Christopher Rech - 124000677

Andrew Allen - 124008219

Task 3 Report: GroupMe Search Bot

Introduction

The GroupMe Search Bot is designed to solve the problem of the lack of an existing search functionality in GroupMe. For background, GroupMe is a messaging app used by most college students in the United States. GroupMe offers the same basic messaging services as other popular messaging apps such as Slack and Facebook Messenger, but GroupMe lacks the functionality to search the messages of a chat.

The lack of a search feature in GroupMe is surprising, given the popularity of the app. The combination of popularity and missing functionality in GroupMe presents a rare opportunity to develop a new feature which would potentially be appreciated by thousands of users. Furthermore, the existing GroupMe API allows for a high degree of freedom for programs to automate user action. Through the GroupMe API, a program can receive and send messages on a regular user account with almost all of the same permissions as a human user. Thus, the development of a search functionality for GroupMe is not only useful, but also feasible.

The search functionality in a typical messaging app is a part of the user interface. For example, there may be a search bar at the top of the screen, just above the content of the group chat. Due to the constraint of only being able to interface with GroupMe, rather than modify its source code, the process of conducting a search must be somewhat different. The idea is to have a bot account which can receive, process, and respond to search queries. When a user wants to

perform a search, they can message the GroupMe Search Bot with the search keywords, and the GroupMe Search Bot will respond with the search results in the chat. If the GroupMe Search Bot is not already in the chat, it can be added by the user. This method results in search results being public to all members of the chat, which reduces privacy but allows for other users in the chat to see the GroupMe Search Bot in action. The other users may then add the GroupMe Search Bot to their own groups, ideally spreading the bot to the entire network of GroupMe chats.

Overall, the GroupMe Search Bot implements a key functionality of messaging apps into GroupMe, a popular messaging app which lacks this basic functionality. Currently, the GroupMe Search Bot can be used by anyone who adds the bot to their group, but the ideal scenario is for the GroupMe Search Bot to spread all across the app.

Prior Work

One example was found of an existing tool to search GroupMe chats, but this tool had severe privacy and security issues due to its method of retrieving information. The existing tool, SearchMe, requires a user to provide their access token prior to initiating a search [1]. The access token is then used by the program, which reads the user's messages in order to perform a search [2]. The issue with this method is that the access token gives permission for the program to read and write to all of the user's chats. In order for the user to search one chat, they must also give up access to all of their chats on GroupMe. If there is any sensitive information in any of their other chats, such information is surrendered to the program, even if the user only intended to give up information for the chat being searched. Additionally, SearchMe would gain the ability to impersonate the user in any of their chats. There is no way of only giving read access while withholding write access; when the access token is given up, read and write access to all of the

user's groups is given up [3]. Even if the developer behind SearchMe can be trusted with one's access token, there is immense danger to the user in event that an attacker compromises the SearchMe database or intercepts the access token being sent to the server. If one of these two scenarios happen, or any other similar attack occurs, the user's access token would fall into the hands of a malicious party. This malicious party would then have read and write access to all of the user's group chats on the entire GroupMe app. For these reasons, users are advised against using SearchMe for the sake of their own privacy and authenticity.

The GroupMe Search Bot completely solves the security issues of SearchMe. At no point in the search process does the user ever give up their access token. Instead, the user simply adds the Search Bot account to their chat. The Search Bot will then have access only to the messages in the chat to which it has been added. Additionally, the Search Bot will only be able to write to its own account, because it will not have write permissions for the user account. The likely reason as to why this method has not been tried by existing applications is because it requires a second GroupMe account for the Search Bot, and GroupMe accounts each require a unique valid phone number. Thus, this method was more work to implement but vastly more secure.

Methodology

The program operates in three steps that continually repeat in a loop. First, the program scans for new messages in its groups and partitions these messages into regular messages and search queries. Secondly, the program inserts the regular messages into the instance of Elasticsearch (the alternative to Solr selected for the project). Lastly, the program responds to search queries by retrieving relevant messages from Elasticsearch and displaying them in the chat.

The program scans for new messages by first retrieving all of the groups in which the Search Bot is a member. If there are any new groups, the program retrieves all of the messages in the new group chat. For the rest of the groups, the program retrieves all of the messages which were posted after the most recent message indexed by the program.

The messages retrieved by the program are then partitioned into regular messages and search queries. The criteria for a search query is that it directly mentions the account name of the GroupMe Search Bot. The rest of the messages are indexed into Elasticsearch, with key information including the message group, ID, sender, text, and timestamp being preserved.

The searches are then responded to by extracting the space-separated keywords from the search text and querying the Elasticsearch instance for messages that match these keywords. Note that only messages from the group being searched are considered, even though the Elasticsearch instance stores messages from multiple groups at once. The program then sorts all matches by timestamp, with the most recent messages being displayed first. A limit of ten messages are displayed such that the bot does not spam the chat with unnecessary information. Lastly, the program constructs a message which returns each match on a separate line. If the search result takes up more than the 1000-character message limit, the response is broken apart into multiple messages.

Results

The program works successfully, in so far as that the Search Bot responds to searches quickly, reliably, and accurately. The results are quick in that the Search Bot responds to user searches within one second. Faster responses would be unfeasible because there are limitations on how frequently the GroupMe API can be queried. Responding faster would require making

API calls more frequently, which will result in the account being timed out. The results are reliable, in that the Search Bot responds to all messages in testing. To prevent the program from crashing, API calls are made within try-except statements, such that if the GroupMe API fails to return a response, the program continues running. If the program terminates, it will be able to respond to messages it missed upon being rebooted. Lastly, the program is run on a virtual machine in Azure rather than on one of our own computers, such that it is very unlikely its server will go down given the high reliability of Azure. Lastly, the results are accurate in that all of the messages returned match at least one keyword, and no documents are missed. However, there is a subjective tradeoff between keyword relevance and recency. If the search returns messages with the highest keyword relevance, through a ranking such as tf-idf, the results may likely be too old to still be relevant to the user. Thus, documents are returned in order of recency rather than pure keyword relevance. This could be an issue if a high percentage of documents were matches, but based on user testing, this is not the case. Since only a small percentage of documents are matches, it makes more sense to return matches in order of recency instead of relevance, in order to provide useful results. Based on user testing in a few different group chats, the results returned by the Search Bot are mostly relevant, and all relevant results are returned if they are among the most recent results. The GroupMe Search Bot is expected to provide a robust implementation of an important functionality missing in a popular messaging app.

References

1. Sweeney, Patrick. “About SearchMe.” *SearchMe,* Oct. 2017, www.searchme.co/.

2. “Authentication Tutorial.” *GroupMe Developers*, https://dev.groupme.com/tutorials/oauth.

3. “Public API.” *GroupMe Developers*, https://dev.groupme.com/docs/v3.