Francis Kigawa
Professor: Joan Feigenbaum
CPSC 610: Computer Science and Law

Final Project Report: Content Moderation System for Nombox

Abstract

Over the course of the semester, an important topic that we have discussed in class has been content moderation: the policies that different platforms employ to monitor content, how effective they are, and their impact on the platforms' reputations. It's been fascinating to do a deep-dive into this iterative process that ultimately forms the company's brand. An important part of Reddit's brand, for example, is the liberty with which its users can express themselves - as long as their comments are on-topic. In contrast, Facebook has built complex (and not wholly transparent) infrastructure to closely monitor political posts on their platform. Concurrently with learning about these platforms' moderation of user content in CPSC 610, I have been working on my senior project (CPSC 490), for which I am building a platform, Nombox, that showcases user-generated recipe walkthroughs. As a result, through our class readings and discussions, I have thought about both the content moderation philosophy that would best align with my platform's brand, as well as the most intuitive way to implement it. For my final project, I decided to build a content moderation system that would work both to empower users to make content on Nombox and to set clear rules for the kind of content that is acceptable. In this report, I plan to discuss my considerations for this content moderation system and to delve into the architecture of its implementation.

Introduction

Learning a new recipe is hard. It's not as easy as just searching up a dish, clicking on the top result for how to make it, and following along. Otherwise everyone would be learning new recipes all the time; in reality, most people have a pretty small cupboard of recipes that they cycle through week after week. It's because, at every point in the learning process, there's an obstacle in place. What do I feel inspired to make? Which recipe out of the 20 in front of me do I choose? Will it actually take me through each step from start to finish in a digestible, manageable way? Do I have the time to go and get the ingredients? These obstacles inevitably restrict people's meal vocabulary and worsen their recipe fatigue. I have been working with my peer, Ethan, to create Nombox: a platform that makes learning a new recipe easy. Nombox enables users to create multimedia, recipe walkthroughs, and to share them with their friends. Our platform's creation suite of digital tools will allow for the creation and consumption of the most engaging recipe walkthroughs available.

Over the course of the past several months, I have been working with Ethan to develop this product. When I first came up with the idea, I made ~30 product validation calls, asking people if they could see themselves using a recipe-sharing platform. It became clear that a product like Nombox could create a lot of value, because people were already sharing their favorite recipes and dishes with their friends informally over generic messaging platforms. This discovery led me to thinking - what would the optimal recipe-sharing experience look like, and how could I build it? At the end of the 2021 spring semester, we launched our first product cycle; we asked 6 people to make video-based, recipe walkthroughs, which they shared, collectively, with 24 of their friends. The reception from both the writers (the people creating the recipe walkthroughs) and the readers (the people following along the recipe walkthroughs) was extremely positive, and led to a significant majority of people actually following along with the recipes they had received. We ran another product cycle in August 2021, this time engaging 21 writers and 63 readers. Again, we received significantly positive feedback, with people wanting to keep using our product. At this point, we decided to gather all the data we had collected and harness it to create an app that would model the product cycle experience at scale.

Over the past four months, Ethan and I have cumulatively dedicated ~700 hours to building this iteration of Nombox's iOS app, which we are excited to launch at the end of this month (December 2021). The work that I did for Nombox's content moderation system in this class' final project is integral to its approval on the app store and to a positive users' experience when it's live.

Background

Given the user-generated nature of the content on Nombox, there are important considerations I have to take into account when making the platform live to users. Most importantly, there are guidelines[1] that Apple has set for apps that have user-generated content:

- A method for filtering objectionable material from being posted to the app
- A mechanism to report offensive content and timely responses to concerns
- The ability to block abusive users from the service
- Published contact information so users can easily reach you

Because of these clear requirements, as well as our platform's important obligation to only showcase relevant, safe content, I set out to create a sufficient content moderation system.

When I first started doing research into implementations of content moderation systems, I found a few compelling approaches. The first was a straightforward flow that enables users to flag content and prompts moderators to review it. This type of system is ubiquitous across user-generated content platforms, from Instagram to Tinder. Empowering users to be a front-line of moderation is an effective and simple way to scale moderation with the content on the

[1] https://developer.apple.com/app-store/review/guidelines/

platform. The second approach to content moderation that I researched was a flow that uses NLP models on open source blacklists to find exact string matches and flag matched content. This kind of system would work, to a certain extent, because the user inputs text for the recipe's metadata. And finally, the last approach was the integration of open source APIs to moderate content - the most compelling API being Google's Content Safety API. This functionality would enable automation with low overhead costs for Nombox, which would provide a huge benefit as the platform scales. Ultimately, I decided to go with the first approach, which enables users to flag content for review. For a platform's first content moderation system, it makes the most sense to have a hands-on approach; seeing the kind of content that is being posted and flagged first-hand provides data that ultimately can inform a more hands-off approach like an NLP model or 3rd party API. That being said, later in my report, I will discuss additional considerations for moderating content as the platform begins to scale.

