Date: 2025 년 8 월 21 일 📅
Prepared by: Tiong-Sik Ng 📝

English 🇺🇸

1. Summary of Accomplishments ✅
   • Implemented **pseudo-streaming** in FastAPI: sentence-level chunking with a strict **generate → upload → generate → upload** pipeline, so parts (part0/1/…/final) land as soon as they're ready and perceived latency drops.
   • Achieved **sequential playback** in Amazon Connect: batches now play **batch 0 → batch 1 → … → batch N** and the call **ends cleanly** when the final sentinel is detected.
   • Fixed flow wiring so **NextIndex** increments properly (no more replay of part1).
   • Hardened Lambda/flow plumbing: unified `_build_batch(..., include_prompts=True)` to return presigned URLs (+ optional PromptARNs), added ready-polling for keys, and ensured `start_stream` can hand back the very first playable part as soon as it appears.
2. Issues & Risks 🔄
   • If **part0** isn't ready within the short poll window after `start_stream`, first playback can still wait for the next loop. (Mitigation: brief Wait + retry before falling back to loop.)
   • Keep **BATCH/limit** in Lambda aligned with the flow; a mismatch can repeat indexes or skip parts in edge cases.
3. Next Steps 📌
   • **Integrate Chatbot + TTS streaming** so a caller "talks to" the chatbot by phone with a **consistent voice** every time.
   – Add a `chat_and_stream` Lambda action (ASR text → CHAT_URL → answer → `start_stream`).
   – Keep first chunk short to start audio fast; continue with `get_next_batch` loop.
   – Optional: barge-in/stop, lifecycle for `part*.wav`, and prompt caching.

한국어 🇰🇷

1. 주요 성과 ✅
   • FastAPI 에 **의사 스트리밍** 구현: 문장 단위로 **생성 → 업로드 → 생성 → 업로드** 흐름을 보장하여 part0/1/…/final 이 준비되는 즉시 업로드되도록 개선.
   • Amazon Connect 에서 **순차 재생** 달성: **배치 0 → 배치 1 → … → 배치 N** 순서로 재생되고, 마지막 센티널 감지 시 **통화가 정상 종료**.
   • 플로우에서 **NextIndex** 증가 문제 해결(동일 파트 반복 재생 제거).
   • Lambda/플로우 보강: `_build_batch(..., include_prompts=True)`로 프리사인 URL(+선택적 PromptARN) 제공, 키 준비 폴링 추가, `start_stream` 후 가능한 한 빠르게 **첫 파트 재생** 연결.
2. 문제/리스크 🔄
   • `start_stream` 직후 **part0** 가 짧은 폴링 시간 내 준비되지 않으면 첫 재생이 다음 루프까지 대기할 수 있음(짧은 대기 후 재시도로 완화).
   • Lambda 의 **BATCH/limit** 값과 플로우의 루프 로직이 어긋나면 인덱스 반복/누락 가능.

3. 다음 단계 📌

   • **챗봇 + 스트리밍 TTS 통합**: 발화 → 챗봇 응답 → `start_stream` → `get_next_batch` 루프 흐름으로, **항상 동일한 음성**으로 통화 경험 제공.

   – `chat_and_stream` 액션 추가, 첫 파트는 빠르게 시작되도록 문장 길이 튜닝.

   – (옵션) 바지인/중지, `part*.wav` 수명주기, 프롬프트 캐싱.