ENGG*4450 - Large Scale Software Architecture

ASSIGNMENT 2 - IMPLEMENTING CHANGE REQUESTS

Instructor:
Dr. Petros Spachos

Teaching Assistant:
Marc Baucas

Group: 3

Pieter Jurgens Krige - ID: 1012072

Andrei Korcsak - ID: 1008518

Disha Singh Nath - ID: 0995702

Harshal Patel - ID: 1032961

Bilal Ayash - ID: 0988616

Neel Bhandari - ID: 1004853

Submission Date: Tuesday, October 27th, 2020

School of Engineering

University of Guelph

Contents

List of Figures

List of Tables

1 Introduction

A Unified Modeling Language (UML) use case diagram is an interactive way to provide a higher level view of the underdeveloped process without requiring any prior experience in programming language. It is a logical flow where only by looking at the diagram one could identify and understand the whole concept. Case diagrams generally includes systems, actors, use cases and relationships. All the intricate ideas could be communicated through Use Case diagrams.

Assignment 2 requires each group to select multiple issues and further analyze, and implement at least two bug fixes for Signal Private Messenger along with their test cases. It stresses on scrutinizing which software program to use and which two change requests to choose for implementation. The group is obsessively focused on creating fixes which are reliable and robust.

2 Development Process

This assignment had obligatory seven steps to go through in completing the assignment productively. They are given as follow:

1. The initial step for each group member was to select an issue or small feature request for Signal Messenger.

2. The next step demanded to draw a use case diagram illustrating all the use cases for each issue on the shortlist where all the group members preferred Astah® UML tool to go with as it offers the feature to make Use Case diagrams and everyone was acquainted with this tool from lectures, labs and assignment 1.

3. The third step was to identify appropriate software process model which would guide us through the rest of this assignment. The group decided to adapt SCRUM framework. SCRUM has many advantages such as high productivity, great flexibility, adaptability, increased team accountability and high transparency among other benefits. The group initially started with Sprint - 15 day iteration compared to the traditional 30 day sprint, due to which the process was scaled down accordingly. This started with allocating 1/4 day to group meetings to maintain a balance between the other courses. The next thing was to decide on Prioritizing Product Backlog which is shown in Table 1. In addition to this, the group also followed the Daily Scrum process which included 15 minutes team meeting each day. Each team member would answer; update the team on what have they done since the last meeting; What they will be doing between now and the next meeting; What obstacles they came across in approaching the task. The visualization of SCRUM process can be seen in Figure 8 in appendix [6].

   The group implemented the divide and conquer approach where group members were divided into 2 groups of 3 depending on their skill set. Each members could seek help from their partners in finishing tasks. The aim was to make it much simpler to tackle 2 selected issues efficiently and commendably.

4. The subsequent step was to select at least two items from the shortlist to implement and test. In order to select two items effectively and in a bias free manner, the group decided on forming a decision matrix for all seven items as shown in Table 2.
   Each team member scored their selected issue. The decision matrix was devised in such a way that it would eliminate any predisposition. The group was divided into two sprints of 3 each to competently settle on an issue. The given scores were analyzed by other two members in the sprint and vice-versa. The scores were not concluded until all the members were satisfied with the given scores and the reasoning behind that. The scores are in range of 1-9 where higher value indicates more significance with the exception of Anticipated Risk factor where lower score indicates more risk and higher score corresponds to low risk in implementing the bug. The group is novice in open-source project, so we preferred bug 10044 that required higher effort in implementation of the solution and 10089 which required moderate. Likewise, understanding of the issue is also important which is further described in topic 3. Importance of fixing the issue was also considered one of the major factor as this would help user experience with the app.

5. The most important step was to implement the selected items. This step required knowledge in java as well as how to work on open source project. It entailed most of the available time with a great team effort.

6. Further, test cases, also called ”Customer Acceptance Test”, were to be written to demonstrate that the changes were implemented correctly . It is a form of a step by step user guide which customer can follow to check the accuracy of the solution.

7. The final step was to describe the steps taken to select and implement the items. How well the development process has worked for the group is given in the conclusion.

Table 1: Product Backlog Items with the estimated time using SCRUM model

Backlog Item	Start Date	End Date	Duration
Selecting issues/small feature requests	Oct 13	Oct 14	2
Use case diagrams for the selected item	Oct 15	Oct 16	2
Selecting software process model SCRUM in this case	Oct 17	Oct 18	2
Selecting 2 items from the shortlist to implement and test	Oct 19	Oct 19	1
Implementing selected items	Oct 20	Oct 23	4
Test cases to demonstrate the changes made correctly i.e. Customer Acceptance Test	Oct 24	Oct 25	2
Report writing to describes the steps followed to select and implement the selected item	Oct 25	Oct 27	3

Table 2: Decision Matrix for Shortlisted Issues

Bug/issue/ feature ID	Effort in Implementation of the solution	Understanding of the issue	Importance of fixing the bug)	Anticipated Risks	Total
09088	7	5	8	2	22
10111	7	6	6	3	22
10109	8	4	6	5	23
10044	8	8	7	4	27
09826	6	6	5	3	20
10089	5	8	6	6	25
10085	6	8	5	4	23

3 Use Cases

3.1 GitHub Issue #09088 - Message Appears in Wrong Order

Use Case diagram in Figure 1 illustrates a potential bug encountered by Android users where the recipient receives the message in the wrong order. In the diagram, the two individuals that are communicating with each other have to first login to the app. The 'Android Sender' will be accessing the contacts to send a message, while the 'Android Recipient' will be accessing it for opening and reading the message. The chat history is displayed to both users whenever attempting to open the messaging chat. The issue part is observed on the receiver's side when the messages received are sometimes not ordered properly based on the timestamps of the messages. According to the bug description, a weak WiFi connection or when attempting to reconnect to WiFi from the airplane mode could be a cause of this issue.

