



**Engineering
& Computing**

School of Computing
& Information Sciences

Patient Safety Comments Topic Analyzer

Product Owner:
Alejandro Arrieta

Team Members:
Diego Hernandez
Bosco Silva
Gunther Alonso
Didier Campos
Timothy Rocha



**Engineering
& Computing**

School of Computing
& Information Sciences

Problem And Motivation:

Problem:

Our Product Manager collected thousands of patient and customer comments from surveys across Latin American hospitals and one of the biggest issues that he commonly faces is having to go through the comments and wanted our team to figure out a way to create a program that could auto evaluate comments with the use of AI in order to find the most frequent topics with regards to patient safety.

Motivation:

Understanding patient feedback quickly leads to safer hospitals, better care quality, improved satisfaction, and faster problem resolution, which ends up benefiting both medical staff and patients.

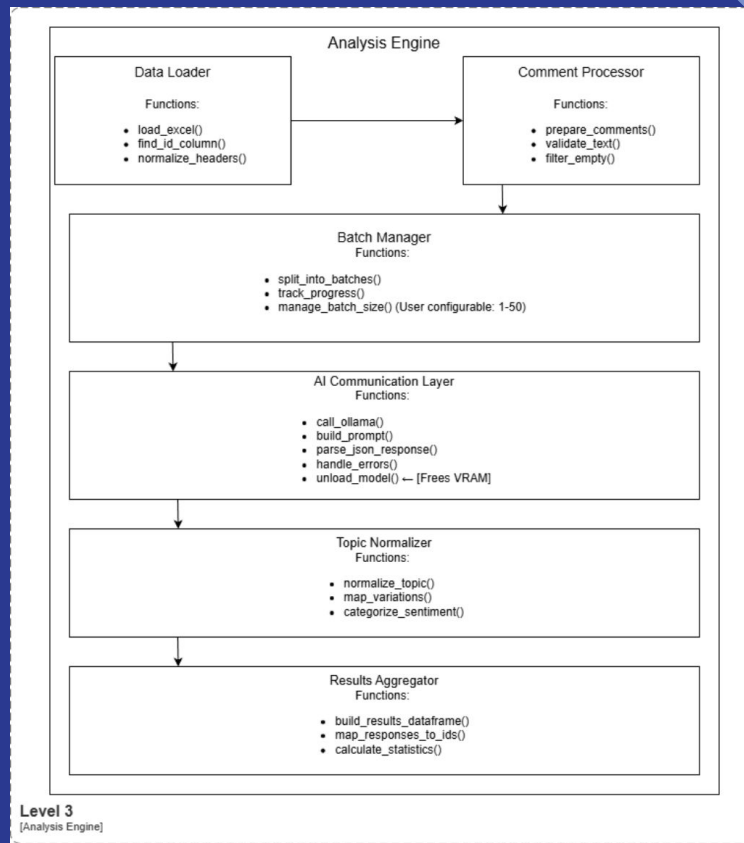
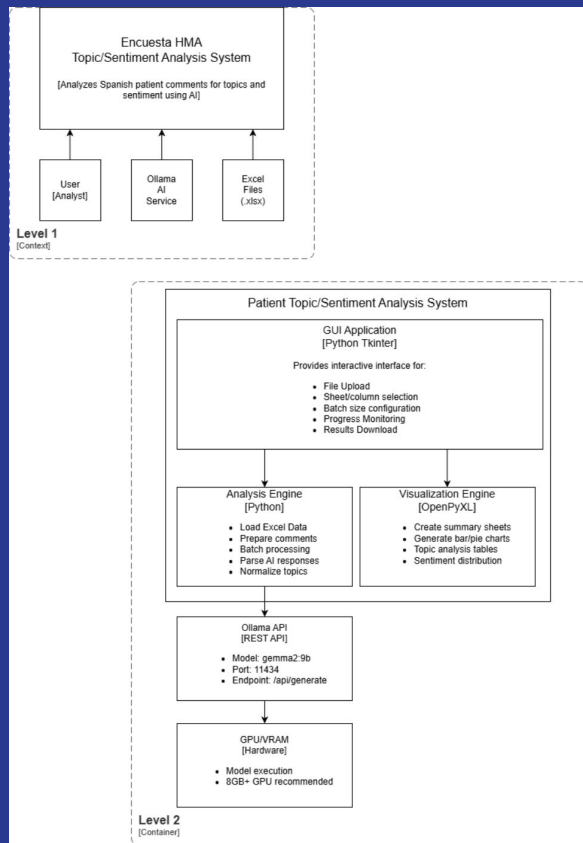
Design overview:

- **Language Model Integration:**
 - Program uses Ollama, an open-source platform for running local language models
- Models used:
 - gemma 2:2b
 - gemma 2:9b
- These models provide high accuracy in Spanish, which is essential for analyzing patient comments from Latin American hospitals

Technologies & Packages Used:

- Python-based program built for large-scale topic analysis
- pandas – Reads Excel files; data processing; DataFrame creation
- requests – Communicates with the Ollama API via REST calls
- argparse – Handles command-line arguments (CLI parsing)
- json – Parses responses returned by the Ollama API
- re – Cleans markdown artifacts from JSON responses
- pathlib – Ensures cross-platform compatibility for file paths
- collections – Tracks topics; provides efficient counting with Counter
- threading – Keeps GUI responsive by running analysis in a background thread
- tempfile – Creates temporary result files before saving
- shutil – Copies generated output files to user-selected locations
- openpyxl – Writes Excel files, applies formatting, generates sentiment charts
- tkinter – Builds GUI elements such as buttons, windows, and dialogs

System architecture:



HydraX9K Update README.md9aaf364 · 3 minutes ago🕒 22 Commits

.idea	Initial Version 1.00	2 months ago
Encuesta HMA - Comments - UTF8 - CLEANED...	Topic Analyzer using Ollama V3	5 minutes ago
Encuesta HMA - Comments - UTF8 - SHORT.xlsx	Topic Analyzer using Ollama V3	5 minutes ago
README.md	Update README.md	3 minutes ago
gemma2_topic_analyzer_v3.py	Topic Analyzer using Ollama V3	5 minutes ago

README

This new version comes with a clean-looking GUI that allows the user to select the excel sheet from their computer. They can also choose the batch size they want the program to use, the default/recommended size is 5. Only increase this number if you have at least 8GB of VRAM or more.

Once the comment analysis has begun, you will visually see progress taking place. It will tell you how many batches have been completed, and how many batches are left. Once the analysis is completed, it will ask you if you want to download the results. You can choose where you want to save these results and you can name the file whatever you want.

NOTE: Make sure the excel comments file has its first row set as 'Response ID', as that is what it currently looks for when scanning and analyzing the comments file. Make sure this is true for all sheets.

Included is a condensed version of the comments file, labeled 'SHORT', for analysis speed and testing. Sheet 1 has 250 comments, sheet 2 has 500 comments.

REQUIRED TO RUN THIS PROGRAM: Python 3.8 or higher.

INSTALL THESE PYTHON LIBRARIES USING THIS LINE: `pip install pandas openpyxl requests tkinter`

INSTALL OLLAMA AND THE GEMMA2:9B MODEL:

1. install Ollama from here: <https://ollama.ai>
2. Open a terminal such as CMD, pull the Gemma 2:9b model by using this line: `ollama pull gemma2:9b`

HARDWARE REQUIREMENTS Minimum: RAM: 8GB GPU VRAM: 6GB (for gemma2:9b) Storage: ~5GB for the model

Recommended: RAM: 16GB+ GPU VRAM: 8GB+ (allows for larger batch sizes) Storage: 10GB+

