Customized Notification-Based Chat Application – Muffle

Pavan Raj Dasari
Department of Information Science and Technology
Missouri University of Science and Technology
Rolla, MO, United States
mdm77@mst.edu

Ethan Moore
Department of Computer Science
Missouri University of Science and Technology
Rolla, MO, United States
eemvzm@mst.edu

Sai Ganesh Pamarti
Department of Computer Science
Missouri University of Science and Technology
Rolla, MO, United States
spbd3@mst.edu

***Abstract*— It's common for someone to wish to receive messages for certain things but not for others. It's okay if your wife is pregnant, and you don't mind if the message that she's going into labor interrupts your meeting. There may be times when you're on call for work, but don't want anything to interrupt your free time. In some cases, a binary on/off notification system is insufficient to meet a user's needs. Considering this, a messaging system with customizable notifications might prove to be beneficial. This would be a form of do not disturb mode, where the user could choose what notifications, they are interested in receiving and what notifications they do not. To do this, they will be able to create a series of whitelists and blacklists for the sources of notifications from which they will receive notifications. At any given time, they can choose which will be active. Additionally, the system can automatically reply to blocked messages with a piece of text the user adds, or the system will generate a message based on the client configuration. As a result, the user can let others know that they are busy and when they are likely to reply to the message. To create a chat application, especially for users who do not want to get interrupted during work by others or when they want to relax or spend time with their family. And to send automatic and customized messages to them whenever they turn on the DND mode. And the user can set who can override the DND mode and send messages to them. And set the DND mode during a certain time.**

***Keywords*— customizable notification, do not disturb mode, whitelists and blacklists, automatic reply, override DND**

I. INTRODUCTION

Information and communication technology development is currently accelerating. The general population may now communicate and obtain information at any time, from anywhere [2]. When referring to computer chat, we mean electronic conversation with other people [3]. But we do not want to get disturbed during a certain period of time, let’s say during some important work or when you are having your time. This is the main motive to create this chat application; to get some time for yourself and spend some free time. In addition to that, we have created an automatic reply message that has to be set by the user when they’re in Do Not Disturb (DND) mode.

Since we wanted to make this chat application available to as many users as possible, we have the app created in Windows, Android, and iOS versions. The technologies used to create this chat application are integrated so that the app can be installed and run on multiple platforms. This was accomplished by using Flutter [11], a cross-platform development framework that can write code once which can then be deployed on various platforms (Android, iOS, and Desktop) [1][4][7]. [9]Firebase is an app development platform that allows you to create and expand popular apps and games. It has systems for distributing messages, pushing notifications [5][8], and logging in to an app through Google Sign-In. Going through Firebase makes this based on the Client – Server Architecture. The user must create an account with their Google mail credentials to use the app.

II. PREVIOUS WORKS

For this chat application, we initially tried to use Tox [10], which is a peer-to-peer instant messaging protocol that emerged in response to the leaks by Edward Snowden's leaks regarding NSA surveillance. It has end-to-end encryption and is designed to allow multiple different client programs to communicate cleanly. We had hoped to be able to use its existing user base to test the functionality of our client. Unfortunately, all of the libraries for Tox in languages our team is familiar with were deprecated out of date, or not yet in a state of sufficient development for our purposes.

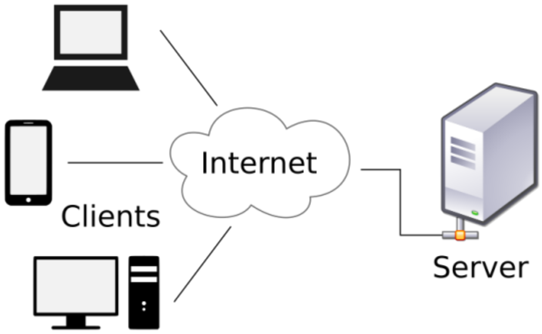
Many chat applications available today have their specifications; yes, this DND mode may also have been done by others earlier. But in this chat application where we had a few users during the testing phase, we achieved good results, and in addition to the message notification feature and the customized and automatic 'Not Available' message feature another unique feature that we have added in this chat application is to override the contacts who can reach out to the user even when they're in DND mode, and the ability to set a specific time for the DND mode so that nobody can contact you at that time, not even the override contacts.

