# Project Overview

This report provides a comprehensive documentation of the development and evaluation process carried out in this project. It summarizes the datasets used, the database design and API integration, and the experiments conducted on multiple models for different natural language processing tasks. The work completed is organized as follows:

1. **Dataset Used** — Description of the datasets collected, cleaned, and prepared for model training.
2. **Database and API** — Explanation of the database structure and its integration through an API to support data storage and retrieval.
3. **Models Training and Testing**
   a. **Named Entity Recognition (NER) Models** — Selection, training, testing, and comparison of different NER approaches.
   b. **Classification Models** — Development, training, and evaluation of models for sentiment and aspect-based classification.
   c. **Speech-to-Text (STT) Models** — Selection, fine-tuning, and performance evaluation of various speech recognition systems.

# Dataset "HEAR"

During the first days of working, I came along a Dataset named HEAR dataset. The project uses a dataset of Arabic patient feedback collected from Saudi Arabian hospitals. The dataset captures real-world patient experiences, including sentiments about hospital services, medical staff, and facilities. It is designed to support aspect-based sentiment analysis and multi-aspect classification simultaneously.

## Dataset Composition

The dataset contains approximately 30,000 records. The data is primarily textual patient feedback written in Modern Standard Arabic or Saudi dialect expressions. The feedback was collected from multiple sources, including hospital surveys, feedback forms, and online reviews (e.g., Google Maps, official hospital websites). The dataset reflects the cultural context of Saudi healthcare, emphasizing local patient expectations, norms, and language nuances.

The dataset is designed for **multi-aspect analysis**, where each feedback is evaluated across several service aspects: **pricing, appointments, medical staff, customer service, and emergency services**. Instead of separating aspect classification and sentiment analysis, both are performed **simultaneously**. For each aspect mentioned in the feedback, the system predicts a sentiment label, producing a vector like `[0, 1, 2, 0, 3]`, where each element corresponds to the sentiment of a specific aspect.

This mixed classification-sentiment approach allows the model to identify which aspects are discussed and how the patient feels about each, in a single pass. Additional fields like doctor name, department, and hospital location are included to support filtering and further analysis. The data is preprocessed for Arabic tokenization and split into training, validation, and test sets to ensure reliable model evaluation.

### Features / Columns

The dataset consists of multiple fields capturing both the feedback content and relevant metadata. Each record is uniquely identified by the `id` field and contains the **full patient feedback text in Arabic**. In addition, several columns represent **aspect-level sentiment scores** for different hospital services, including pricing, appointments, medical staff, customer service, and emergency services. These sentiment scores are numeric labels corresponding to positive, neutral, or negative evaluations for each aspect.

### Classification & Sentiment Analysis

The dataset enables **multi-aspect classification** and **aspect-level sentiment analysis simultaneously**. Each feedback is classified across multiple hospital service aspects, including **pricing, staff, appointments, customer service, and emergency services**. For each aspect, the model predicts a **sentiment score**, allowing both classification and sentiment evaluation in a single pass.

