Facebook Messaging

Implementing Video Sending for Facebook iOS

Capstone Project, 2021

Emerson Berlik
Computer Science
The University of Virginia
School of Engineering and Applied Science
Charlottesville, Virginia USA
Elb5esn@virginia.edu

Abstract

Facebook's iOS application includes a messaging tool widely used in developing countries. Previously this messaging tool lacked support for sending video messages, which led users to move to other messaging platforms. My project added video-sending support with a real-time progress bar and a camera with filters by modifying the iOS Facebook app built in Objective-C. I was able to support video uploading through the use of a data querying tool called GraphQL. In addition, I worked closely with backend engineers to add server-side support for videos. After completing full functionality for video sending, my tool underwent successful internal testing. By the end of my internship, my tool was nearing the end of public testing and was almost in the release stage. This was possible due to a slightly positive delta in message-sending through nearly a million exposures to the video-sending capability. The addition of video-sending will continue to improve the messaging experience within Facebook iOS.

1. Introduction

Social media is all about being able to share information. With the rise in social media that focus on video sharing, such as Snapchat, it becomes increasingly important to add video capabilities to not lose users, especially teens, who tend to be the primary users of these platforms. Similarly, flashy technology such as the filters that Snapchat employs is targeted towards younger users in an attempt to increase teen engagement. Facebook loses about 3% of its teen users a year and this can be attributed largely in part to Snapchat and other competing platforms [2]. To minimize this loss, Facebook has sought to improve its Facebook and Instagram platforms by implementing similar features as these other platforms.

2. Background

Facebook Messenger is the primary platform used for instant messaging by Facebook. While all users used to have the ability to access Messenger through the Facebook iOS app, Facebook now requires users in countries with a high frequency of messaging to install the Facebook Messenger iOS app to use the tool. As a result, the Messenger app has many more daily users than the in-app Facebook messaging and thus has more maintenance and improvement.

The Messaging in Blue team at Facebook which I was a part of was responsible for bringing the same user experience to Facebook iOS as the Messenger app. One of the core missing functionalities was video sending. By adding video sending, my team hoped to increase the number of total daily users. In addition, my team had a young adult initiative that aimed towards increasing young adult engagement in messaging. One of Instagram and Facebook's most widely

used features is stories, which allows users to use an innovative camera that includes different filters and capture modes. I was in charge of adding this camera into messaging to both improve the user experience and make the messaging feature be more appealing to the teen audience.

3. Related Work

In *Framing Public Policy for Internet Data Privacy*, Bevara discusses building an end-to-end encrypted (E2EE) chat service [1]. While my project involved mobile message sending, Bevara and his team built a service targeted at those without access to mobile devices and states that the majority of encrypted chats occur on mobile. Although our projects were built on different platforms, Bevara provides strong evidence about the importance of encryption of messages in general, such as privacy from third parties. The video sending involved in my project is currently encrypted, but does not support E2EE. Adding E2EE will be a very beneficial future step for my project.

4. Project Design

My project included three main components:

- The first component added support for a video object to be sent through a message.
- The second component added a progress bar user interface to indicate the upload progress of a video in real-time.
- The last component replaced the current camera with a new one with filters that supported video capture.

4.1 Implementation

My implementation was done in Objective-C on the existing code base for Facebook iOS. Sending videos required both frontend and backend work. I was responsible for completing the frontend implementation while communicating the backend tasks to a backend engineer on my team. To send videos, I had to convert an iOS media video item into the path for the actual mp4 file stored on the phone. Following this, I had to allow users to select videos from the gallery in addition to images, which were already supported. I used an existing framework for media uploads to the backend to create an application programming interface (API) for uploading videos. At this stage, I worked with my team's backend engineer to allow the server to accept mp4 uploads. I used this same API to track the real-time progress of the upload and displayed it on a user interface progress bar I created underneath the video. Lastly, I used the code for the existing camera in the Facebook Stories feature to support the same camera in the messaging view.

