Alex Stamos, Stanford Internet Observatory
Shelby Grossman, Stanford Internet Observatory
Jeffrey Hancock, Stanford Internet Observatory

This document presents a slightly modified version of the project that students work on for Trust and Safety Engineering (a computer science course taught by Alex Stamos) and The Politics of Internet Abuse (a political science course taught by Shelby Grossman). After add/drop has ended, students are put into groups that include a mix of students from both classes. The ideal group size is 4-5 students. The first year that we had this cross-course project, Milestone 1 was completed in groups. The second year, Milestone 1 was completed by students independently. Either way works.

If you are teaching a course to students with technical skills, you could use this document as is, though you might consider cutting some of the policy components depending on group size. If you are teaching this course to students without technical skills, Milestone 1 and the first component of Milestone 2 are appropriate.

Project Overview

For the course project, you and your group will be the Trust and Safety team at a major social media or consumer cloud platform. You will be assigned to a group at the end of April. The team will focus on a particular type of abuse, proposing policies to the executives at your company as well as researching and implementing relevant technological solutions within a content moderation bot in Discord.

The project is split into three milestones, which will be completed over the course of the quarter. The first milestone will be completed individually. The second and third milestones will be completed with your group. The final milestone will culminate in a presentation that your group will give to the teaching team and guest judges from industry, the evening of Wednesday, June 5th.

Please do not generate the text you submit using ChatGPT or any other LLM. You will have plenty of opportunities to use these systems to generate test data or perform abuse detection in future milestones, but anything submitted by students in this assignment should be written by humans.

Milestone 1: Abuse Study and Content Policy

Percent of Final Grade: 20%

Deliverables:

- A PDF containing two major sections:
    - 1) Abuse Research Report (2000-4000 words)

- 2) Policy Comparison Table

Submission*:* This first milestone will be completed independently. CS152 students should upload the document to their Canvas site and POLISCI143 students should upload the document to their Canvas site.

Note: The abuse type you choose to focus on for this milestone may or may not be the abuse type your group focuses on for milestones 2 and 3.

Part 1: Abuse Research Report (**75%** of assignment grade)

Description: The execs recently came to your team and tasked you with looking into a significant type of online abuse, researching the current best options available for dealing with such abuse, and making specific recommendations to the company of how to detect and mitigate it.

In writing this paper, please use citations (choose your preferred style) for factual information and feel free to add your own original interpretations and suggestions. Please follow the structure below for the paper overall, but you are welcome to add additional information in whatever sections you see fit. You are also welcome to use graphics, charts, or diagrams as long as you cite the original source.

Example Abuse Types: You are welcome to write about any of these abuse types or one of equivalent importance. If you want to move away from topics that are covered in the syllabus, please check in with the teaching team first.

- Suicides driven by bullying on Instagram
- Murder-Suicides on Facebook Live
- Live streaming of terrorist attacks
- Government propaganda against domestic minorities
- Disinformation in online ads
- Coordinated harassment of journalists on Twitter
- Sextortion (on many platforms)
- Trading of Child Sexual Abuse Materials (on many platforms)
- Online cryptocurrency scams on Twitter
- Hate speech on a streaming platform
- Terrorist recruitment on Twitter
- Fraudulent identification on Airbnb
- Grooming on Snapchat or Instagram
- Catfishing on dating apps

A note on choosing abuse type: You may end up working with this abuse type for the entire quarter, so we encourage you to pick a topic that you care about. Note that when you later work to implement technological solutions we will use stand-ins for illegal and/or very harmful material (e.g. pictures of kittens instead of CSAM). Don't let the potential technical challenge of a topic scare you away from tackling it; your milestones will be graded on effort and thoughtfulness, not whether or not you're able to effectively solve these problems. They're still problems in the real world because they're quite hard, after all!

Required Sections:

- *Description of the Abuse Type* - Provide a summary of the kind of abuse, including citations to known examples. This can include linking to reporting in the media, academic research, talks by professionals, or links to legal documents like indictments.

