İstanbul
Bilgi Üniversitesi

Complaint Management System

by

Mustafa Deniz Demirhas, 121200144
Emre Cem Kenarci, 121200150

Supervised by

Dr. Emel Küpçü

Submitted to the
Faculty of Engineering and Natural Sciences
in partial fulfillment of the requirements for the

Bachelor of Science

in the
Department of Computer Engineering

Jun, 2025

Abstract

This project mainly focuses on the design and development of an AI-enhanced complaint management system that aims at improvement in customer satisfaction and business processes. Machine learning algorithms are used for the classification of Turkish user complaints, where Linear Support Vector Classifier (LinearSVC) [1] appeared as a highly promising algorithm, achieving a test accuracy of 0.9564 after hyperparameter optimization based on experimentations.
Problems such as insufficient data were prominent in the making of the project. This problem was overcome by AI-assisted synthetic data generation strategy, utilizing multiple Large Language Models (LLMs like ChatGPT, Gemini, and Grok), with methods such as persona-based prompting and tuning the LLM's temperature parameter to create a dataset with approximately 3900 samples across six categories. The system encompasses integrated feedback collection, AI-powered classification, and LLM-based (Groq with Llama-3)[2] summary report generation that would help an organization find out recurring issues and thereby work at improving its services.
The main goal is to ensure that a system which can be easily integrated into different domains and achieve better complaint management for customer retention, can be built. This work will help in effective feedback management and better customer retention strategies.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

AdaBoost	Adaptive Boosting
AI	Artificial Intelligence
API	Application Programming Interface
BERT	Bidirectional Encoder Representations from Transformers
CNN	Convolutional Neural Network
FAQs	Frequently Asked Questions
id	Identifier
JakLapor	Jakarta Public Complaint Classification Channel
LLM	Large Language Model
NYC	New York City
NYC 311	New York City's 311 Complaint Dataset
RNN	Recurrent Neural Network
SVM	Support Vector Machine
TF-IDF	Term Frequency-Inverse Document Frequency

1 Introduction

Acquiring a new customer costs five to 25 times more than keeping an existing one, according to research by Fred Reichheld for Bain & Company [3]. This fact shows how important it is to focus on keeping customers happy to achieve lasting business success. In the same study, it was also found that in the financial services industry, increasing customer retention by just 5% can lead to a profit increase of over 25%. These findings clearly show the financial advantages of keeping existing customers loyal, making customer satisfaction a key part of any successful business strategy.

Customer complaints, which are seen as problems, are a range of feedback that is very important for companies because they come directly from the customer. If managed correctly they don't just solve the immediate problem, they can increase customer satisfaction and loyalty. Also, they provide very important information about where the business might be having chronic problems.

Managing these complaints is more than just solving the problems as they arise. It is about examining each complaint individually and then carefully organizing, analyzing, and resolving the larger and more important issues that the company faces. By learning from complaints, companies can improve their products and services and become more in line with customer expectations. This approach ensures that decisions are made with the customer in mind, helping the business grow and improve over time.

The effective handling of these complaints is not only important for the reputation of the company. It also provides a positive impression for word-of-mouth marketing. According to a 2018 study by Sitel Group [4], almost half (49%) of customers who had a positive experience shared it online, compared to 30% who had a negative experience. Importantly, handling complaints properly can turn 30% of dissatisfied reviews into positive ones. However, failing to address these complaints and resolving problems can lead to loss of customers and damage to the company's reputation.

This report details the work on a web-based AI-enhanced complaint management system that will be developed to simplify complaint management, solve the problem of storage, and help organizations identify more fundamental problems. The system allows users to provide feedback anonymously, which will be categorized using text classifications and can generate graphs, charts, and reports.

Organizations can use this feedback to track and resolve recurring issues, improve services and products, and increase customer satisfaction. This solution, which we aim for organizations to integrate into their existing systems, provides a method for managing feedback.

2 Related Works / Literature Review

2.1 Review on Importance of Complaint Management

Complaints mean a lot to an organization in terms of feedback, and thus they affect the levels of customer satisfaction and retention. Therefore, this section has looked at works that emphasize complaint management and how it impinges on the operation of a business.

Various views and arguments develop a substantial meaning out of Moshe Davidow's work titled *Organizational Responses to Customer Complaints: What Works and Doesn't* [5]. He goes ahead and elaborates on basic practices when it comes down to considering complaints on the merit that should realize improvement in buyer relations towards longitudinal commitment.

In the context of Turkey, Mehmet Nurettin Alabay discusses effective complaint resolution strategies along with cultural and operational variables that are critical for success in *MÜŞTERİ ŞİKÂYETLERİ YÖNETİMİ* [6].

Statistics and research prove that unresolved complaints are one of the top few reasons for customer churn. Thus, complaint management has to be a serious component of a company's customer relationship management strategy.

2.2 Review on Works of Classification

The technical papers reviewed here give insight into methodologies, challenges, and developments in text classification and machine learning for helping in our complaint classification project.

2.2.1 Approaches to Text Classification

Short Text Classification: A Survey by Ge Song et al.[7] discusses some unique challenges of short text classification, including sparse representation, semantic ambiguity, and limited contextual information. The authors have pointed out the importance of embedding techniques, such as Word2Vec, GloVe, and BERT, which improve semantic understanding. The key takeaway from this paper is that when processing brief customer complaint narratives, embedding techniques are very helpful and can be utilized.

Text Classification Algorithms: A Survey by Kamran Kowsari et al.[8] compares text classification tasks using classic machine learning algorithms, such as SVM, Naive Bayes, and Random Forest, and deep learning models, including CNNs, RNNs, and Transformers. This paper presents the identification that deep learning models are much better at capturing nonlinear relationships within texts, while traditional algorithms perform much better with engineered features in small datasets. The key takeaway from this paper is that the size and complexity of the data should drive the choice of which algorithm to use.

2.2.2 Complaint-Specific Use Cases

Automating Public Complaint Classifications through JakLapor Channel[9] uses machine learning to classify complaints in Jakarta. In this regard, the JakLapor dataset has been experimented on with a set of algorithms like Random Forest, SVM, and Naive Bayes. Among these, Random Forest was the most robust, with an accuracy of 84%, indicating the strength of the classifier for a real-world complaint dataset. The key takeaway from this paper is that Random Forest and SVM algorithms were the two most successful algorithms for this multilingual data.

Data Mining for Classification of Crowdsourced Citizen Complaints[10] compares algorithms such as k-NN, SVM, and AdaBoost on local government complaint data. It points to really powerful tools like SVM, promising for text materials, and gives a sign of advantages in AdaBoost for integrating weak learners. The key takeaway from this paper is that techniques like AdaBoost can improve model performances, especially when the individual performances are not that good.

Data Science and Machine Learning Capstone Project - NYC 311 Complaint Dataset[11] [1] defines two major uses of complaint classification with NYC's data at 311. Extensive, multi-step preprocessing may involve features such as removal of 'stopping' words, usage of stemming, lemmas, etc., or feature engineering using, not limited to, term frequencies-inverse document frequency (TF-IDF) vectors. Repetitive training and tuning further improved the models to develop better accuracy. The key takeaway from this project is that the steps of pre-processing will assure that text is clear, and this is a vital process for achieving high accuracy.

Topic Classification and Clustering on Indonesian Complaint Tweets[12] compares the performance of various supervised and unsupervised learning methods, considering the classification of complaints on Indonesian tweets. It identifies the performance of the supervised techniques, such as Logistic Regression and Support Vector Machine over clustering-based approaches, and also improves interpretability by feature selection. The key takeaway from this paper is that supervised learning remains the best option when dealing with structured data.

[1]https://www.kaggle.com/code/korfanakis/capstone-project-nyc-311-complaint-dataset/notebook

Automatic Classification of Complaint Reports regarding City Parks[13] proposes domain-based key-terms for effective feature selection that leads with simpler Bayes approaches to excellent accuracy. The key takeaway from this paper is that domain-specific keywords will improve the precision of models in cases where complaints are from a well-defined domain.

2.3 Summarization

The technical papers that were reviewed here provide important lessons on which we can build our project upon and how the methodology and implementation of text classification can be made. Some of the key lessons here include:

- **Model Selection:** Some of the traditional algorithms, like SVM and Random Forest, provide very good performance for structured datasets. Traditional algorithms also perform better with small datasets when the features are engineered properly.
- **Preprocessing:** Full preprocessing can improve the accuracy, especially the integration of linguistic features. Preprocessing can be tuned if the accuracy isn't what it's desired to be.
- **Methodologies For Different Domains:** Some methodologies can improve performance when used with different domains. Different methodologies can be used to find which works best for that specific domain.

These will form the basis for the development of our project to a reliable, accurate, and scalable solution for managing consumer complaints.

These will form the basis for the development of our project to a reliable, accurate, and scalable solution for managing consumer complaints.

Algorithm	**Accuracy/Performance %**	**Data Set / Project**
Support Vector Machine (SVM)	80–85%	JakLapor, NYC 311
Random Forest	84%	JakLapor
Naive Bayes	70–75%	Indonesian tweets, NYC 311
Logistic Regression	78–82%	Indonesian tweets, NYC 311
AdaBoost	80%	Crowdsourced complaints

Table 1: Algorithm Accuracy/Performance across Various Datasets

3 Methodology

This chapter outlines the tools, technologies, and approaches used during the development of the Complaint Management System. It presents the technologies selected, how they were applied, and the rationale behind these choices—particularly for processing, classifying, and analyzing feedback data.

Each major component is discussed, including backend, frontend, database, and machine learning modules. This chapter also provides an overview of the challenges encountered during dataset creation and the methods used to address them, with the aim of ensuring clarity and reproducibility for future work.

3.1 Technologies Used

In this project, various technologies, tools, and methodologies will be used to develop a scalable, efficient, and user-friendly system. These tools will help to build both the backend and frontend, store and process feedback, and communicate between the components of the system.

1. **Python.**
   The Python language will be used in this project for two major purposes. The first one is to implement AI-based classification algorithms. With its libraries and resources, it is a highly preferred language in the field of machine learning. The second will be to use Django for developing the backend of the website. Django, based on Python, ensures rapid development along with efficient data management features that will eventually help build the web application. This will keep it simple for classification and for web development too, avoiding unnecessary technology additions, hence coherence in mind.

2. **Angular.**
   The complaint management system will be using Angular on the frontend. It provides a way for users to lodge complaints or suggestions through a UI. In Angular, it's easy to organize the UI with its component-based architecture. Furthermore, it would be easy to integrate graphs, charts, and reports through which the organizations can analyze this information very easily.