## Database and It's API

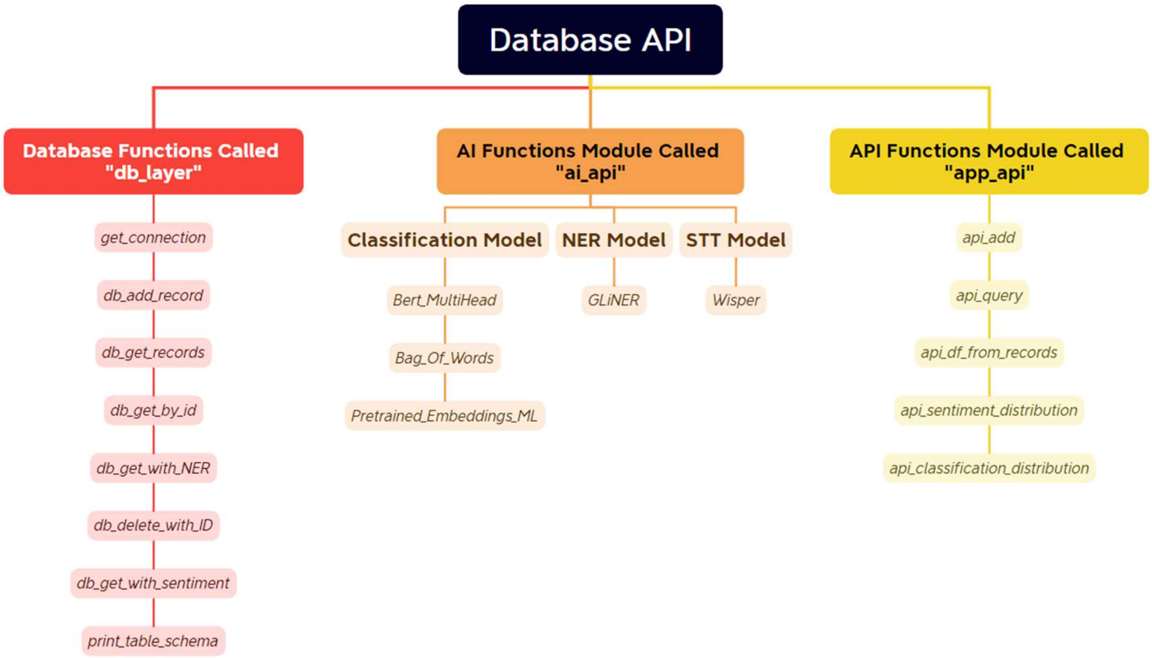
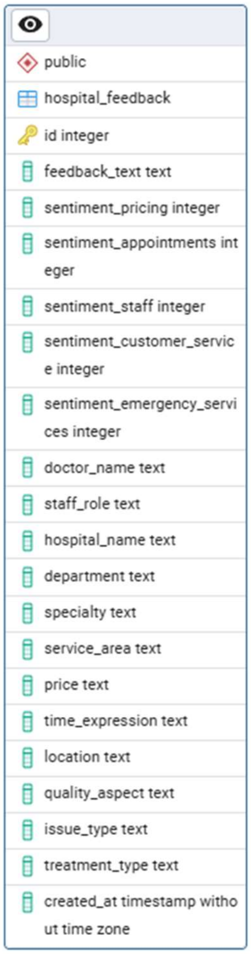


*Figure 1: Modular structure of the project backend. The diagram shows the three main modules—db_layer, ai_api, and app_api—along with their internal functions and interactions. This visual illustrates the separation of responsibilities and the flow of operations*

The project's backend is organized into three main modules: **`db_layer`**, **`ai_api`**, and **`app_api`**. Each module encapsulates a specific layer of functionality. The **`db_layer`** handles all database interactions, including querying, inserting, and updating feedback records. The **`ai_api`** module manages the AI models, performing tasks such as classification, sentiment analysis, and NER. The **`app_api`** module integrates the previous two layers, exposing functions that are directly used by the **UI** to interact with both the database and the AI models seamlessly.

To illustrate this structure, an **XMind diagram** is provided, showing each module and its inner functions. This visual representation helps clarify the modular organization and the flow of function calls between layers.

The underlying **database schema** is implemented in **PostgreSQL**. An image exported from **pgAdmin4** shows the `hospital_feedback` table, including all columns and their types. The database stores textual patient feedback along with metadata such as hospital name, department, staff role, timestamp, and aspect-level sentiment labels. Together with the module diagram, this image provides a complete overview of how the application's backend is structured and how data flows from storage to analysis.



public
hospital_feedback
id integer
feedback_text text
sentiment_pricing integer
sentiment_appointments integer
sentiment_staff integer
sentiment_customer_service integer
sentiment_emergency_services integer
doctor_name text
staff_role text
hospital_name text
department text
specialty text
service_area text
price text
time_expression text
location text
quality_aspect text
issue_type text
treatment_type text
created_at timestamp without time zone