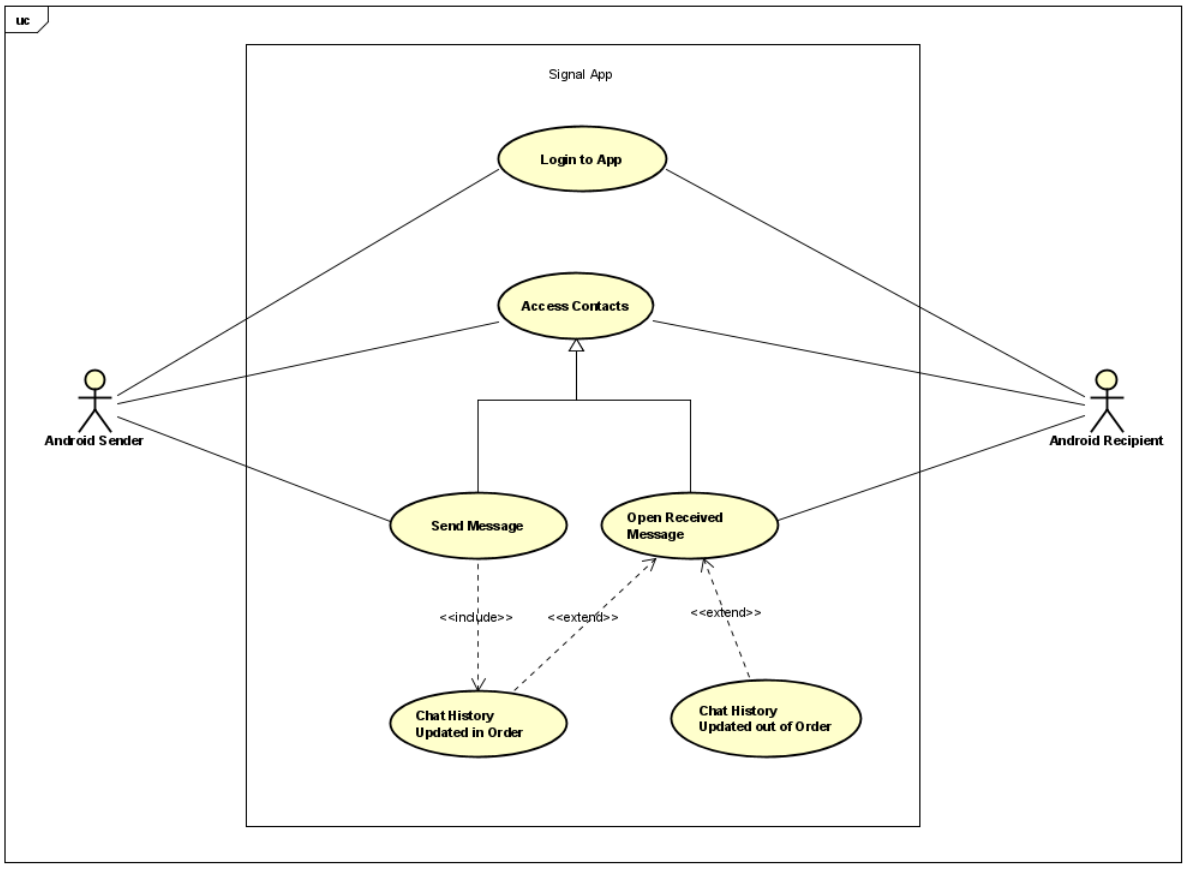

Figure 1: Use Case for Issue #09088

3.2 GitHub Issue #10111 - Failing to Send SMS Longer Than 160 Characters

Figure 2 shows use case diagram for issue 10111. When sending a SMS message of longer than 160 characters from an android device to a non-registered android user, the app will fail to send the message and resend option will appear whenever this issue occurs. In the error message it displays, ” Some issues need your attention.” along with the resend option. The expected result is the message must be fragmented into 160 or fewer characters and be sent to the recipient.

Upon systematic investigation of the issue from the origin, a conclusion could be made that the ’OutgoingTextMessage.java’ file in SMS folder has no issue related to outgoing text message to either registered or non-registered user as the code does not explicitly restrict messages to 160 characters. This suggests that the potential problem could be on the server side more predominantly on the outgoing servers.

The prospective solution to this problem would be to debug the outgoing servers. Another interesting thing to note is in error log, the debug log shows the message being split but there is something wrong with the system implementation of sendMultipartTextMessage method which is in SmsSendJob.java file under jobs folder [5].

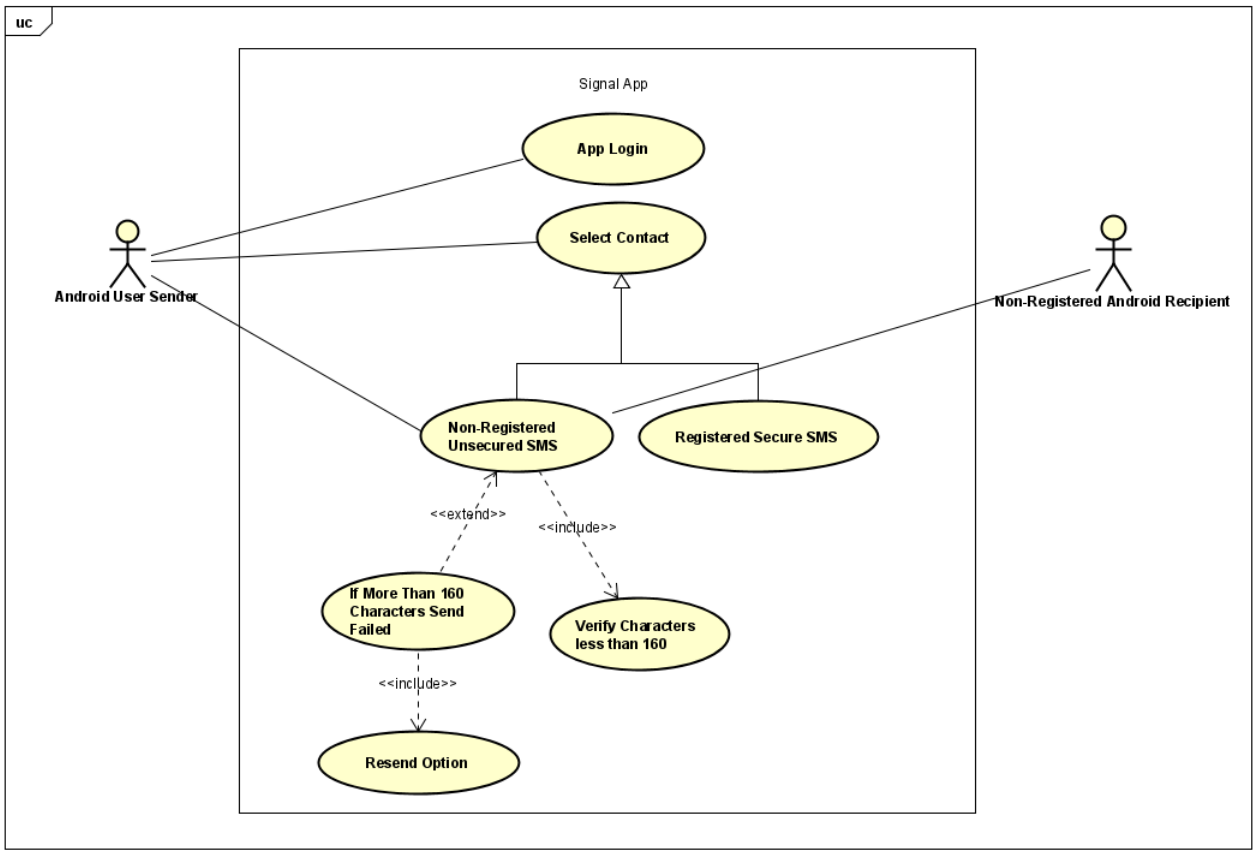

Figure 2: Use Case for Issue #10111

3.3 GitHub Issue #10109 - Cannot Remove a Group Already Left

Figure 3 shows the use case diagram for issue 10109. This Bug deals with the issue that occurs when adding other users to your existing group. Currently if the user creates a group, leaves that group, deletes it from their conversations and then tries to add another user to another group, they run into the issue where they can still see the previous group that they already left and deleted. The user should not be able to still see the group when they try to add other users to other groups, rather they should only see the groups that they are still apart of.