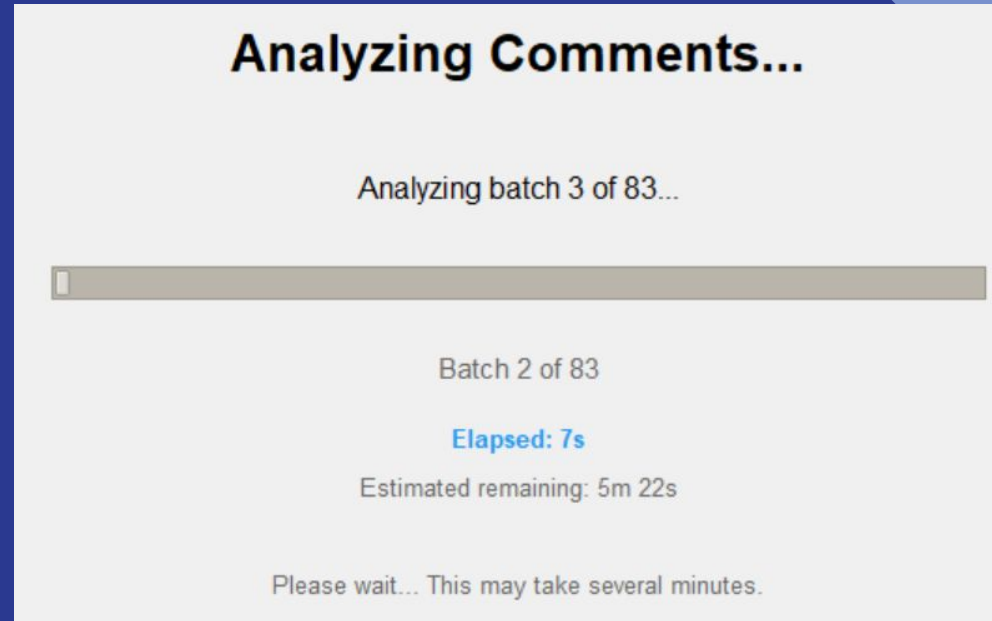
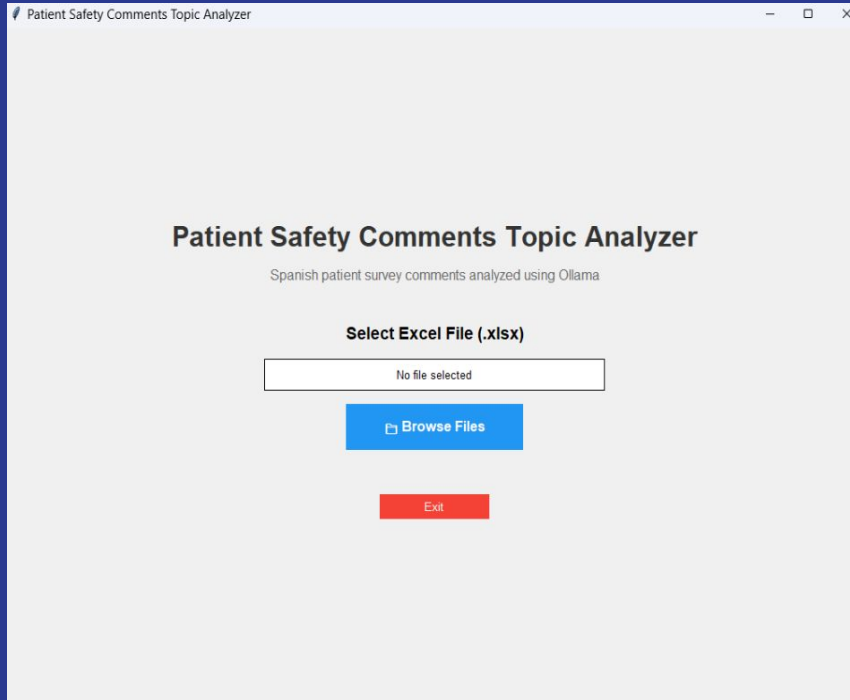
TO RUN THE PROGRAM, USE THIS LINE: `python gemma2_topic_analyzer_v3.py`

Implementation:

Our repository structure is rather simple. It includes two comment files that are used for testing and performance monitoring. One comment file is a condensed version containing fewer comments, so the program can process and complete the task faster.

The 'README' file contains concise directions on how to run the latest version of the program and is updated as needed when the main files are updated, or new files are uploaded.

Processing UI:



Result UI:



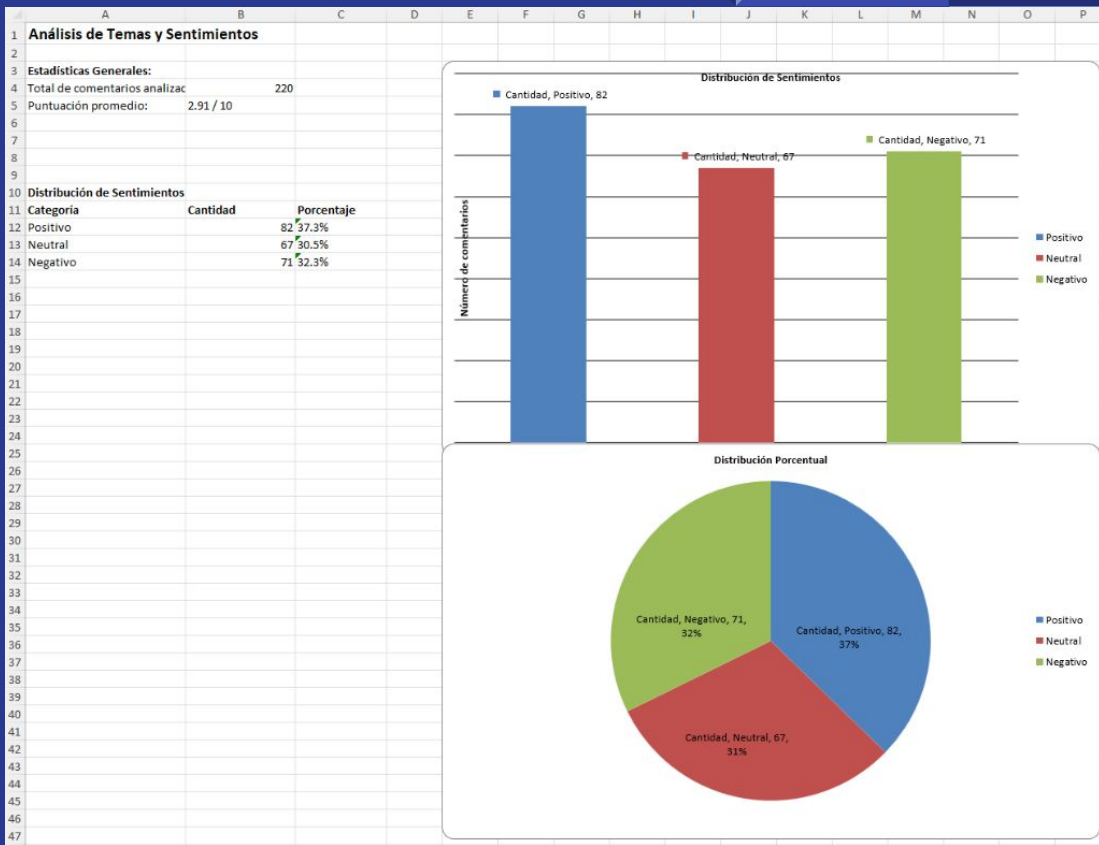
Analysis Complete!

Total time: 4m 25s

Your results are ready to download

 Download Results

Return to Main Page



Testing and Results:

Unit Testing:

We have performed unit testing on the individual components of the program. For instance, when developing the comment summary and comment sentiment analysis features, we ran the program testing with many different AI models that each delivered different results. Ultimately, we landed on the open source Ollama gemma 2:2b and 2:9b models which have given the best results for the project needs.

Integration Testing:

In the integration testing phase, the comment summary and comment sentiment analysis feature code was integrated into the main program and tested to confirm functionality requirements.

E2E Testing:

We have also then end to end tested the program on more than one machine to confirm functionality. From downloading and installing the model locally, then pulling either of the Ollama gemma models for testing and to be downloaded locally through command prompt, tested running the Patient Topic Analyzer exe file, uploading an .xlsx file, tested running the program, and analyzing the results the program returns.

Performance Testing:

We have done performance testing to determine system requirements, the results of which are listed under deployment and operations.

We have tested and ran a full sheet of comments (about 1000) and confirmed the processing times to be 15-20 minutes for a full sheet of comments. We have also tested and run the program on other machines.

Challenges:

The known Challenges of our product are:

- Model Size: Gemma-2 requires significant memory (4-8GB), limiting deployment on edge devices
- Language Support: Optimized for Spanish; other languages may produce suboptimal results
- Context Length: Maximum input limited to model's context window (8K tokens for Gemma-2)
- Real-time Processing: Single-document latency may be high for very long texts

Lesson Learned:

We Learned so many valuable lesson throughout the creation of this project, such as the importance of teamwork, the importance of constant testing as well as keeping records of the different ways we test, and keeping documentation of our day to day work. These lesson will provide invaluable help in our future careers.

Future Roadmap:

Q1 2026:

- 1) Multi-language support (fine-tune on multilingual corpus)
- 2) Batch API endpoint with async job processing
- 3) Performance optimization
- 4) Testing of newer models (gemma3)

Q2 2026:

- 1) Dynamic topic modeling (adaptive topic count)
- 2) Topic trend analysis over time
- 3) Fine-tuning pipeline
- 4) Begin working on a design for Front end of Website

Q3 2026:

- 1) Finish Working on front-end and design of website for product