*Figure 2: PostgreSQL database schema for the hospital_feedback table.*

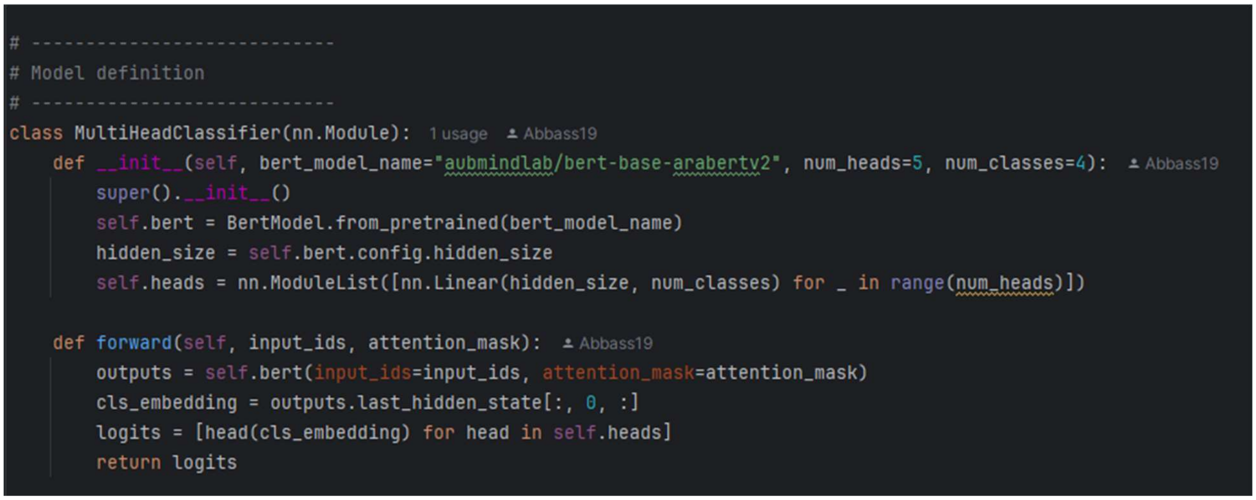
# Models Training and Testing

## Classification Models

### Multi-Head BERT-Based Classifier

Architecture:

The project employs a multi-head BERT-based classifier that performs aspect-level sentiment analysis and classification simultaneously on patient feedback. Using aubmindlab/bert-base-arabertv2 as the base model, it extracts the hidden representation of the [CLS] token from the input text and passes it through five separate linear heads, each corresponding to a hospital service aspect—Pricing, Appointments, Medical Staff, Customer Service, and Emergency Services. For each aspect, the model predicts a sentiment label coded as 0 for Positive, 1 for Negative, 2 for Neutral, and 3 for Not Mentioned, producing a vector that represents all aspects in one pass. This architecture allows the system to efficiently handle multiple aspects at once, enabling both classification and sentiment analysis within a single model, which is particularly suitable for healthcare feedback that often covers multiple service dimensions in the same text.



```
# ---------------------------
# Model definition
# ---------------------------
class MultiHeadClassifier(nn.Module):
    def __init__(self, bert_model_name="aubmindlab/bert-base-arabertv2", num_heads=5, num_classes=4):
        super().__init__()
        self.bert = BertModel.from_pretrained(bert_model_name)
        hidden_size = self.bert.config.hidden_size
        self.heads = nn.ModuleList([nn.Linear(hidden_size, num_classes) for _ in range(num_heads)])

    def forward(self, input_ids, attention_mask):
        outputs = self.bert(input_ids=input_ids, attention_mask=attention_mask)
        cls_embedding = outputs.last_hidden_state[:, 0, :]
        logits = [head(cls_embedding) for head in self.heads]
        return logits
```