- *Actor and Victim* - What do you know about the people behind this kind of abuse? How about the victims? Is this something anybody can experience, or is the abuse tied to a specific part of the victim's identity? Are there forums or other platforms where these kinds of abusers congregate and can we learn more about them?
    - *In-Depth Profile Piece on an Actor or Victim* - Research someone who has personal experience (as an actor, victim, bystander, or content moderator) with the abuse type you are analyzing. What are their experiences with this type of abuse? How has the abuse shaped or influenced their post-abuse experience (if at all)? What do they have to say about the way content moderation on major tech platforms should handle this type of abuse? Include these findings in your final paper. You will receive extra credit on this section if you are able to do an "in-person" (remote) interview with a real person who experienced (or perpetrated) this kind of abuse.

- *Details of the Abuse* - Describe the immutable aspects of this kind of abuse and how they could be detected. What might differ between attackers and victims? Dive into at least one real-world example and pull out specific moments at which the abuse could be detected or mitigated.

- *Relevant Technologies* - What technologies currently exist that are/can be used to combat this kind of abuse? What are their strengths and weaknesses? On what platforms are they used now and to what levels of success?

- *Specific Recommendations* - What policy, product, engineering, or operational changes do you recommend to deal with this type of abuse?

Length: 2000-4000 words. We will penalize papers that exceed this word limit.

Part 2: Policy Comparison Table (**25%** of assignment grade)

Create a policy table outlining what platform policies are currently in effect that relate to this kind of abuse. Compile language, pulled from the policies of other platforms, that you think is relevant or appropriate. Please do this for **three platforms**. An example table for an investigation into coordinated harassment of journalists on Twitter is below, but you may adjust as appropriate for different abuse types. Think critically about what types of columns you think are theoretically important. Please make sure all table cells that reference policy hyperlink to the exact website. Note that there may be cases where you need to write "unclear." That's fine, just justify the reasoning. You can see other examples in Figure 1 here, Table 1 here, and here.

In addition to the table, in two or three paragraphs, please explore any research on the effects of these policies, including whether they help mitigate the abuse or enable it.

Coordinated harassment of journalists on Twitter (Note: this table includes made up data) Note to instructors: Including this example table will lead many students to focus on this exact abuse type.

Platform	**Does the platform have a policy on harassment?**	**What is the policy?**	**Is the policy specific?**	**Is there a specific policy for coordinated harassment?**	**Does the platform state publicly how they will enforce this policy?**
YouTube	No	NA	NA	NA	NA
TikTok	Yes	"Harassment is prohibited on TikTok."	No	No	No
Twitter	Yes	"[lots of language about various types of harassment]"	Yes	Yes, discussed under their policy about brigading	Yes, they have a three strike rule
etc.					

Milestone 2: Content Moderation Bot

Percent of Final Grade: 20%

This milestone has 3 components and 4 deliverables:

1. **Design a user reporting flow and a behind-the-scenes moderator flow**
    a. Deliverable: A user reporting flow and a behind-the-scenes moderator flow, in pdf format
2. **Implement these flows into your Discord bot**
    a. Deliverable: All code files from your backend implementation checked into a forked Github repository (submit the link to your repo)
    b. Deliverable: Short video (around 6 minutes) demoing your bot's functionality + discussing examples
3. **Writeup**
    a. Deliverable: Writeup about the work you've done so far to handle your specific abuse type (~500 words)

Submission*:* Please have **one** group member from CS152 upload all documents to the Canvas CS152 site before the deadline. Make sure to include your team number in the name of each document.

For milestone 3 you will be asked to make your bot "smart," e.g. train a classifier on your abuse type. For this milestone, milestone 2, it is fine if the back end moderator flow is manual.