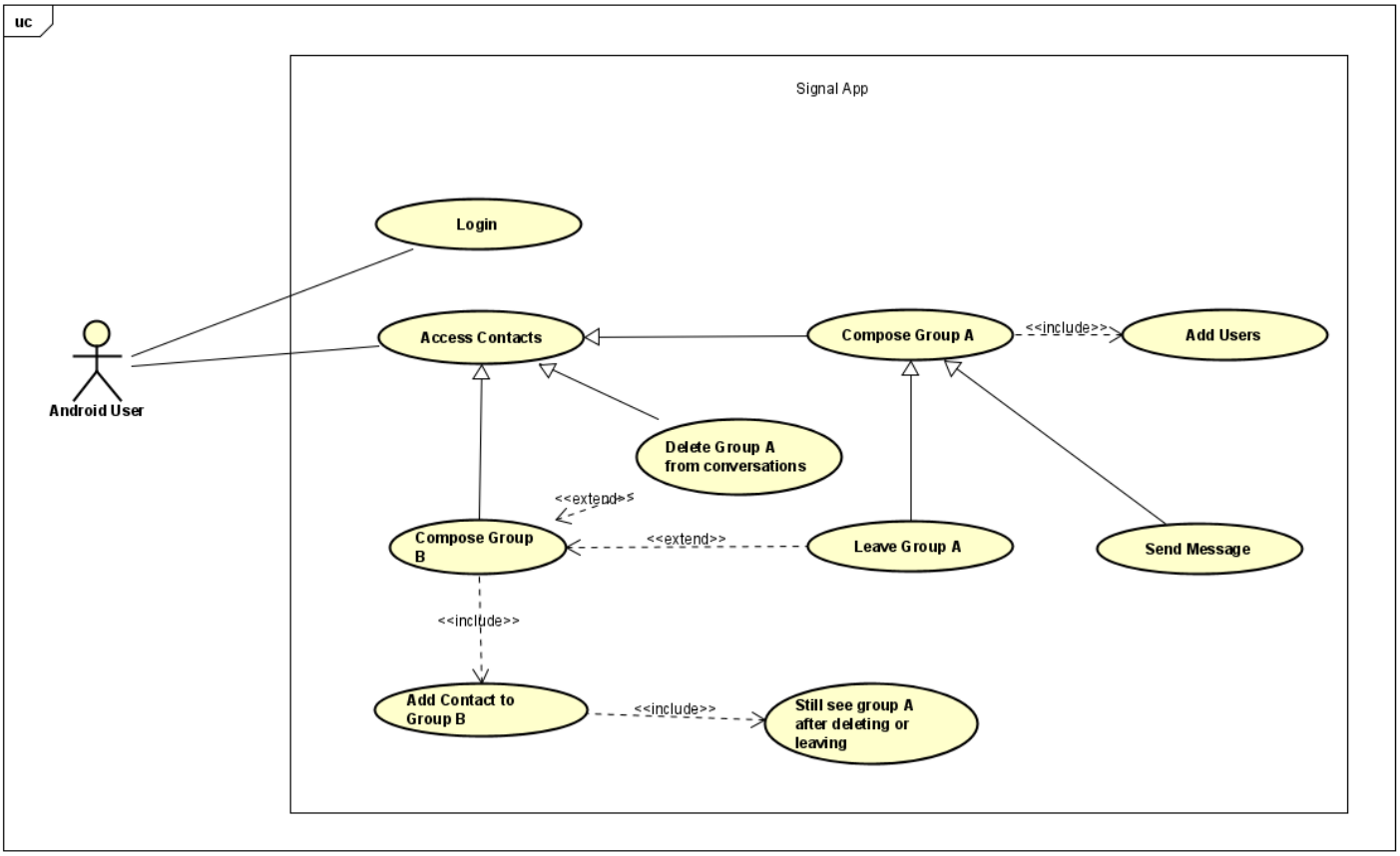

Figure 3: Use Case for Issue #10109

3.4 GitHub Issue #10044 - Message Text Lost After Removing Image

Figure 4 depicts the use case diagram for bug 10044 of the Signal App for Android. This bug concerns the ability to add and remove pictures from a message containing text. In the current implementation if text has already been added into a message, an image can also be added, almost as if inline with the text. Then, once added, the user is able to add text which should be added to the previous text. The problem presents itself when the user then removes the 'inline' image. Instead of keeping the text which followed the image, all of that text is also lost with the removal. This means, if an incorrect image is added to the message and text has been added after it, the user has no choice but to recompose the message instead of just re-adding another image.

To replicate this bug, when the messaging app was made to run, it accessed files MediaSendActivity.java and ConversationActivity.java. It is believed the bug is embedded within these java files, however, given the team size, time constraint and complexity of the bug, the group decided to explore other options for implementation.

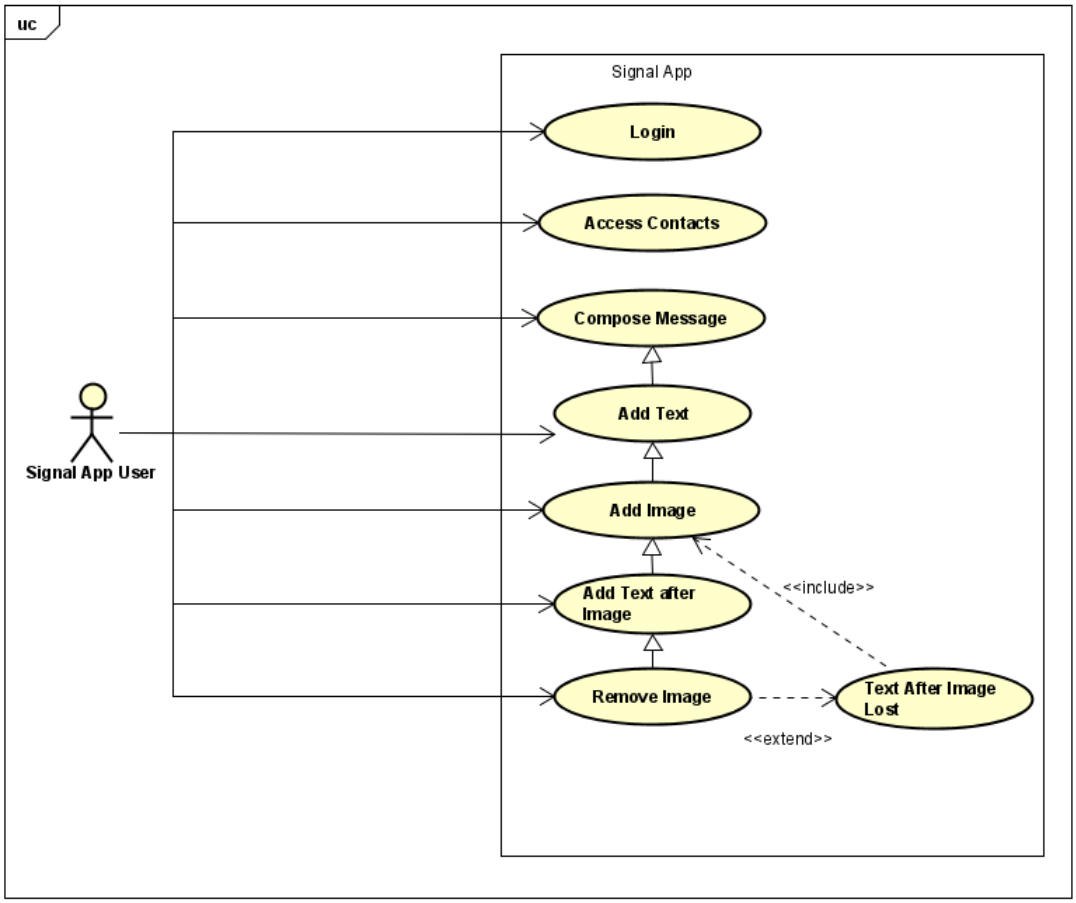

Figure 4: Use Case for Issue #10044

3.5 GitHub Issue #09826 - Media Messages Get Send Twice

Figure 5 presents a Use Case diagram for the #9826 issue. This issue ticket highlights the problem faced when a media message is sent over the Signal Messenger app. When a user shares a media message to a Signal group or a Signal contact, the application ends up sending the message twice even if its multiple images. The expected result is only one message. This issue was discovered on the Nokia 6 device under Android version 9. The signal app version in which this issue occurs is 4.65.2. A screenshot of the issue is highlighted in the appendix Figure 9.

