WhisperX Pipeline Setup Checklist

Word-level timestamps + speaker diarization on your own infrastructure

1 · The four-stage pipeline

#	Stage	Model	What it gives you
1	Voice activity detection	pyannote VAD	Find speech; Cut & Merge into ~30 s chunks on silence
2	Transcription (batched)	Whisper / faster-whisper	What was said; up to 70x real time
3	Forced alignment	wav2vec2	When each word was said (~50 ms)
4	Diarization	pyannote.audio	Who said it — Speaker A / B labels

2 · Self-host WhisperX vs a hosted API

Criterion	WhisperX (self-host)	Hosted API
Cost (batch)	GPU time only (free software)	Per-hour fee, vendor-tuned
Word timestamps	~50 ms via forced alignment	Built in
Diarization	pyannote, free, needs HF token	Built in, one flag
Data location	Stays on your servers	Sent to vendor cloud
Streaming	No — batch only	Yes
Best when	On-prem; steady high volume	Ship fast; spiky volume; streaming

3 · Pre-flight checklist before you ship

- ☐ Confirmed you actually need word timing or speaker labels — otherwise plain Whisper / faster-whisper is simpler.
- ☐ Created a Hugging Face token, accepted the gated pyannote model terms, and wired the --diarize flag.
- ☐ Handled the alignment fallback: aligned words carry tight times; numbers/symbols inherit the segment time — flag those as approximate.
- ☐ Tested timing on audio full of numbers, prices, and proper nouns, not just clean prose.
- ☐ Cleaned noisy input first — wav2vec2 alignment degrades faster than transcription on noisy audio.
- ☐ Checked data residency: can the audio legally leave your servers, or must it stay on-prem?
- ☐ Planned a human check on diarization where wrong speaker labels matter (medical, legal).
- ☐ Modelled cost at real batch volume (GPU-hours = audio-hours / realtime-factor) and compared to a hosted API.
- ☐ Load-tested the GPU pipeline at expected peak; confirmed token does not expire silently in production.