Moody Bot Application

By: Erica Moisei

Intermediate Java Programming - New Brunswick Community College

Author Note

Submitted to the Information Technology Department in partial fulfillment of the requirement of PROG1090J. The school may make copies of this document and any associated computer files for non-profit purposes without further permission of the author.

Submitted by:

_____

Author, Erica Moisei

Accepted by:

_____

Instructor, Chris Cusack

Contents

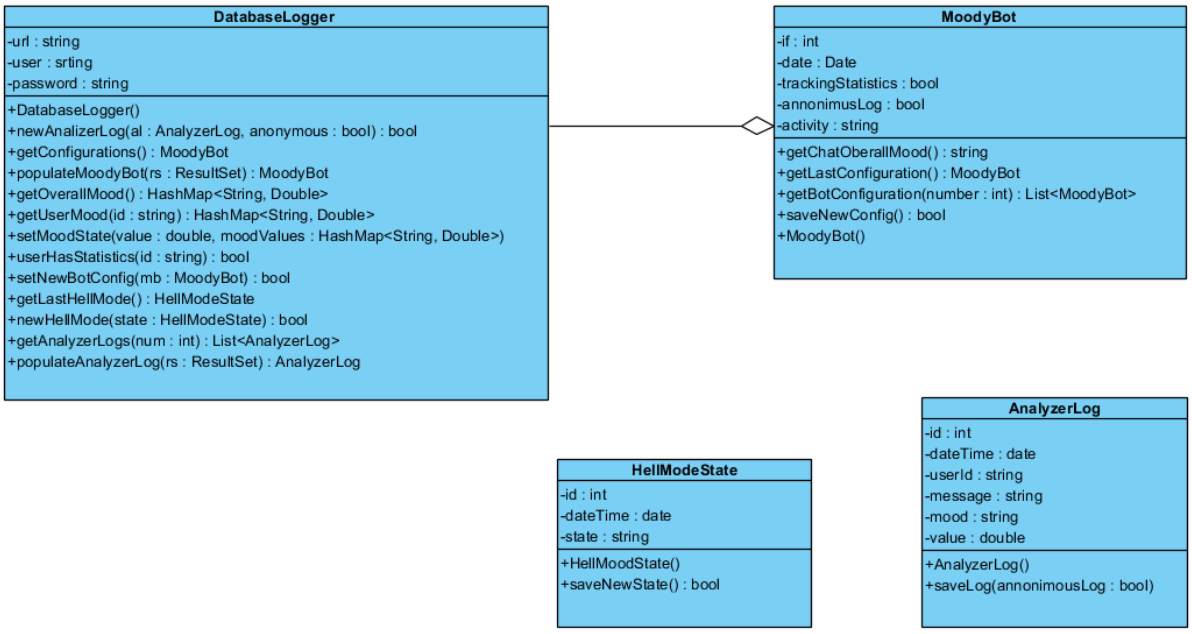
Abstract

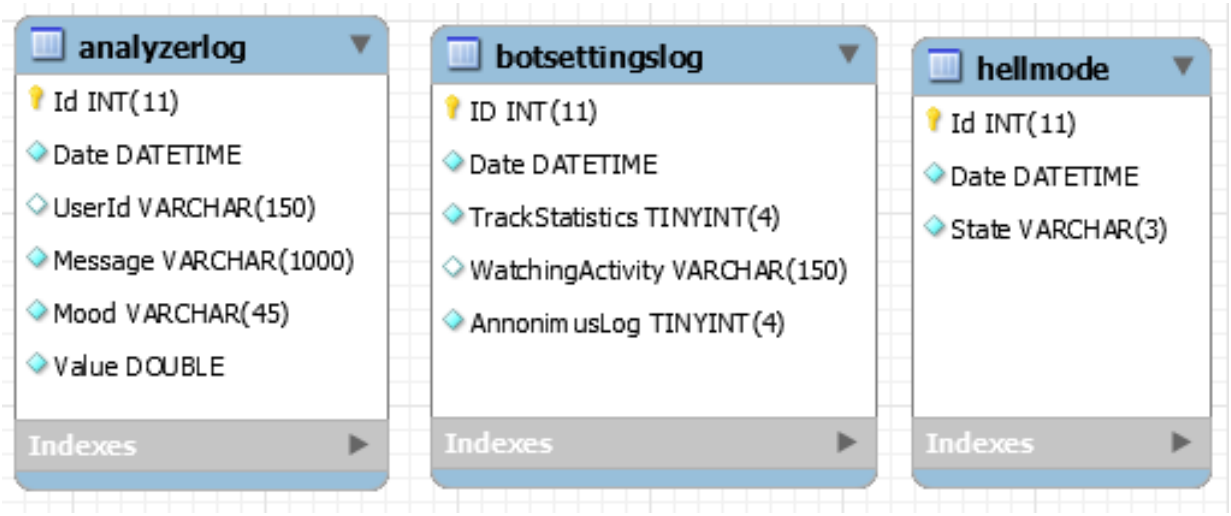
This report included detailed documentation about the project I created such as information about the project, resources used, possible extensions, summaries and conclusions, user manual and some of the project source code.
The project I have created is called Moody Bot Application. Initially I wanted to create a program that would analyze the sentiment of the email message, this idea came to my mind after I attended DevCon 2019 in Fredericton. This idea was already implemented by a company and I wanted to do something similar. I did a research and ended up with the whole code solution. I still wanted to use the AI library for sentiment identification I found, so I decided to do a Discord chat bot that will analyze user's sentiments by the message they send and display the result.

