Multimodal LLM Road Safety Platform

CITS5206 Group 8 Project

22941307 | Sarath Pathari

23633858 | Yuanfu Cao

23743373 | Yuxin Gu

23832048 | Gnaneshwar Reddy Bana

23870387 | Pedro Wang

23959947 | Kanishk Kanishk

1. INTRODUCTION

The Multimodal LLM Road Safety Platform aims to enhance road safety by analyzing both textual and visual data to provide insights and recommendations. Using advanced AI techniques, the platform allows users to upload images and text descriptions of road conditions, which are processed to generate safety suggestions tailored to specific driving environments. The project follows Agile methodologies, ensuring user-centered development. The system has been successfully deployed, and all required documentation, including the user manual and test suites, are provided for seamless operation.

2. SYSTEM OVERVIEW

2.1 Key Features

- **Text and Image Input for Analysis**: Users can provide both text prompts and image inputs for safety analysis.
- **Gemini Model Integration**: The system supports both Gemini 1.5 Flash and Gemini 1.5 Pro models.
- **Image Distortion Options**: Includes effects such as blur, brightness, contrast, sharpness, color adjustments (saturation, hue shift), rain effect, custom overlay, and warp (wave, bulge effects).
- **Batch Processing**: Supports bulk image uploads and folder paths for centralized or individual settings.
- **Customizable Prompts and AI Instructions**: Users can set predefined or custom prompts for analysis.
- **AI-Generated Insights**: The system provides AI-driven safety recommendations.
- **Structured CSV Output**: Results are exported in a customizable CSV format for analysis.

2.2 High-Level Architecture

This Multimodal LLM Road Safety Platform is built and deployed using Streamlit, a Python framework for creating dynamic data applications. The system is composed of several key layers (Fig 1.) :

1) **User Interface Layer**

Streamlit is an open-source Python framework for data scientists and AI/ML engineers to deliver dynamic data apps with only a few lines of code. Using Streamlit components the software provides:

- Image upload (both single and batch mode),
- Text input as prompts
- Distorting controls: apply different kinds of distortion
- Analysis: display image preview and results of the analysis

2) **Backend Layer**

Python backend: handles the core logic and processing of the web application. It manages communication between the user inputs and the various processing modules.

3) **Processing Layer**

- **Image Processing using PIL/Pillow:** Use the python image processing library for hanlding and manipulating images.
- **AI integeration:** Incorporates AI capabilities through LLM API for tasks such as image and text to provide road safety insights.
- **Data Management (Pandas):** for some special case of distorting, we use pandas to handle transforming of images.

4) **Runtime Environment**

Python 3.x Runtime Environment: The system runs on a Python 3.x environment, ensuring compatibility with the libraries and modules mentioned.

5) **LLM API:** The platform currently relies on Gemini API but with few changes, it can support more LLM models.

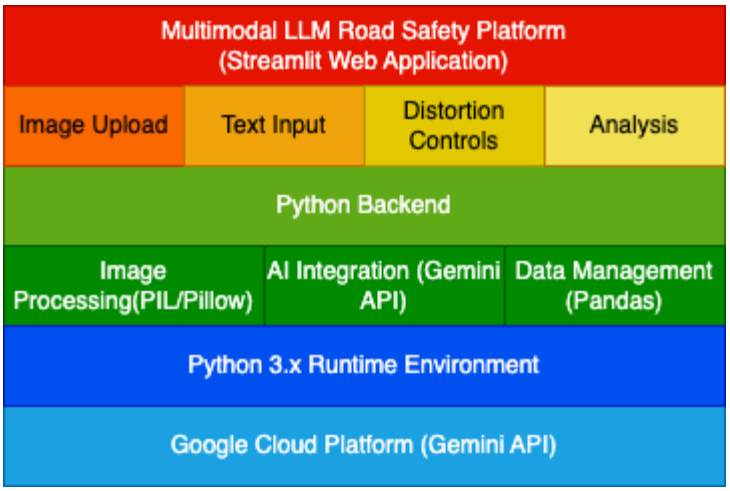


Fig 1.

2.3 System Workflow

The workflow Fig 2. represents the steps of the system which involves image and text inputs being processed and analyzed through LLM.

1) **Image upload and text inputs:** The system starts by allowing users to upload images and enter text inputs.
2) **Image processing:** After uploading the images, users can add some distortion effects to images.
3) **LLM Analysis:** The backend will forward the processed images and the text input to LLM and get the results back.
4) **Output processing:** The output is displayed on GUI interface and (if in batch mode) users can download the results in form of CSV.

