1. Data Collection

- **Core Principle:** Prioritize capturing the full spectrum of speech as it occurs in the real-world Grab driving context.
- **Elements:**
    - **Language(s):** Collect data in all relevant languages used by Grab drivers in the target region (e.g., Malay, English, Chinese).
    - **Speaking Style:**
        - Formal: Drivers reading navigation prompts or scripted interactions.
            - Occurrences: Clear pronunciation, consistent speed, minimal disfluencies.
        - Spontaneous: Drivers giving directions to passengers, asking questions, or reacting to road events.
            - Occurrences: Colloquial expressions, slang, faster speech, hesitations, "ums" and "ahs," sentence fragments.
    - **Speech Context:**
        - Commands: "Navigate to...", "Call the passenger," "Cancel the trip."
            - Occurrences: Concise phrasing, specific keywords.
        - Requests: "Can you take the Lebuhraya...", "Where is the nearest...", "I need to avoid...".
            - Occurrences: Politeness markers, indirect requests.
        - Passenger Interaction: Greetings, confirmations, clarifications.
            - Occurrences: Highly variable phrasing, background passenger speech.
    - **Acoustic Environment:**
        - Clean: Parked car, engine off, minimal external noise.
            - Occurrences: Baseline recordings for comparison.
        - In-Car Noise:
            - Engine noise (idling, accelerating, braking).
            - Road noise (different speeds, road surfaces - asphalt, gravel, highway, city).
            - Wind noise (windows up/down).
            - Rain noise (light to heavy).
            - Music/Radio.
            - Other passengers talking.
            - Phone call audio (both driver and other end).
            - In-cabin ambient noise (music)
            - Device generated sounds (navigation prompts).
            - Occurrences: Overlapping noise sources, varying noise levels (quantified by SNR).
    - **Driver Variability:**

        - Accents and Dialects: Capture regional and individual variations.
            - Occurrences: Differences in pronunciation, intonation, word choice.
        - Speech Speed: Fast, normal, slow.
            - Occurrences: Hurried vs. relaxed speech.
        - Emotional State: Calm, stressed, hurried, frustrated.
            - Occurrences: Changes in pitch, volume, speech rate.
    - **Recording Setup:**
        - Microphone position: Built-in car mic, dashboard mic, wearable mic.
            - Occurrences: Variations in audio quality and noise capture.
- **Output:** A large, diverse dataset of in-car audio recordings with corresponding transcriptions and metadata tags.

2. Audio Preprocessing for Clarity

- **Core Principle:** Enhance the audio signal to improve its quality and reduce noise before feature extraction.
- **Techniques:**
    - **Noise Reduction:**
        - Spectral Subtraction: Reduce stationary background noise.
            - Occurrences: Engine noise, fan noise.
        - Wiener Filtering: Minimize noise based on statistical properties.
            - Occurrences: Broadband noise.
        - Adaptive Filtering: Remove periodic noise.
            - Occurrences: Some types of engine whine.
        - Voice Activity Detection (VAD): Identify and remove non-speech segments.
            - Occurrences: Silence, coughs, etc.
    - **Filtering:**
        - High-Pass Filtering: Reduce low-frequency noise.
            - Occurrences: Rumble from the engine.
        - Band-Pass Filtering: Isolate the frequency range of speech.
            - Occurrences: Reduce both low and high-frequency noise.
    - **Signal Enhancement:**
        - Automatic Gain Control (AGC): Normalize the amplitude of the audio.
            - Occurrences: Varying driver speaking volume.
    - **Occurrences:** Preprocessing steps and their occurrences are intricately linked with the noise types and recording conditions encountered during the data collection phase.
- **Output:** Cleaned audio signals, ready for feature extraction.