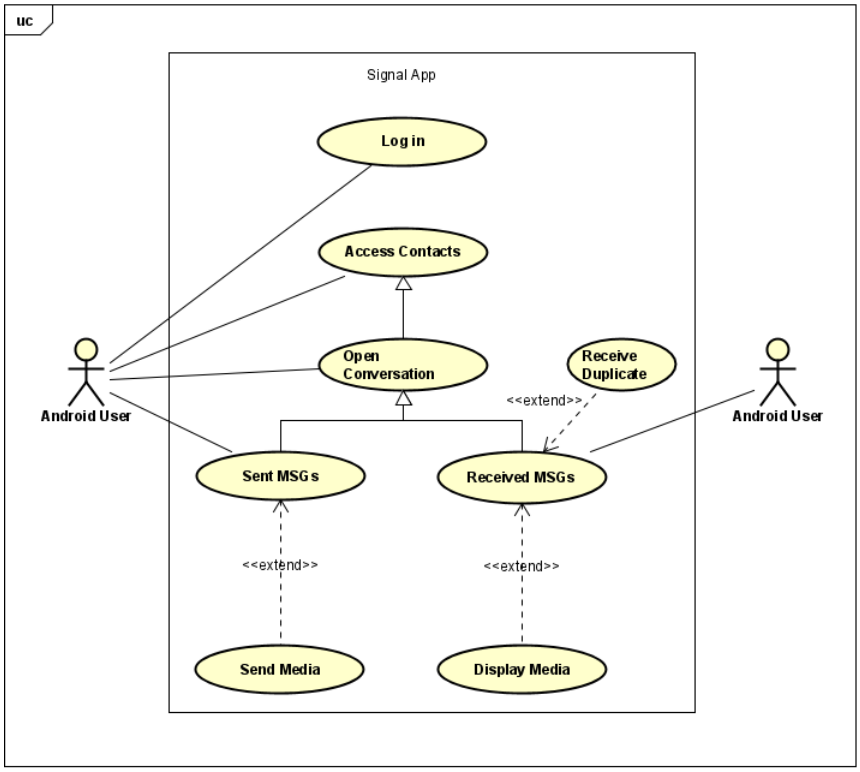

Figure 5: Use Case for Issue #09826

3.6 GitHub Issue #10089 - Dark-skinned Emoji Have Low Contrast in Dark Mode

Use Case diagram for issue 10089 can be seen in Figure 6. This bug occurs when a dark skinned emoji is displayed while the dark mode is active on the Signal app or on the system. The contrast of dark skinned emoji is seen to be very low which makes it harder to see on the black screen with a dark message background. This issue is more noticeable when the user sends a message or when the recipient reacts to a message with dark skin emoji's. It was also seen that the issue was less pronounced for reactions entered by the user as the background for user reactions is lighter in colour. The group collectively decided to choose this bug to implement as it was believed to be very relevant to user experience. Everyone uses emoji's in text and quick reactions to messages, having a bug like this could potentially be seen as offensive to some users which makes it an important parameter to fix for any application. Looking more in depth, the solution to the code was also analyzed to be feasible for the targeted timeline. More details can be found in the implementation process section.

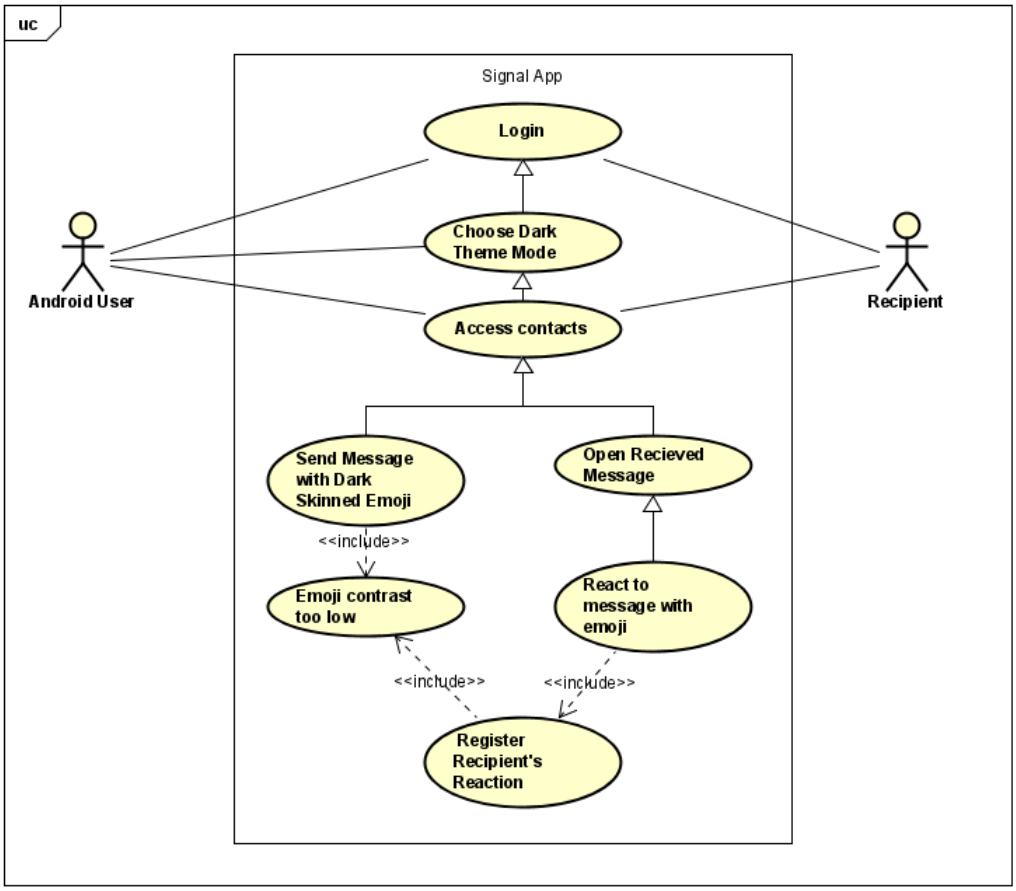

Figure 6: Use Case for Issue #10089

3.7 GitHub Issue #10085 - Group Name Can't Be Made as Long on Editing as on Creation

The diagram in Figure 7 represents the use case diagram for replicating the issue regarding group names. The group name creation page and group name editing page have different character length limits. This causes the display of the group name in edit mode to be cut-off if it approached the maximum number of characters allowed. The solution to this problem would require the name limit length to be consistent at all times throughout the application.