3. **Libraries & Algorithms.**
   For building the module of complaint classification using AI, various important Python libraries were used:

- **Pandas:** Used to manipulate and analyze data, specifically to load and preprocess the complaint datasets.
- **NumPy:** Used for numerical computations, essential to machine learning work.
- **Scikit-learn (sklearn):** Library providing core implementation of machine learning algorithms. Includes:
  - `TfidfVectorizer` to turn text features into numerical features.
  - `LabelEncoder` to encode class labels.
  - `TrainTestSplit` to split the data into a training and a test set.
  - These include classification algorithms such as `MultinomialNB`, `LogisticRegression`, and `LinearSVC` [1].
  - `Pipeline` to streamline the vectorization and classification workflow.
  - `GridSearchCV` to tune hyperparameters of the models.
  - Some of the metrics available at `sklearn.metrics` (e.g., `accuracy_score`, `classification_report`, `confusion_matrix`) for evaluation.
- **Matplotlib Seaborn:** Used to plot the class distributions, the distributions of the text lengths, and confusion matrices.
- **Joblib:** Saving and loading the machine learning models that have been trained and the label encoder.

4. **Database & Storage.**
   The complaints, suggestions, and employee data will be stored in and managed by MySQL. The database schema will be described in greater detail later in the report.

5. **Development Tools, Testing, and Version Control.**
   Testing of the backend API endpoints will be done using Postman to ensure that they are working correctly. For developing Angular, VS Code will be used, and IntelliJ IDEA will be used for Django backend. Also, GitHub will be used for version control and collaboration. You can find the project repository in: `https://github.com/SakaGibi/CMPE491---Complaint-Management-System`.

4 Design

This section summarizes the work done in the project. Steps such as developing the database schema, designing and implementing the backend services and frontend components, integrating the AI-based classification and LLM modules, and designing the web routing scheme will be discussed.

4.1 System Overview Architecture

In this section, we describe the design and architectural decisions taken for the complaint management system, the overall system structure, the workflow, and the interactions between the components of the system.

4.1.1 System Architecture

The system will be a web-based platform for easy integration with the organization's own systems. It follows the structure of client-server architecture. A client communicates with the backend to submit complaints and retrieve data.

The frontend will be designed using Angular, which will be used for a dynamic, responsive UI through which users can submit their complaints or suggestions, track their status if desired, and get feedback. It will be coupled with the backend developed in Django, which manages employee authentication, complaint classification, and other core functionalities. MySQL handles the feedback and employee data.

The integration of the AI model is in process, and for complaint classification, the testing has been done using machine learning algorithms like SVM, Naive Bayes, and Random Forest. The results of the testing will be explained later in the report.

4.1.2 Workflow

The workflow of the system may be divided into 4 major groups:

1. **User Interaction:** The front-end includes users interacting with the system. It includes customers and employees. Customers will be allowed to submit their complaints or suggestions, while employees shall view complaints and generated reports. This information will be passed via API calls to the backend for the submitted feedback.

2. **AI Processing:** Machine learning models are embedded into the backend to classify various kinds of complaints consumers provide. It pro-

cesses the text, performs feature extraction, and allocates the complaints into categories for analysis.

3. **Data Storage:** The submitted complaints will be stored with metadata in the MySQL database, such as the time of submission, the category of complaint, and information about the submitting user.

4. **Feedback and Reporting:** After processing at the backend, the complaints will be relayed back to the frontend. The user will receive feedback through the system on the status of their complaint, while organizations will benefit from the generation of various reports on recurring issues/trends for informed action.

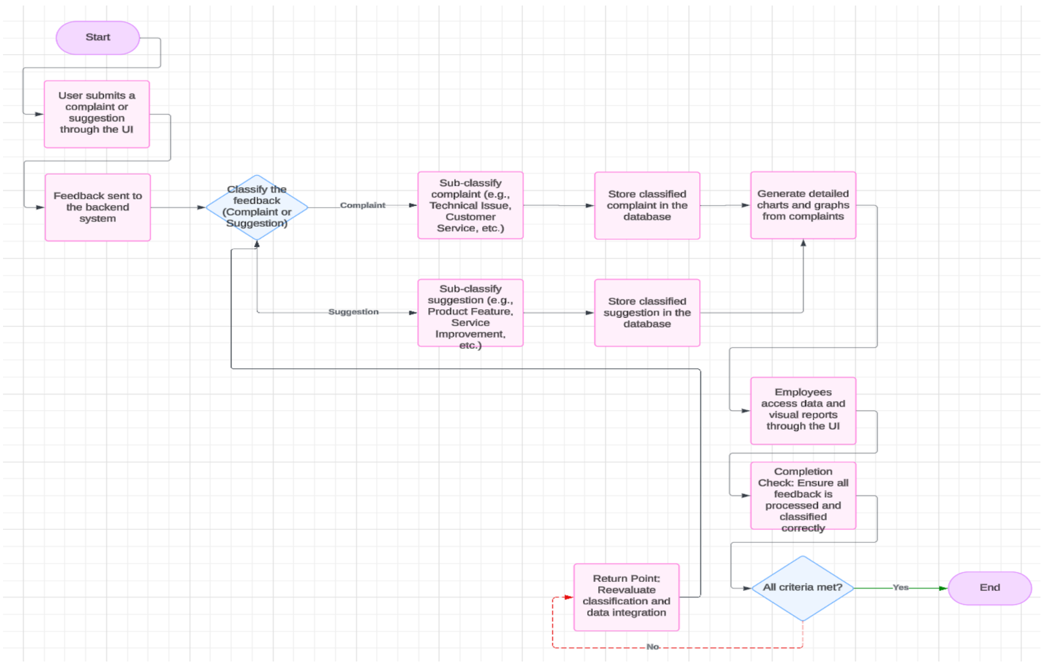

Start
User submits a complaint or suggestion through the UI
Feedback sent to the backend system
Classify the feedback (Complaint or Suggestion)
Complaint
Suggestion
Sub-classify complaint (e.g., Technical Issue, Customer Service, etc.)
Sub-classify suggestion (e.g., Product Feature, Service Improvement, etc.)
Store classified complaint in the database
Store classified suggestion in the database
Generate detailed charts and graphs from complaints
Employees access data and visual reports through the UI
Completion Check: Ensure all feedback is processed and classified correctly
All criteria met?
Yes
No
End
Return Point: Reevaluate classification and data integration

Figure 1: System Workflow Diagram

To better illustrate how the system works, we have created a workflow diagram that explains how the four main categories work and how they interact with each other.(see Figure 1)

4.1.3 Scalability and Maintenance

The system is designed with scalability in mind. By using modular architecture, it will be easy to add new features or improve existing components without affecting the entire system. The separation of concerns between the frontend, backend, and AI components ensures that updates to one part of the system do not disrupt the others.

Frontend Scalability: In the Angular, component-based architecture, UI scaling is quite straightforward. Other features, like notification handling or adding new complaint categories, don't require much rework.

The Backend Scaling: The Django backend is made scalable. Any number of servers or load balancers are required simply. It keeps increasing as per the needs that will arise. API endpoints are designed efficiently to use and handle an ever-increasing amount of data/traffic.

Improvement of AI Models: More complaints coming in would also allow retraining of the AI models with more data, thus improving their accuracy over time. It is also easy to integrate new algorithms or improvements of existing models without major changes to the backend.

This design approach has made the complaint management system flexible, scalable, and able to adapt to future needs.

4.2 Backend Design

Django and the Django REST Framework are used in the development of the complaint management system's backend. It is responsible for handling the business logic, database operations, user authentication, classification processes, and communication with external APIs. Each feature is implemented as part of a separate app to ensure maintainability, and the backend is made to be scalable and modular. The backend provides structured responses in JSON format, and all communication between the frontend and backend is handled via RESTful API endpoints.

4.2.1 API Endpoints and Routing

The backend includes multiple endpoints grouped under different modules. Each module is implemented as a separate Django app, and their routes are defined in individual `urls.py` files. These apps are included in the main `urls.py` file of the project using Django's `include()` function.

There are three main groups of endpoints:

- **Employee Endpoints** (`/api/employees/`): These endpoints allow employees to log in, be added or removed, and change roles.
- **Complaint Endpoints** (`/api/complaints/`): These endpoints handle complaint submission, classification, tracking, deletion, status updates, statistical analysis, and report generation.

- **Support Endpoints** (`/api/support/`): These endpoints handle incoming support messages or FAQ-type questions. They allow submitting, listing, answering, and deleting messages.

Each endpoint is built as a function-based API view using the Django REST Framework. Endpoints accept and return data in JSON format. Authentication and permission checks are handled at the application level based on user roles.

4.2.2 Complaint Classification Module

The module is at the core of machine learning-enabled backend that is responsible for automatically classifying incoming textual complains to specified classes. The module facilitates streamlined routing of complains, identification of trends, and rapid actioning by administrator. Design of the module is a standard machine learning pipeline, comprising collection and preparation of the data, feature engineering, learning of a model, and performance evaluation.

Dataset Design and Augmentation The core piece of the classification module is the dataset used for training and evaluating the machine learning models. Given the challenge of data scarcity for specific building management complaints, an approach with 2 steps was adopted for dataset design:

- **Preliminary Experiment Dataset:** In preparation to first make pipelines and compare algorithms, Kaggle's public "Consumer Complaint Database" was utilized. While this dataset is about banking complaints, its very large size (approximately 800,000 records) and textual nature were a suitable platform to try out text classification methodologies with "Product" as the class and "Consumer complaint narrative" as the text.

- **Custom Turkish Complaint Dataset:** In line with the given project domain, a customized Turkish complaint-oriented dataset was prepared. In order to create this dataset and diversify its content, a new dataset augmentation method using a range of Large Language Models (LLMs) – including ChatGPT, Gemini, and Grok – was followed. This dataset augmentation using machine learning comprised:

  – Persona-Based Prompting: Specific roles (e.g., "as an office employee, write a complaint about...") were given to the LLMs to create contextually suitable and stylistically diverse complaint writings.

  - Temperature Control: The `temperature` of the LLMs was tuned (between 1.0 and 2.0) at generation to promote originality and avoid repetitive results.
  - Iterative Review and Generation: It was an iterative process whereby generated samples were assessed on quality, relevance, and realism. This review loop was applied to refine prompts on subsequent generation cycles.

  The resulting augmented dataset (`sikayet_v1.csv`), used for model training in this project, consists of 3899 samples. This dataset was structured around six predefined target categories: *Temizlik (Cleaning)*, *Bakım ve Onarım (Maintenance & Repair)*, *Gürültü (Noise)*, *Güvenlik (Security)*, *Ortak Alan Kullanımı (Common Area Usage)*, and *Yönetim (Management)*. Efforts were made to ensure a relatively balanced distribution across these classes.

Text Preprocessing and Feature Engineering Raw text complains have to be preprocesssed before input into machine learning algorithms. The developed preprocessing pipeline is comprised of:

- **Lowercasing:** Converting all text to lowercase.
- **Punctuation and Number Removal:** Basic cleaning to remove numbers and punctuation was achieved with regular expressions. Advanced tokenization and stop word removal is handled by the vectorizer TF-IDF itself.
- **TF-IDF Vectorization:** The preprocessed text data is then turned into numerical feature vectors using the Term Frequency-Inverse Document Frequency (TF-IDF) technique. The `TfidfVectorizer` from Scikit-learn was configured with parameters optimized through GridSearchCV:
  - `ngram_range=(1, 2)`: Considers both unigrams and bigrams.
  - `max_df=0.85`: hose terms that occur more frequently than in 85% in the documents are excluded.
  - `min_df=2`: Terms appearing in fewer than 2 documents are excluded.