3. Feature Extraction for Model Input

- **Core Principle:** Convert the preprocessed audio into a numerical representation that the ASR model can process.
- **Techniques:**
    - **Mel-Frequency Cepstral Coefficients (MFCCs):**
        - Dominant feature extraction technique in ASR.
        - Calculate the frequency spectrum of the audio and then transform it to the Mel scale (which aligns more closely with human hearing).
        - Apply Discrete Cosine Transform (DCT) to decorrelate the features.
        - Occurrences: Captures the essential characteristics of speech sounds.
    - **Log-Mel Spectrogram:**
        - Visual representation of the audio signal where the frequency scale is transformed to the Mel scale and the intensity scale is transformed to a log scale.
        - Can be used directly as input to the ASR model (Whisper).
        - Occurrences: Represents the energy of different frequencies over time.
    - **Other Features:**
        - Perceptual Linear Prediction (PLP)
        - Linear Predictive Coding (LPC)
        - Occurrences: May be used in conjunction with MFCCs or for specific model architectures.
- "The whisper feature extractor works in two steps. First, the whisper feature extractor pads the audio signal to match the 30-second duration. If audio exceeds the 30-second limit, then audio will be sliced up into two parts. If the audio signal is less than the 30-second limit, then the audio signal will be padded with zero or silence into 30 seconds duration. The next step is converting audio into an 80-channel log-mel spectrogram visual. The audio signal will first be converted into a Fast Fourier Transform (FFT) measurement. Then, this measurement will be inserted into the mel-filter bank and apply the logarithm operation. The result is a visualization of the log-mel spectrogram."
- **Output:** A sequence of feature vectors representing the audio input.

4. Data Augmentation

- **Core Principle:** Artificially expand the dataset to cover edge cases and improve the model's ability to generalize.
- **Techniques:**
    - **Realistic Noise Injection:**
        - Combine clean speech with recorded in-car noise at varying SNRs.

        - Occurrences: Simulate different driving scenarios.
    - Mix multiple noise sources (e.g., engine + rain + radio).
        - Occurrences: Create complex, real-world noise conditions.
- **Speech Modification:**
    - Vary speech speed (±10-20%).
        - Occurrences: Simulate hurried or relaxed speech.
    - Shift pitch (± a few semitones).
        - Occurrences: Simulate different driver voices.
    - Adjust volume levels.
        - Occurrences: Simulate different distances from the microphone.
- **Simulate Signal Degradation:**
    - Add reverberation.
        - Occurrences: Simulate different car interiors.
    - Introduce artificial distortion or packet loss.
        - Occurrences: Account for Bluetooth or wireless mic issues.
- **Colloquialism Synthesis:**
    - Use TTS to generate variations of phrases with different slang terms.
        - Occurrences: Expand the dataset of colloquial expressions.

- **Output:** An even larger and more varied dataset, where each original audio sample may now have several augmented versions.

5. Segmentation and Detailed Labeling

- **Core Principle:** Prepare the audio data and its corresponding text in a way that the ASR model can learn effectively.
- **Process:**
    - **Segmentation:**
        - Divide continuous audio recordings into shorter, meaningful segments.
            - Occurrences: Utterances, commands, responses, or short conversational turns.
        - Use voice activity detection (VAD) to help automate segmentation.
            - Occurrences: Identify speech vs. silence.
        - Handle overlapping speech (if possible).
            - Occurrences: Passenger interrupting driver.
    - **Transcription:**
        - Accurately transcribe each audio segment.
            - Occurrences: Pay close attention to colloquialisms, disfluencies, and words affected by noise.

        - Use a standardized orthography while noting pronunciation variations (if needed).
            - Occurrences: Balance consistency with capturing the nuances of speech.
    - **Metadata Tagging:**
        - Add tags to each segment to describe its characteristics.
            - Occurrences:
                - Language
                - Speaking Style (Formal, Spontaneous)
                - Speech Context (Command, Request, Interaction)
                - Noise Type (Engine, Traffic, Rain) and SNR
                - Driver Accent
                - Speech Speed
                - Emotional State
                - Colloquialisms Present (and which ones)
                - Microphone Position
- **Output:** A dataset of segmented audio files and corresponding text transcriptions, enriched with detailed metadata tags.

6. Data Splitting

- **Core Principle:** Divide the dataset into subsets for training, validation, and testing in a way that ensures fair model evaluation.
- **Splits:**
    - **Training Set (e.g., 70-80%):**
        - Used to train the ASR model's parameters.
        - Occurrences: The largest portion of the data, containing a wide variety of all the tagged categories.
    - **Validation Set (e.g., 10-15%):**
        - Used during training to monitor performance and tune hyperparameters.
        - Occurrences: A representative sample of the training data, used to prevent overfitting.
    - **Testing Set (e.g., 10-15%):**
        - Used to evaluate the *final* trained model's performance.
        - Occurrences: Unseen data that mimics real-world scenarios as closely as possible.
- **Considerations:**
    - **Representative Distribution:** Ensure each split contains a similar distribution of all the tagged categories (language, noise, style, colloquialisms, etc.).

