Purdue University

Sprint 3 Retrospective
MonsterChat Application

Team #14:
Rayten Arnold Rex, Cole Baughn, Vishal Gill, Zeyu Pan, Andrew Sytsma, Michael Young

1. What Went Well?

In Sprint 3, we were able to most of our user stories mentioned in the Sprint 3 planning document successfully. In the background of Android, creating and joining private chat rooms went well. Also, in the background of Android the spam filter went well; however, properly resetting the user's messages to not be spam on a time interval was a little awkward. In the background of Android, writing and reading blocked users went well; however, actually blocking the messages had a few hiccups but was later resolved. Encryption and decryption, in the background of Android, took quite a bit of work in order for it to operate properly; there was not a lot of good resources on making encryption and decryption work for Android, but now, the encryption and decryption work as expected. In the background of Android, making pictures messages went well as did resizing them to fit the dimensions that we wanted. Deleting logs, in the background of Android, went smoothly. In the GUI for Android, creating and displaying private chat rooms went well. In the GUI for Android, blocking and unblocking users went well. In the GUI for Android, the creation of an about page to display contact information went well. In the GUI for Android, giving the user the ability to insert images went well. In the GUI for Android, adding buttons to delete logs went well. For the iOS side, we were able to do very well with creating and joining chat rooms and displaying available chat rooms. Unlike the first two sprints, we switched over to using a better paradigm in the MCF that allowed us to implement functionality that worked well with a chat room type of application and allowed us to hide all available peers unless they were advertising a chat room. Blocking users also went very well, we were able to block and unblock users both while in the chat and inside the block list in the settings page. If the user blocked any incoming messages, they would not be displayed unless they were unblocked. Encryption and decryption went well, every time the data was sent it was encoded and then decoded once it reached peers. We were able to make all the GUIs including the about page for users to send us feedback. Even though we were not able to receive images when the user tried to send an image it did not clog up the chat and kept messages well formatted. For iOS, having low battery strain went well.

We implemented the following tasks successfully:

- **As a user, I want the ability to create and/or join a private chat room because I want to chat with a specific group of people without others being able to join.**
    - We were able to successfully allow users to create private chat rooms by clicking on the create chat room button and entering a password. Join the chat rooms also worked successfully by allowing the users to click on a private chat room from the list and join it by entering the correct password.
- **As a user, I want to be able to view private chat rooms because I want to know what private chat rooms are available to me.**
    - We were able to successfully view private chat rooms in our list of available chat rooms. The private chat rooms are all denoted by a picture of a lock next to their names.
- **As a user, I want some kind of filter to minimize spam because I do not want to have my chat experience ruined by other users (Android).**
    - Filtering spam message worked successfully by not allowing a user to enter the same message three or more times in a row in less than three seconds. Also, as a

second layer of protection, messages are not allowed to be sent back to back if they are within two hundred fifty milliseconds of each other.

- **As a user, I want the ability to block/unblock the messages of other users because I do not want my chat experience ruined by other users.**
    - Blocking and unblocking worked successfully by allowing users to click on another user's message in chat and choosing to block them. They could subsequently be unblocked if the user went to settings and removed the blocked user from the block list.
- **As a developer, I want to be able to receive feedback from users because I can use the feedback to better the app.**
    - An about page was added to allow users to email us with feedback they have about the app. Included in the page is our names and a shared email that the users can send to.
- **As a developer, I want a secure method to transfer data because I do not want my users' personal information to be accessed by a third party.**
    - Data transferred was secure successfully through the process of encrypting and decrypting the messages. Encryption worked by using AES encryption, which is an encryption that the NSA views as strong enough to use for secret documents. This helps to make sure that no one besides the intended receiver of the message can view the data that is being sent.
- **As a user, I want to be able to send a picture because I want allow other people to see them (Android).**
    - Sending and receiving picture messages worked successfully. The app allows users to add one picture to their message, which can be chosen from somewhere on their device; they can then send the message with the picture, and it will display the picture properly in the chat room.
- **As a user, I don't want pictures to clog up my chat because I do not want my chat experience to be cluttered by large images.**
    - We avoided clogging up the chat room by resizing images to be a certain dimension. This would guarantee that large pictures would not take up absurds amount of space so that the chat experience is not ruined by them.
    - Even though we could not send and receive images in iOS, when the user tried to send the image it did not clutter up the chat in their respective chat views.
- **As a user, I want to be able to delete my chat logs (Android).**
    - Deletion of chat logs was implemented successfully by allowing users to hold their finger down on a chat log in the list, which would in turn bring up a message asking them if they would like to delete it. Also, if a user was already in a chat log, they could delete the chat log by clicking the button at the top of the screen. Furthermore, users are allowed to delete all their chat logs at once, if so desired, by clicking the button at the top of the chat logs list page.
- **As a developer, I want to delete old chat logs to minimize space usage on the phone (Android).**
    - Deletion of old chat logs was handled by checking the amount of space that was taken up by the logs when writing new ones. It would check to see if the space taken up was greater than the specified amount, and if it was, would delete chat

logs starting from the oldest chat log until the space taken up by the chat logs is under the specified threshold.