III. SYSTEM MODEL

In this part, we have described about the client-server model that our application works on, and the message flowchart in the application.

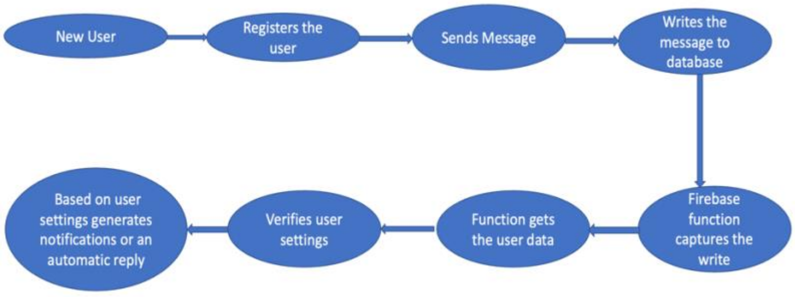
A. Client Server Architecture

We have used the Client-Server model for this chat application. The client-server paradigm is a distributed application structure that divides tasks or workloads between resource or service providers, known as servers, and service requesters, known as clients. In the client-server architecture, when the client computer sends a request for data to the server through the internet, the server accepts the requested procedure and delivers the data packets requested back to the client.



1. By using this model, we can enhance data security, scalability, and accessibility.
2. Due to the high processing power of workstations and information sharing capabilities offered by LANs, LAN servers, database servers, and other application servers are becoming very useful and popular in 'downsizing' and 'upsizing' applications of mainframe onto PC servers and LANs using co-operative processing environment and platforms. This client-server architecture is well supported by new Network Operating Systems (NOS). Database architecture, Query languages, and Graphical User Interface (GUI) tools [18].
3. In our application, the Firebase will act as the server, and the clients would be the devices that have the application (mobiles, laptops).
4. Firebase is widely supported and has native integration of push notifications and existing secure login infrastructure through the widely used Google Sign-In

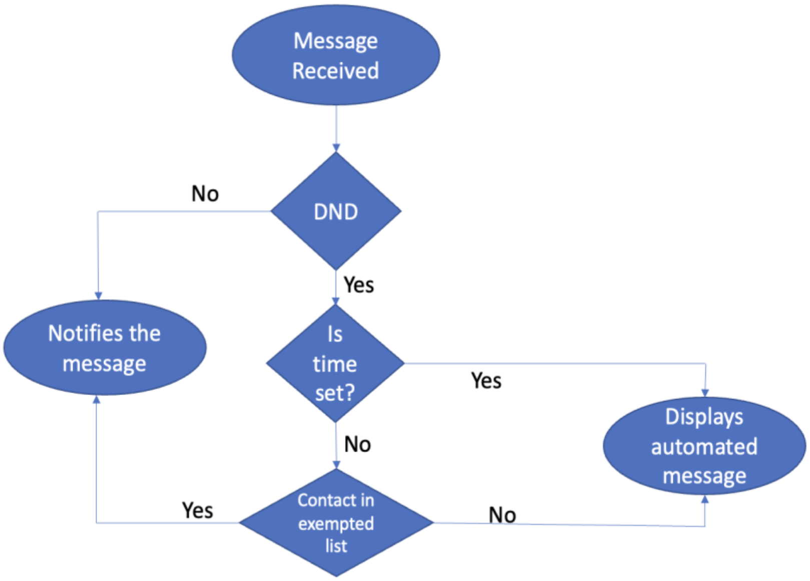
B. Application Working model


New User
Registers the user
Sends Message
Writes the message to database
Firebase function captures the write
Function gets the user data
Verifies user settings
Based on user settings generates notifications or an automatic reply

The detailed explanation for the flowchart is as follows,

1. When new user opens the app they need to register first. A unique hash is generated once the user is registered, and information is written into the firebase database with the default notification settings.
2. Then they can send messages to all of the registered users. And these sent message is written into the database.
3. The Firebase function monitors the write to the database and it starts execution once the user message has been written to the database.
4. The function gets the user data and his custom notification configuration, performs additional checks, and tries to either send an automated reply back to the sender or a notification to the receiver.