        - Occurrences: Avoid having all the noisy data in the training set and all the clean data in the testing set.
    - **Speaker Independence:** Ideally, ensure that different speakers are present in different splits to improve generalization.
        - Occurrences: Avoid training on a specific driver's voice and only testing on that same voice.
    - **Stratified Sampling:** Use stratified sampling techniques to guarantee that less frequent occurrences (e.g., specific colloquialisms, high noise levels) are present in all splits.
        - Occurrences: Prevent the model from failing on rare but important events.
    - **Testing Set Categorization:**
        - Further subdivide the testing set into categories based on the metadata tags.
            - Occurrences:
                - Test set for "High Noise"
                - Test set for "Colloquialism X"
                - Test set for "Fast Speech"
                - Test set for "Specific Accent Y"
        - This allows for granular evaluation of the model's strengths and weaknesses.
            - Occurrences: Identify if the model struggles with a specific type of noise or a particular colloquialism.
- **Output:** Three distinct datasets (training, validation, testing), with the testing set further categorized for detailed analysis.

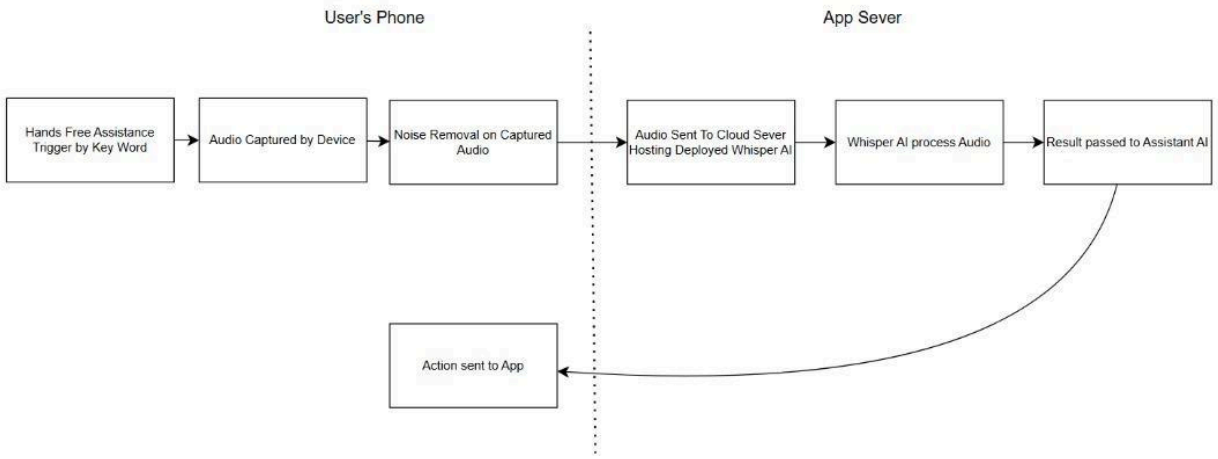
7. Model Fine-Tuning

- **Core Principle:** Adapt a pre-trained ASR model (Whisper) to the specific characteristics of the Grab driver's speech while minimizing computational cost.
- **Process:**
    - **Pre-trained Model Selection:** Choose a strong pre-trained model as a starting point.
        - Occurrences: Whisper is a good candidate due to its multilingual capabilities.
    - **Quantization :** As computational resources are limited, apply quantization to the model to reduce its size and memory footprint.
        - Occurrences: 8-bit quantization can speed up training and inference.
    - **Parameter-Efficient Fine-Tuning (PEFT):** Use techniques like LoRA to fine-tune only a small subset of the model's parameters.

        - Occurrences: Reduces the computational cost compared to fine-tuning the entire model.
    - **Fine-tuning:** Train the model on the training set.
        - Occurrences: The model learns to recognize the specific language, accents, colloquialisms, and noise conditions present in the data.
    - **Validation Set Monitoring:** Monitor the model's performance on the validation set during training.
        - Occurrences: Use validation loss and other metrics to tune hyperparameters (learning rate, batch size, etc.) and prevent overfitting.
- **Output:** A fine-tuned ASR model optimized for the Grab driver's speech patterns and in-car environment.

