# Requirements Gathering: Sentiment Analysis with Streamlit

Project: Deployed Complaint & Feedback Analysis System

Document Version: 2.0

Author: Abdelrahman

Date: November 1, 2025

### 1. Stakeholder Analysis



### 

| Stakeholder | Role | Description | Key Interest / Need |
| --- | --- | --- | --- |
| End User | Public user or employee. | Submits feedback/complaint via the Streamlit application. | A simple way to submit a report and have it routed to the correct department and categorized by the correct issue type. |
| Development Team | The four team members. | Responsible for building and deploying the project. | A clear dataset and requirements for this new dual-classification model. |
| Project Evaluator | Instructor or judge. | Assesses the project's quality and functionality. | A functional application that correctly classifies both the department and the issue type. |
| (Hyphetical) Operations Manager | A city or business manager. | Uses the tool's data to identify trends. | An accurate dashboard to see which types of problems are happening and which departments are responsible. |

### 

### 2. User Stories & Use Cases

These stories define the system's functionality from a user's perspective.

* **US-1 (Data Collector):** "As the Data Collector, I need to source or create a dataset of public/internal feedback, where each entry is labeled with the responsible **department** (e.g., 'Traffic') AND the **issue type** (e.g., 'Nuisance')."
* **US-2 (Data Pipeline Engineer):** "As the Pipeline Engineer, I need to build a function that cleans and preprocesses the raw feedback text so it's ready for the classification models."
* **US-3 (ML/DL Specialist):** "As the ML Specialist, I need to design and train a machine learning system that can accurately predict **both the department and the issue type** from a single piece of text."
* **US-4 (Streamlit Developer):** "As the Streamlit Developer, I need to create a simple web page with a text box and a button so a user can submit their feedback."
* **US-5 (End User):** "As a user, I want to type my complaint (e.g., 'There's a broken power line sparking on Main Street') and have the system automatically understand this is for the **'Electricity'** department and the issue type is **'Safety Hazard' / 'Vandalism'**."
* **US-6 (Project Evaluator):** "As an evaluator, I need to access the live app and test it with various inputs to see if it correctly identifies both outputs."

### 3. Functional Requirements (FRs)

These are the specific actions the system **must** perform.

| **ID** | **Requirement Description** | **Priority** |
| --- | --- | --- |
| FR-1 | The system must be trained on a dataset where each entry has feedback\_text, a department\_label, and an issue\_type\_label. | High |
| FR-2 | The data pipeline must clean raw text (e.g., lowercasing, remove punctuation, normalize text). | High |
| FR-3 | The data pipeline must tokenize and process the text into a format suitable for the ML models (e.g., TF-IDF or BERT embeddings). | High |
| FR-4 | The system must train a machine learning model to perform **Department Classification** (e.g., 'Electricity', 'Traffic', 'Health', 'Sanitation'). | High |
| FR-5 | The system must train a machine learning model to perform **Issue Type Classification** (e.g., 'Harassment', 'Nuisance', 'Vandalism', 'Outage', 'Query'). | High |
| FR-6 | The trained model(s) must be saved as artifacts that can be loaded into the Streamlit application. | High |
| FR-7 | The Streamlit application must feature a user interface with a text area for input and a button to trigger the analysis. | High |
| FR-8 | Upon analysis, the application must clearly display **both** the predicted **Department** and the predicted **Issue Type**. | High |
| FR-9 | The application should display a confidence score or probability for **both predictions**. | Medium |
| FR-10 | The final application must be publicly deployed and accessible via a URL. | High |

4. Non-Functional Requirements (NFRs)

These requirements define the quality, performance, and operational standards.

| **Category** | **ID** | **Requirement Description** |
| --- | --- | --- |
| **Performance** | NFR-1 | The sentiment prediction for a typical review (50-100 words) should be displayed in under 3 seconds. |
| **Usability** | NFR-2 | The user interface must be clean, intuitive, and require no special instructions to use. |
| **Reliability** | NFR-3 | The deployed Streamlit application must maintain 99% uptime during the evaluation period. |
| **Deployability** | NFR-4 | The project's GitHub repository must contain a requirements.txt file with all necessary dependencies for a one-click deployment on Streamlit Cloud. |

### 5. Data Requirements

The primary data structure to be used across the project.

| **Category** | **ID** | **Requirement Description** |
| --- | --- | --- |
| **Performance** | NFR-1 | The dual-classification prediction for a typical complaint (50-100 words) should be displayed in under 3-4 seconds. |
| **Usability** | NFR-2 | The user interface must be clean, intuitive, and clearly present the two separate outputs. |
| **Reliability** | NFR-3 | The deployed application must maintain 99% uptime during the evaluation period. |
| **Deployability** | NFR-4 | The project's GitHub repository must contain a requirements.txt file for one-click deployment. |

### 6. System Architecture (High-Level Consideration)

The development team must decide on the model architecture to achieve the dual-classification goal.

**Multi-Task Learning:** Instead of training two completely separate models, you can train one single, large model (like a BERT model) with two different "output heads." One head learns to predict the department, and the other head learns to predict the issue type. This is very efficient because the model learns the core meaning of the text once and uses that knowledge for both tasks.