Methods

A content moderation system needs to have rules that dictate what is and is not acceptable to post. Since their creation, social media giants have continuously iterated on the rules that they set for users and moderators to follow. Many of them boast, at their scale, complex and lengthy codes of conduct. But they each started from a simple and short set of guidelines. Similarly, to align Nombox's content as closely as possible with the brand I envision for it, I set my own rules for the platform in its Terms of Service, which is included below:

```
Welcome to Nombox!

Here, we want to make explicit what you can create and share on Nombox. We want people to use Nombox to create and share food and drink recipe content that they find meaningful to them. As a result, any content unrelated to food and drink recipes will be taken down. Further, any content that puts the safety and well-being of others in our community at risk will also be removed. By using our platform, you agree to not create or share anything that violates our Community Standards (informed by Facebook's Community Standards[2]), which are outlined as follows:

- Content unrelated to food and drink recipes
    - Creating content that does not involve a step-by-step instruction process to make food or drink
- Hate Speech
    - Using verbal language that harms or makes users on the platform feel uncomfortable
```

[2] https://transparency.fb.com/policies/community-standards/

- Violent and Graphic Content
    - Displaying any content that contains violence, whether intentional or accidental
- Explicit Sexual Content
    - Displaying any content that contains sexual or suggestive content

```

Unlike other user-generated content platforms that we have discussed in class, Nombox is meant, specifically, for the cooking niche. As a result, its scope for permissible content is much narrower than, say, Instagram or Twitter. If the content is not related to making food or drink, it's not meant for the platform. This initial content filter will make moderators' decisions much more streamlined and will remove significant grey area within which other platforms need to arbitrate. That being said, I can still foresee difficult decisions within videos that are in the scope of cooking. For example, if someone curses while recording a step in their recipe walkthrough, has it violated the Terms of Service as detailed? It would depend on whether the moderator thought the cursing was harmful to readers. It is for early examples of moderation, perhaps like this case, that make a more hands-on approach useful.

Now that I have laid out the Terms of Service that act as the guidelines for permitted content, I will delve into the content moderation system - first as an overview, and then as a technical review of its features.

First, I will discuss the high-level UI/UX of Nombox's content moderation system and how it fulfills Apple's guidelines for user-generated content platforms. While following along with a walkthrough, a user at any point can flag content that they believe violates Nombox's Terms of Service. This experience addresses Apple's guideline for "a mechanism to report offensive content." Once a walkthrough passes a certain threshold of flags, moderators are notified. Within 24 hours of being notified, moderators will review the flagged walkthrough and decide whether or not to remove it. This moderator experience fulfills Apple's need for Nombox to have "a method for filtering objectionable material from being posted to the app" and for "timely responses to concerns." Finally, users that pass a certain threshold of banned walkthroughs are banned and are given Nombox's customer service contact for questions or appeals. This flow addresses Apple's protocol for "the ability to block abusive users from the service" and "published contact information so users can easily reach you."

Next, I will detail the database models that I made to support the content moderation system's functionality. I created two database models, one for the user's content moderation policy (UCMPolicy) and one for the walkthrough's content moderation policy (WCMPolicy). The models and their attributes exist as documented below:

- WCMPolicy
    - numberOfFlags: Number
    - flaggedReasons: [... Enums]
```

    - isSetForReview: Boolean
    - hasBeenReviewed: Boolean
    - isBanned: Boolean
- UCMPolicy
    - numberOfBannedWalkthroughs: Number
    - isBanned: Boolean

My design philosophy in making these models was not only to create an architecture that was sufficiently robust for Nombox's current content moderation needs, but to also take into consideration the best implementation for easy adjustments and additions as the platform scales. For example, by using the numberOfFlags or numberOfBannedWalkthroughs attributes, I can enable adjustment to how punitive the system should be. At what number threshold does the server take action? I could envision, once we first launch Nombox, making a lower threshold in order to have a more hands-on, controlled process. But, perhaps as the platform scales and there is more of a demand for moderation (and less of a hands-on ability to meet demand), the threshold will need to be higher to trigger moderator action. An example of a threshold for a platform at scale that we discussed in class was Apple's CSAM violation threshold, which they documented to be ~30.

