Attack Capital Mobile Engineering Challenge

About Attack Capital

We're a venture studio that builds and scales software companies at breakneck speed. We don't do MVPs—we ship production systems that handle real users from day one. Our mobile engineers are full-stack hackers who can build, deploy, and scale apps that actually work when things go wrong.

The Assignment: Medical Transcription App

Build a Flutter app that doctors can trust with their patient consultations.

The app records audio during medical visits and streams it to a backend for AI transcription. Sounds simple? Here's the catch—it must work flawlessly when:

- Doctors get phone calls mid-recording
- They switch to other apps to check drug databases
- The hospital WiFi drops out
- Their phone dies at 60% battery (happens more than you'd think)

Core Requirements

1. Real-Time Audio Streaming

- Stream audio chunks to backend during recording (not after)
- Continue recording with phone locked or app minimized
- Handle chunk ordering, retries, and network failures
- **Must demonstrate native microphone access with proper gain control**

2. Bulletproof Interruption Handling Must survive without losing data:

- Phone calls (auto pause/resume)
- App switching (EMR, calculator, camera)
- Network outages (queue locally, retry when back)
- Phone restarts (recover unsent chunks)
- Memory pressure (when system kills other apps)

3. Theme & Language (State Management Test) Must survive with no restart required:

- Manual + system **dark/light mode** (persisted)
- **English/Hindi** full UI language switching (persisted, no restart required)

Technical Stack

- Flutter (no Expo/React Native - we need native performance)
- Platform channels for native features when needed
- Android foreground service + iOS background audio

API Reference & Resources

📚 **Full API Documentation**: https://docs.google.com/document/d/1hzfry0fg7qQQb39cswEychYMtBiBKDAqIg6LamAKENI/edit?usp=sharing

🔧 **Postman Collection** (mock backend): https://drive.google.com/file/d/1rnEjRzH64ESlIi5VQekG525Dsf8IQZTP/view?usp=sharing

Key endpoints you'll use:

None
```
# Session Management
POST /v1/upload-session          # Start recording
POST /v1/get-presigned-url      # Get chunk upload URL
PUT  {presignedUrl}              # Upload audio chunk
POST /v1/notify-chunk-uploaded   # Confirm chunk received

# Patient Management
GET  /v1/patients?userId={userId}
POST /v1/add-patient-ext
GET  /v1/fetch-session-by-patient/{patientId}
```

Deliverables

1. Working Mobile App

- GitHub repo: `github.com/[you]/ai-scribe-copilot`
- **Android**: APK download link in README
- **iOS**: Loom/screen recording showing all features
- Build instructions that actually work

2. Platform Requirements

Android (APK Required)

```shell
flutter build apk --release
# Upload to GitHub Releases
# Include direct download link in README
```

iOS (Loom Video Required)

- Record comprehensive Loom showing all features on iPhone
- Include simulator or real device demo
- Show native features working (camera, mic, share)
- Link video prominently in README

Alternative iOS Options:

- TestFlight link (if you have Apple Developer account)
- Web build deployed to Vercel/Netlify for basic testing
- Codemagic CI/CD artifacts

3. Mock Backend

- Use the postman collection as reference to build a backend for the app
- One-command Docker deployment: `docker-compose up`
- Deploy live

4. Demo Video (5 minutes) Single Loom showing both platforms:

- **3-5 minute recording** with phone locked
- Phone call interruption with auto-recovery
- **Native features**: camera for patient ID, microphone levels, share sheet
- Network dead zone with queued uploads
- Heavy multitasking without data loss

Pass/Fail Test Scenarios

Test 1: Start 5-minute recording → Lock phone → Leave locked
Pass: Audio streams to backend, no data loss

Test 2: Recording → Phone call → End call
Pass: Auto-pause, auto-resume, no audio lost

Test 3: Recording → Airplane mode → Network returns
Pass: Chunks queue locally, upload when connected

Test 4: Recording → Open camera → Take photo → Return
Pass: Recording continues, proper native integration

Test 5: Recording → Kill app → Reopen
Pass: Graceful recovery, clear session state

Native Feature Requirements

Critical: Show you can use phone hardware properly

Microphone

- Audio level visualization
- Gain control (not just on/off)
- Handle Bluetooth/wired headset switching

System Integration

- Native share sheet (not custom UI)
- System notifications with actions
- Haptic feedback on key actions
- Respect Do Not Disturb mode

Bonus Points (+30pts)

On-Device Speech Recognition (+15pts)

- Live transcription preview during recording
- Uses platform speech APIs (iOS Speech, Android SpeechRecognizer)
- Shows you understand native platform capabilities

Professional Polish (+15pts)

- Adaptive icons (Android) / alternate icons (iOS)
- Platform-specific UI (Material You on Android 12+)
- Accessibility: Screen reader support, dynamic type

How We'll Test Your Code

Shell
```shell
# Clone and build - must work first try
```

```
git clone https://github.com/[you]/ai-scribe-copilot
cd ai-scribe-copilot
flutter pub get

# Android - install APK from your GitHub releases
# iOS - watch your Loom video for functionality

# Build locally if needed
flutter build apk --release
flutter build ios --simulator
```

Your README must include:

- 📱 **Android APK download link** (GitHub Releases)
- 🎥 **iOS Loom video link** (showing all features)
- 📚 Link to API documentation
- 🔧 Link to Postman collection
- Flutter version: `flutter --version` output
- Backend deployment URL

Evaluation (15 minutes)

1. Download and install APK (2 min)
2. Watch iOS Loom video (3 min)
3. Test recording with interruptions (5 min)
4. Verify streaming to backend (3 min)
5. Check native features work (2 min)

Scoring

- **Native Platform Mastery** (35pts): Proper hardware access
- **Real-time Streaming** (25pts): Chunks upload during recording
- **Interruption Resilience** (20pts): Survives all scenarios
- **Cross-Platform** (20pts): APK + iOS Loom provided
- **Code Quality** (15pts): Builds first try, clean code
- **Polish** (15pts): Feels native, not web wrapper

Instant Fail:

- No APK provided

- No iOS demonstration (Loom/video)
- Can't build from source
- Fake streaming (uploads after recording ends)
- Native features don't work properly

Required Submission Checklist

✅ GitHub repo with Flutter source code
✅ Android APK download link
✅ iOS Loom video demonstrating all features
✅ Live backend URL (deployed)
✅ Docker setup for backend (`docker-compose up`)
✅ 5-minute demo video showing interruption handling
✅ README with all links and setup instructions

Resources Recap

- **API Documentation**: https://docs.google.com/document/d/1hzfry0fg7qQQb39cswEychYMtBiBKDAqIg6LamAKENI/edit?usp=sharing
- **Postman Collection**: https://drive.google.com/file/d/1rnEjRzH64ESIIi5VQekG525Dsf8IQZTP/view?usp=sharing

---

Time estimate: 3 days. We're looking for engineers who understand native mobile development, not just cross-platform frameworks. Show us you can ship production apps that leverage platform capabilities.

Submit your APK and iOS Loom—we want to see it working, not just read about it.