Model Selection, Training, and Optimization A systematic approach was used to select and tune the classification model:

- **Baseline Model Evaluation:** Three classical machine learning algorithms were initially evaluated with 5-fold cross-validation on the augmented dataset: Multinomial Naive Bayes (MNB), Logistic Regression (LR), and Linear Support Vector Classifier (LinearSVC). LinearSVC achieved the best initial performance with a cross-validation F1 Macro of 0.9331.

- **Hyperparameter Tuning with GridSearchCV:** For the LinearSVC model, an hyperparameter search was performed using `GridSearchCV` with 5-fold cross-validation, optimizing for the F1 Macro score. The parameter grid included variations in `tfidf__max_df`, `tfidf__min_df`, `tfidf__ngram_range`, and the classifier's parameters `clf__C`, `clf__loss`, `clf__penalty`, and `clf__dual`. The best parameters found were: `C=1`, `dual=True`, `loss='squared_hinge'`, `penalty='l2'`, `tfidf__max_df=0.85`, `tfidf__min_df=2`, and `tfidf__ngram_range=(1, 2)`. This resulted in an improved cross-validation F1 Macro score of 0.9553.

- **Final Model Performance:** Best-performing LinearSVC model's ultimate test accuracy was 0.9564 on the test set that was held out (20% of the data).

- **Model Finalization and Storage:** The final optimized LinearSVC model and the `LabelEncoder` were saved using `joblib` for deployment in the backend API.

Classification Logic in Backend Backend The classification module that is deployed on the Django backend does the following on a new complaint:

1. Input the raw complaint text.

2. Applies the same preprocessing as used during training (e.g., lowercasing, removal of punctuation and numbers).

3. Converts the preprocesssed text to a TF-IDF vector with the loaded `TfidfVectorizer`.

4. Uses the trained LinearSVC model loaded to make the class prediction.

5. **Sub-Category Determination:** In order to pick potential sub-classes, the module makes use of the `decision_function` scores of the LinearSVC of all classes. The class with the greatest score is the primary `category`. A second-highest scoring class is also determined. If

the gap between the top scoring class and the second-highest scoring class is smaller than a threshold (0.35 was used for the project), a `sub_category` is determined. In all other cases, the `sub_category` is the top category.

6. Returns the primary category and sub-category via the API.

This design aims to provide administrators with both a primary classification and an indication of closely related secondary category for more comprehensive complaint handling.

4.2.3 LLM API Integration

To enhance the functionality of the Complaint Management System, with a focus on generating actionable insights and summaries to aid administrators, a Large Language Model (LLM) API interface was conceptualized and developed. This component facilitates the ability to create AI-powered reports on shortlists of complaints, to help understand trends, repeating problems, and potential areas of improvement.

LLM Selection and Rationale Several LLM APIs were evaluated for this integration, including Hugging Face Inference API (with models like BART-Large-CNN and Mistral-7B-Instruct), Google Gemini API (e.g., Gemini 1.5 Flash), and the Groq API (leveraging models such as Llama-3.1-8b-instant and Llama-3.3-70b-versatile). After initial testing, the Groq API utilizing Llama-3[2] based models (specifically, `llama-3.1-8b-instant` was chosen as the primary LLM service for the backend due to its rapid response times, reliable output quality for summarization, and notably generous free-tier API limits, and notably generous free-tier API limits.

Prompt Engineering for Report Generation Successful interaction with LLMs is also very much dependent on carefully prepared prompts. A dynamic generation strategy of the prompts was developed.The function generates the prompt by:

1. Precise description of work: e.g., "Aşağıda belirtilen bina şikayetlerini analiz ederek bir rapor hazırla." (Task: Analyze the building complaints listed below and prepare a report.)

2. Incorporating the administrator-applied filters (e.g., category, status, date range)

3. Listing the relevant complaints (up to a specified maximum, e.g., the latest 10-1000) in a structured format: ”- ID:id, Kategori: category, Durum: status, Tarih: date, Açıklama: shortened_description”. Descriptions are truncated to 150 characters.

4. Defining the desired output format of the report to cover headings such as ”Genel Durum (Overall Situation)”, ”Öne Çıkan Temalar (Prominent Themes)”, and ”Öneriler (Varsa) (Recommendations (If any))”.

5. Providing explicit negative constraints (e.g., avoid vague statements) and positive guidance (e.g., ensure recommendations are actionable and clear) to shape the LLM’s output towards being clear and useful for administrators.

4.3 Frontend Design

Angular is used in the development of the system’s frontend. For both system employees and end users, it offers a simple user interface. The structure is designed to be modular, with each view (or page) handled by a separate component. Data binding, routing, and HTTP communication are implemented using Angular features like services, observables, and two-way binding via `ngModel`. The interface is responsive and focuses on usability, ensuring that both complaints and internal tasks can be handled efficiently. In the following subsections, each main page of the application is described in detail.

4.3.1 Main Menu Page

This page is designed to submit complaints or suggestions and optionally track their status. Users can type their message into a large text box, choose whether they want to track it, and if so, enter their email address. Upon submission, the form sends the data to the backend via an API call and resets the interface.

There is also a tracking button that opens a modal window for users to track their complaints by entering the complaint ID they received through mail if the complaint is trackable.

The page also includes buttons to navigate to the management panel login screen and help & support page.

4.3.2 Management Panel Login Page

This page is designed for employees to log in to the management panel. It includes two input fields for username and password. There is also a checkbox that allows the user to log in as admin if they are. When the form is submitted, a login request is sent to the backend for authentication. If login fails an error message is shown to the user.

The interface also includes two navigation buttons at the top: One to return to the main menu and another to go to the help & support page.

4.3.3 Management Panel Page

The management panel is the main interface used by employees to monitor, filter, and analyze complaints. The page is divided into three main sections: Complaint statistics and trends (left), complaints list (center) and report generation (right).

On the left side, employees can view quick previews of the complaint distribution and trend graph. Clicking on these opens a fullscreen modal with more detailed charts and filters. These visuals help users understand the patterns of complaints over time.

The center section lists all complaints with filtering options based on status, category, whether it is trackable, and sort order. When a complaint is selected, a detailed modal opens with all its information. Employees can update its status, view category classifications, and delete it if necessary.

The right sections allows employees to generate AI-based reports. The report creation form includes filters such as category, status, date, range and the number of complaints to analyze. submitted reports are listed below and can be viewed or deleted. These reports are generated by sending the filtered complaint data to a Large Language Model (LLM) via API.

4.3.4 Help Support Page

This page is designed to help users who have questions or need assistance. It consists of two main sections. A list of Frequently Asked Questions (FAQs) is shown in the first section. Every FAQ displays the question along with, if available, the answer.

The second section provides a contact form for users to reach out to the support team directly. Users can write their questions and mail address, then submit the form. The submitted message is sent to the backend and stored for review by an admin.

This page ensures that users can qucikly find answers or get additional helps when needed.

4.3.5 Admin Panel Page

The admin panel is designed for managing system users and viewing incoming suggestions or questions about the system from end-users.

Admins can add new users by entering a username, password, email address, and selecting the role. Existing users are listed on the right, and clicking on any user opens a modal where their details can be reviewed, their role can be changed, or the user can be deleted.

Below the user management section, there is a list of submitted suggestions of support messages. Each message displays a short preview and timestamp. Admins can click on a message to view its full details in a modal and delete it if necessary.

4.3.6 Routing Structure

Routing in the frontend is managed using Angular's Router. Each page of the application is mapped to a specific route, enabling transitions between views.(see Figure 2)

The main routes are:

- `/main-menu` – Main Menu Page
- `/login` – Management Panel Login Page
- `/admin-panel` – Admin Panel Page
- `/management-panel` – Management Panel Page
- `/support` – Help & Support Page

These routes are defined in the application's routing module and triggered via navigation buttons located in each view. Role-based redirection is handled after login, directing users either to the admin panel or management panel depending on their role and preference.

The diagram below illustrates the routing structure and the navigation flow between pages:

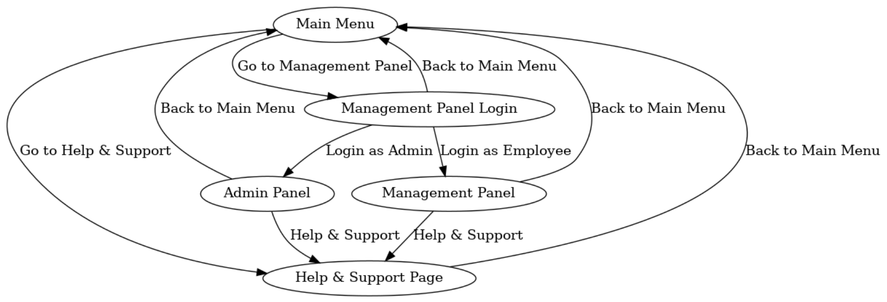


Figure 2: Frontend Routing Structure

The routing system contributes to a seamless user experience and keeps the navigation logic consistent throughout the application.

4.4 Database Design

The database to be used in the project is designed to securely store and manage complaints and suggestions made by users. A MySQL-based structure was preferred. This structure enables the storage of feedback and employee information.