C. *Application Usage*



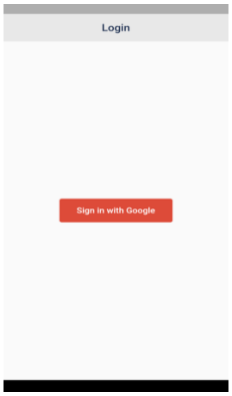
The detailed explanation for the flowchart is as follows,

1. Once the message is sent the application looks after the receiver configuration settings.
2. If the user has not activated the DND, they'll get a message notification.
3. If the receiver has DND on, and the sender is not on the override list, they will get the automated reply set by the receiver. The message will be received, but no notification will be sent.
4. Otherwise, if the sender is on the override list, they will get the message notifications.
5. If the receiver has DND on, and if the receiver sets the time period for the DND mode, no one can message them, even if they're on the override list.

Every user can be able to log in using Gmail only so that we can get to know if the user is real or not. Therefore, every user will get to know about the other user they're texting with.

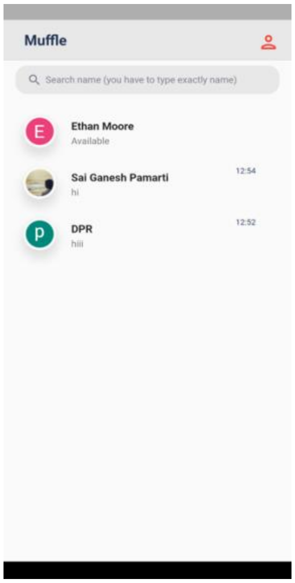
IV. DEMONSTRATION

1. When we first open the application, it will ask us to sign in. Then we have to sign in using Google mail.



Login

2. After successfully signing in, we can see the users that are using the app. We can start a conversation by just clicking on their tab. And also, we can search for any user who has installed the app to see the chat with them.



Muffle

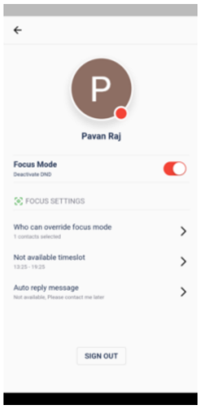
Ethan Moore
Available

Sai Ganesh Pamarti 12:54
hi

DPR 12:52
hiii

3. If we click on the top right of the home screen, we'll go to the account page where we can turn on/off the DND mode, set the 'Auto reply message', and also select the override contacts.

4. We can also set the time on the account page, where we just have to set the from time, and, to time. During this time, no one can be able to contact the user.



Pavan Raj

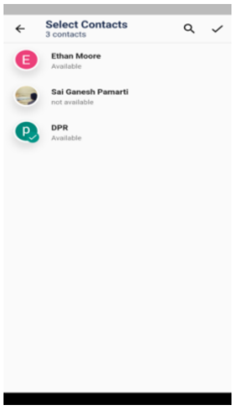
Focus Mode
Deactivate DND

FOCUS SETTINGS

Who can override focus mode
1 contacts selected

Not available timeslot
13:25 - 19:25

Auto reply message
Not available, Please contact me later

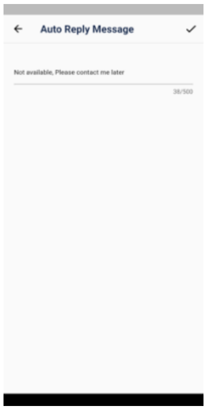


Select Contacts
3 contacts

Ethan Moore
Available

Sai Ganesh Pamarti
not available

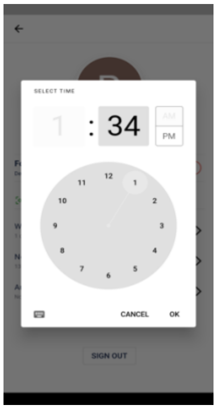
DPR
Available



Auto Reply Message

Not available, Please contact me later

38/500



SELECT TIME

1 : 34 AM PM

V. CONCLUSION

Many online messaging programs have emerged as a result of the fast-paced development of information technology, and some of these messaging applications are highly well-liked by mobile device users. Globally, chat apps are gaining popularity as a form of communication. But before creating, it's important to comprehend how consumers will be disturbed by each notice. The potential of creating a chat application based on personalized notification settings is therefore discussed in this study. So, we have created a chat application that sends notifications, sets the DND mode including a customized time duration setting, and overrides the contacts who can send messages anyway. Regardless of how many chat applications will come in the future, we can develop this app to send multimedia files, and we can get responses from the users into making this application more of a personal assistant.