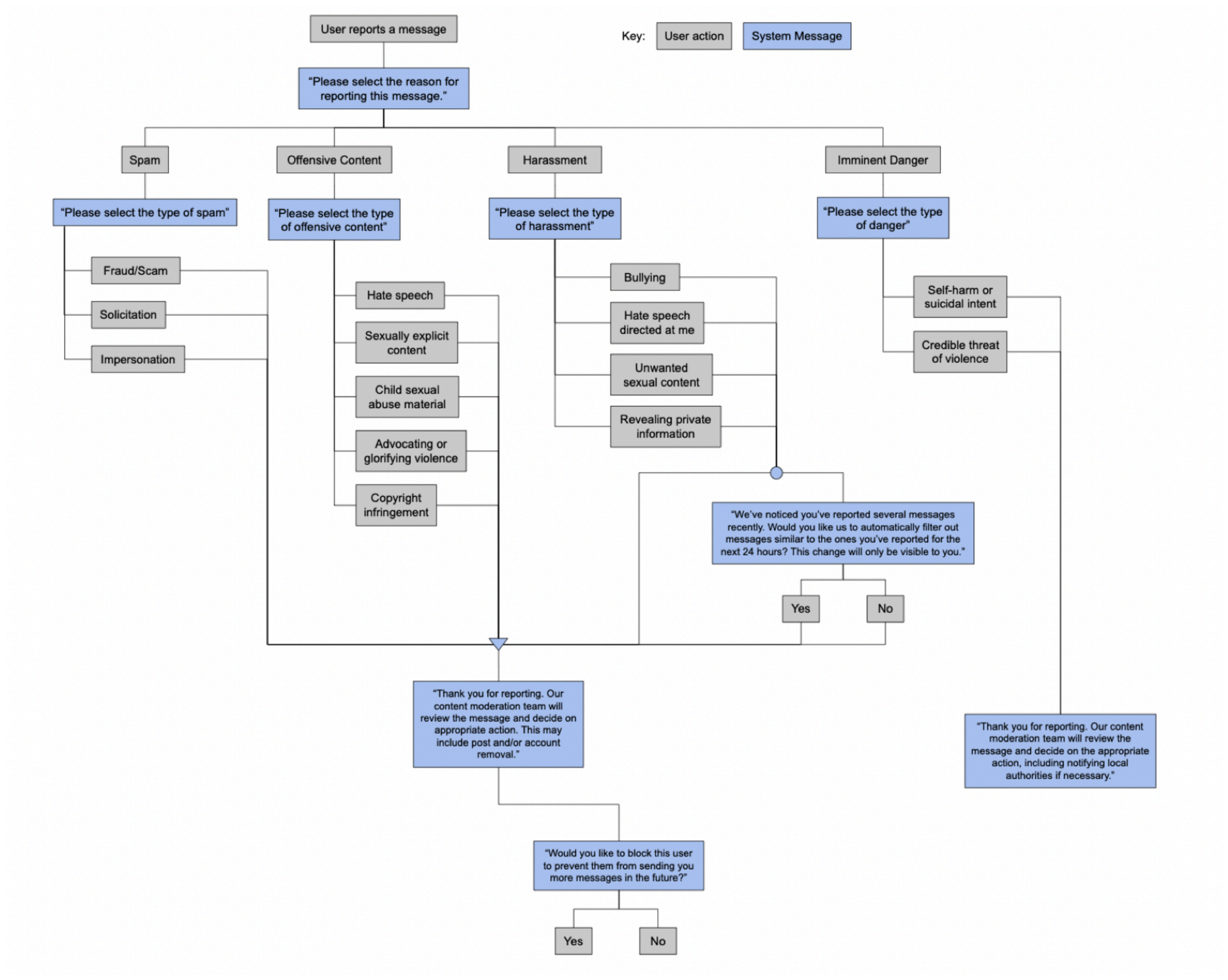
Part 1: User reporting flow and behind-the-scenes moderator flow (40%)

Select an abuse type to focus on for the group project. This should probably be an abuse type investigated by a member of your group for milestone 1, so that you are starting with a good understanding of the abuse and potential mitigations. While choosing an abuse type to focus on, please keep in mind that milestone 3 will ask you to create some kind of automated detection mechanism for that abuse type, so choose something for which automated detection is at least tractable.

You are allowed to choose an abuse type where realistic testing is illegal or would be extremely difficult for the team, such as the trading of child sexual abuse material. In these cases, you will use a type of proxy content to train and test your detection system. For example, we have used "naked" photos of kittens (versus adult cats) as stand-ins for CSAM detection in the past. If you want to scan for extremely violent videos, you can use cartoon violence instead of content such as beheading videos. For topics where testing might require the use of upsetting content, such as racist language or misogynistic threats, make sure that the entire team is ok with the topic before proceeding with milestone 2.

Your group will design two reporting flows: one based on user reports and the other a behind-the-scenes flow for content moderators. You will implement both these flows (within Discord's constraints) in your bot for this milestone. The best way to represent these flows is using a flowchart that includes both the user action and the system's response. The reporting flow should have a variety of abuse types at the highest level, but the more detailed flow (after the first or second prompt) only needs to be built out for your specific abuse type.