Introduction

Moody Bot Application is a Discord chat bot Java Application that includes two 2 applications. First is the Bot Applications, second is the UI controller to configure bot's settings.
Moody Bot Application is a console app that configures the bot and connects to Discord via web socket using JDA library. The app will be listening for the events that will be coming from the chat and perform different action if the command matches one of the existed commands. All the commands are manually configured in a separate class that extends ListenerAdapter (JDA class). UI app gives you the ability to configure bot and see statistics that is saved to the database. This report is a documentation of the application including review of resources, implementation details, testing, possible extensions, summation conclusion, bibliography, user manual and program source code.

Class Diagram and ERD





Literature Review

In order to create Moody Bot Application, I have researched some topics that weren't covered by in Intermediate Java Programming course.

JLabel Image Icon in Swing

To create an image in swing you must use JLabel and add an image icon. There is no other way to implement images in swing as far as I know. I added a Moddy Bot Image I created in Adobe Photoshop last Term.
Resource: https://www.geeksforgeeks.org/jlabel-java-swing/

Java Sockets (TCP)

To establish connection between 2 Java projects I choose to use a TCP connection. Moody Bot App is the server, UI is the client.
When the server is launched the port 4444 (it can be almost any port) opens and the server is listening for the connections and a specific command. The only one command we need will indicate that the new bot configuration has been saved to database, and Moody Bot will perform an update from database. After a client disconnects the server keeps listening for new connections.
When the UI app is launched it connects to the server on a 4444 port and keeps the connection opened until the app closes. On the button click (Save Configuration) I increment a counter, so the while loop will get into an if statement in order to send a message to the server about the new configuration that has been recently saved. Wee need the while loop and if statement to keep the connection opened while the app is running. ON the app closing, there is thread sleep method that gives the program to close the TCP connection between the server and the client.
Resource: https://www.baeldung.com/a-guide-to-java-sockets

JDA – Java Discord API

To implement the functionality, I choose JDA (Java Discord API) Library that uses web sockets for connection with the chat using bot token, which you van get at the Discord Developer Portal.
The WebSocket API is used to receive events from Discord, including message creation, message deletion, user kick/ban events, user permission updates, and many more. Communication from a bot to the WebSocket API on the other hand is more limited.
With this library I have: established connection with the Discord chat; configured message receive events; guild join/leave events; styled the response message appearance; sent a response back in the chat, and private message to the user that triggered the event; configure bot's status and its activity; assigned roles to users.

Resource: https://www.toptal.com/chatbot/how-to-make-a-discord-bot

Stanford NLP library

To analyse messages, I used a free and open source Stanford Natural Language Processing library. Stanford CoreNLP provides a set of human language technology tools. It can give the base forms of words, their parts of speech, whether they are names of companies, people, etc. To use the library, I needed to use a Maven application, which was not covered in the course as well. To be able to use the CoreNLP Dependency I needed to create 2 classes: *SentimentAnalyzer* and *SentimentHandler*. The Analyzer uses the Stanford NLP library to determine the sentiment of the text and produces a string containing the sentiment annotations for each sentence in the text. If the message will have more than one sentence each sentence will be analyzed separately. The results are saved in a list and then the overall mood is calculated using values between -15 and 15.