*Figure 3: MultiHead Classifier Class*

| ***Aspect*** | ***Accuracy*** | ***Precision*** | ***Recall*** | ***F1-score*** | ***mAP*** |
|---|---|---|---|---|---|
| *Pricing* | 0.96 | 0.82 | 0.58 | 0.65 | 0.68 |
| *Appointments* | 0.94 | 0.77 | 0.73 | 0.75 | 0.73 |
| *Medical Staff* | 0.84 | 0.77 | 0.74 | 0.75 | 0.82 |
| *Customer Service* | 0.82 | 0.74 | 0.69 | 0.71 | 0.76 |
| *Emergency Services* | 0.97 | 0.66 | 0.63 | 0.64 | 0.67 |
| ***Overall Mean*** | - | - | - | 0.70 | 0.73 |

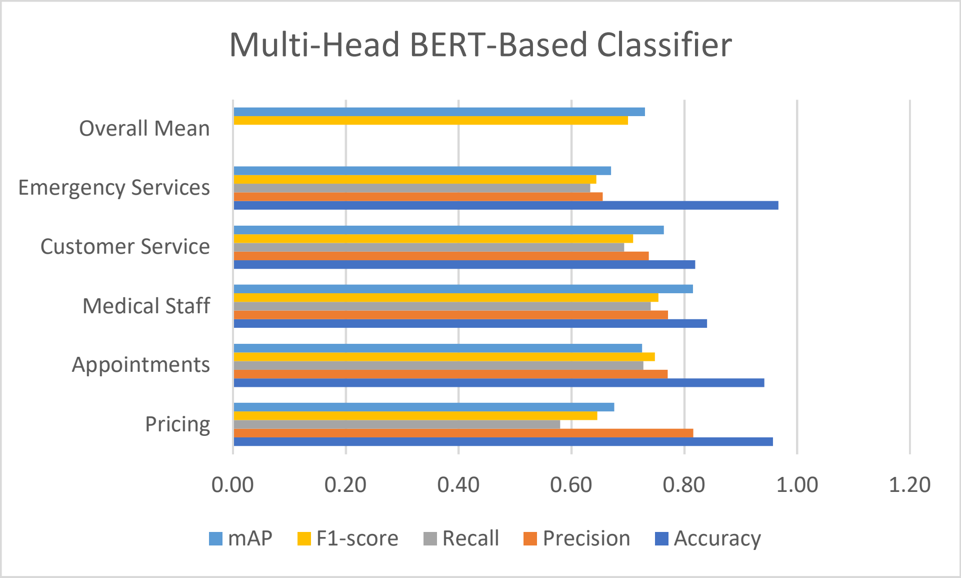


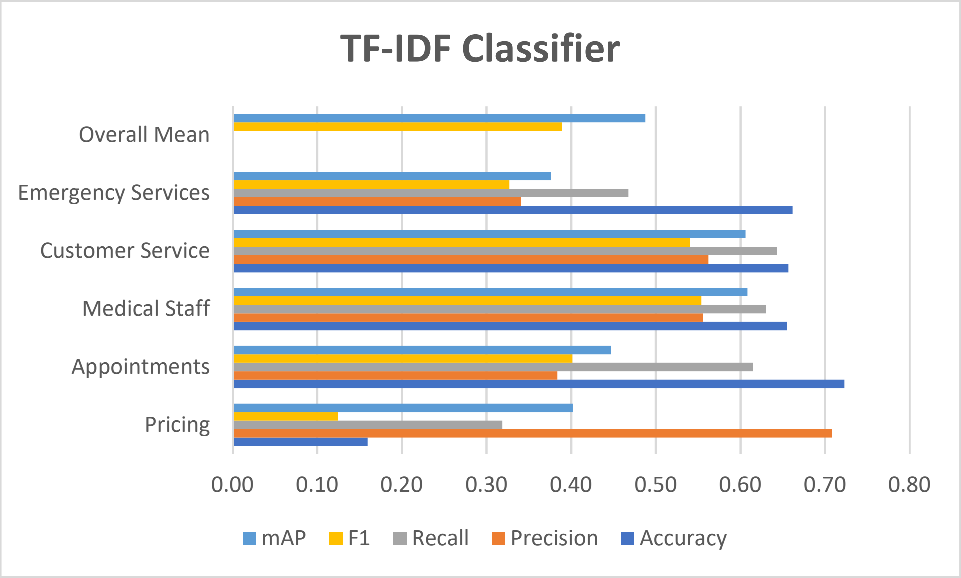
*Figure 4: Aspect-level evaluation of the multi-head feedback classification model. The chart shows Accuracy, F1-score, and mean Average Precision (mAP) for each hospital service aspect*

