Store AI Assistant Project Report

1. Implementations

1. Use Mederation feature provided by openai to check if user input, and ai output are appropriate.

```
client.moderations.create(input="...")
```

2. Use a series of functions stored in utils.py to extract relevant product information from user's original questions.
3. Use LLM to check if the respond model gave before answer user's questions.

```
user_message = f"""
    Customer message: {delimiter}{user_input}{delimiter}
    Agent response: {delimiter}{final_response}{delimiter}

    Does the response sufficiently answer the question?
    Answer "Y" for yes, "N" for no
    """
```

4. Use log_interaction() to log AI-User interaction into Json file. A new file will be created each day, by setting the file name as the current date.

```
date_of_day = datetime.datetime.now().strftime("%Y_%m_%d")
filename = f"{date_of_day}_log.json"
```

2. Challenges

The one challenge I faced is to apply my process_user_message() function to Gradio interface while keeping the chat history with chain prompt system. I uses three ways to try to figure it out. First, look through gradio block examples to trying to understand the log. Second, ask ai to help. Third, write out a basic logic outlines to clearify the process.

3. Potential improvements

1. Add a clear button that can clear the chat history (both on the interface and stored in variable) so that user won't have to refresh to page for a brand new chat.
2. Add some of the example prompts to give a better sence of what this chat assistant can do.