Android Developer - Take Home Assignment

Overview

Build a voice recording app with transcription and summary generation that handles real-world edge cases.

Timeline: 2 days (48 hours)
Tech Stack: Kotlin, Jetpack Compose, MVVM, Coroutines & Flow
Minimum SDK: API 24 (Android 7.0)

What You'll Build

3 core features only:

1. **Record Audio Robustly** - Background recording with explicit interruptions handling
2. **Generate Transcript** - Convert audio to transcript.
3. **Generate Summary** - Create structured summary from transcript.

Requirements

1. Record Audio

Recording Service:

- Foreground service that records audio
- Split into 30-second (audio) chunks
- Save chunks to local storage
- Persistent notification with Stop action

Critical Edge Cases to Handle:

1. **Incoming/Outgoing phone calls**
    - Pause recording when call starts
    - Show status: "Paused - Phone call"

    - Resume when call ends
2. **Audio focus loss**
    - Pause when other apps take audio focus
    - Show ‘Paused – Audio focus lost’ as a persistent foreground notification with Resume/Stop actions.
    - Resume when focus regained
3. **Microphone source changes**
    - Bluetooth headset connect/disconnect → continue recording
    - Wired headset plug/unplug → continue recording
    - Show notification when source changes
4. **Low storage**
    - Check storage before starting
    - Stop gracefully if storage runs out
    - Show error: "Recording stopped - Low storage"
5. **Process death recovery**
    - Persist session state in Room and enqueue a termination worker to finalize the last chunk and resume transcription on restart.
6. **Silent Audio Detection**
    - Recording is silent (no audio input)
    - Detect after 10 seconds of silence
    - Show warning: "No audio detected - Check microphone"
7. **Record 30-second chunks with ~2-second overlap to preserve speech continuity across chunk boundaries.**
8. **Live Updates - Android 16**

    Show live recording status on lock screen:

    - Recording timer (updates every second)
    - Current status ("Recording" / "Paused - Phone call")
    - Pause/Stop actions
    - Visual indicator (recording icon)

UI:

- Single recording/post meeting screen
- Record/Stop button
- Timer (00:00)
- Status indicator:
    - "Recording..."
    - "Paused - Phone call"
    - "Paused - Audio focus lost"
    - "Stopped"
- List all the meeting on the Dashboard

---

2. Generate Transcript

- Upload chunks as and when the 30 second chunk is ready to transcription API
- Use **OpenAI Whisper** or **Google Gemini 2.5 Flash** or **mock**
- Save to Room database and keep as the single source of truth.
- Transcript must be in correct order.
- Retry transcribing **ALL** the audio if there is some failure.
- Don't lose audio chunks

---

3. Generate Summary

- Send transcript to LLM API
- Generate structured summary and **stream it in the UI.**
- Update the UI as and when the summary response comes.
- Show specific error message
- The summary should get generated even if the user kill the app while generating the summary

UI:

- Summary screen with 4 sections:
    - Title
    - Summary
    - Action Items
    - Key Points
- Loading state: "Generating summary..."
- Error state with Retry button

Technical Requirements

Architecture

- **MVVM:** ViewModel → Repository → DAO/API
- **Hilt:** Dependency injection
- **Room:** Local database
- **Retrofit:** API calls (or mock)
- **Compose:** 100% Jetpack Compose

- **Coroutines + Flow:** Async operations

Submission Checklist:

1. Android APK (Debug Build)

2. Public GitHub Repository (Link to the code)

3. A screen recording demonstrating the app flow:

Video walkthrough of the app:

TwinMind Tutorial Video - YouTube