Progress Report #2	
Course Code: CPE201L	**Program:** Computer Engineering
Course Title: Data Structures and Algorithm	**Date Performed:** September 13, 2025
Section: CpE - 2A	**Date Submitted:** September 13, 2025
Members: Balana, Jerkielle Roen O. Balaoro, Judge Wayne B Barbas, Steven Jade Dispo, Lei Andrew Sorellano, John Kenneth	**Instructor:** Engr. Maria Rizette Sayo

1. Objectives

- To create and implement a queue-based chatbot system for UCC AIMS portal assistance
- To demonstrate practical application of queue data structures in actual situations
- To create a prototype that can handle basic academic inquiries from students
- To create or to provide a foundation for future enhancements using more complex data structures

2. Discussion

Building on the planning phase from our first report, we have now moved into the development phase. Our project, **AIMSSIST**, has been created using Python and is built around two main classes: **QueueManager** and **AIMSKnowledgeBase**.

The **QueueManager** serves as the core of the system, showing how queues can be used in practice. It uses three queues **input_queue**, **processing_queue**, and **response_queue** to manage the flow of a user's query. When a user sends a message, it first goes into the input_queue. From there, it is moved to the processing_queue where the system simulates handling the request. Once a response is generated, it is placed in the response_queue and then displayed to the user. This step-by-step process ensures that all queries are answered in the order they are received.

3. Materials and Equipment

- PyCharm
- Streamlit
- Python
- Github

4. Procedure

1. Open Python(Google Colab, PyCharm, etc.) and Streamlit and create a code.
2. Create a code that manages incoming questions using a queue, ensuring they are handled in order.
3. We programmed the chatbot in Python using the Streamlit package with answers to common questions about enrollment and grades.
4. We also used Streamlit to create the chat window and the "Queue Status" display.
5. We linked the user interface to the queue system and the chatbot's knowledge so it could process questions and give answers.
6. Ran the program to check if the chat and queue display worked correctly.

5. Output

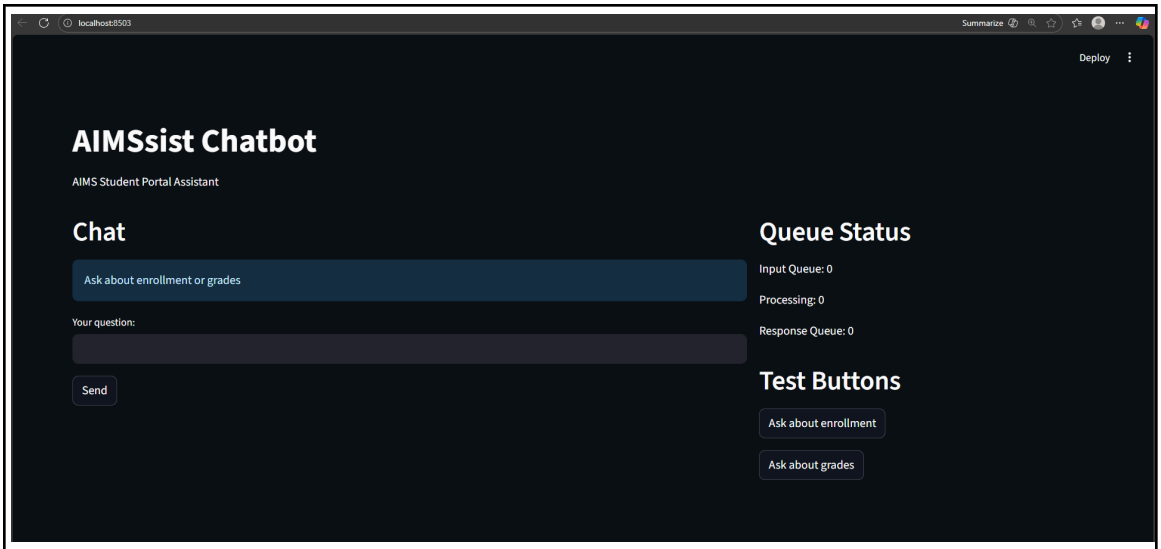

Figure 1. Interface Progress

In the image above, it shows the running prototype of the AIMSsist chatbot. This interface is built using the streamlit package in python. On the left, there's a “Chat” section that displays the conversation and a text box for user input. On the right, it will show the real-time update of count of messages, and the implementation of Queues. On the test buttons, if you click the buttons above it, it will act as a shortcut to send messages without typing it.

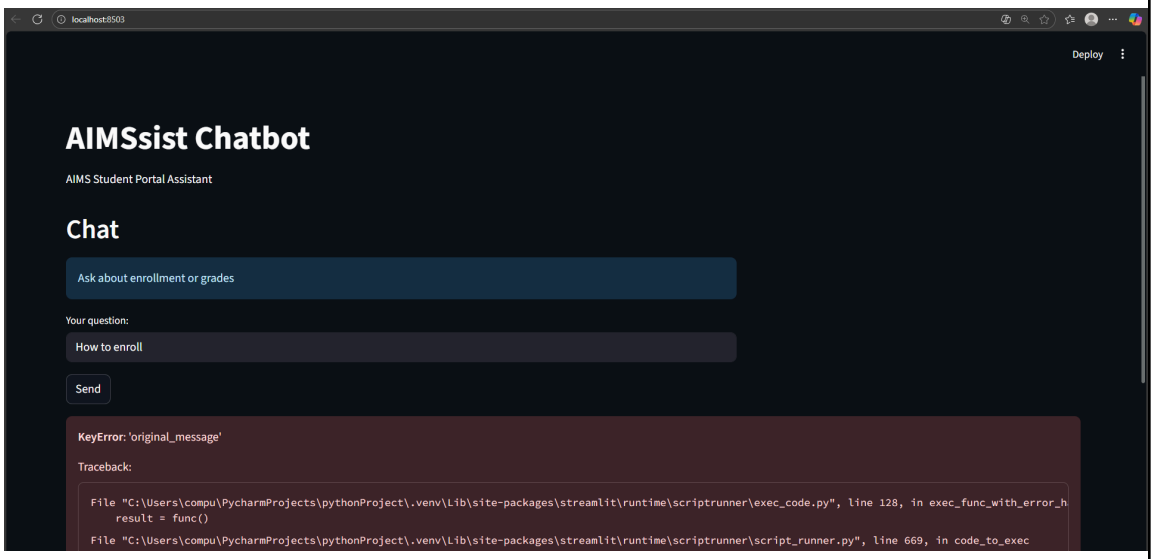

Figure 2. Error on Interface

This interface however has many errors as the purpose of this is to just act as the foundation for the next codes.

6. Conclusion

This second progress report shows that we have properly moved from the planning stage to a practical execution. A functional prototype of the AIMSsist chatbot has been created, and it efficiently handles customer inquiries via a queue-based mechanism. It has been shown that combining Streamlit for the frontend user interface with Python for the backend functionality is an effective strategy.

The existing prototype provides a strong basis for future improvements while meeting the goals for this phase. To be clear, this version is the initial version of the application. The interface is purposefully simple as a result, and there might still be a few small bugs to fix. By enhancing the user interface and enhancing the chatbot's general stability, we intend to address these issues in our upcoming progress reports.