Date: 2025 년 8 월 14 일 📅
Prepared by: Tiong-Sik Ng 📝

English 🇺🇸

1) ✅ Summary of Accomplishments

- **End-to-end capture & caching**: Successfully captured caller utterances, generated Chat → TTS text, and saved **`input_text`** / **`final_text`** / **`bucket`** / **`key`** / **`prompt_arn`** into DynamoDB. Instant reuse for identical utterances (cache hit).
- **Audio + Prompt pipeline**: Generated audio, uploaded to S3, and created/updated the **Amazon Connect Prompt**. Implemented sidecar writes (`.txt`, `.prompt`) next to the `.wav` for fast lookups.
- **Contact flow**: Minimal flow operational. Able to **retrieve the exact same response** for repeated utterances and **play immediately**.
- **Bug fixes & hardening**
  - Fixed DynamoDB **reserved word** issue (`bucket`) by aliasing in `UpdateExpression`.
  - Taught `start` to return **top-level `prompt_arn`** on cache hit (so the player can read `$.External.prompt_arn`).
  - Hardened text extraction so **`user_input`** is reliably captured (avoids `{}` when maps are passed).

2) 🔄 Issues Faced

- **Cold utterances**: First-time phrases require generation time; current flow ends the call before audio is ready.
- **Near-duplicate/fuzzy inputs**: Variants like “테스트 / 데 / 데스트” each generate new cache entries and audio (cache fragmentation).
- **Latency work pending**: Need to integrate **FishSpeech quick inference** once the call flow is fully stabilized.

3) 📌 Next Steps

- **Wait-loop in flow**: After `start`, use **Check contact attributes** on `$.Attributes.ready == true`; else **Invoke `check` → Wait 1–2s → loop** with a max-attempt counter and a polite “please hold” prompt. On timeout, route to agent/callback.
- **Use `pk` in polling**: Pass `pk` from `start` to `check` so `check` can read DDB and return the **ARN immediately** when `state=ready`, even if S3 HEAD lags.

- **Error surfacing**: Ensure worker marks `state="error"` with `error_message`; in flow, branch on `$.Attributes.error == 'worker_failed'` → fallback message/escalation.
- **Fuzzy-match canonicalization**: Add a lightweight **`phonetic_key()`** (Hangul NFKC + jamo simplification + small edit-distance threshold) so near-duplicates map to the same pk.
- **Prewarm top phrases**: Small scheduled job to pre-generate frequent prompts to avoid first-call latency.
- **Finalize flow**: Clean transitions, increase Lambda timeouts where needed, and confirm Connect prompt read permissions on the S3 prefix.

한국어 🇰🇷

1) ✅ 주요 작업 요약

- **발화 캡처 및 캐시**: 고객 발화를 성공적으로 수집하여 챗봇 → TTS 텍스트 생성 후 **`input_text`** / **`final_text`** / **`bucket`** / **`key`** / **`prompt_arn`** 를 DynamoDB 에 저장. 동일 발화는 **즉시 재생**(캐시 히트).
- **오디오 + 프롬프트 파이프라인**: 오디오 생성 → S3 업로드 → **Amazon Connect 프롬프트** 생성/업데이트. `.wav` 옆에 `.txt`, `.prompt` 사이드카 파일 저장으로 조회 속도 향상.
- **컨택트 플로우**: 최소 플로우 동작. **동일 발화 재생은 즉시** 가능.
- **버그 수정 및 보강**
    - DynamoDB **예약어(`bucket`)** 문제 해결(속성 이름 별칭 처리).
    - 캐시 히트 시 `start` 가 **최상위 `prompt_arn`** 반환(플레이어가 `$.External.prompt_arn` 로 바로 참조).
    - 텍스트 추출 로직 보강으로 **`user_input`** 정상 반영(맵 전달 시 `{}` 저장 문제 방지).

2) 🔄 문제점

- **신규 발화 대기**: 최초 발화는 생성 지연이 발생하며, 현재 플로우가 기다리지 못하고 통화가 종료됨.
- **유사/오타 처리**: “테스트 / 데 / 데스트” 등 유사 발화가 각각 새 항목을 만들어 캐시가 분산됨.
- **지연 최적화 미적용**: **FishSpeech 빠른 추론** 연계가 남아 있음.

3) 📌 향후 계획

- **대기 루프 도입**: `start` 이후 `$.Attributes.ready == true` 검사, 아니면 **`check` 호출** → **1–2 초 대기** → **반복**(최대 시도 횟수 및 안내 음성 포함). 타임아웃 시 상담원 연결/콜백 제공.

- **pk 활용**: `start` 가 반환한 `pk` 를 `check` 에 전달하여 DDB 상태가 `ready` 면 **즉시 ARN 반환** (S3 HEAD 지연 무시).
- **에러 처리 노출**: 작업자 실패 시 `state="error"`와 `error_message` 저장, 플로우에서 `$.Attributes.error == 'worker_failed'` 분기 후 대체 안내/상담원 연결.
- **유사 발화 정규화**: 가벼운 **`phonetic_key()`**(한글 정규화 + 자모 단순화 + 소거리 편집 거리)를 적용해 유사 발화를 **같은 pk** 로 매핑.
- **상용 문구 프리워밍**: 자주 쓰는 문구를 주기적으로 미리 생성하여 최초 지연 제거.
- **플로우 마무리**: 전이 정리, 람다 타임아웃 점검, Connect 의 S3 읽기 권한 재확인.

Notes / Remarks

- 캐시 히트 경로에서는 `start` 만 호출해 **즉시 `prompt_arn`** 를 받아 플레이 가능. 콜드 미스 시에는 `check` 기반 폴링이 필요.
- FishSpeech 최적화 적용 전까지는 **프리워밍 + 대기 루프** 조합이 체감 지연을 가장 잘 줄일 전망.