The database contains the following 4 basic tables:

1. **Employees:** Keeps the identity and role information of employees logging into the system. Employees can manage complaints and suggestions by logging into the system through this table. (see Figure 3)

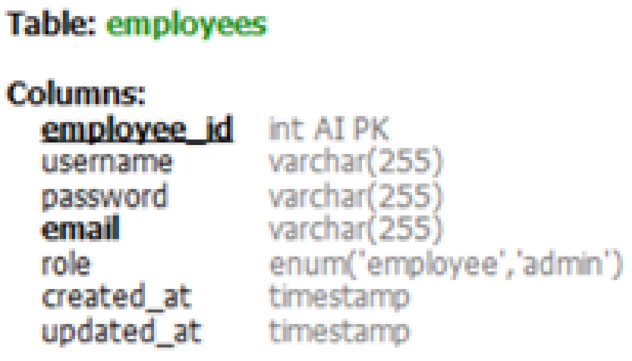


Table: employees

Columns:

employee_id	int AI PK
username	varchar(255)
password	varchar(255)
email	varchar(255)
role	enum('employee','admin')
created_at	timestamp
updated_at	timestamp

Figure 3: Employees Table

2. **Complaints_Suggestions:** Stores complaints and suggestions from the customer. Each complaint or suggestion contains id, sender id, category, status, and system response. If the user is an anonymous user, a predefined value of 1 is returned for the sender_id field. Also, for anonymous users, the email, response, and response_at fields remain NULL. (see Figure 4)

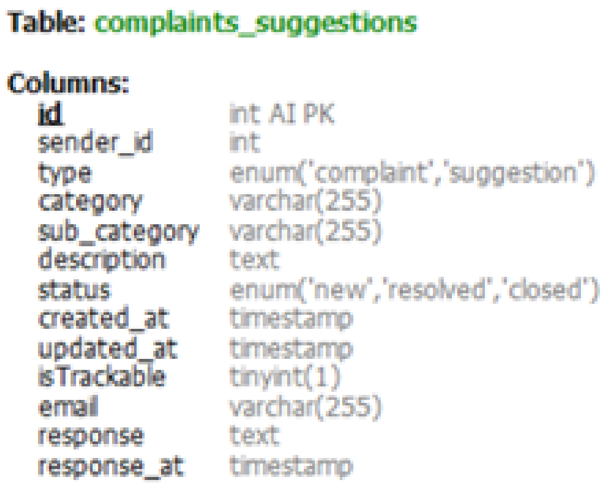


Table: complaints_suggestions

Columns:

id	int AI PK
sender_id	int
type	enum('complaint','suggestion')
category	varchar(255)
sub_category	varchar(255)
description	text
status	enum('new','resolved','closed')
created_at	timestamp
updated_at	timestamp
isTrackable	tinyint(1)
email	varchar(255)
response	text
response_at	timestamp

Figure 4: Complaints and Suggestions Table

3. **Support_Questions_Messages:** Is used to manage system-related questions and support requests from both customers and employees. (see Figure 5)

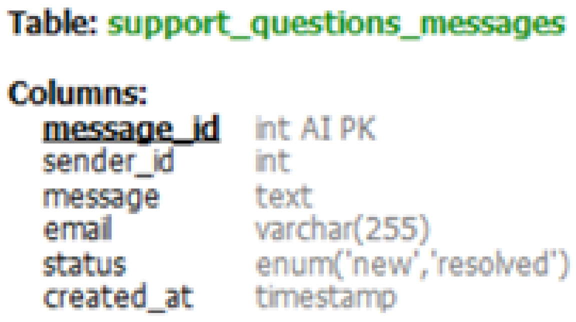


Table: support_questions_messages

Columns:

message_id	int AI PK
sender_id	int
message	text
email	varchar(255)
status	enum('new','resolved')
created_at	timestamp

Figure 5: Support Questions and Messages Table

4. **Report_Recommendation:** Stores AI-generated summary reports created by the system. Each report includes a type, the filters applied

during its generation, and the full content. The filters are saved in JSON format to allow flexible querying. This table has no relational links with other tables, making it modular and easy to manage. (see Figure 6)

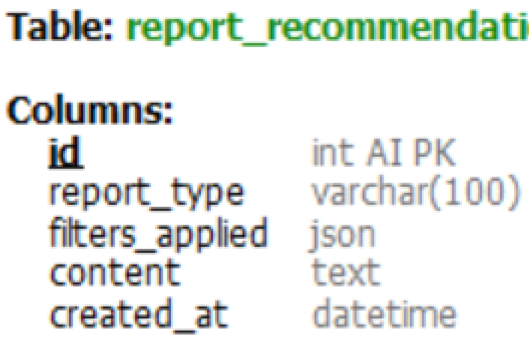


Table: report_recommendation

Columns:

id	int AI PK
report_type	varchar(100)
filters_applied	json
content	text
created_at	datetime

Figure 6: Report Recommendation Table

This database structure is not relational but is open to future growth and additional tables that may be needed. As the requirements of the system change, the database structure will be easy to extend and optimize. This flexibility supports the system's scalability and adaptability to future needs.

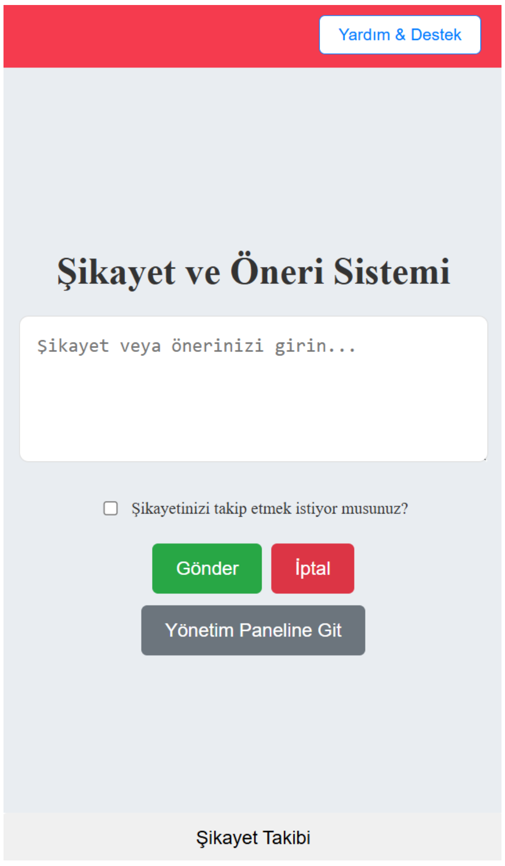
4.5 UI Design

The user interface design of the project was created to provide functionalities such as enabling customers to easily submit complaints and suggestions, track the status of their existing complaints (if they wish), and enable employees to control and manage these complaints and see visual forms related to them. The design was created with the user experience in mind, aiming to send complaints as quickly and simply as possible.

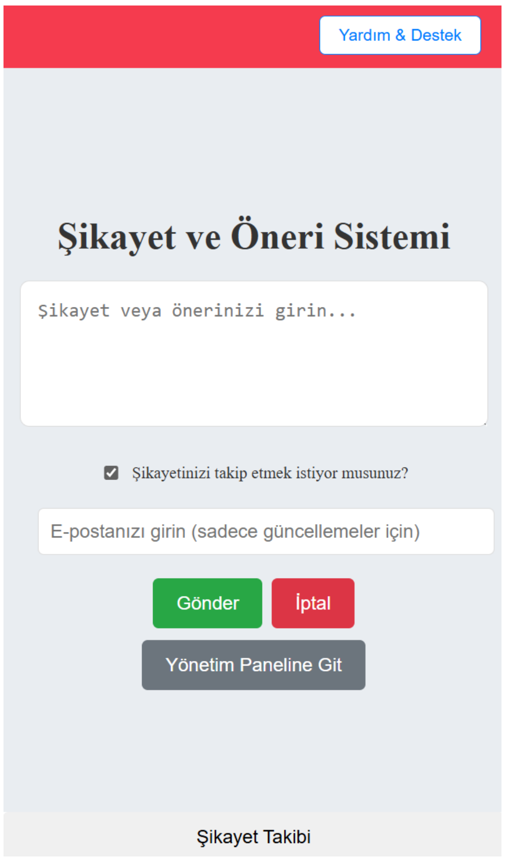
The user interface is divided according to customer and employee needs. Customers have a simple and straightforward format for submitting complaints and suggestions, while employees can view and manage these complaints. There is a dashboard that provides detailed information on the status of feedback and provides graphs and management tools for employees.

1. **Main Menu:** This screen provides a text field where users can type and submit their complaints or suggestions. Users can also choose whether they want to track their complaints. Users who wish to track their complaints can do so by entering their email address, the id of the

complaint sent to them, and clicking on the complaint tracking button at the bottom of the page, filling in the required fields. The screenshots for this page were taken in mobile view, considering that most users are expected to access the site from their smartphones (see Figures 7a-7b- 8a-8b).

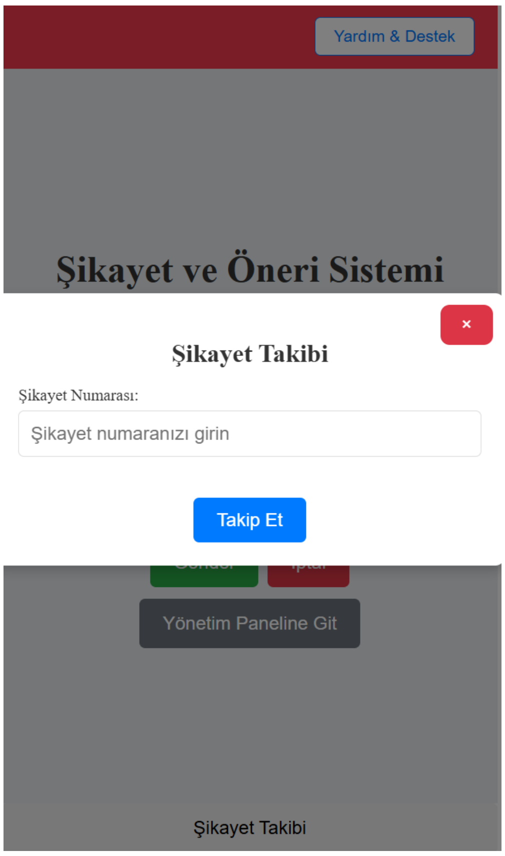


(a) Default View (Tracking Off)


Yardım & Destek
Şikayet ve Öneri Sistemi
Şikayet veya önerinizi girin...
Şikayetinizi takip etmek istiyor musunuz?
E-postanızı girin (sadece güncellemeler için)
Gönder
İptal
Yönetim Paneline Git
Şikayet Takibi

(b) Tracking Option Enabled

Figure 7: Main Menu Page Views

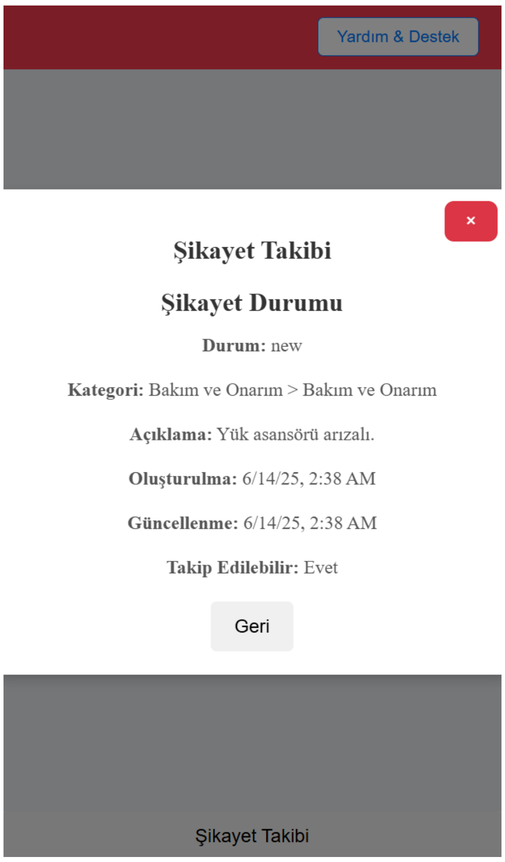


Şikayet Takibi

Şikayet Numarası:

Şikayet numaranızı girin

(a) Initial Modal View



Şikayet Takibi

Şikayet Durumu

Durum: new

Kategori: Bakım ve Onarım > Bakım ve Onarım

Açıklama: Yük asansörü arızalı.