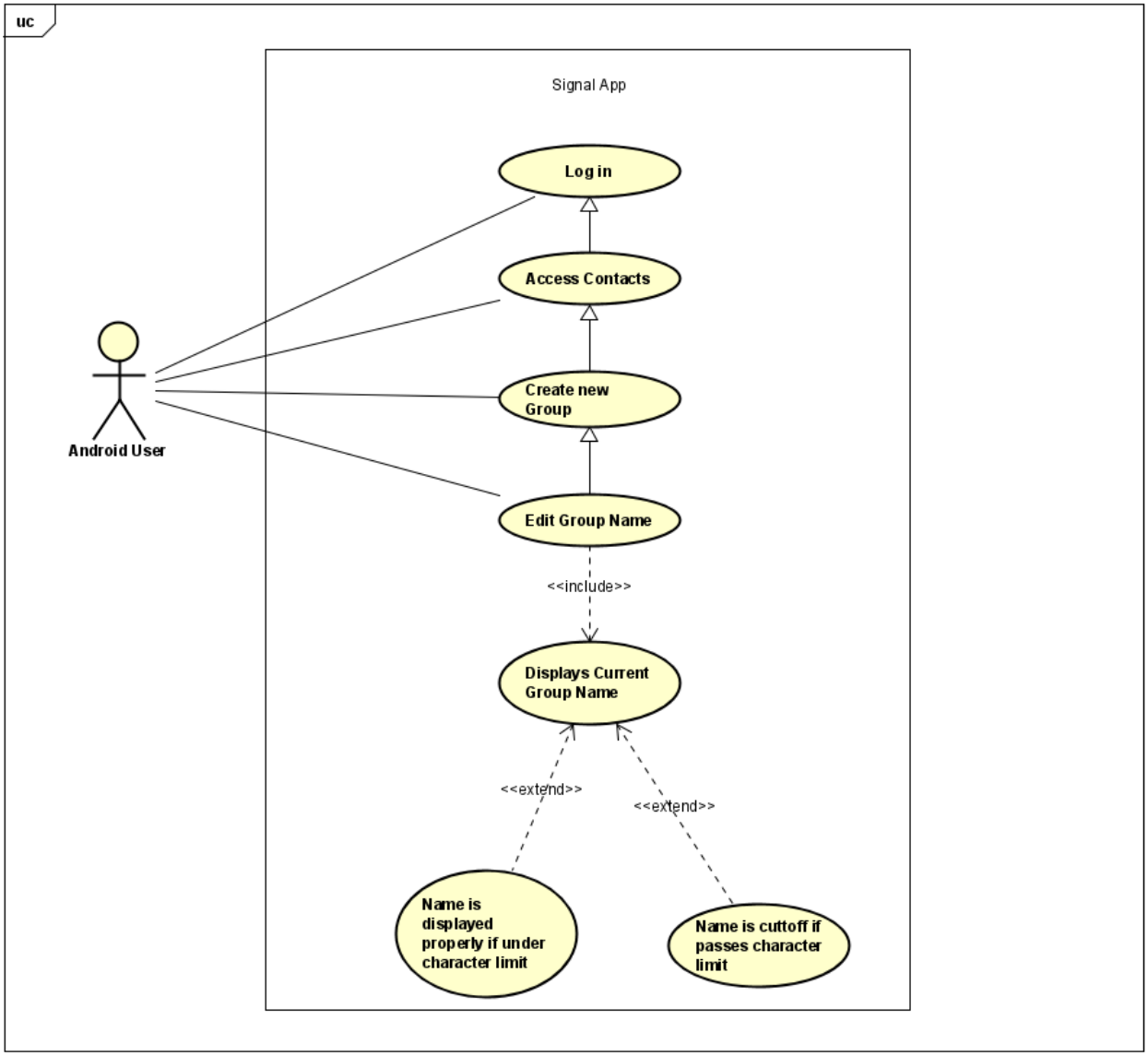

Figure 7: Use Case for Issue #10085

4 Implementation Plan

4.1 Issue #10089

When it was decided which bugs to choose it was not clear which might be the simplest to implement. When our shortlisted bugs were compared, multiple bugs seemed to be fixable, however, when inspected more closely this was not always the case.

In order to begin the process of understanding how the methods of the app worked together, a trace of the program was performed. This was done in order to provide insight into when objects are created, thus hopefully making clear where in the code the bug is taking place. Although, when the trace was done, it did not yield the information which was anticipated so a new route needed to be taken.

When the problem description was read, it described that the emojis were difficult to see due to low contrast between the emoji, as in Figure 10, and its background element. When tested it was determined that the emojis themselves are appearing as expected. If the emojis were the element causing the contrast issue there would need to be different rules for different emojis resulting in a very complex bug to fix. It was established that the emojis couldn't easily be modified so the plan moved to focus on how we could increase the contrast between the emoji element and the surface it appeared on.

Since this issue was to do with how things were being displayed on screen, some research needed to be done into how Android does this [2]. By looking through the documentation it was found that if an element is going to be displayed, it would be of the type 'Drawable'. As per the Android documentation, a drawable element can be defined either by directly using a bitmap file or by describing the element to be displayed using a XLM file [1]. This meant that the drawable element in our bug would also needed one of these files in the project's resources folder.

Once the resources folder was located, the drawable folder was found and examined. It was found that the Signal app describes its drawable elements using XML files, which meant that in order to change the colour it would be as simple as changing which colour was used in the XML description. First, we focused on improving the contrast of the reaction emojis as they had the worst readability. The project tree was scanned and those XML files which were associated with reactions were opened. After it was determined which of the files needed to be changed, the stroke colour was adjusted to be lighter in order to increase the contrast.

When the fix was implemented for the inline emoji's it was more difficult to find where the colour would need to be adjusted. It was now understood that since this was being displayed on the screen, it would need to have a drawable resource. Based on this knowledge, the resource manager was explored further for any resources which looked similar to a sent message (since they appeared as dark grey). The first route explored was again the drawable folder where multiple XML files describe sent message bubbles shown in Figure 11. When these XML's were inspected the background colour used what just white, however a test was done in order to see if the background colour was changed the contrast would be better, shown in Figure 12. However, as depicted in Figure 13, this resulted in even lower contrast. With this knowledge, it could be understood that rather than these files only define the shape and background colour, which are then coloured over with the colours of the theme which is selected, hence why it is the 'background'. Once the XML was located where the theme colours were defined, themes.xml, the Dark Theme was found and it was found that the background of the conversation bubbles was being overwritten with a darker grey. A new grey was chosen which better suited dark skinned emojis, and once recompiled the result in Figure 18.

4.2 Issue #10085

Part of our process into determining a good choice of a bug to fix, involved analysing the complexity of the bug and the ability to reproduce the issue. While analyzing the bug description, the issue seemed to have been a minor character limit issue that deserved a better look. When editing a group after being created, the name displayed would become cut-off after a certain number of characters as shown in Figure 14 and Figure 15.

The debugging process involved observing the data provided in the logs when executing the app. After creating a new group and going to the editing option where the bug was occurring, we could observe in the console the functions being utilized during the process. When we accessed the group editing page we were given information about ManageGroupActivity and ManageGroupFragment. After doing some research about what this information could mean, it turns out that the app was accessing the group_manage_fragment.xml and group_manage_activity.xml files. The process is illustrated in Figure 16 and Figure 17

When taking a closer look at the files we discovered that the error was not in the code, but in the layout section of the application. However, these were not the exact files contributing to the error, so we searched for the add_group_details_fragment.xml file used for the creation of a new group. We modified the maxLength variable to 32 characters from the original of 34 characters. The issue was that the group creation page and the group editing option had different character limits for the group name which lead to the group name being cut-off if exceeded the limit. Having both set to 32 as the maximum number of characters has solved the issue.