As a future development of our research, several goals exist. Detecting user pulse using daily surveys regarding their interest to be contacted by someone and their availability time based on scheduled usage of the application we try to integrate a machine learning algorithm to build a model that would predict the user response and automatic information gathering by the system from user utterances will be used by the system in future talks and try to respond to the received messages [16]. So as a result, our model will work as a personal assistant to the user which can talk on behalf of the user [17].

VI. REFERENCES

[1] Dewi,M.M.(2012). Mobile Application Development Feasibility Studies: A Case Study in Universiti Teknologi MARA.

[2] Raymond B.J, et al(2006).A study of Instant Messaging and chat protocols.

[3] Reynolds.P.(2015). Benefits of the online chat system

[4] Harsha Sinha, Payal Mishra Analysis of Cross-Platform Mobile App Development Tools 2019 IEEE 5th International Conference for Convergence in Technology (I2CT)

[5] Stohy, R. A., Ghetany, N. e., & Gharib, H. A. (2016). A Proposed System for Push Messaging on Android. International Journal of Interactive Mobile Technologies, 10(3), 29-34. https://doi.org/10.3991/ijim.v10i3.5567

[6] Yilmaz, Y. S., Aydin, B. I., & Demirbas, M. (2014). Google Cloud Messaging (GCM): An Evaluation. 2014 IEEE Global Comm. Conf. 2847-2852. Austin, TX, USA: IEEE. https://doi.org/10.1109/GLOCOM.2014.7037233

[7] Martinez, M., & Lecomte, S. (2017). Towards the Quality Improvement of Cross-Platform Mobile Applications. IEEE/ACM 4th International Conference on Mobile Software Engineering and Systems (MOBILESoft) (pp. 184-188). Buenos Aires, Argentina: IEEE. https://doi.org/10.1109/MOBILESoft.2017.30

[8] Ding, J., Song, W., & Zhang, D. (2014). An Approach for Modeling and Analyzing Mobile Push Notification Services. 2014 IEEE International Conference on Services Computing (pp. 725-732). Anchorage, AK, USA: IEEE. https://doi.org/10.1109/SCC.2014.99

[9] Firebase – Firebase firebase.google.com

[10] Tox about page - https://tox.chat/about.html

[11] Flutter - Flutter - Build apps for any screen

[12] Design and implementation of web-based real-time chat interfacing server | IEEE Conference Publication | IEEE Xplore

[13] Entao WANG, Xiang LI.,The communication research of mobile phones and database server based on Socket. Technology and Development of Computer, vo17. 2,pp.82-84,2007

[14] Tilson, D., Sorensen, C., & Lyytinen, K. (2012). Change and Control Paradoxes in Mobile Infrastructure Innovation: The Android and iOS Mobile Operating Systems Cases. 45th Hawaii International Conference on System Sciences. 1324-1333. Maui, HI, USA: IEEE. https://doi.org/10.1109/HICSS.2012.149

[15] Brito, H., Gomes, A., & Bernardino, Á. S. (2019). JavaScript in mobile applications: React native vs ionic vs NativeScript vs native development. 14th Iberian Conference on Information Systems and Technologies (CISTI). Coimbra, Portugal, Portugal: IEEE. https://doi.org/10.23919/CISTI.2018.8399283

[16] J. Weizenbaum, ”ELIZA—a computer program for the study of natural language communication between man and machine,” Communications of the ACM, vol. 9, pp. 36-45, 1966.

[17] J. Jia, ”CSIEC: A computer assisted English learning chatbot based on textual knowledge and reasoning,” Knowledge-Based Systems, vol. 22, pp. 249-255, 2009

[18] G. S. Hura, "Client-server computing architecture: an efficient paradigm for project management," Proceedings for Operating Research and the Management Sciences, 1995, pp. 146-152, doi: 10.1109/IEMC.1995.523924.

VII. CONTRIBUTION BREAKDOWN

Member name	Percent contribution	Activities completed by the member
Pavan Raj Dasari	34%	End-to-end coding
Ethan Moore	33%	Research and proof-of-concept prototypes
Sai Ganesh Pamarti	33%	Testing, prototyping, and coding