Resources:

1. https://www.toptal.com/java/email-sentiment-analysis-bot
2. https://stanfordnlp.github.io/CoreNLP/

Implementation of Moody Bot Application

Moody Bot Application has two application that are connected to each other providing a full functionality to the user. Moody Bot Application is a console app and Moody Bot UI is a JFrame Swing app. When both applications are running at the same time they are connected via TCP.

Moody Bot Application

The main functionality of the Moody Bot is to receive messages from the chat members with a specific command, perform sentiment analyze and send a response message back to the chat with the identified mood and a random predefined message for each type of mood. There are 5 types of the mood: Very Positive, Positive, Neutral, Negative, and Very Negative. The Bot can be configured in the UI, and when the statistics configuration is on, the analyzer log will be saved to the database (MySql) with the time, message, result, value, and user id. Although, if the anonymous configuration is on, the author of the message won't be recorded.
Currently there is more functionality added for a better user experience. The commands are triggering the bot to perform different actions. Also, on member join / leave events a (random) notification message is sent to the chat.
Chat commands overview:
~info – general information about the bot that shows all the available commands and their short description. Very helpful for new members.
~mood [message] – identifies sentiments using provided message and sends a random response message with the mood identified.
~chat statistics – shows the overall chat mood and value representation
~my statistics – personal mood statistics, send a private message with a response
~UpdateActivity – updates watching activity from the latest bot configuration
~HellMode? – shows current Hell Mode state (on/off)
~HellMode [on/off] – turn on/off Hell Mode (admin only)

Moody Bot UI

User Interface Application gives you the ability to configure the bot. Set its Watching activity, turn on and off tracking statistics, and anonymous log. Also, you can retrieve history logs from the database to see the bot configurations, analyzer logs and the overall chat mood.

Testing

Functionality of the project was tested iteratively. Step by step new functionality was added, tested and modified when needed.
Most of my testing I was using a debug tool. To test the connection to the database I was inserting and receiving the test data using both applications and checking the results on the database side and application. For the bot events tests I, and Ben were sending commands from Discord chat. I was comparing the expected results with the once we were getting. If at any point of testing the errors were thrown, I fixed them all.

Possible Extensions

The main purpose of Moody Bot Application was to create a Discoed chat bot that would analyze sentiments from the users' messages.

Critique

The applications work properly. The main goal of the bot has been achieved. The Moody Bot Application analyzes the messages by a specific command and gives the user a response with the result. Also, bot has extra functionality that helps to track chat and users' statistics. The latest added feature was the Role classification, which depends on the personal user statistics. The member with a mood value over 10 will be given an Angel role, between 10 and -10 – Folks, and below -10 – Devils role. The Devils chat restrictions: no emojis, no reactions, no sending files, no voice, no live. The UI app is also functional. It saves bot's configurations to the database and provides some statistics.
With some small changes and improvements this application can be deployed on a server as a fully functional Discord chat bot.

Improvement

For any Discord chat bot new functionality and improvements can be added constantly. For the functionality I have right now, I would still add a few improvements:

1. Add extra privileges for Angels role
2. Provide more help messages about each command (how to use a command)
3. Add more various response messages
4. Add a time limited mute for the members with a very bad statistics (close to -15)
5. Try to switch from a TCP connection between java applications to GRPC. It is a Google open source framework for Java applications that will establish the connection between the apps. Also, it will allow you to use methods from another application if you specify the method you want to give access to. It will be helpful with not having the same methods written in two applications.

Summary and Conclusions

The Moody Bot Application was a very fun and interesting experience for me as a new Java developer. The result is satisfactory. I implemented the analysis of the messages using Stanford CoreNLP library and recorded statistics. Also, extra features were added to give the admin, and users more functionality.

I have learned how to manage a project with a free subject; follow the plan and perform research in new areas; ask instructor for help when stuck in one place for a long time. I also, discovered dependencies, that can be used in java projects to help you in achieving the results using other open source libraries.