Oluşturulma: 6/14/25, 2:38 AM

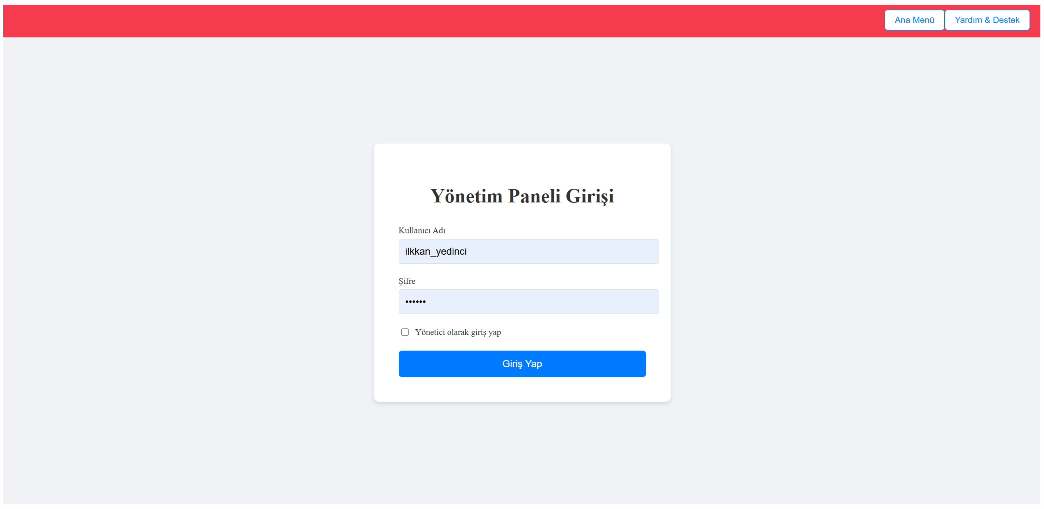
Güncellenme: 6/14/25, 2:38 AM

Takip Edilebilir: Evet

(b) Modal After Submission

Figure 8: Complaint Tracking Modal Views

2. **Management Login:** By entering their username and password, employees can log in to the management panel, manage complaints, and access reports. There is also an option for admin users. If this option is checked and the user has the admin role, they can access the admin panel. Non-admin users cannot access this area (see Figure 9).



Yönetim Paneli Girişi

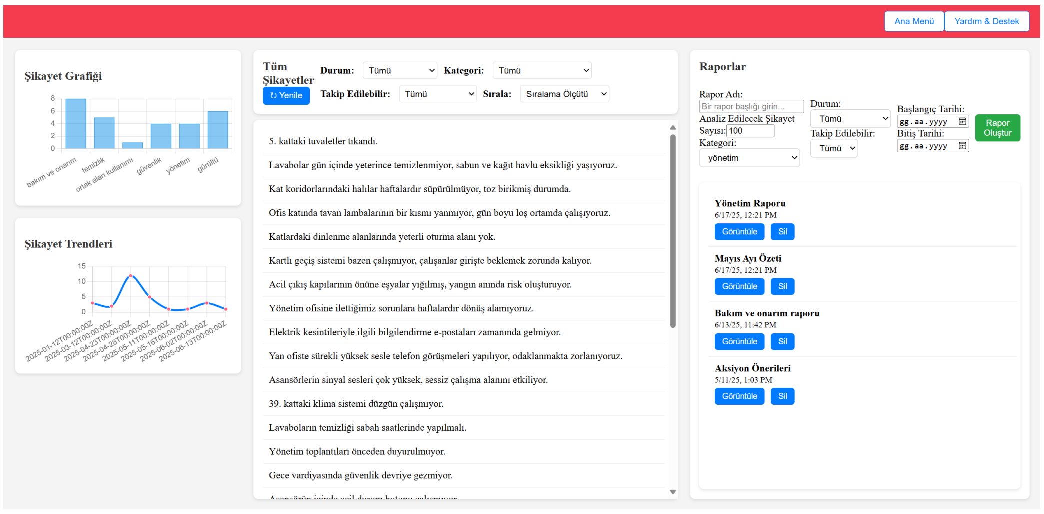
Kullanıcı Adı: ilkkan_yedinci

Şifre: ••••••

☐ Yönetici olarak giriş yap

Figure 9: Management Panel Login Page

3. **Management Panel:** On this page, employees can view complaints and suggestions, review each one in detail, and update their status. They can access charts and diagrams on the left side of the screen, change their filters, and create new customized charts or diagrams according to their needs. On the right side of the screen, they can also generate custom reports using various filters, powered by the LLM API (see Figures 10– 11–12–13–14).



Tüm Şikayetler

Durum: Tümü **Kategori:** Tümü
Takip Edilebilir: Tümü **Sırala:** Sıralama Ölçütü

5. kattaki tuvaletler tıkandı.

Lavabolar gün içinde yeterince temizlenmiyor, sabun ve kağıt havlu eksikliği yaşıyoruz.

Kat koridorlarındaki halılar haftalardır süpürülmüyor, toz birikmiş durumda.

Ofis katında tavan lambalarının bir kısmı yanmıyor, gün boyu loş ortamda çalışıyoruz.

Katlardaki dinlenme alanlarında yeterli oturma alanı yok.

Kartlı geçiş sistemi bazen çalışmıyor, çalışanlar girişte beklemek zorunda kalıyor.

Acil çıkış kapılarının önüne eşyalar yığılmış, yangın anında risk oluşturuyor.

Yönetim ofisine ilettiğimiz sorunlara haftalardır dönüş alamıyoruz.

Elektrik kesintileriyle ilgili bilgilendirme e-postaları zamanında gelmiyor.

Yan ofiste sürekli yüksek sesle telefon görüşmeleri yapılıyor, odaklanmakta zorlanıyoruz.

Asansörlerin sinyal sesleri çok yüksek, sessiz çalışma alanını etkiliyor.

39. kattaki klima sistemi düzgün çalışmıyor.

Lavaboların temizliği sabah saatlerinde yapılmalı.

Yönetim toplantıları önceden duyurulmuyor.

Gece vardiyasında güvenlik devriye gezmiyor.

Raporlar

Rapor Adı: Bir rapor başlığı girin...
Analiz Edilecek Şikayet Sayısı: 100
Kategori: yönetim
Durum: Tümü
Takip Edilebilir: Tümü
Başlangıç Tarihi: gg.aa.yyyy
Bitiş Tarihi: gg.aa.yyyy

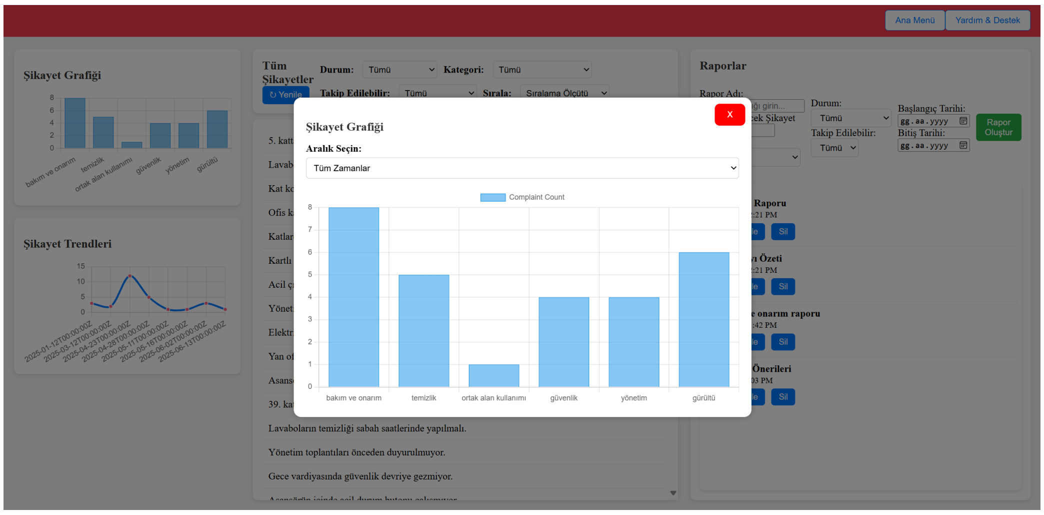
Yönetim Raporu
6/17/25, 12:21 PM

Mayıs Ayı Özeti
6/17/25, 12:21 PM

Bakım ve onarım raporu
6/13/25, 11:42 PM

Aksiyon Önerileri
5/11/25, 1:03 PM

Figure 10: Management Panel – Main View

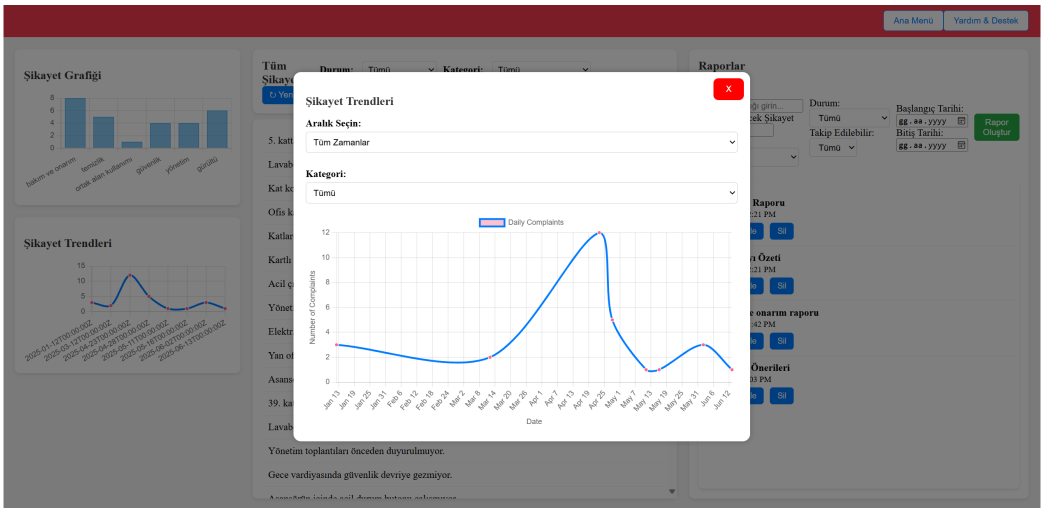


Lavaboların temizliği sabah saatlerinde yapılmalı.

Yönetim toplantıları önceden duyurulmuyor.

Gece vardiyasında güvenlik devriye gezmiyor.