5 Tests Cases

5.1 Issue #10089

To verify whether the issue is resolved and the changes are truly implemented, multiple tests were run to ensure accuracy. These steps were also taken to ensure while fixing the issue any other related issue did not arise. For the following cases the user first needs to go to Signal apps setting and select 'Dark' mode under 'Appearance' or go to system setting and set the device into dark mode. Table 3 shows the process followed for customer acceptance test cases.

When the group followed the given steps, the test output was successfully achieved. The dark-skinned emoji's could be seen to have higher contrast than what was before the fix. Both cases, emoji as a message and emoji as a reaction were working as desired. The result obtained can be seen in Figure 18.

Table 3: Customer Acceptance Test Case for Issue #10089

Test Number	Test Description	Test Output	Pass(Yes/No)
1	Open a conversation with a contact. Send a light-skinned emoji as a message	The emoji should have normal contrast and user should be able to see it clearly in dark mode	
2	Open a conversation with a contact. Send a dark-skinned emoji as a message	The emoji should have normal contrast and user should be able to see it clearly in dark mode	
3	Open a conversation with a contact. Send a dark-skinned emoji as reaction to a message	The emoji should have normal contrast with the background and user should be able to see it clearly in dark mode. The background for reaction is lighter in colour than the conversation bubble	
4	Open a conversation with a contact. Tell them to react to a message with dark-skinned emoji	The emoji should have normal contrast with the background and user should be able to see it clearly in dark mode	

5.2 Issue #10085

In order to ensure the accuracy of the features that were implemented, it is important to test the boundaries as well as the different locations where the group name is changeable. The test cases shown in Table 4 were used to check if the boundary conditions of the group name text box worked in the expected manner. The test cases also checked that the name in the 'edit group name' text field. The checking of 'edit group name' text field was used to ensure that new issues were not created when fixing bug 10085. The result can be seen in Figure 19, 20, 21 and 22.

Table 4: Customer Acceptance Test Case for Issue #10085

Test Number	Test Description	Test Output	Pass(Yes/No)
1	Create a group chat amongst other users, give the group chat a name that is 32 characters long	The user should be allowed to create a group with a name that is 32 characters long	
2	Create a group chat amongst other users, give the group chat a name that is 33 characters long	The user should not be allowed to create a group with a name that is 33 characters long. The name should cut off after the 32nd character	
3	Create a group chat amongst other users, give the group chat a name that is 31 characters long	The user should be allowed to create a group with a name that is 31 characters long. The user should be able to give the group a name that is less than or equal to 32 characters	
4	Create a group chat amongst other users, give the group chat a name that is 32 characters long. Send a message in the chat, and shortly after edit the group name. Check to see if you can edit the group name to be longer than 32 characters. This will be done by adding characters	The user should not be allowed to edit the group name to be 33 characters long. The name should cut off after the 32nd character	
5	Create a group chat amongst other users, give the group chat a name that is 32 characters long. Send a message in the chat, and shortly after edit the group name. Check to see if you can see all 32 characters in the edit group name text field	The user should see the 32-character name given to the group that was initially created in the edit group name text field	

6 Conclusion

The SCRUM framework selected by the team worked out splendidly. Since, there were seven steps required to complete the assignment a combination of techniques were used. The group allotted 15 days to this assignment. The team began by all selecting an issue from the Signals App and implemented the use of the case diagram for their respective issues. The group then used a decision matrix to shortlist the list of the problems and ultimately decided to fix bug 10085 and bug 10089. Bug 10085 deals with the issue of creating the group name. Users were able to create a group name that is 34 characters long, but when the user went to edit the group name they could see a maximum of 32 characters. To eliminate this issue, it was decided to limit the initial group naming to 32 characters. Now when users initially create their groups, they will only be able to give the group a name that consists of 32 or less characters. Bug 10089 deals with the issue of not being able to properly see dark coloured emoji's when in dark mode. Initially, when dealing with bug 10089 it was difficult to locate where the emojis contrast would be changed. After extensive research, the file was found and the contrast was adjusted to better see the dark coloured emojis in dark mode. After fixing and testing issues 10085 and 10089 we were able to make a pull request to implement our changes.

The group faced many challenges along the way. All members were new to open-source projects and none of us has worked on any issues before. The most difficult thing to choose was which bug to implement. The decision matrix came in handy as it was a proficient way to finalize on the issues. The next thing was to decide on what development process to use so that the group can approach the problem in uniform manner. The divide and conquer approach where the group was apportioned into 2 sprints of 3 and using the SCRUM model the problem got much manageable. The timeline was created to allocate duration to the backlog items as in table 8 according to the SCRUM model. The group underestimated the time taken to implement the selected issues instead of 4 it took group 5 days to finish that task which put more stress on the group later. This is a good lesson for the future assignments and the team is looking forward to minimize any miscalculations and focusing on planning accurate timeline.

In conclusion, the SCRUM approach used to plan the workflow, manage and distribute tasks among the team members proved to be an excellent asset for resource management and enabled the group to complete the tasks required efficiently, while maintaining a high quality of work.

7 References

[1]“Drawables overview nbsp;: nbsp; Android Developers,” Android Developers. [Online]. Available: https://developer.android.com/guide/topics/graphics/drawables. [Accessed: 27-Oct-2020].

[2]“Drawable resources nbsp;: nbsp; Android Developers,” Android Developers. [Online]. Available: https://developer.android.com/guide/topics/resources/drawable-resource. [Accessed: 27-Oct-2020].

[3] Radu Muresan and Kevin Dong - ENGG4420: Real-Time Systems Design Lab Manual Sixth Edition. (2020, Fall). ENGG4420. Guelph, Canada: University of Guelph

[4] Wittenstein ”What is an RTOS” https://www.highintegritysystems.com/rtos/what-is-an-rtos/ (accessed Oct 19,2020)

[5] ”Github SignalApp issues” [Online]. Available: https://debuglogs.org/23c96d5dcd43949dcf9c0b2cde808c069a828da630217ed2d3c7281a5e5c6beb [Accessed: 20-Oct-2020]

[6]”Scrum Process: why what you’re doing might be not Scrum at all”, Magora Systems, 2020. [Online]. Available: https://magora-systems.com/scrum-vs-scrumbut/. [Accessed: 27- Oct- 2020].