References and Bibliography

JLabel Image Icon in Swing
https://www.geeksforgeeks.org/jlabel-java-swing/

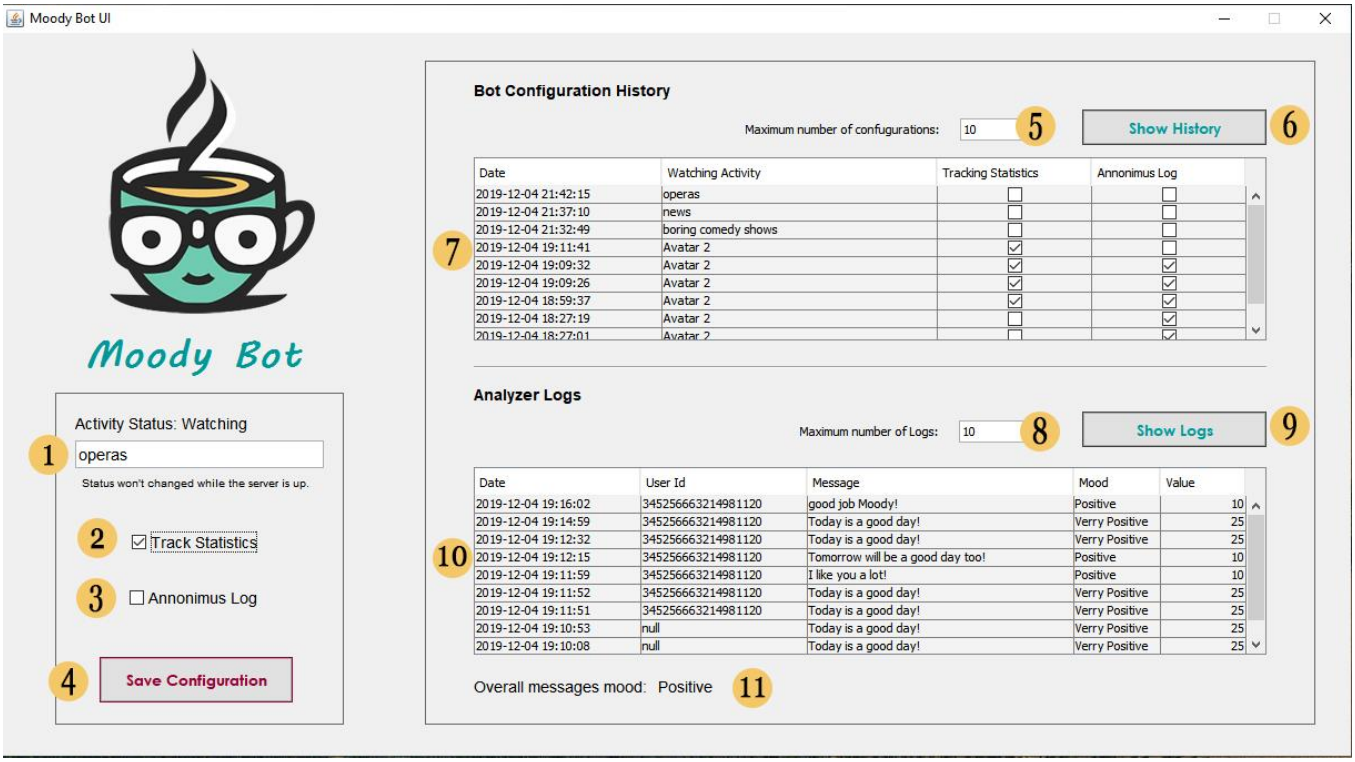
Java Sockets (TCP)
https://www.baeldung.com/a-guide-to-java-sockets

JDA – Java Discord API
https://www.toptal.com/chatbot/how-to-make-a-discord-bot

Stanford NLP library
https://www.toptal.com/java/email-sentiment-analysis-bot
https://stanfordnlp.github.io/CoreNLP/

Appendix A - User Manual

Moody Bot UI

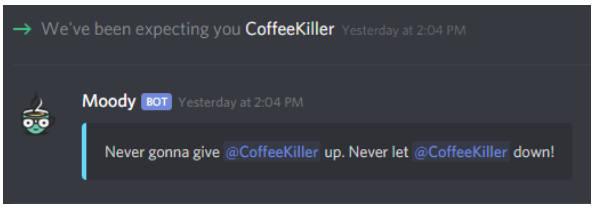


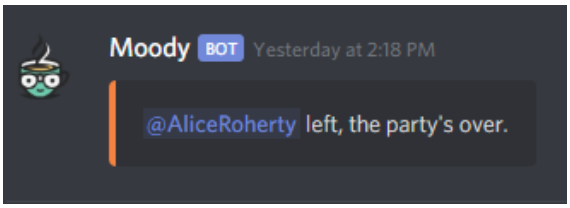
1. Add a watching activity
2. Turn on/off Tracking Statistics
3. Turn on/off Anonymous log
4. Click to save new bot configurations and notify the Moody Bot Application (console app) about new configurations that were recorded to the database.
5. Provide number or records you want to retrieve to see bot configurations history.
6. Click to retrieve bot configurations history.
7. Table with the data requested. Includes: date, watching activity, Tracking statistics, anonymous log.
8. Provide number or records you want to retrieve to see analyzer logs history.
9. Click to retrieve analyzer logs history.
10. Table with the data requested. Includes: date, user id, message, mood and value.
11. Shows overall chat mood as a string representation.