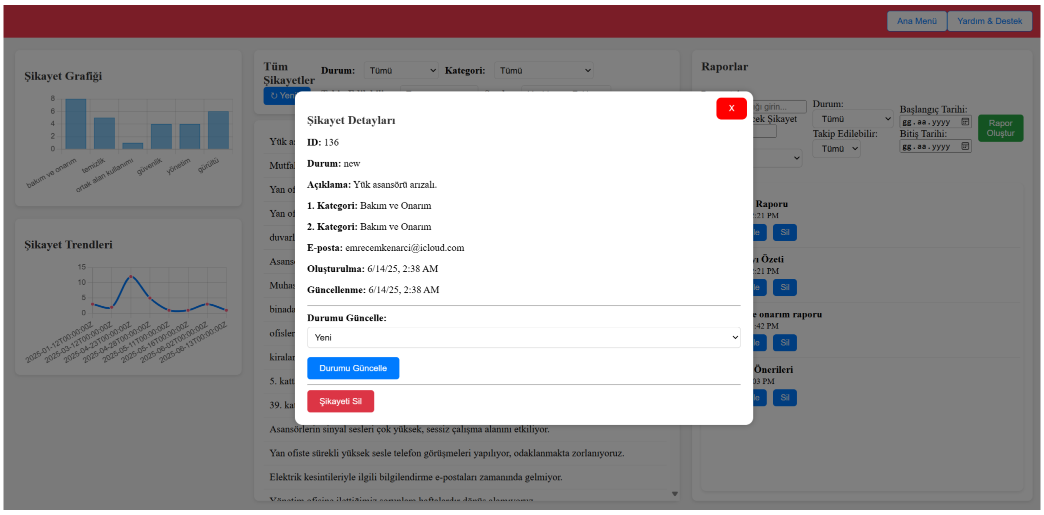
Figure 11: Management Panel – Chart Modal



Yönetim toplantıları önceden duyurulmuyor.

Gece vardiyasında güvenlik devriye gezmiyor.

Figure 12: Management Panel – Trend Modal



Şikayet Detayları

ID: 136

Durum: new

Açıklama: Yük asansörü arızalı.

1. Kategori: Bakım ve Onarım

2. Kategori: Bakım ve Onarım

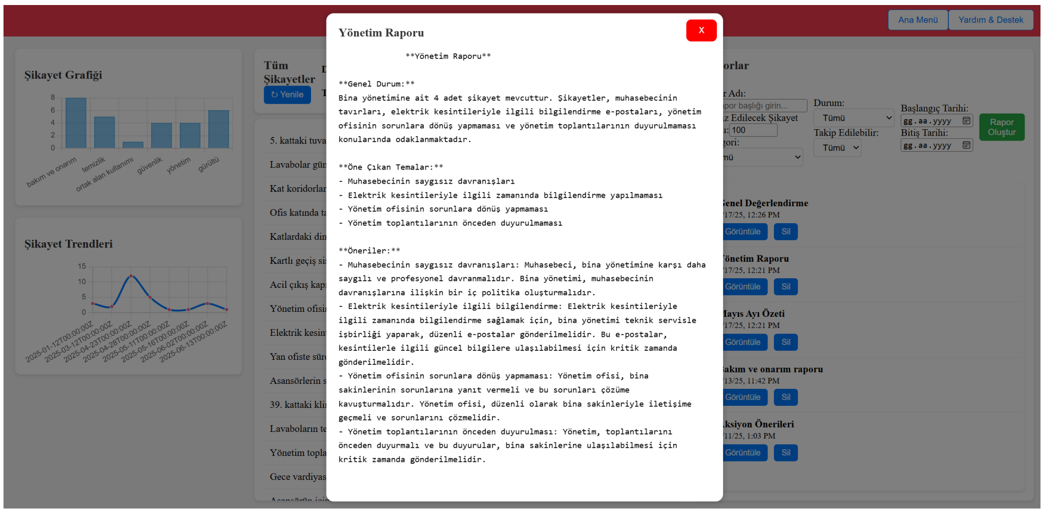
E-posta: emrecemkenarci@icloud.com

Oluşturulma: 6/14/25, 2:38 AM

Güncellenme: 6/14/25, 2:38 AM

Durumu Güncelle: Yeni

Figure 13: Management Panel – Complaint Details Modal

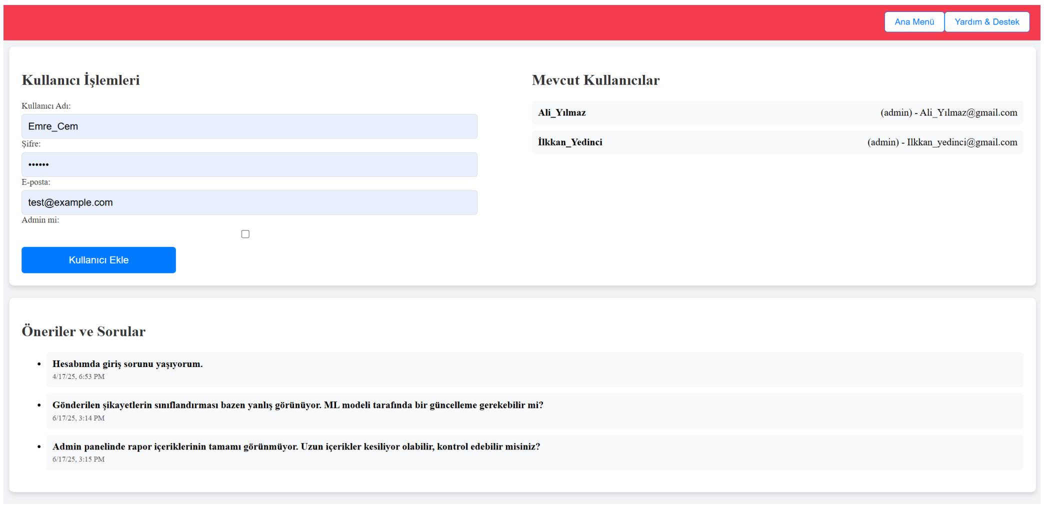


Yönetim Raporu

```
            **Yönetim Raporu**

**Genel Durum:**
Bina yönetimine ait 4 adet şikayet mevcuttur. Şikayetler, muhasebecinin
tavırları, elektrik kesintileriyle ilgili bilgilendirme e-postaları, yönetim
ofisinin sorunlara dönüş yapmaması ve yönetim toplantılarının duyurulmaması
konularında odaklanmaktadır.

**Öne Çıkan Temalar:**
- Muhasebecinin saygısız davranışları
- Elektrik kesintileriyle ilgili zamanında bilgilendirme yapılmaması
- Yönetim ofisinin sorunlara dönüş yapmaması
- Yönetim toplantılarının önceden duyurulmaması

**Öneriler:**
- Muhasebecinin saygısız davranışları: Muhasebeci, bina yönetimine karşı daha
saygılı ve profesyonel davranmalıdır. Bina yönetimi, muhasebecinin
davranışlarına ilişkin bir iç politika oluşturmalıdır.
- Elektrik kesintileriyle ilgili bilgilendirme: Elektrik kesintileriyle
ilgili zamanında bilgilendirme sağlamak için, bina yönetimi teknik servisle
işbirliği yaparak, düzenli e-postalar gönderilmelidir. Bu e-postalar,
kesintilerle ilgili güncel bilgilere ulaşılabilmesi için kritik zamanda
gönderilmelidir.
- Yönetim ofisinin sorunlara dönüş yapmaması: Yönetim ofisi, bina
sakinlerinin sorunlarına yanıt vermeli ve bu sorunları çözüme
kavuşturmalıdır. Yönetim ofisi, düzenli olarak bina sakinleriyle iletişime
geçmeli ve sorunlarını çözmelidir.
- Yönetim toplantılarının önceden duyurulması: Yönetim, toplantılarını
önceden duyurmalı ve bu duyurular, bina sakinlerine ulaşılabilmesi için
kritik zamanda gönderilmelidir.
```

Figure 14: Management Panel – Report Creation Modal

4. **Admin Panel:** Admin users can perform simple user operations such as adding and deleting employees. They can also view and respond to questions and support requests from all users in the system (see Figures 15–16).



Kullanıcı İşlemleri

Kullanıcı Adı: Emre_Cem

Şifre: ••••••

E-posta: test@example.com

Admin mi: ☐

Mevcut Kullanıcılar

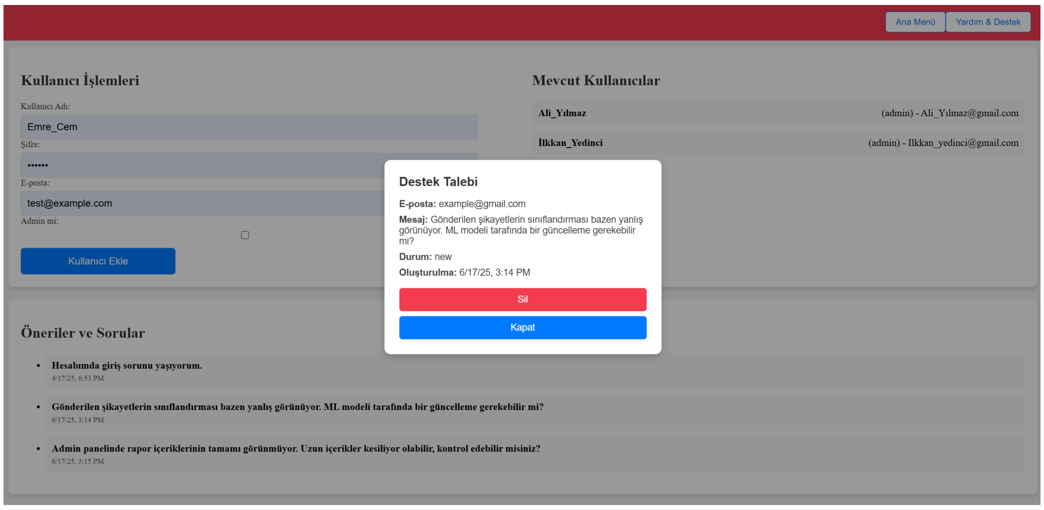
Ali_Yılmaz (admin) - Ali_Yılmaz@gmail.com

İlkkan_Yedinci (admin) - Ilkkan_yedinci@gmail.com

Öneriler ve Sorular

- **Hesabımda giriş sorunu yaşıyorum.**
  4/17/25, 6:53 PM
- **Gönderilen şikayetlerin sınıflandırması bazen yanlış görünüyor. ML modeli tarafında bir güncelleme gerekebilir mi?**
  6/17/25, 3:14 PM
- **Admin panelinde rapor içeriklerinin tamamı görünmüyor. Uzun içerikler kesiliyor olabilir, kontrol edebilir misiniz?**
  6/17/25, 3:15 PM

Figure 15: Admin Panel – Main View



Destek Talebi

E-posta: example@gmail.com

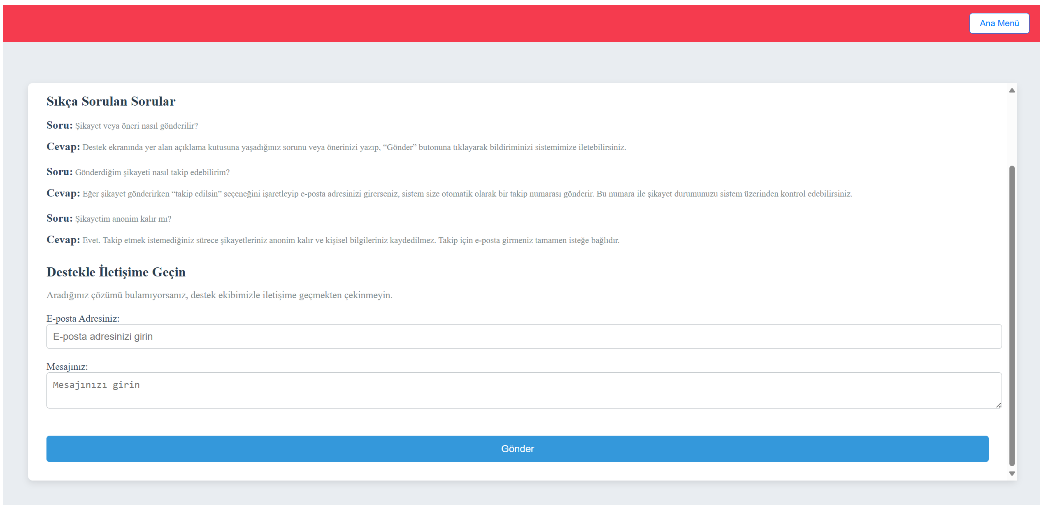
Mesaj: Gönderilen şikayetlerin sınıflandırması bazen yanlış görünüyor. ML modeli tarafında bir güncelleme gerekebilir mi?