A Appendix

A.1 Development Process

Figure 8: Visualization of SCRUM process

A.2 Issue #09826 - Media Messages Get Send Twice

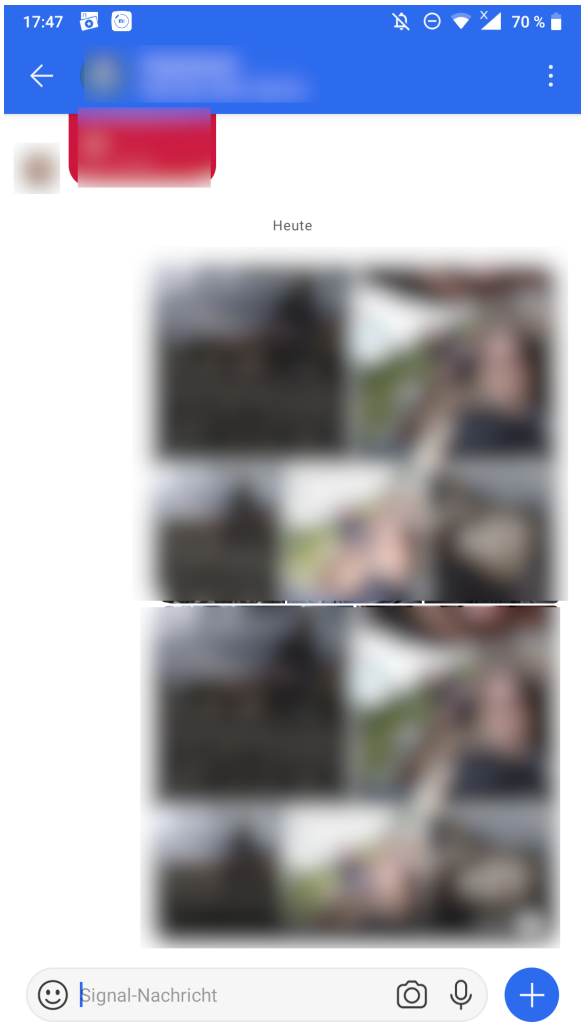

Figure 9: Issue Screenshot #9826

A.3 Implementation of Issue #10089

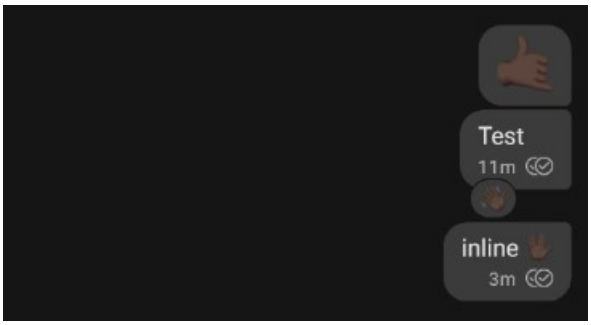

Figure 10: Broken Contrast

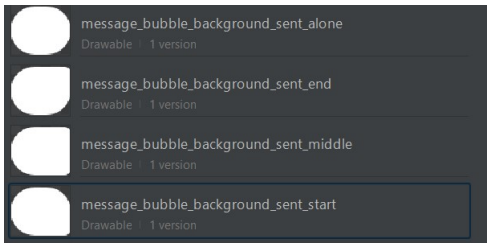

Figure 11: Sent Message Bubbles

```
<!--            <solid android:color="@color/white" />-->
<solid android:color="@color/core_grey_50" />
```

