# Model Monitoring Pipeline

*The question's vagueness and the difficulty of proposing a generalised model pipeline have motivated me to base my response on what I know about the Mercury speech-to-text pipeline's potential applications and the audio data annotation process.*

## Setting

A speech-to-text model pipeline, such as Mercury after deployment, must be monitored for performance-degrading Model Drift before it becomes functionally insufficient. Model drift can be defined as a model's training data or its relationships to output variables no longer representing the current reality. Some concrete examples of such changes include,

- New unseen themes in police transcripts, such as K-pods being an emergent topic despite rarely occurring in training data gathered a year ago (Covariate Drift)
- The word "transfer" becoming similar to "pay" after mobile payments became culturally ingrained (Concept Drift)

The proposed pipeline aims to detect and inform a team of potential model drift before it causes functional issues that affect end-users in general transcription use cases. It is designed considering the following contextual issues.

1. Data annotation makes regular labelled data generation prohibitively costly.
2. Singapore's unique and complex speech patterns make alternative data scarce.
3. Privacy concerns inhibit data access, resulting in infrequently sent batches of data.

## Model Drift Tracking Pipeline

Performance measurements are intuitive measures of a model's drift given their inherent ties to a model's value proposition. This is because varying or decreasing accuracy in production is a common and undesirable symptom of model drift. However, issue #1 precludes accuracy metrics since they depend on a ground truth, meaning result-oriented metrics must therefore be derived from a proxy.

The pipeline shall start from a tool for users to check and adjust freshly generated transcriptions, which logs finalised transcriptions and the user interactions required to create them. Unchanged or minimally changed sections indicate an accurate transcription, while larger changes indicate the opposite. Relating to the aforementioned drift examples, unfamiliar terms or semantics associated with them may reduce the accuracy of a language model's output, therefore requiring changes, which makes change tracking valuable. These may be used in simple metrics (e.g., percentage of unchanged segments) to track model drift.

Additionally, the completed transcript may act as a simple ground truth, enabling more complex metrics like word error rate. Its position as a proxy ground truth also partially alleviates issue #2.

Logs are also textual, consuming significantly less storage than raw audio, allowing batches of data to be transferred via secure SSDs inhibited by issue #3.

Adhering to security requirements, logs shall be processed on an air-gapped machine to produce summary metrics and visualizations that are less sensitive and safe for reporting, before being stored in a shared location to track metrics over time. Automatic alerting is unnecessary in this case because reading batched data from SSDs is manual and infrequent, though thresholds should be established to distinguish performance drops from natural fluctuations. However, for less sensitive or on-premise deployments, data aggregation and visualization components could be deployed on a continuously updating service, providing real-time access to tool logs and statistics and summary visualizations.

Line graph visualizations shall allow teams to track potential drift over time without exposing raw transcripts. To support deeper investigation when degradation is suspected, each batch of statistics will be tagged with identifiers correlating them with original audio files, allowing investigators to make tightly-scoped requests for poorly performing audio files.

Overall, this pipeline extracts insights from a transcription editing tool to generate ground truth proxies for performance metrics, thereby tracking model drift while considering data privacy and scarcity.