Durum: new

Oluşturulma: 6/17/25, 3:14 PM

Figure 16: Admin Panel – User Detail Modal

5. **Help & Support:** In the Help and Support section, users can access frequently asked questions (FAQs) and find the answers they are looking for. If their questions are not answered, users can submit their own questions through the system and create a support ticket (see Figure 17).



Sıkça Sorulan Sorular

Soru: Şikayet veya öneri nasıl gönderilir?

Cevap: Destek ekranında yer alan açıklama kutusuna yaşadığınız sorunu veya önerinizi yazıp, “Gönder” butonuna tıklayarak bildiriminizi sistemimize iletebilirsiniz.

Soru: Gönderdiğim şikayeti nasıl takip edebilirim?

Cevap: Eğer şikayet gönderirken “takip edilsin” seçeneğini işaretleyip e-posta adresinizi girerseniz, sistem size otomatik olarak bir takip numarası gönderir. Bu numara ile şikayet durumunuzu sistem üzerinden kontrol edebilirsiniz.

Soru: Şikayetim anonim kalır mı?

Cevap: Evet. Takip etmek istemediğiniz sürece şikayetleriniz anonim kalır ve kişisel bilgileriniz kaydedilmez. Takip için e-posta girmeniz tamamen isteğe bağlıdır.

Destekle İletişime Geçin

Aradığınız çözümü bulamıyorsanız, destek ekibimizle iletişime geçmekten çekinmeyin.

E-posta Adresiniz: E-posta adresinizi girin

Mesajınız: Mesajınızı girin

Figure 17: Help and Support Page

4.6 Deployment and Environment

Currently, the system is running in a local development environment. Both the frontend and backend are hosted and tested on the developers' personal machines. The backend runs on a Django server using Python, and the frontend runs on Angular's local development server. MySQL is used as the local database.

The development environment includes:

- Operating System: Windows 11
- Python 3.11.6 and Django 4.2
- Angular 17 and Node.js 18
- MySQL 8.0 (local instance)

The project has not yet been deployed to a public server. However, it is fully deployable to a cloud environment such as AWS, Azure, or Heroku. A production deployment would require:

- A public domain name
- Cloud-hosted MySQL or managed database service
- A backend server (e.g., Gunicorn + Nginx for Django)
- A frontend hosting platform (e.g., Firebase Hosting, Netlify, or Vercel)

If the system is to be made available online, basic security measures such as HTTPS, authentication token protection, and database backups must also be considered.

5 Data

In this project, a specially constructed, domain-specific dataset was used to develop and assess the AI-driven complaint classification module. The primary focus remained on the custom dataset designed for building management complaints, although an experimental dataset was initially used for preliminary algorithm exploration.

5.1 Experimental Dataset (Banking Complaints - Summary)

The "Consumer Complaint Database" from Kaggle was used to conduct preliminary research on text classification pipelines. By testing TF-IDF vectorization and baseline models like SVM and Logistic Regression, this sizable dataset (roughly 800,000 banking-related complaints) confirmed the general applicability of these models for text classification tasks. Nevertheless, its application was restricted to these initial tests because of its domain irrelevance to building management.

5.2 Custom Turkish Building Management Complaint Dataset

The main dataset for training and assessing the complaint classification model is `sikayet_v1.csv`, a specially selected set of Turkish complaints pertaining to building and residential management issues.

AI-Assisted Generation About 150 handwritten Turkish complaint texts served as the dataset's original source. A multi-LLM approach involving models like ChatGPT, Gemini, and Grok was used to greatly augment this dataset in order to raise this amount to acceptable levels and improve diversity for reliable model training. The following is the augmentation procedure:

- **Persona-Based Prompting:** Generating complaints from various viewpoints (e.g., manager, office worker).
- **Temperature Parameter Adjustment:** Using a range of temperature settings (1.0 to 2.0) in LLMs to obtain a variety of textual outputs.

- **Iterative Curation:** Examining and improving produced samples to guarantee their relevance and quality.

The final `sikayet_v1.csv` dataset, which contained 3899 distinct complaint instances, was the outcome of this process.

Dataset Structure and Categories There are two primary columns in the dataset:

- **Şikayet Metni (Complaint Text):** The complaint's textual content in Turkish.
- **Sınıf (Class):** The category of the complaint.

Six categories relevant to building management are used to classify complaints: *Temizlik (Cleaning)*, *Bakım ve Onarım (Maintenance & Repair)*, *Gürültü (Noise)*, *Güvenlik (Security)*, *Ortak Alan Kullanımı (Common Area Usage)*, and *Yönetim (Management)*.

Dataset Characteristics and Distribution Complaints are proportionally distributed across the six categories, allowing the model to receive sufficient examples of each class. This was intentional, and one of the considerations when creating the dataset. The category counts can be found in Table 2.

Table 2: Class Distribution in `sikayet_v1.csv` Dataset

Sınıf (Class)	**Örnek Sayısı (Sample Count)**
Temizlik	671
Bakım ve Onarım	664
Gürültü	656
Ortak Alan Kullanımı	650
Güvenlik	648
Yönetim	610
Toplam (Total)	**3899**

A visual representation of this distribution is provided in Figure 18.

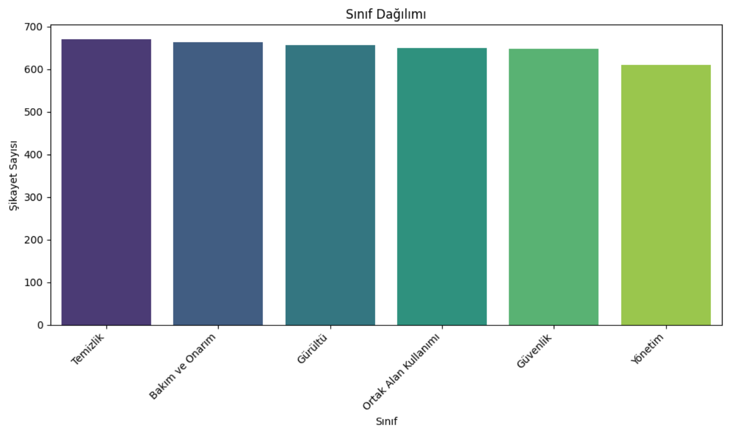


Figure 18: Class Distribution Bar Chart for `sikayet_v1.csv`

Analysis of the complaint texts (Figure 19) shows that the average complaint length is approximately 10 words, while the mean character length is around 80.

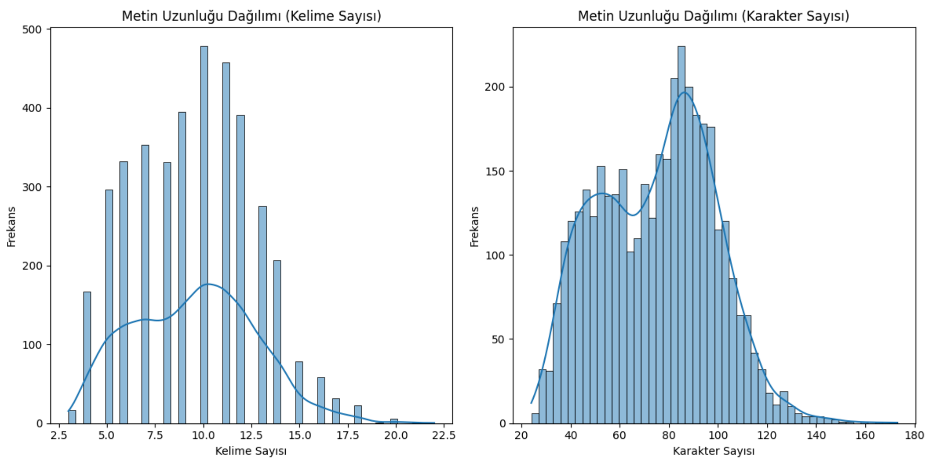


Figure 19: Text Length Distribution (Word and Character Count) for `sikayet_v1.csv`

The customized, AI-enhanced dataset is key in developing a classification model which will effectively classify real-world, Turkish complaints in the domain of building management. Since no datasets existed for this purpose, it was necessary to create a dataset. Sample complaint texts for each category in the dataset can be found in Table 3.

Table 3: Sample Complaint Texts from `sikayet_v1.csv`

Sınıf (Class)	**Örnek Şikayet Metni (Excerpt)**
Temizlik	Ne zamandır koridor temizlenmiyor, pis kokuyor...
Bakım ve Onarım	Binanın dış cephesindeki bazı reklam panolarının aydınlatması gece boyunca yanıyor, enerji israfı...
Gürültü	Diğer katlardan gelen ve mesai saatleri dışında bile devam eden yüksek sesli müzik...
Ortak Alan Kullanımı	Ortak alanlarda temizlik eksikliği var...
Güvenlik	Apartmanın bodrum katında izinsiz kişiler dolaşıyor...
Yönetim	Şikayetimizle ilgili yönetim toplantı yapmıyor...

6 Experimental Setup and Results

This section describes the experimental design used for training and testing the complaint classification models, along with a representation and discussion of those results thus obtained, primarily for the self-created Turkish Building Management Complaint Dataset (`sikayet_v1.csv`).

6.1 Experimental Setup

Dataset and Preprocessing The primary dataset was the custom-made `sikayet_v1.csv` dataset, which contained 3899 Turkish complaints in six classes as noted in Section Data. The preprocessing of texts involved lowercasing and basic cleaning (number and punctuation elimination). The feature creation occurred using TF-IDF vectorization with an n-gram range of (1, 2), `max_df` = 0.85, and `min_df` = 2, which were optimizations from GridSearchCV.

Train-Test Split The `sikayet_v1.csv` dataset was divided into 2 sets: a training set (80%, 3119 samples) and a test set (20%, 780 samples). The data was split in a way so that it maintained the same proportion of class labels in both the training and testing sets.