Figure 12: Sent Message Bubble, Darker Background

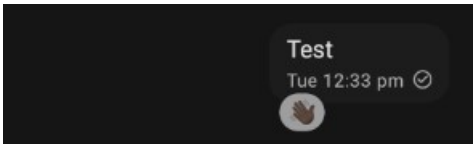

Figure 13: Sent Message Bubble, Darker Background

A.4 Implementation of Issue #10085

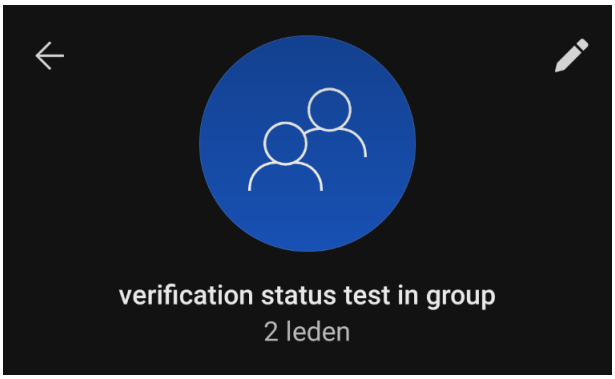

Figure 14: Creating Group Name

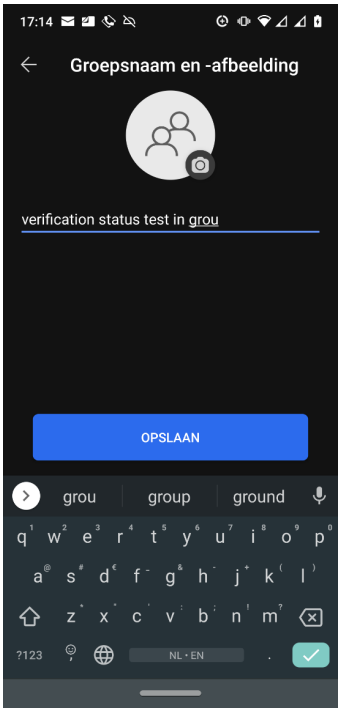

Figure 15: Name Cut-Off in Group Name Edit Mode

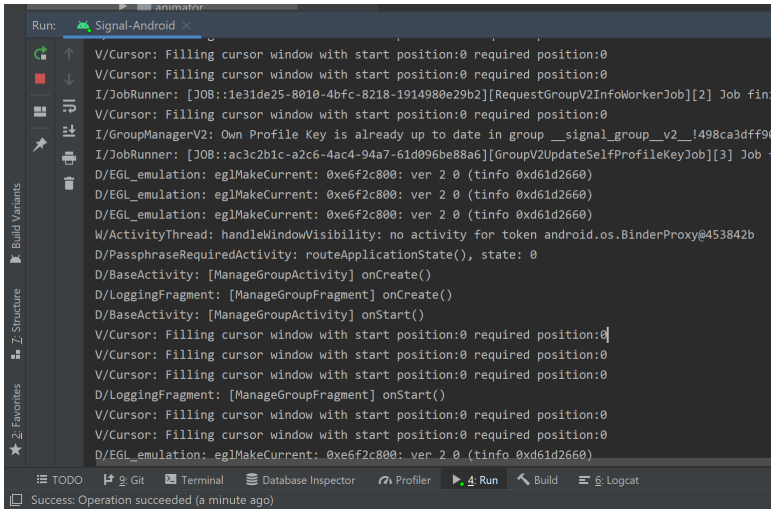

Figure 16: Logs for Debugging

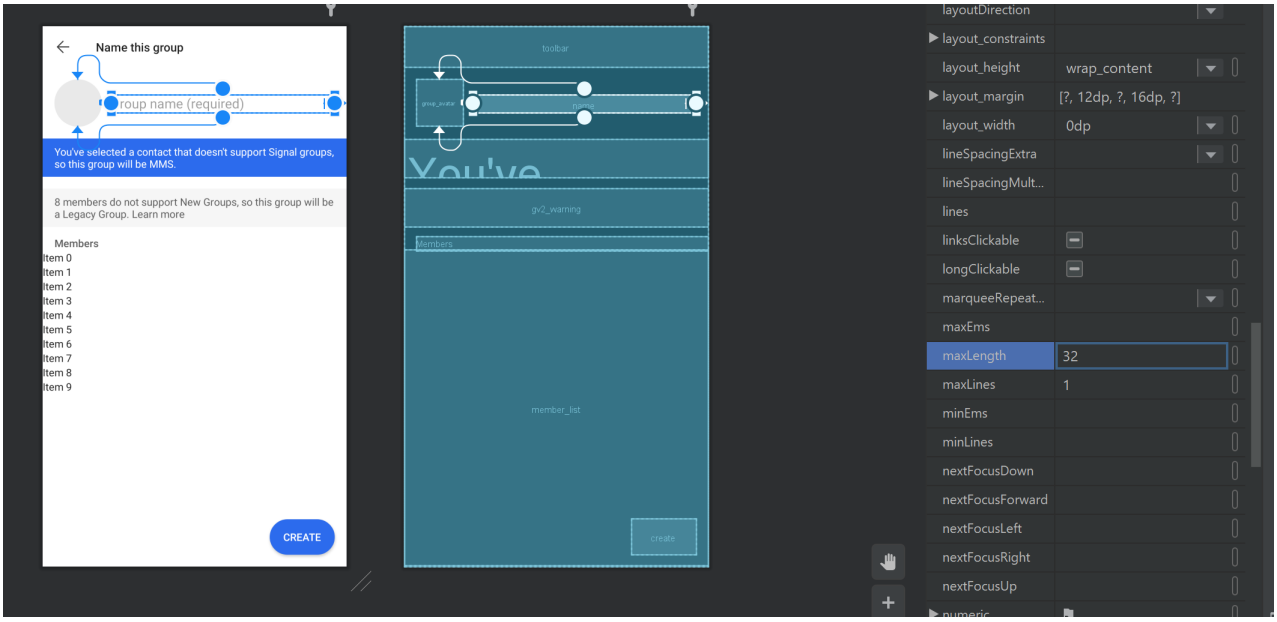

Figure 17: maxLength Variable Modified to 32 Characters as Highlighted

A.5 Test Cases Issue #10089

Figure 18: Fixed Contrast for Reaction and Inline

A.6 Test Cases Issue #10085

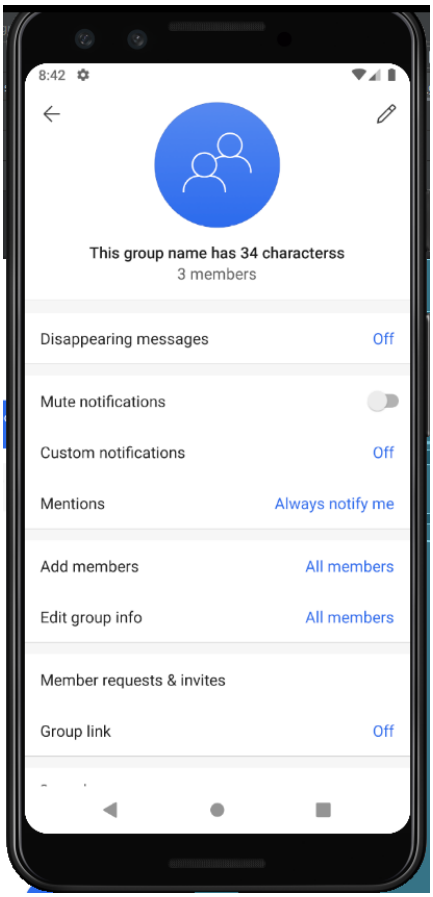

Figure 19: Group Name Before Fixing Bug

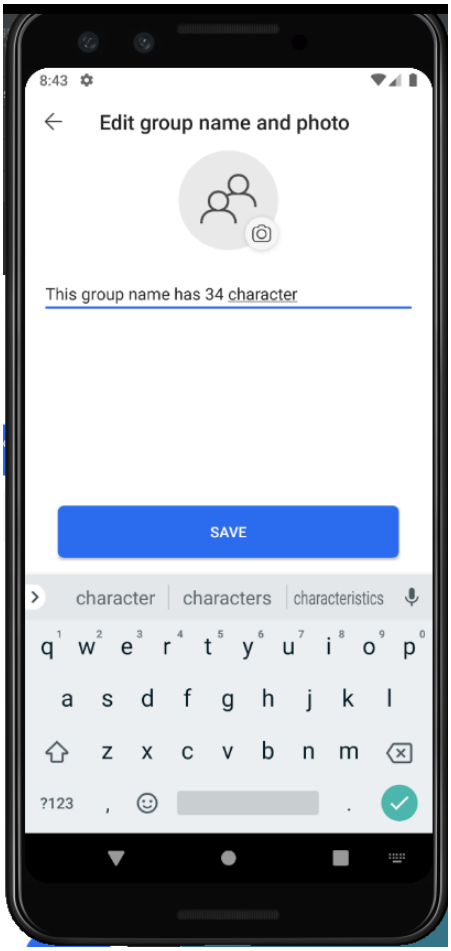

Figure 20: Editing Name Before Fixing Bug

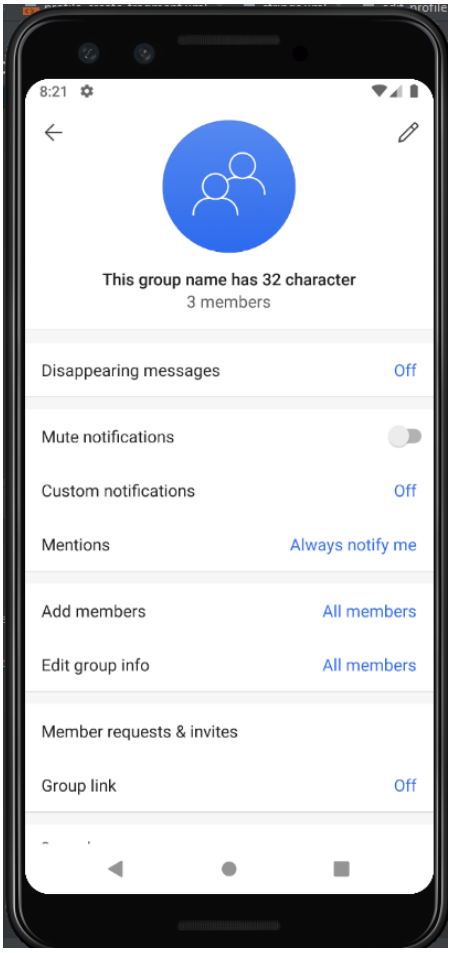

Figure 21: Group Name After Fixing Bug

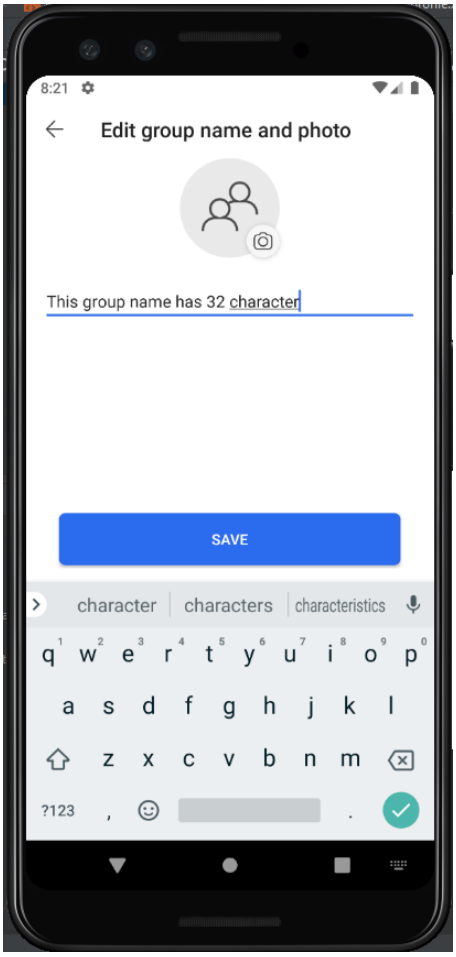

Figure 22: Editing Name After Fixing Bug