An example of a User Reporting Flow focused on hate and harassment that received a high score last year is included here, however as noted above you only need to show a variety of abuse types at the highest level.


User reports a message
Key:
User action
System Message
"Please select the reason for reporting this message."
Spam
Offensive Content
Harassment
Imminent Danger
"Please select the type of spam"
"Please select the type of offensive content"
"Please select the type of harassment"
"Please select the type of danger"
Fraud/Scam
Solicitation
Impersonation
Hate speech
Sexually explicit content
Child sexual abuse material
Advocating or glorifying violence
Copyright infringement
Bullying
Hate speech directed at me
Unwanted sexual content
Revealing private information
Self-harm or suicidal intent
Credible threat of violence
"We've noticed you've reported several messages recently. Would you like us to automatically filter out messages similar to the ones you've reported for the next 24 hours? This change will only be visible to you."
Yes
No
"Thank you for reporting. Our content moderation team will review the message and decide on appropriate action. This may include post and/or account removal."
"Thank you for reporting. Our content moderation team will review the message and decide on the appropriate action, including notifying local authorities if necessary."
"Would you like to block this user to prevent them from sending you more messages in the future?"
Yes
No

Some additional examples from industry:

- Facebook: https://blog.heyo.com/wp-content/uploads/2012/06/FB-Reporting-Guide.png

- Twitter: https://help.twitter.com/en/safety-and-security/report-abusive-behavior

User reporting flow

Your user reporting flow should outline the process that a user is taken through when they attempt to report an instance of your abuse type on your platform. It should do the following:

- Offer users the ability to specify the detailed type of abuse
- Note steps of the process that require review (automated or manual)
- Clearly identify potential outcomes of a report (nothing, post is removed, shadow block, etc.)

Manual review flow

Your manual review flow should outline the process that a content reviewer goes through when they review a piece of content submitted by a user as abusive using the flow you just created. It should do the following:

- Handle reports coming both from users and automated flagging (though automated flagging need not be complete until milestone 3)
- Outline the manual review process of flagged messages - what options are given to reviewers? What information do they have access to? Make sure to clearly identify potential outcomes of a report (nothing, post is removed, user is banned, etc.)
- Are there multiple levels of reviewers? Are there situations where a first-tier content reviewer can engage their management or a specialized investigations team?

Some questions you should think about when designing your flows:

- How many steps should there be? How does this balance warding off malicious reporters/spammy reporters while still encouraging real reports?
- How specific should the options be? What's the tradeoff between offering many different options and only a few? How will this affect user experience?
- What characteristics make content able to be moderated automatically, and what content should go through human review?
- In a perfect world, what outcomes might exist to help keep users safe? (E.g. shadow blocking, user rehabilitation programs, etc.) How can you work those ideas into the flows?

Part 2: Creating an anti-abuse moderation tool (40%)

Your company's T&S-minded CEO has approved your reporting flows and asked that you implement them for this abuse type, commending you on the thought that went into them. Success! She gives you the green light to build out a skeleton of the system for some A/B testing with real users.

You will design and implement a reporting flow within the context of Discord. We are using Discord for the class project because their bot framework is very powerful; many communities build the functionality they want on top of Discord by using bots to greet users as they join servers, auto-respond to messages, moderate chat/ban swear words, and much more. Discord bots are written in asynchronous Python - don't worry if you haven't worked with it before, it shouldn't be too hard to pick up (and the TAs are here to help!). Please consult the Discord Bot Setup Guide at the bottom of this doc.

Context

Your group will be given two pre-configured channels: `group-#` for general chatting and `group-#-mod` to serve as a back-end for human moderators (from now on we'll call them the "main" channel and the "mod" channel). User generated content will go in the main channel, and that content will either be manually reported or automatically flagged by your bot and potentially be sent to the mod channel for human review.

Although we have given you the setup of these two channels, you are not necessarily restricted to the context of a group chat! It is up to you to specify **what context** your moderation tools exist in, and style your main `group-#` channel accordingly. For example, you could say it is a feed of content that one specific user is scrolling through (e.g. Instagram/Twitter style), or say the channel is actually a DM between two users. There's a lot of flexibility here, and as long as you're clear about what you imagine and how you are simulating that within Discord, you're welcome to adapt in whatever way makes sense for the abuse type you have chosen. Please reach out to the TAs if you have ideas that you aren't sure how to adapt; they can also potentially give you additional channels if it would be helpful.