Models Evaluated The following machine learning models were implemented using Scikit-learn and integrated into a Pipeline with the TF-IDF vectorizer:

- Multinomial Naive Bayes (MNB)
- Logistic Regression (LR)
- Linear Support Vector Classifier (LinearSVC)

Evaluation Metrics Model performance was assessed using standard classification metrics:

- Accuracy
- Precision (macro and weighted averages)
- Recall (macro and weighted averages)
- F1-Score (macro and weighted averages)
- Classification Report (per-class metrics)
- Confusion Matrix

Hyperparameter Optimization GridSearchCV and 5-fold cross-validation (aiming for the best F1 Macro score) were used to get the best settings for the LinearSVC model. This model was seen as the best starting point, based on first tests on the train data.

Experimental Environment The model was built and tested in Google Colab, using Python and the tools listed in the Methodology part.

6.2 Results and Discussion

Initial Model Comparison (Cross-Validation) First, the first part of the `sikayet_v1.csv` dataset was checked with 5-fold cross-validation on t. This gave us a basic idea of how the chosen methods work. We put the findings in Table 4.

Table 4: 5-Fold Cross-Validation Performance Summary on Training Data (`sikayet_v1.csv`)

Model	**Mean CV Accuracy**	**Mean CV F1 Macro**
Naive Bayes	0.9108	0.9103
Logistic Regression	0.9200	0.9198
Linear SVC	**0.9331**	**0.9331**

LinearSVC achieved the highest F1 Macro score in the initial cross-validation, thus it was selected as the model to get further hyperparameter optimization.

Optimized Model Performance on Test Set After hyperparameter tuning using GridSearchCV, the optimized LinearSVC model was evaluated on the remaining test set. The best parameters identified were `clf__C=1`, `clf__dual=True`, `clf__loss='squared_hinge'`, `clf__penalty='l2'`, `tfidf__max_df=0.85`, `tfidf__min_df=2`, and `tfidf__ngram_range=(1, 2)`.

The optimized LinearSVC model achieved a final accuracy of **0.9564** and a weighted average F1-score of **0.9561** on the test set. The detailed classification report is shown in Table 5, and the confusion matrix is shown in Figure 20.

Table 5: Classification Report for Optimized LinearSVC on Test Set (`sikayet_v1.csv`)

Sınıf (Class)	**Precision**	**Recall**	**F1-Score**	**Support**
Bakım ve Onarım	0.9516	0.8872	0.9183	133
Gürültü	0.9549	0.9695	0.9621	131
Güvenlik	0.9248	0.9462	0.9354	130
Ortak Alan Kullanımı	0.9773	0.9923	0.9847	130
Temizlik	0.9478	0.9478	0.9478	134
Yönetim	0.9839	1.0000	0.9919	122
Accuracy			**0.9564**	**780**
Macro Avg	0.9567	0.9572	0.9567	780
Weighted Avg	0.9564	0.9564	0.9561	780

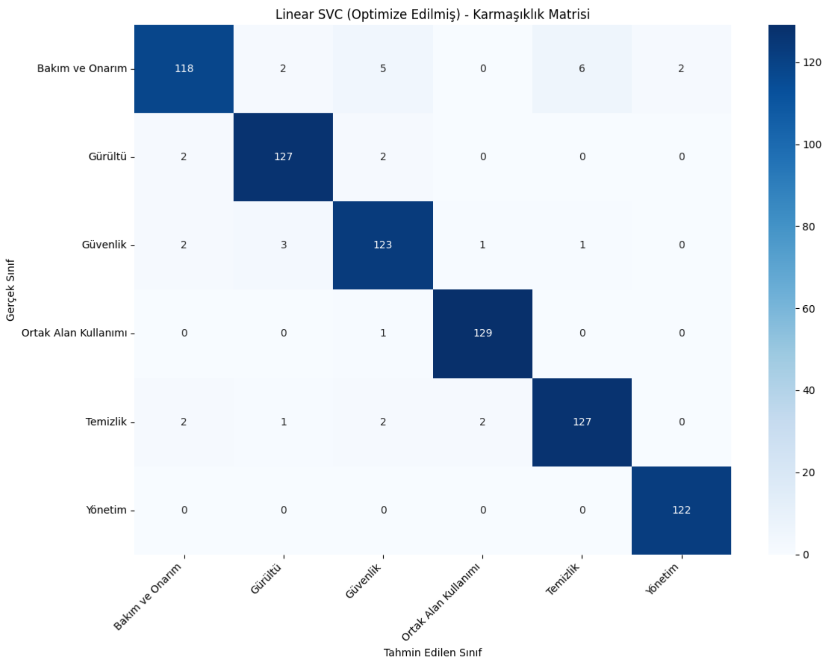


Figure 20: Confusion Matrix for Optimized LinearSVC on Test Set (`sikayet_v1.csv`)

Discussion of Results The improved LinearSVC model works very well in sorting out complaints about building management. It scores high in precision, recall, and F1-scores in most areas. With an accuracy of 0.9564 on the test data, this shows it's a strong model that can handle new, unseen data well. The confusion matrix (Figure 20) shows that the model gets it right for most cases in each group. For example, the 'Yönetim' group got a perfect recall of 1.0000, and 'Ortak Alan Kullanımı' also had very high precision (0.9773) and recall (0.9923).

Small mix-ups are most seen when that have some overlaps with other groups; for example, 'Bakım ve Onarım' had a recall of 0.8872, hinting some of the data might have been mixed with other like groups. Yet, the very high F1-scores (a mean of 0.9567) show a good balance in how well and how often the right groups were picked for all classes.

The use of AI to expand data, along with good feature engineering (TF-IDF with bigrams) and fine-tuning hyperparameters, has helped reach this good result. By hand trying it with made-up complaint texts during the build phase showed good broad use aswell. The way it sorts out sub-category complaint types, by looking at score changes with a set point of 0.35, makes a way to spot detailed complaints. Yet, setting this point just right in the system needs a lot of careful work.

The use of AI to add more data, along with feature engineering (TF-IDF with bigrams) and tuning hyperparameter settings, has clearly helped bring about this strong result. Hands-on tests with made-up complaint texts, as tried in making the tool, also gave good results. The logic to find subcategories (using score diffs with a checked threshold of 0.35) gives a way to spot fine-detail complaints, though this cut-off needs right tuning at the backend.

The tool's skill to spot the right group for many kinds of complaints is a key part of the system's auto work plan, making the sorting and handling of user feedback quicker and better.

7 Conclusion & Future Work

Conclusion This project succeeded in its premise of successfully developing an AI-enhanced complaint management system, including an automatic classification module for Turkish building management complaints. An important addition of the project was the AI-assisted generation of a custom synthetic dataset (`sikayet_v1.csv`) with 3899 samples using multiple LLMs, persona-based prompting, and temperature variation in its creation. This dataset was then used to train a LinearSVC model which, after TF-IDF vectorization and GridSearchCV optimization, achieved test accuracy score of **0.9564** and a weighted F1-score of **0.9561** across six complaint categories. The system also includes a built-in LLM (Groq with Llama-3) support for generating AI-based summary reports from filtered complaints.

This study demonstrates that AI-augmented data can be used as a valuable alternative, especially in scenarios where there is high data scarcity. Data scarcity usually blocks lots of people from developing their own projects, but as demonstrated in this research, some level of success can still be found by utilizing these methods. While high accuracy was achieved, ensuring comprehensive coverage of real-world complaint diversity remains as a greater challenge than getting high accuracy for models that utilize synthetically generated data. The system provides a solid foundation for efficiently processing complaints and allowing administrators to gain valuable insights.

Future Work Future enhancements will focus on evolving the system into a more versatile and solid platform: In the future, by implementing some new features, the project can evolve into a more versatile and solid platform:

- **Live Deployment & Iteration:** The system can be deployed for live use in order to gain user feedback and improve UI/UX and the ML model.

- **Advanced Dataset Improvements:** The dataset used in training the model can be further enhanced by adding real-world complaints and some more diverse AI-generated complaints.
- **Multilingual Support:** The model and the website can be developed to support multilingual usage, increasing the system's reach.
- **Domain Adaption:** System can be integrated with different data (e.g., classification of university student complaints) to demonstrate its ability to be used as a framework.

Addressing these areas would allow the project to develop further and increase its value.

References

[1] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "LIBLINEAR: A library for large linear classification," *Journal of Machine Learning Research*, vol. 9, pp. 1871–1874, Jun 2008.

[2] Meta AI, "Llama 3: The next generation of open source large language models." Blog Post, Apr. 2024.

[3] F. Reichheld, "Prescription for cutting costs," *Bain & Company. Harvard Business School Publishing*, 2001.

[4] S. Group, "2018 cx index: Brand loyalty & engagement." `https://cdn2.hubspot.net/hubfs/5196934/40502861-0-2018-CX-Index-Sitel-.pdf`, 2018.

[5] M. Davidow, "Organizational responses to customer complaints: What works and what doesn't," *Journal of service research*, vol. 5, no. 3, pp. 225–250, 2003.

[6] C. Yilmaz, K. Varnali, and B. T. Kasnakoglu, "How do firms benefit from customer complaints?," *Journal of Business Research*, vol. 69, no. 2, pp. 944–955, 2016.

[7] G. Song, Y. Ye, X. Du, X. Huang, and S. Bie, "Short text classification: a survey.," *Journal of multimedia*, vol. 9, no. 5, 2014.

[8] K. Kowsari, K. Jafari Meimandi, M. Heidarysafa, S. Mendu, L. Barnes, and D. Brown, "Text classification algorithms: A survey," *Information*, vol. 10, no. 4, p. 150, 2019.

[9] S. M. Intani, B. I. Nasution, M. E. Aminanto, Y. Nugraha, N. Muchtar, and J. I. Kanggrawan, "Automating public complaint classification through jaklapor channel: A case study of jakarta, indonesia," in *2022 IEEE International Smart Cities Conference (ISC2)*, pp. 1–6, IEEE, 2022.

[10] E. D. Madyatmadja, C. P. Sianipar, C. Wijaya, and D. J. Sembiring, "Classifying crowdsourced citizen complaints through data mining: Accuracy testing of k-nearest neighbors, random forest, support vector machine, and adaboost," in *Informatics*, vol. 10, p. 84, MDPI, 2023.

[11] O. Konstantinos, "Capstone project - nyc 311 complaint dataset." Online, Available at: https://www.kaggle.com/code/korfanakis/capstone-project-nyc-311-complaint-dataset, 2020.

[12] T. Pratama and A. Purwarianti, “Topic classification and clustering on indonesian complaint tweets for bandung government using supervised and unsupervised learning,” in *2017 International Conference on Advanced Informatics, Concepts, Theory, and Applications (ICAICTA)*, pp. 1–6, IEEE, 2017.

[13] Y. Sano, K. Yamaguchi, and T. Mine, “Automatic classification of complaint reports about city park,” *Information Engineering Express*, vol. 1, no. 4, pp. 119–130, 2015.