Sprint 1 Report - ASR Baseline and Limitations Study

1 Objective

The goal of this sprint was to benchmark existing automatic speech recognition (ASR) systems and establish a Whisper-based baseline for multilingual meeting transcription. We evaluated leading ASR tools on **English**, **Arabic**, and **code-switch** (English–Arabic) recordings under both clean and noisy conditions to identify limitations and define the scope for model improvement in future sprints.

2 Experimental Setup

Audio Samples	3 short meeting recordings (English / Arabic / code-switch)
Models Evaluated	Google Speech-to-Text (Pro API v2), Google Docs Voice Typing, Otter.ai (free demo), Whisper Large (open-source)
Evaluation Metrics	Word Error Rate (WER), Noise robustness, Code-switch handling, Speaker separation (qualitative)

Whisper Inference Settings

```
result = model.transcribe(
    audio_path,
    language=language,
    prompt=prompt,
    temperature=0.0,
    beam_size=5,
    best_of=5,
    patience=2,
    condition_on_previous_text=True
)
```

3 Results

Language / Condition	Google Pro	Google Docs	Otter.ai	Whisper Large
English (no noise)	**0.18**	0.20	0.20	0.20
English (+ noise)	0.21	0.19	**0.17**	0.29
Arabic (no noise)	0.21	0.21	–	**0.19**
Arabic (+ noise)	0.22	0.22	–	**0.21**
Code-switch (no noise)	0.55	–	–	**0.35**
Code-switch (+ noise)	0.45	–	–	**0.28**

4 Observations & Insights

- **English recordings:** Google Pro achieved the lowest WER in clean settings, while Otter.ai was most resilient under noise. Whisper slightly underperformed in noisy conditions.

- **Arabic recordings:** Whisper Large marginally outperformed Google models, demonstrating better handling of Arabic phonetics and dialectal variations.

- **Code-switch recordings:** Whisper Large outperformed Google Pro, achieving lower WER (0.35 vs 0.55 in clean settings, 0.28 vs 0.45 in noisy settings). Google sometimes incorrectly transcribed English words using Arabic letters, while Whisper preserved most English words correctly.

5 Limitations & Gaps

Limitation	Observation	Impact	Improvement Goal
Noise Robustness	Whisper WER ↑ from 0.20 → 0.29 with noise	Lower accuracy in real meetings	Integrate noise suppression or augmentation
Code-Switch Support	Google fails on mixed-language speech	Misinterpretation of bilingual users	Fine-tune Whisper using code-switch corpora
Speaker Diarization	Missing in all baseline models	Harder to extract per-speaker insights	Add diarization using WhisperX/Pyannote
Latency / Speed	Whisper Large = slow inference	Delayed output	Explore smaller or quantized models
Context Retention	Whisper resets context across segments	Fragmented transcripts	Enhance with prompt conditioning

6 Next Steps

Sprint 3 will focus on:

- **Speaker Diarization Integration:** Implement WhisperX or Pyannote.audio for speaker segmentation and diarization, enabling speaker-labeled transcripts in meetings (Sprint 3a).

- **Summarization & Task Extraction:** Generate concise meeting summaries and extract action items using T5/BART with rule-based task extraction (Sprint 3b).

- **Code-Switch Fine-Tuning:** Improve bilingual transcription using the ArzEn Speech Corpus (Egyptian Arabic–English) or the Arabic-Whisper-CodeSwitching-Edition model (subject to discussion).

- **Noise & Accent Robustness:** Apply preprocessing (noise suppression, filtering) and data augmentation (speed, pitch, volume) for real-world meeting conditions.

- **Streaming Optimization (pre-Sprint 7):** Explore lightweight or quantized Whisper variants for near-real-time transcription.

7 Conclusion

This sprint established a strong empirical baseline for multilingual ASR performance in realistic meeting conditions. Whisper Large demonstrated superior multilingual and code-switch capabilities compared to commercial baselines, while Google and Otter.ai showed better noise resilience.

Future improvements will focus on adaptation rather than architecture: fine-tuning for bilingual/dialectal contexts, integrating speaker-aware transcription pipelines, and enhancing noise robustness. The combination of WhisperX/Pyannote and fine-tuning on ArzEn or the Arabic-Whisper-CodeSwitching-Edition model is expected to bridge the gap between baseline performance and a practical, speaker-aware meeting assistant.