8. Evaluation

- **Core Principle:** Assess the model's performance comprehensively, paying close attention to its ability to handle challenging speech variabilities.
- **Process:**
    - **Overall Evaluation:** Calculate standard ASR metrics (Word Error Rate - WER) on the entire testing set.
        - Occurrences: Provides a general measure of the model's accuracy.
    - **Categorical Evaluation:** Evaluate the model's performance on each of the categorized testing subsets.
        - Occurrences:
            - Calculate WER for the "High Noise" test set.
            - Calculate WER for the "Specific Colloquialism X" test set.
            - Compare WER between "Formal" and "Informal" test sets.
    - **Error Analysis:** Analyze the types of errors the model makes.
        - Occurrences:
            - Identify common confusions (e.g., "to" vs. "two").
            - Determine if the model struggles with specific words or phrases.
            - See if errors are correlated with specific noise types or colloquialisms.
    - **Qualitative Evaluation:** Manually listen to and transcribe a sample of the model's output.
        - Occurrences: Gain a deeper understanding of the model's strengths and weaknesses.
- **Output:** A detailed report on the ASR model's performance, including overall metrics, categorical results, and error analysis, guiding further model improvement.



Scenario: A Grab driver, Adi, is navigating through heavy traffic in KL and wants to quickly reroute to avoid a traffic jam.

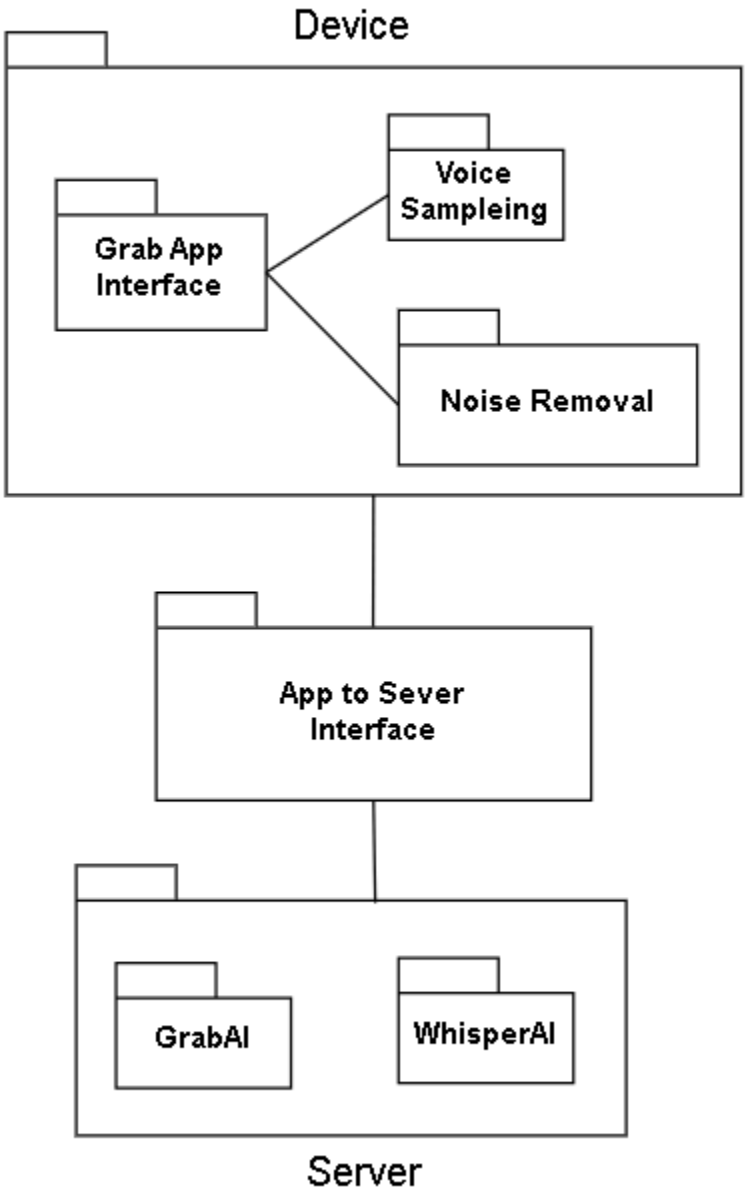
DAX Assistant - Handsfree: Behind the Scenes Workflow