Requirements

For this part, your bot should be able to do the following:

- Allow users to report content (and/or other users) and follow your reporting flows to completion, including outcomes.
    - Note: if your outcomes include banning users, you can just simulate that with messages; we won't be banning from the 152 server in the interest of keeping it functional.
- Allow moderators to moderate reported content using your moderation flow and implement the outcomes to their completion (with the exception of banning users, which you can simulate with a message).

A note on sensitive content here; the TAs can see all messages sent in your group channels, but we won't be actively monitoring them. In order to properly do this milestone, you may have to engage with perverse and hateful content; that is the nature of this kind of abuse-fighting. It is important that you test your bots, but this is not an excuse to behave inappropriately to other students, so make sure that it is clear to your team when you are testing and don't actually target individuals in a way that might be emotionally harmful. Please take care of yourselves and each other, and reach out to the teaching team if you're finding that this work places any undue emotional burden.

Starter code

For this milestone we have given you the skeleton of a moderation bot with the following capabilities:

- Automatically forward every message to the mod channel
- Allow users to report messages from the main channel (reports are initiated via DMs)

Please set up your bot within the first week of the assignment going out so we can catch any potential problems. The TAs will be ready to answer any questions and help debug during section!

Submission

For this part of the milestone you will be submitting a short (up to 6 minute) video demoing all of the functionality of your Discord bot as well as talking through your edge cases. Be sure to begin with a clear description of the context you've chosen for your channels.

Resources

Below are some resources we think might be useful to you for this part of the milestone.

Here is the documentation for discord.py, Discord's python package for writing Discord bots. It's very thorough and fairly readable; this plus google (in addition to the TAs) should be able to answer all of your functionality questions!

Discord bots frequently use emoji reactions as a quick way to offer users a few choices - this is especially convenient in a setting like moderation when mods may have to make potentially many consecutive choices. Check out on_raw_reaction_add() for documentation about how to do this with your bot. You also might want to look into on_raw_message_edit() to notice users editing old messages.

Discord offers "embeds" as a way of getting a little more control over message formatting. Read more about them in this article or in the docs.

unidecode and uni2ascii-janin are two packages which can help with translating unicode characters to their ascii equivalents.

Part 3: Write up (20%)

In approximately 500 words, summarize:

- Your abuse type

- An explanation of the user reporting flow and behind-the-scenes moderator flow, and the rationale for the decisions you made

Discord bot setup guide

For this milestone your group will be making your very own Discord bot. Discord bots are implemented in Python (or Javascript) - don't stress if you haven't written Python before! It's a pretty readable language, so you should be able to pick it up as you go, and the TAs are always here to help.

If you're not familiar with Discord, that's totally okay! Check out this short video which overviews Discord's features and quirks.

[All group members] Joining your group channels

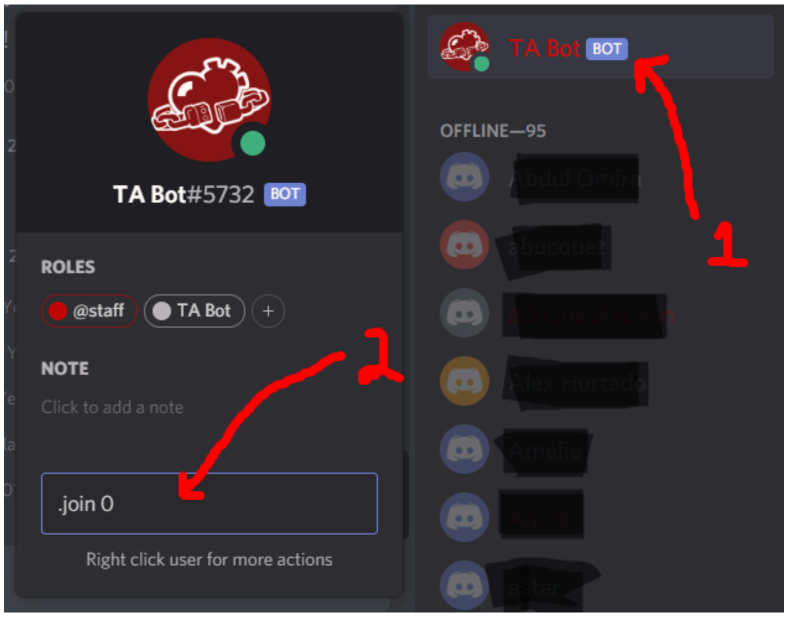
First, every member of the team (both CS and POLISCI) should join the Discord server using this invite link:

[insert link here]

Discord can be used in your web browser, although most people prefer the thick client apps.