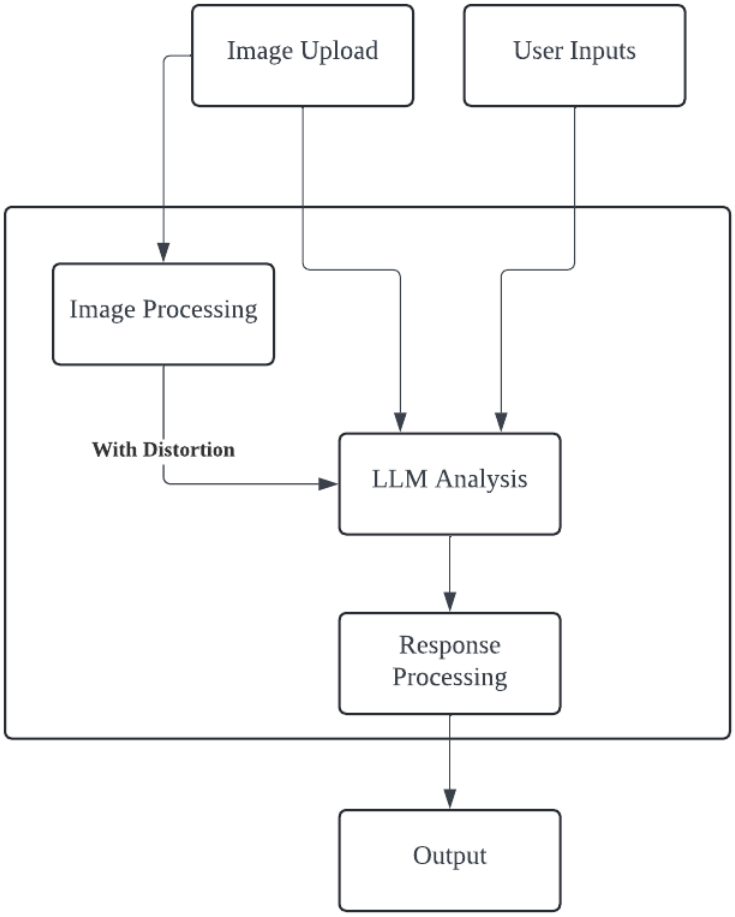


Fig 2.

2.4 Further development

1) **Support more LLM models -** The code can be refactored to allow integration of more LLM models.
2) **Support sequential images analysis –** several images as one input to test AI's cognitive ability on perception of video like inputs.
3) **More user-friendly structured outputs -** In *alternative* branch, users are able to add extra columns to csv output and can also specify how to extract the information using python regex. It can be integrated into *main* branch.
4) More distortions can be integrated into the platform.

2.5 System Deployment and Access

The Multimodal LLM Road Safety Platform has been successfully deployed and is accessible via a live Streamlit web application. Users can interact with the system by uploading images, inputting text, and applying image distortions, all of which are processed by the AI model. The deployed system ensures that both single and batch uploads are supported for testing the platform's capabilities.

Access the platform here: https://llmroadsafetyplatform.streamlit.app

3. SOURCE CODE

The full system, including source code, technical documentation, unit tests, and other project materials, is available in the group's GitHub repository. This repository also contains the user manual and testing documents, ensuring easy access to all necessary components.

Access the source code and additional resources here:

GitHub Repository (https://github.com/AlteredOracle/CITS5206)