I will now discuss the user-facing UI/UX and corresponding server flow with lower-level detail. When a user is following along with a walkthrough, a flag icon will be present and clickable in the case that the user deems the content harmful. On click, a modal will pop up with a selection of categories for the user to choose to detail the type of harmful content. Once the user selects a category and submits it, they will see a message appear stating that the walkthrough will be reviewed within 24 hours. On flag submit, a series of logical steps occur on the server side that calculates, updates, and saves the flag to the database. The server:

- Increments WCMPolicy.numberOfFlags
- Appends to WCMPolicy.flaggedReasons array
- If WCMPolicy.numberOfFlags > threshold and WCMPolicy.hasBeenReviewed is false
    - Sets WCM.isSetForReview to true
    - Sends an email to Nombox customer service to review the walkthrough

Now, I will delve into the moderator-facing UI/UX and triggered server flow. When a moderator is notified that a walkthrough has been set for review, they review it and check to see if it violates Nombox's Terms of Service. Because the moderator has moderator permissions, when they are reviewing the walkthrough, a review icon will be present and clickable. If the moderator decides to take down the walkthrough, they click on the review icon and choose the take-down option. Conversely, if the moderator decides not to take down the walkthrough, they click on the review icon and choose the reviewed option. It's important that, even if the moderator chooses not to take down the walkthrough, it is still marked as reviewed; that way,

even if another user flags the same walkthrough, a moderator will not be notified for its review again. On review submit, a series of logical steps occur on the server side to update the walkthrough and user content moderation policies:

- When a moderator submits a review action to take down the walkthrough, the server:
    - Sets WCMPolicy.isSetForReview to false
    - Sets WCMPolicy.hasBeenReviewed to true
    - Increments UCMPolicy.numberOfBannedWalkthroughs
    - If UCMPolicy.numberOfBannedWalkthroughs > threshold
        - Sets UCMPolicy.isBanned to true
- When a moderator submits a review action not to take down the walkthrough, the server:
    - Sets WCMPolicy.isSetForReview to false
    - Sets WCMPolicy.hasBeenReviewed to true

Once the content moderation flow is completed, there are effects for banned walkthroughs and banned users. In the case that WCMPolicy.isBanned is true, anytime a user tries to access the walkthrough, an error will state that the walkthrough has been taken down and give the user the customer service information to ask questions if they would like. Additionally, in the case UCMPolicy.isBanned is true, anytime the user tries to login, an error will state that their account has been banned and that they can reach out to a customer service contact for questions or appeals. The desired user experience, after a walkthrough is banned, is for the writer to be able to reach out to Nombox customer service to better understand why their walkthrough violated the Terms of Service. The hope is that the user will better understand their violation and will be sure to upload acceptable content in the future. In the case that they continue to make violations, the desired user experience is for them to not be able to generate any content on the app. We made a deliberate choice in making a phone number verification for Nombox, in order to increase the difficulty with which abusive users would be able to make new accounts.

Conclusion

This first iteration of the content moderation system for Nombox takes a more hands-on approach for both users and moderators to maximize only acceptable content on the platform. It's appropriate given the newness of the platform; I want to make sure that our first users feel like there is transparency on the app for them to generate or follow content. And, in the case that they decide certain content is unacceptable, I want to make sure that they feel empowered to themselves be the front line of moderators. As the platform grows and scales, though, so will the content moderation system. I could see next steps being a separate interface entirely for moderators - where they would be able to receive tickets from users asking for appeals, or where they would be able to revert a decision that had been successfully appealed. Also, I could see new steps of automation that could be especially effective as the demand for moderation increases. As I had mentioned in the background section of this report, Nombox could use NLP

models to scan through the user-input text and check to see if there are any words that violate the Terms of Service. Finally, outsourcing AI scanning of video content to open source software like Google's Content Safety API could be a robust solution that would immediately flag walkthroughs before they are even put in front of other users. Similarly to other platforms, however, these automated flags would have to be reviewed by actual moderators before any final say, to ensure that there are not any wrong decisions being made.

More generally, this iteration of Nombox has all the necessary functionality for a recipe creation and sharing platform. Ethan and I just need to edgecase and remove any bugs from the software before submitting to the app store (by the end of December 2021). We are so excited to onboard new users in 2022 and have them start to make their own recipe walkthroughs, which they will be able to share with their friends. I am so excited to see Nombox and its content moderation system live and in action, and I look forward to incorporating feedback from users to keep refining and improving our software and policies.