For the next two milestones, you and your group will have two channels to test and develop your bot in: `group-#`, and `group-#-mod`, where # is your group's number. We will give you and your bot a special role such that only you and the staff can see those channels; that way, everyone will have their own small workspace.

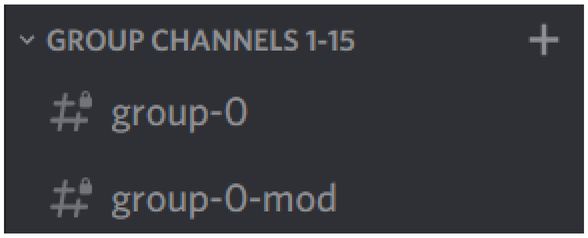
To get the role for your group, click on the **TA Bot** user to bring up this window.
Type in: `.join #` where # is replaced by your group number.



```
.join 0
```

If all goes according to plan, you should receive a message back saying that you have been given a role corresponding to your group number and you should see a new role on your user in the server.

Additionally, you should be able to see two new channels under one of the “Group Channels” categories:



If you accidentally join the wrong group, just message the TA Bot `.leave #` to have the role removed and leave those channels.

Please let [TA] know if something goes awry in this process!

[One student per group] Setting up your bot

Note: **only ONE** student per group should follow the rest of these steps.

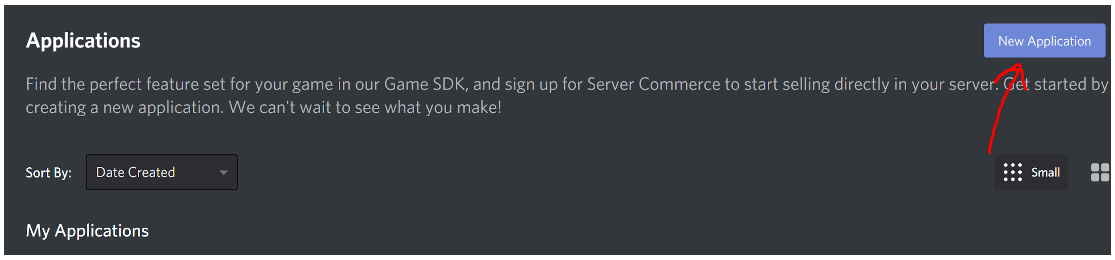
Download files

Fork and clone the GitHub repository here. For instructions on how to fork a github repo, see this article. In order for your group to be able to collaborate effectively on this project, we recommend you create a shared GitHub repository; when you do, make sure you use the .gitignore file included in the starter code so that you don't accidentally upload your tokens to GitHub. Our GitHub repository already has tokens.json in its .gitignore file. When you clone your project from there, you will have to create your own tokens.json file in the same folder as your bot.py file. The tokens.json file should look like this, replacing the "your key here" with your key. In the below sections, we explain how to obtain Discord keys.

```
{
      "discord": "your key here"
 }
```

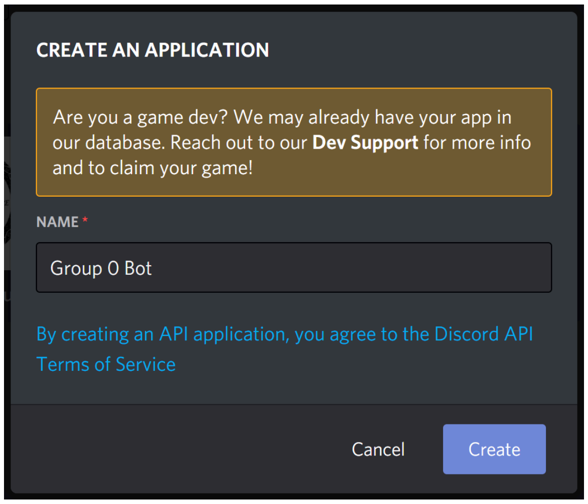
Making the bot

The first thing you'll want to do is make the bot. To do that, log in to https://discord.com/developers and click "New Application" in the top right corner.