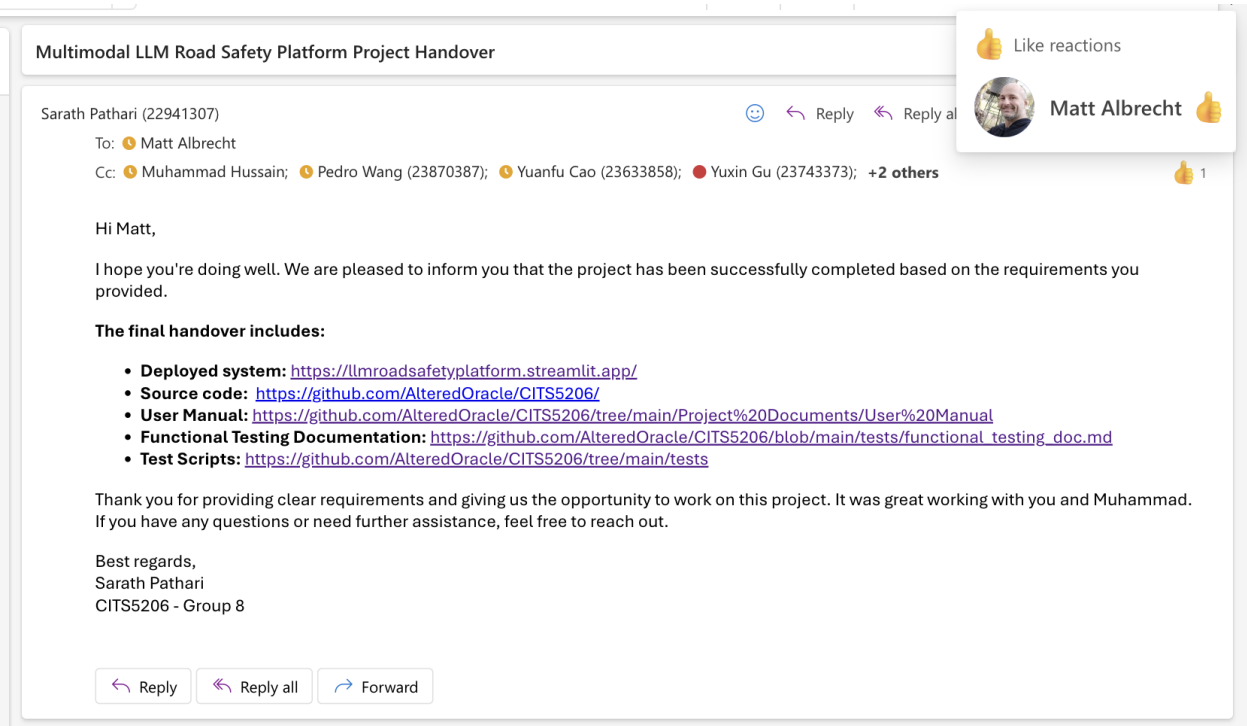
4. PROJECT HANDOVER

The final handover of the Multimodal LLM Road Safety Platform was successfully completed on **17th October 2024**. The client has been provided with access to the deployed system, source code, and all necessary documentation.

Handover Details:

- **Deployed System**: Link to deployed system
- **Source Code**: GitHub Repository
- **User Manual**: User Manual Link
- **Functional Testing Documentation**: Functional Testing Documentation Link
- **Test Scripts**: Test Scripts Link

A formal email was sent to the client confirming the successful project handover.



The email confirms the deployment and all deliverables. The client has acknowledged receipt of the system and its documentation.

5. Documentation and User Manual

The Multimodal LLM Road Safety Platform is supported by comprehensive documentation to guide users and developers. The documentation includes:

- **User Manual**: A detailed guide for setting up, using, and troubleshooting the system. It covers image upload, batch processing, and AI analysis.
- **Functional Testing Documentation**: Describes the tests performed to validate the system, ensuring all features work as expected.
- **Test Scripts**: Provides the test scripts used to perform automated tests.

All documentation can be accessed via the following links:

- User Manual (https://github.com/AlteredOracle/CITS5206/tree/main/Project%20Documents/User%20Manual)

- Functional Testing Documentation (https://github.com/AlteredOracle/CITS5206/blob/main/tests/functional_testing_doc.md)
- Test Scripts (https://github.com/AlteredOracle/CITS5206/tree/main/tests)

6. TEST SUITE

The primary goal of this testing process is to validate the core functionalities of the "Multimodal LLM Road Safety Platform," ensuring that the platform operates efficiently and reliably during real-world use. The following areas were the focus of the testing:

- **Image Processing & Distortion Effects**: The platform successfully applies different types of image distortion effects, such as blur, brightness, contrast, sharpness, color transformations, etc., ensuring accurate image transformation.
- **Integration with the Gemini API**: The platform integrates seamlessly with the Gemini AI model and generates correct AI analysis results based on the provided inputs.
- **Error Handling Mechanism**: For incorrect inputs or exceptional scenarios, the platform properly captures and handles errors, preventing crashes and providing useful error information.

6.1 How to Run Tests

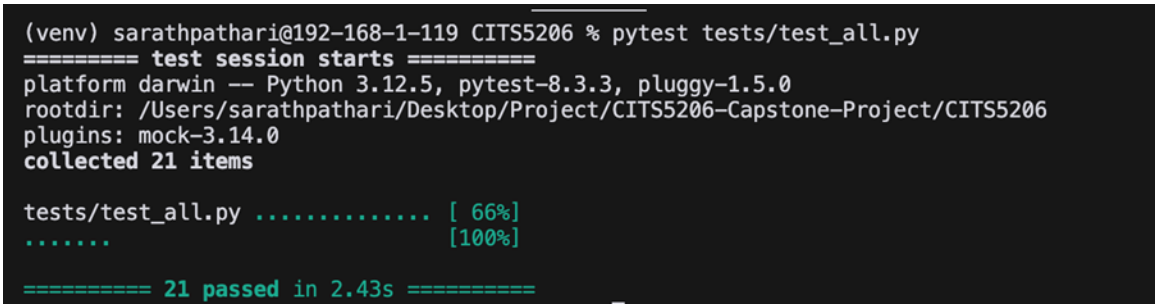
To run the complete test suite, follow these steps:

- **Install Dependencies**: Before running the tests, ensure that all dependencies are properly installed. You can install the required Python packages by running the following command:

  ```
  pip install -r requirements.txt
  ```

- **Run Tests**: Execute the following command to run all test cases:
  pytest tests/test_all.py
  This command will run all unit tests in the project and return the test results.
- **View Test Results**: Once the tests are completed, the terminal will display the pass/fail status of each test. Ensure that all test cases pass to confirm that the platform's functionality is working as expected.

6.2 Test Results



All 21 test cases were executed successfully, covering the core functions of image processing, AI integration, and error handling. The pass rate is 100%, indicating that the platform is functioning correctly without any issues.

6.3 Functional Testing Document

This document outlines the functional testing performed for the "Multimodal LLM Road Safety Platform."

CITS5206/tests/functional_testing_doc.md at main · AlteredOracle/CITS5206 (github.com)

7. PROJECT MANAGEMENT AND PLANNING

Task Management: Trello

- **User Stories & Task Breakdown**

  The team utilized Trello as a central tool for task management, breaking down all project requirements into clear and actionable *User Stories*. These stories ensured

each task was well-defined with specific goals and criteria for completion. Each member was assigned specific tasks, which made it easy to track responsibilities.

- **Board Structure**

  The Trello board was organized into the following columns:

  - ***To Do*:** Tasks yet to be started
  - **Doing:** Tasks actively being worked on
  - ***Done*:** Finished tasks, ready for sign-off

  This structure allowed for clear visibility into the project's workflow and provided insight into progress at every stage.

- **Task Details & Priority Management**

  Each task card included detailed descriptions, deadlines, and relevant links (such as code repositories or documents). Priority management was facilitated using colored labels, with critical or time-sensitive tasks marked in red. Due date reminders were also implemented to ensure timely task completion.

- **Sprint Review & Planning**

  At the end of each sprint, the team conducted a review to assess incomplete tasks and adjust priorities accordingly. This iterative process enabled effective reallocation of resources and timely adaptation to changing project requirements.

  Trello Board

Version Control and Collaboration: GitHub

- **Commit & Branch Management**

  GitHub served as the team's primary platform for version control. Regular commits were made, each accompanied by detailed messages outlining the purpose and nature of changes. This provided transparency and traceability, ensuring all team members stayed informed on project development.

- **Pull Requests & Code Review**

The team utilized *Pull Requests (PRs)* to propose code changes and updates. Before any PR was merged into the main branch, it underwent thorough peer review to ensure that it adhered to the project's coding standards, was free of bugs, and met performance expectations. This process was crucial in maintaining code quality and preventing integration issues.

Closed Pull Requests

- **Issue Tracking**

The team employed the *Issues* feature in GitHub to log tasks, bugs, and feature requests. Each issue was assigned a specific priority and tagged with relevant labels, making it easier to track progress. This systematic approach ensured that all issues were resolved in a timely manner and that nothing was overlooked during development.

GitHub Issues

Adaptability and Client Feedback

- **Client-Centric Adjustments**

The project team demonstrated a high level of adaptability in response to client feedback. Whenever the client proposed new changes or additional features, these requests were swiftly incorporated into the task management system on Trello. The team reassessed and adjusted priorities to reflect the updated project scope.

- **Real-Time Changes and Issue Resolution**

The team efficiently handled changes by logging new requirements and bugs as issues on GitHub, ensuring that any additional tasks or revisions were swiftly addressed. This ensured the project remained aligned with the client's expectations while maintaining a clear path to completion.

- **Problem-Solving and Flexibility**

Faced with technical challenges and occasional time constraints, the team's quick problem-solving abilities ensured steady progress. The team's flexibility in managing unforeseen challenges contributed significantly to meeting deadlines and maintaining client satisfaction.

Meeting Minutes

1. 26-07-2024-MOM.docx
2. 03-08-2024-MOM.docx
3. 09-08-2024-MOM.docx
4. 16-08-2024-MOM.docx
5. 19-08-2024-MOM.docx
6. 27-08-2024-MOM.docx
7. 29-08-2024-MOM.docx
8. 08-09-2024-MOM.docx
9. 19-09-2024-MOM.docx