1. **Microphone Input and Initial Buffering**
    - The car's microphone array (or a dedicated microphone) constantly captures the raw audio stream.
    - The incoming audio is initially stored in a small buffer. This buffer holds a few seconds of audio data.
    - Why? This buffer is crucial for:
        - Wake word detection: Giving the system a short history to analyze for the wake word.
        - Noise reduction: Some noise reduction algorithms need a short segment of audio to estimate noise characteristics.
2. **Wake Word Detection**
    - The system continuously analyzes the audio buffer for the "Hey Grab" wake word.
    - Process:
        - Feature Extraction: The audio in the buffer is converted into features suitable for wake word detection (MFCCs).
        - Wake Word Model: A small, efficient wake word detection model (Picovoice) processes these features.
        - Detection: The model calculates the probability that the wake word is present.
    - Occurrences:

        - True Positive: "Hey Grab" is correctly detected.
        - False Positive: Background noise or other speech is incorrectly identified as "Hey Grab."
        - False Negative: Adi says "Hey Grab," but it's missed by the system.
3. **Audio Preprocessing (Triggered by Wake Word)**
    - Once the wake word is detected:
        - The system may pull a slightly larger segment of audio from the buffer, including a bit before and after the wake word, to ensure the entire command is captured.
        - Audio preprocessing is applied to this segment.
    - Steps:
        - Resampling: Audio is converted to a standard sample rate (e.g., 16kHz) if necessary.
        - Bandpass Filter (300Hz–3400Hz): Frequencies outside the typical human speech range are attenuated.
        - Noise Suppression: Algorithms like RNNoise estimate the noise profile and subtract it from the audio, enhancing Adi's voice.
        - Voice Activity Detection (VAD): VAD precisely identifies speech segments, potentially discarding any remaining silence or non-speech sounds.
4. **Feature Extraction (for ASR)**
    - The preprocessed audio is transformed into a representation that the ASR model can understand.
    - Whisper Feature Extractor:
        - Padding/Slicing: The audio is padded or sliced to a fixed length (e.g., 30 seconds).
        - Log-Mel Spectrogram Conversion:
            - FFT: The audio signal is converted into its frequency components using the Fast Fourier Transform.
            - Mel Filter Banks: The frequencies are mapped to the Mel scale, which aligns better with human hearing.
            - Logarithm: The power of each frequency is converted to a logarithmic scale.
5. **Whisper ASR (Speech-to-Text)**
    - The log-Mel spectrogram is fed into the Whisper model.
    - Encoder: The encoder part of Whisper processes the spectrogram and creates a hidden representation of the audio.
    - Decoder: The decoder generates the text transcription, one word (or token) at a time, conditioned on the encoder's output and the previously generated words.

    - Output: A text transcription of Adi's command.
6. **NLP Intent & Entity Extraction**
    - The transcribed text is processed by the NLP module.
    - Tokenization: The text is broken down into individual words or sub-word units.
    - Intent Classification: The model determines the *goal* of Adi's utterance (e.g., `change_destination`, `call_passenger`).
    - Entity Recognition: The model identifies and extracts relevant pieces of information (e.g., the location "Jalan Sudirman" is extracted as a `destination` entity).
7. **Action Execution**
    - The system takes the extracted intent and entities and performs the appropriate action.
    - API Calls: This often involves calling external APIs:
        - Navigation API: To change the destination.
        - Phone API: To initiate a call.
    - Logic: The system may need to perform some internal logic:
        - Checking for errors (e.g., "Jalan Sudirman" not found).
        - Retrieving information (e.g., finding the passenger's phone number).
8. **TTS (Text-to-Speech Response)**
    - The system formulates a response to Adi.
    - Text Generation: The response is created as a text string (e.g., "Navigating to Jalan Sudirman").
    - TTS Engine: A TTS engine (Coqui TTS or Google TTS) converts the text into an audio waveform.
    - Prosody: The TTS engine adds appropriate intonation, rhythm, and emphasis to the speech.
9. **Speaker Output**
    - The generated audio is played through the car's speakers.
    - Volume Adjustment: The system may dynamically adjust the volume based on the level of background noise in the car.
    - Audio Mixing: The assistant's voice might be mixed with other audio (e.g., navigation prompts)

DAX Assistant - Handsfree: Deployment Architecture



The deployment of the application is split into device side and server side.