4.2 Testing

With each new feature, there are two stages of testing before public release. This includes internal testing, which allows users within the company to test the feature; and external testing, which exposes a percentage of users to the feature to see if any regression occurs. For each type of testing, there is a control and test group. Hundreds of statistics, such as daily usage, are compared between the two groups. If regression occurs in any of these statistics, a deeper analysis of the cause occurs. My feature successfully finished internal testing with no regression and was undergoing public testing at the end of my internship.

4.3 Challenges

The largest challenge I faced in my implementation was a video parsing error. Any video sent from Facebook iOS and viewed in Facebook iOS would display a generic error. Because my team did not work with the code in which the error was generated, I was required to reach out to many different teams to deep dive into the issue. In addition, I was a frontend engineer and did not have experience or an understanding of the underlying backend code which could have

been causing the issue. Ultimately, I was able to isolate the root cause of the issue, which involved having the MOOV atom, which is the part of a video file with metadata, in the wrong location of the file. Through my investigation, I realized the file issue also existed for Android. Through my fix, the overall code quality was improved for both iOS and Android. In addition, my project became unblocked and I was able to implement the other aspects of video sending.

5. Results

One goal of my feature was to get a positive delta in daily messages sent (DMS). I was unable to witness results due to my project being unreleased at the end of my internship. While my DMS was neutral at the time of my departure, it could increase after a release.

6. Conclusion

With the addition of video sending in Facebook iOS, the millions of daily users who use the Facebook iOS platform will be able to share the same experience as those who use Messenger, the better maintained alternative. In addition, the filter camera will increase young user engagement and assist with decreasing the number of teen users lost every year. Once the video sending tool is publicly released, I hope to see a positive delta in daily messages to corroborate these expected results.

7. Future Work

While the core functionality of my project was entirely built out at the time I ended my internship, many improvements could have been made. Currently, if a large video is uploading and the app is closed, the video upload is stopped entirely. An improvement involves pausing the upload upon closing the app and resuming it once the app is reopened. There is currently also a video size restriction on the backend that is not reflected on the frontend. Any video uploaded that is over 25 MB will ultimately fail to send without properly notifying the user of the cause. Lastly, videos can only be played inline. If users could full-screen videos, the user experience would be better. All of these improvements are currently underway.

8. CS Program Evaluation

The relevant UVA coursework that helped me complete this project includes Mobile App Development, Advanced Software Development Techniques, and Program and Data Representation. Mobile App development taught how to structure and design a mobile app. I used a similar design process to the one I used in class when I synthesized an approach to my internship project. Advanced Software Development Techniques greatly emphasized the importance of working productively in a team setting and touched on an Agile style of development. I used these exact skills when communicating to my team and other teams to add backend support to my project as well as fix bugs.

Lastly, Program and Data Representation went over proper coding practices to maximize efficiency. I also worked in Objective C, which is a superset to the language taught in that class. Using this knowledge gave me insight into how to write efficient and readable code. While these courses gave me experience that allowed me to succeed at my internship, a class for iOS development would have been greatly beneficial to my experience since I had to learn it after beginning my internship.

References

[1] Bevara, S., Behl, Madhur (advisor), Foley, Rider (advisor), Rogers, Hannah (advisor), & Sun, Yixin (advisor) (2021). *E2-Chat: A Web-Based End-to-End Encrypted Messaging Service; Framing Public Policy for Internet Data Privacy*. Charlottesville, VA: University of Virginia, School of Engineering and Applied Science, BS (Bachelor of Science), 2021. Retrieved from https://doi.org/10.18130/31bn-7p44

[2] Kantrowitz, A. (2020, April 7). *Snapchat was 'an existential threat' to facebook - until an 18-year-old developer convinced Mark Zuckerberg to invest in Instagram Stories*. Business Insider. Retrieved October 18, 2021, from https://www.businessinsider.com/how-developer-mark-zuckerberg-invented-instagram-stories-copied-snapchat-2020-4.