## TF-IDF Classifier

Architecture:

This model uses TF-IDF features from Arabic patient feedback and a MultiOutputClassifier with Logistic Regression estimators, one per hospital service aspect. It predicts sentiment labels for all aspects simultaneously. Class weights handle imbalance, enabling efficient aspect-level classification and sentiment analysis in a single model.

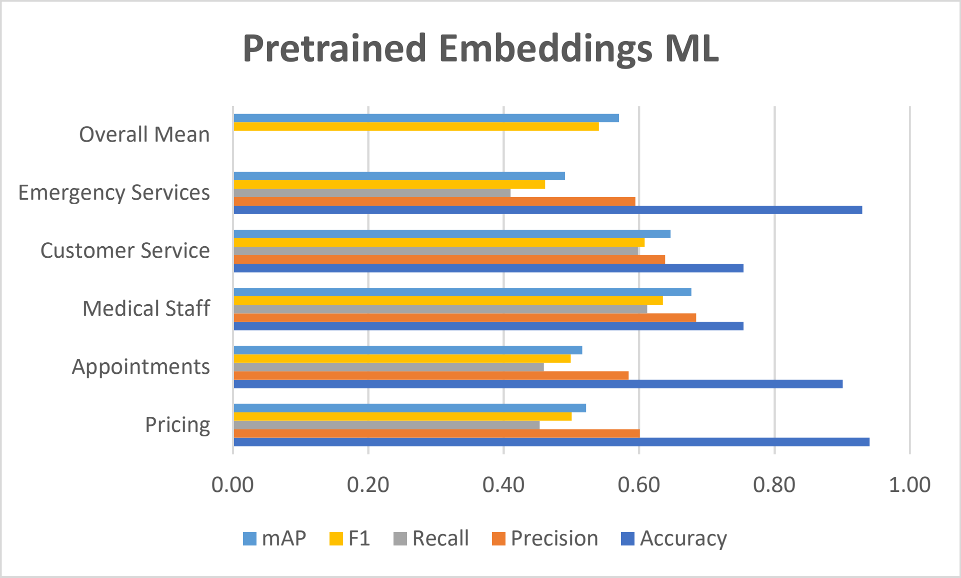
| ***Aspect*** | ***Accuracy*** | ***Precision*** | ***Recall*** | ***F1-score*** | ***mAP*** |
|---|---|---|---|---|---|
| *Pricing* | 0.16 | 0.71 | 0.32 | 0.13 | 0.40 |
| *Appointments* | 0.72 | 0.38 | 0.62 | 0.40 | 0.45 |
| *Medical Staff* | 0.65 | 0.56 | 0.63 | 0.55 | 0.61 |
| *Customer Service* | 0.66 | 0.56 | 0.64 | 0.54 | 0.61 |
| *Emergency Services* | 0.66 | 0.34 | 0.47 | 0.33 | 0.38 |
| ***Overall Mean*** | - | - | - | 0.39 | 0.49 |

