1. Methodology

- **Dataset** → A small sample dataset was manually created to simulate customer & workshop feedback (e.g., “steering wheel leather peeling”, “ABS sensor failure”, “seatbelt fraying”).
- **Preprocessing** → lowercase, remove punctuation, clean typos → cleaned_text.
- **Embeddings** → Sentence-BERT (all-MiniLM-L6-v2) converts texts into semantic vectors.
- **Clustering** → HDBSCAN (Takes Vector Embeddings as an Input)
  - Automatically detects an unknown number of clusters.
  - Handles noise/outliers (-1).
  - Groups semantically similar failure descriptions (e.g., “steering leather peeling” + “steering leather discolored”).
- **Cluster Labeling** → Generate short, meaningful headlines for each cluster:
  - **Option 1:** KeyBERT keywords.
  - **Option 2:** Hugging Face bart-large-cnn summarization → short headlines.
  - **Option 3:** WG-BERT (Warranty and Goodwill) specifically for domain.
- **Output** → CSV with feedback_id, cleaned_text, cluster_id, cluster_label.

2. Result

I built an automated pipeline that cleans messy automotive feedback, converts it into semantic embeddings, and groups similar issues using HDBSCAN—ideal since the number of failure types is unknown and the data is noisy. Each cluster is then given a clear label using keyword extraction, summarization, or W&G-BERT to identify failure type and location. The result is homogeneous clusters with meaningful headlines, scalable across any car component, and achieved without manual annotation.

3. Scalability Note

I have built a sample project based on the Objective to demonstrate the approach.

- This pipeline can be scaled up depending on dataset size and requirements.
- Embedding and labeling models can be upgraded (e.g., mpnet-base-v2, GPT-based summarization) to improve accuracy for large datasets.
- However, the workflow remains the same:

Preprocessing → Embeddings → Clustering → Labeling → Output.