Start Sync Button Pulse**: When all paths are ready, it pulses between two greens to signal interactivity.
* **Dynamic tooltips**: File tooltips show full path or placeholder (“No video selected”)
* **Relief toggling**: Live button changes depth based on active/inactive state

**🧩 Key Internal Flags**

| **Variable** | **Purpose** |
| --- | --- |
| self.pause\_flag | Indicates pause status during sync |
| self.stop\_flag | Triggers graceful interruption |
| self.auto\_scroll\_right | Controls TreeView auto-scroll |
| self.live\_scroll\_enabled | Tracks live button state |

**🌱 Planned V2 Features (as outlined)**

* File Properties menu (video/subtitle metadata via FFmpeg and stat)
* Filter and extract TV episode info from filenames
* Manual sync using previously generated Whisper + subtitle
* In-pane editing of subtitle lines
* Show/hide Whisper/final output in either pane
* Filter selections for matching series/episode across folders
* Session persistence and cleaner modular workflows

Let me know if you'd like this exported as a .txt or .md file — or if you’d prefer it formatted as an inline help menu within the app. We can easily evolve this into real documentation or tooltips for users down the line.