- **As a user, I want to be able to create and join chat rooms because I want to be able to chat with other users (iOS).**
    - This part went very well because we switched over to using different classes in the MCF and used a different paradigm that came along with those classes. That way we were able to advertise chat rooms with a specific name and when clicked on that chat room we were able to join it with the owner's approval.
- **As a user, I want to be able to view chat rooms because I want to know what chat rooms are available to me (iOS).**
    - This part also went very well because we switched over to using different classes in the MCF and used a different paradigm that came along with those classes. That way we were able to send and receive a dictionary object that contained information about the chat rooms advertised and display it accordingly.
- **As a user, I want to be able to use this app without excessive battery strain because I want the app to run a long time without my battery running out (iOS).**
    - When the user exited out of the app the MCF services would stop trying to advertise and browse chat rooms. This saved battery life because the peer to peer wifi only ran when the user was actively using the application and not in the background.

2. What Did Not Go Well?

Although we were able to implement most of our user stories, still there were some tasks that were unsuccessful. In the background of Android, using little RAM did not go well because something was wrong with the way old chat log messages were retrieved from the file. For networking, switching over to Bluetooth was not done in the end due to the disadvantages of Bluetooth outweighing the disadvantages of WiFi Direct. We also could not implement battery saving measures with the network adapters. iOS logging did not go well because the code was left untested until the very end. The code for the application was not updated on GitHub every time something was implemented, so code that could have been tested by running the application under tests was not testable. Also, spam prevention was left out because, when the design of the application was changed significantly, the responsibility of implementing the spam filter fell into an unclear area, so it was forgotten. In the GUI for Android, limiting RAM usage did not go well. Limiting RAM usage worked in chat rooms, but in logs it caused an error when trying to scroll up through the log. In iOS, limiting RAM usage did not go well because we did not put a cap on how many messages we wanted inside the array that was holding all the messages and also since we were not able to do logging we could not empty the array after a certain number of messages.

- **As a user, I want to be able to send and receive messages with other Android devices because I want to communicate with others.**
    - Going into switching from WiFi Direct to Bluetooth, we knew that we would lose certain things, such as distance of the WiFi radio. However, there are other unforeseen disadvantages. We were unfortunately still stuck with the same issue that plagued us with WiFi Direct, where one phone must act as a server, which means we could not achieve a true mesh network still. We also lose the speed of the WiFi radio, which would have made sending images a bad idea. Finally, Bluetooth can apparently only support seven devices being connected at a time, versus WiFi Direct having 15 connected at a time. While we couldn't actually verify this due to not having that many devices, it was something to take into consideration. It was determined that, while WiFi Direct has it's problems, Bluetooth would be worse overall, and we remained with WiFi Direct.
- **As a user, I want to be able to use this app without excessive battery strain because I want the app to run a long time without my battery running out (Android).**
    - The original idea was to turn off WiFi Direct when the application closed. Because we could not achieve a true mesh network, this was actually impossible to implement seamlessly. The original idea was that if a single device dropped out, the chat would continue as normal. However, by having one device as a server, having a single device drop out will either do nothing or destroy the entire network.
- **As a user, I want the application to run with as little RAM usage as possible.**

    - On the Android side, we were able to limit the number of messages printed to screen to effectively limit RAM usage. However, this caused an error when trying to scroll up while viewing a Log.
    - On the iOS side, limiting RAM usage did not go well because we did not put a cap on how many messages we wanted inside the array that was holding all the messages and also since we were not able to do logging we could not empty the array after a certain number of messages.
- **As a user, I want to be able to view my past conversation(s) because I would like to be able to access information that I may have forgotten (iOS).**
    - Code was developed for logging, but it did not necessarily work. Logging was untested until the end of the sprint due to the fact that the code that could be used for testing was unavailable until the end of the sprint.
- **As a user, I want to be able to delete my chat logs (iOS).**
    - Code was developed to handle deleting chat logs, but it appears that logging itself does not work. Code that could be used for testing was not available until the very end of the sprint due to lack of commits.
- **As a developer, I want to delete old chat logs to minimize space usage on the phone (iOS).**
    - Logging was not well developed, so we did not get to this point in the sprint. It would not take much to fix this when deleting logs and logging starts to work.
- **As a user, I want some kind of filter to minimize spam because I do not want to have my chat experience ruined by other users (iOS).**
    - With the way the application was originally designed, the responsibility for implementing preventing spam would have fallen to the background. Since a lot of the application changed to the point it appeared unrecoverable, many of the background tasks were changed, moved, or removed to adapt to the new design. The spam filter, however, was forgotten. Spam filtering should have been moved to the same place as sending messages, as that is where it's implemented in the Android application, but it did not move with the sending of messages, and there was no way to implement it in the background tasks anymore.

3. What Should Be Improved?

In the future, there are ways we can work to improve ourselves as a group and as individuals. We still need to utilize GitHub better and more often. As soon as a feature is implemented, it should be available on GitHub so others can use it. Similarly, the trello board needs to be utilized more, as well. Our Application resizes images to a fixed height and width that we define which can cause some picture to look weird when sent. Our Application currently uses the default Icons which is not unique.

- To fix the issue of not using GitHub well enough and often enough, we can play around with it more to get a better feel for the features that it has to offer, which will also boost expertise. Also, it is important that we work to commit early and often.
- To fix the issue of not using the trello board enough, we can make sure to be more active by putting up the information about what we are working on and updating it upon completion.
- To fix the issue of some images looking weird, when they are displayed, we could resize images based on their original dimensions, instead of just always resizing them to a 200x200 square.
- To fix the issue of using the default icon, we could create our own icon and use it as our Applications icon.