Device Side: Responsible for detecting, recording and preprocessing audio input.

- Grab App Interface: Integrates the voice processing programs.
- Voice Sampling: Responsible for capturing audio input and detecting wake up words.
- Noise Removal: Responsible for clearing noise from input before being sent for processing.

Server Side: Responsible for speech to text and appropriate actions.

- WhisperAI: Performs speech to text with processed audio input.
- GrabAI: The AI assistant responsible for taking actions after receiving inputs from WhisperAI.

Benefits of Speech Emotion Recognition for a Driver Assistant: (BONUS FEATURE)

- **Driver Fatigue Detection:** Detecting signs of drowsiness or exhaustion in the driver's voice (e.g., monotonous tone, slow speech rate) could prompt the assistant to suggest taking a break.
- **Stress Monitoring:** Identifying stress or agitation (e.g., raised pitch, rapid speech) could allow the assistant to offer calming suggestions or adjust its communication style to be more supportive.
- **Emergency Alert:** In extreme cases, detecting panic or distress might trigger an automatic alert to emergency contacts or a dispatch center.
- **Personalized Assistance:** Understanding the driver's emotional state can help the assistant provide more personalized and contextually appropriate responses.

Module: Speech Emotion Recognition (SER)

1. **Module Description**
    - This module enhances the DAX Assistant by enabling it to detect the driver's emotional state from their speech.
    - This capability allows for more context-aware and proactive assistance, particularly in scenarios related to driver safety and well-being.
2. **Input**

- Audio signal, preprocessed as described in the core workflow (including noise reduction, filtering, etc.).
- Specifically, the audio segment corresponding to the driver's speech after the wake word is detected and before the assistant's response.

3. **Processing**
    - **Feature Extraction (SER-specific):**
        - In addition to or instead of the features extracted for ASR, this module extracts features relevant to emotion recognition.
        - Key features include:
            - Prosodic features:
                - Pitch (fundamental frequency) and its variations
                - Energy (amplitude) and its dynamics
                - Speaking rate
                - Pauses and silences
            - Spectral features:
                - MFCCs (if not already extracted)
                - Spectral centroid
                - Spectral bandwidth
        - Feature extraction methods may include:
            - Short-time Fourier Transform (STFT)
            - Wavelet transform
    - **Emotion Classification:**
        - The extracted SER features are fed into a trained emotion classification model.
        - Model types:
            - Machine learning models: Support Vector Machines (SVM), Random Forests
            - Deep learning models: Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), Transformer networks
        - The model assigns the input audio to one or more emotion categories.
        - Typical categories:
            - Neutral
            - Happy
            - Sad
            - Angry
            - Fearful
            - Surprised
            - Drowsy / Fatigued

4. **Output**
    - The detected emotion category or a probability distribution over emotion categories.
    - A confidence score indicating the model's certainty in its prediction.
5. **Integration with Core Workflow**
    - The SER module operates in parallel with the ASR and NLP modules.
    - The emotion output is passed to the Action Execution module.
    - The Action Execution module uses this emotion information to:
        - Modify the assistant's response (content and tone)
        - Trigger specific actions (e.g., suggesting a break)
6. **Considerations**
    - **Dataset:**
        - Requires a labeled dataset of speech with emotional annotations.
        - Dataset should be representative of the target user group (e.g., accounting for cultural differences in emotional expression)
    - **Accuracy:**
        - SER accuracy is influenced by factors like noise, speaker variability, and the complexity of emotions.
        - The system should be designed to handle uncertainty in emotion predictions.
    - **Real-time Performance:**
        - Emotion recognition needs to be performed with low latency to provide timely assistance.
    - **Ethical Implications:**
        - Privacy: Handling emotion data requires careful consideration of privacy.
        - Bias: Emotion recognition models can be biased, leading to unfair or inaccurate predictions for certain groups of people.

Prototype Figma Link

https://www.figma.com/proto/f2eURAthQShgERNExYUo7S/DAX-Assistant-%E2%80%93-Handsfree?node-id=101-108&t=I5GMvqZFPDZqMdK5-1