Moody Bot Application

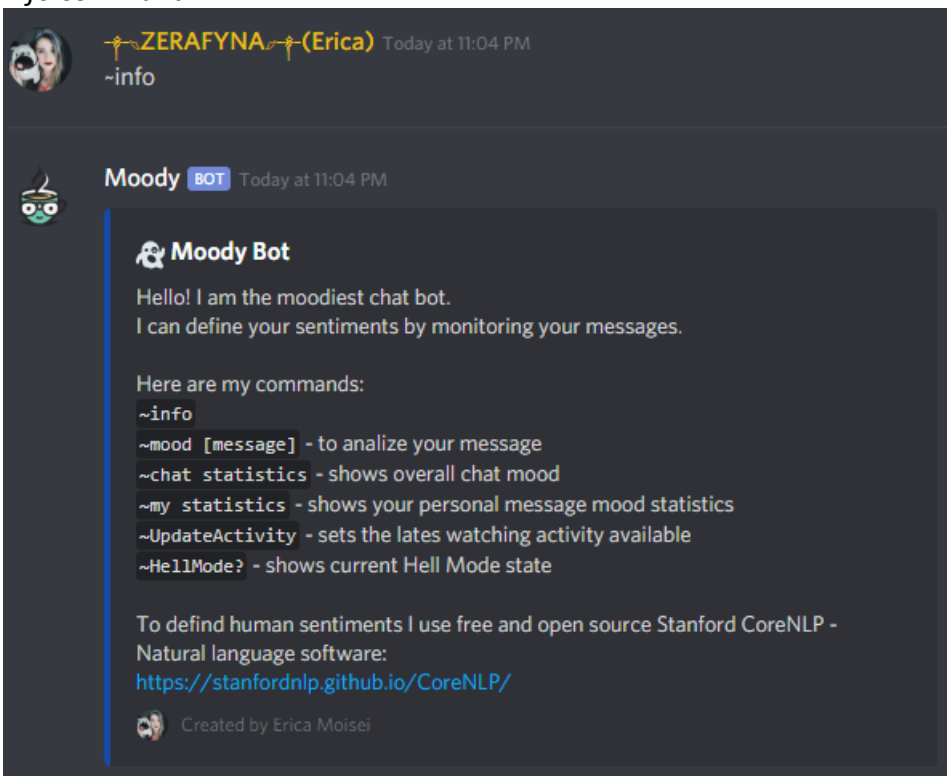
Events and commands and responses:

Join and leave messages:

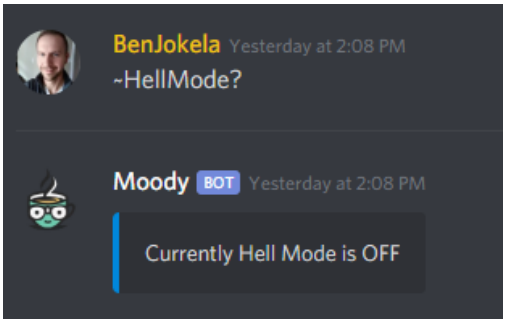


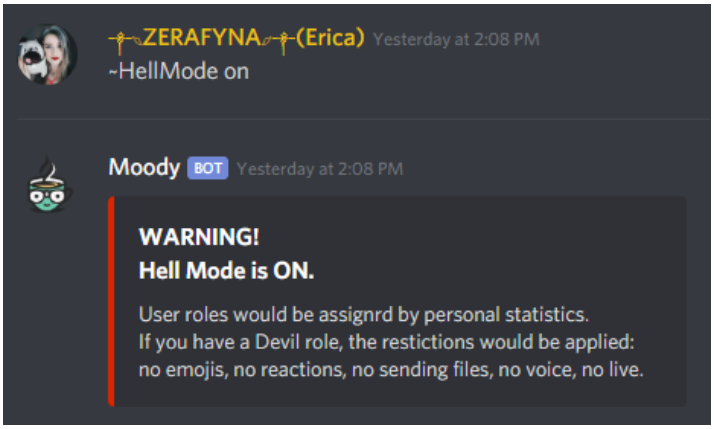


Info command

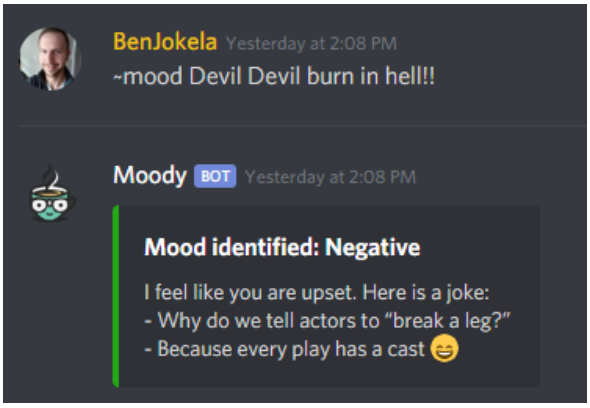


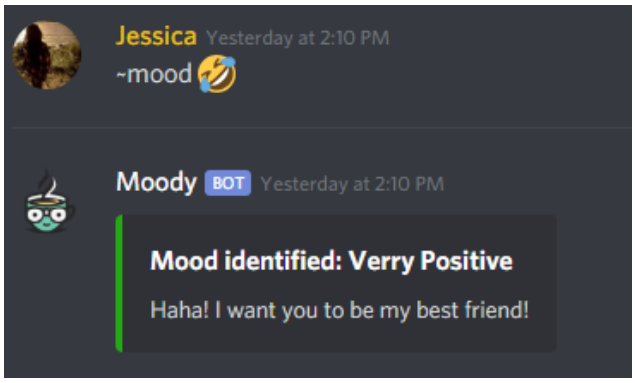
HellMode? And HellMode on/off commands:



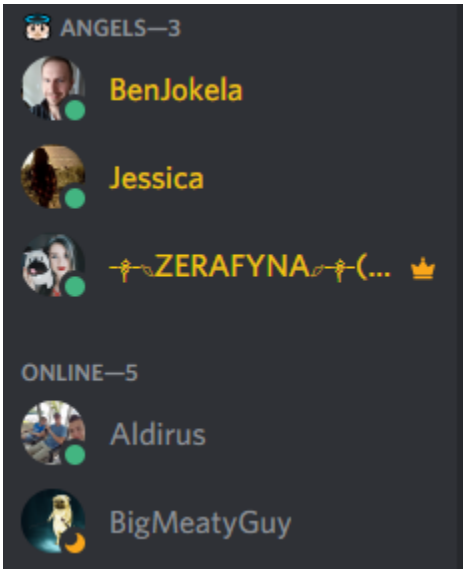


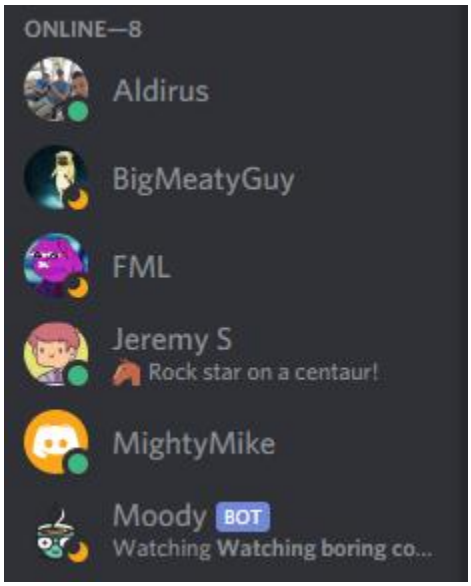
Mood command with positive and negative responses



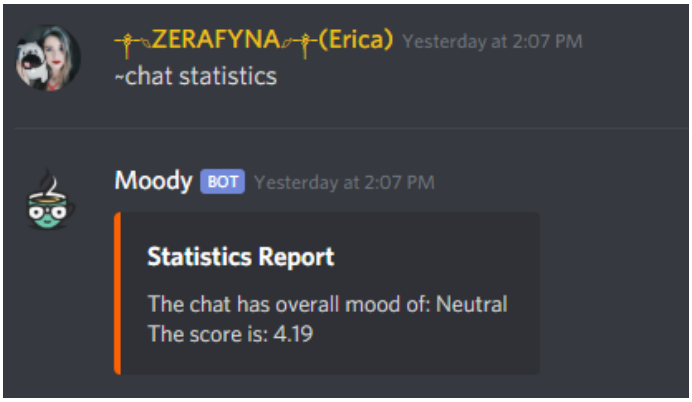


Angel's role, and Bot Watching Activity





Chat statistics command



my statistics command with private responses