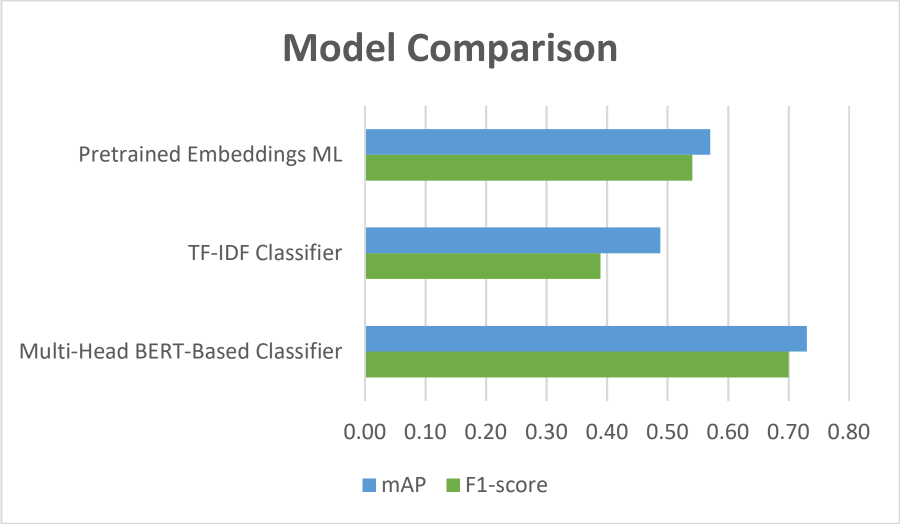


## Pretrained Embeddings ML

This model uses pretrained Arabic Sentence-BERT embeddings to encode reviews, with a separate Logistic Regression classifier per aspect (Pricing, Appointments, Medical Staff, Customer Service, Emergency Services). It predicts labels from the embeddings and evaluates Accuracy, Precision, Recall, F1-score, and mAP, reporting overall mean F1 and mAP. The saved model and embedder allow local inference, capturing nuanced Arabic reviews (e.g., positive about doctors, negative about pricing). Compared to TF-IDF, SBERT embeddings capture semantic context, making the model more robust for varied Arabic phrasing.

| ***Aspect*** | ***Accuracy*** | ***Precision*** | ***Recall*** | ***F1-score*** | ***mAP*** |
|---|---|---|---|---|---|
| *Pricing* | 0.94 | 0.60 | 0.45 | 0.50 | 0.52 |
| *Appointments* | 0.90 | 0.58 | 0.46 | 0.50 | 0.52 |
| *Medical Staff* | 0.75 | 0.68 | 0.61 | 0.64 | 0.68 |
| *Customer Service* | 0.75 | 0.64 | 0.60 | 0.61 | 0.65 |
| *Emergency Services* | 0.93 | 0.59 | 0.41 | 0.46 | 0.49 |
| ***Overall Mean*** | - | - | - | 0.54 | 0.57 |





Although the BERT based model is slower, though it has better results, and that is why it was chosen.

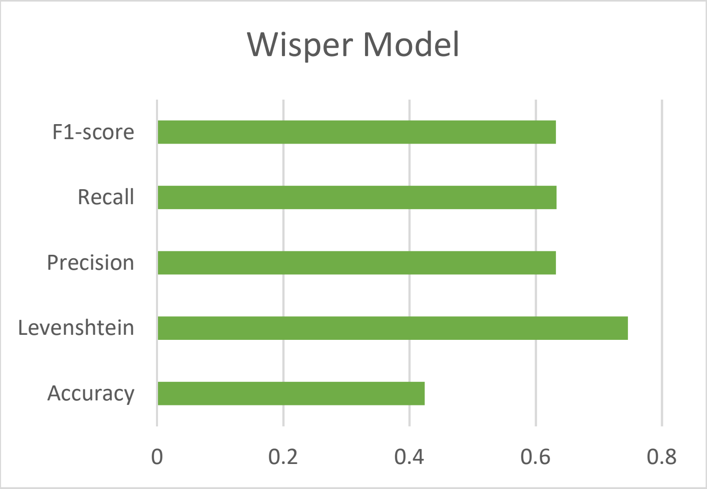
The scripts for the training the models and Testing them are present on GitHub Repo. The models were trained of coarse on Google Colab.

## NER Model: GLiNER

Used but honestly not tested yet!

## STT Model: Wisper

I used the Wisper model. Also, I recorded 20 audio tracks in order to test the accuracy of this model. Here is the result.



Overall, Whisper performed **reliably for clear Arabic speech**, but accuracy dropped noticeably for clips containing **background noise, dialectal variation, or unclear pronunciation**. This makes it a suitable baseline STT model, with potential improvement through **fine-tuning or noise-robust preprocessing**.