Find the perfect feature set for your game in our Game SDK, and sign up for Server Commerce to start selling directly in your server. Get started by creating a new application. We can't wait to see what you make!

Name your application **Group # Bot**, where # is replaced with your group number. So, for instance, Group 0 would name their bot like so:



Are you a game dev? We may already have your app in our database. Reach out to our **Dev Support** for more info and to claim your game!

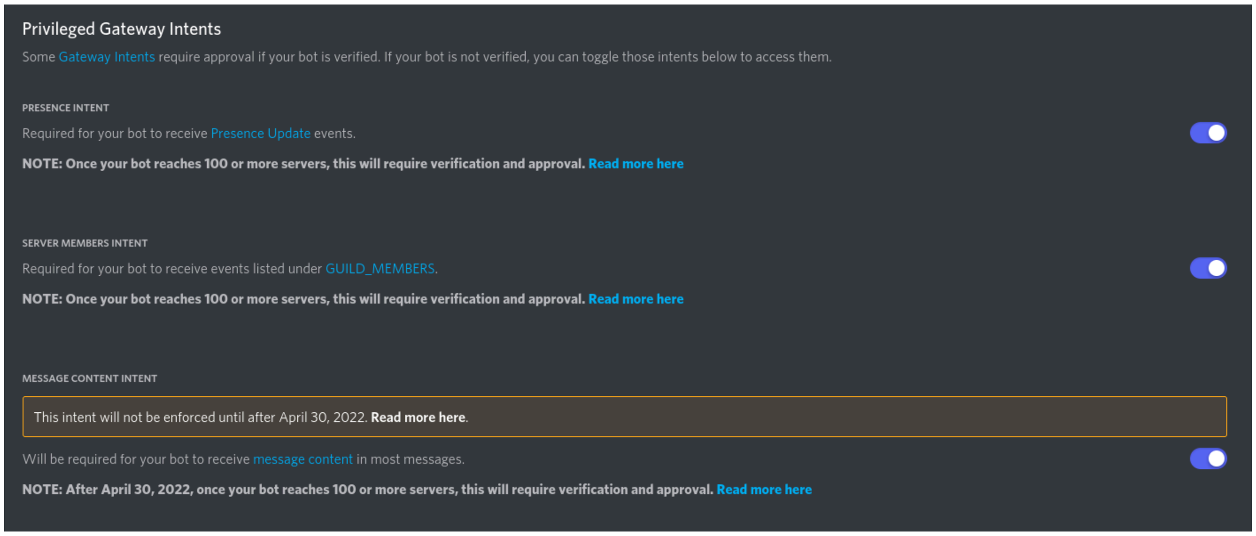
NAME *

Group 0 Bot

By creating an API application, you agree to the Discord API Terms of Service

It is **very important that you name your bot exactly following this scheme**; some parts of the bot's code rely on this format.

1. Next, you'll want to click on the tab labeled "Bot" under "Settings."
2. Click "Copy" to copy the bot's token. **If you don't see "Copy", hit "Reset Token" and copy the token that appears (make sure you're the first team member to go through these steps!)**
3. Open tokens.json and paste the token between the quotes on the line labeled "discord".
4. Scroll down to a region called "Privileged Gateway Intents"
5. Tick the options for "Presence Intent", "Server Members Intent", and "Message Content Intent", and save your changes. See the image for what it should look like.



Privileged Gateway Intents

Some Gateway Intents require approval if your bot is verified. If your bot is not verified, you can toggle those intents below to access them.

PRESENCE INTENT

Required for your bot to receive Presence Update events.

NOTE: Once your bot reaches 100 or more servers, this will require verification and approval. Read more here

SERVER MEMBERS INTENT

Required for your bot to receive events listed under GUILD_MEMBERS.

NOTE: Once your bot reaches 100 or more servers, this will require verification and approval. Read more here

MESSAGE CONTENT INTENT

This intent will not be enforced until after April 30, 2022. **Read more here.**

Will be required for your bot to receive message content in most messages.

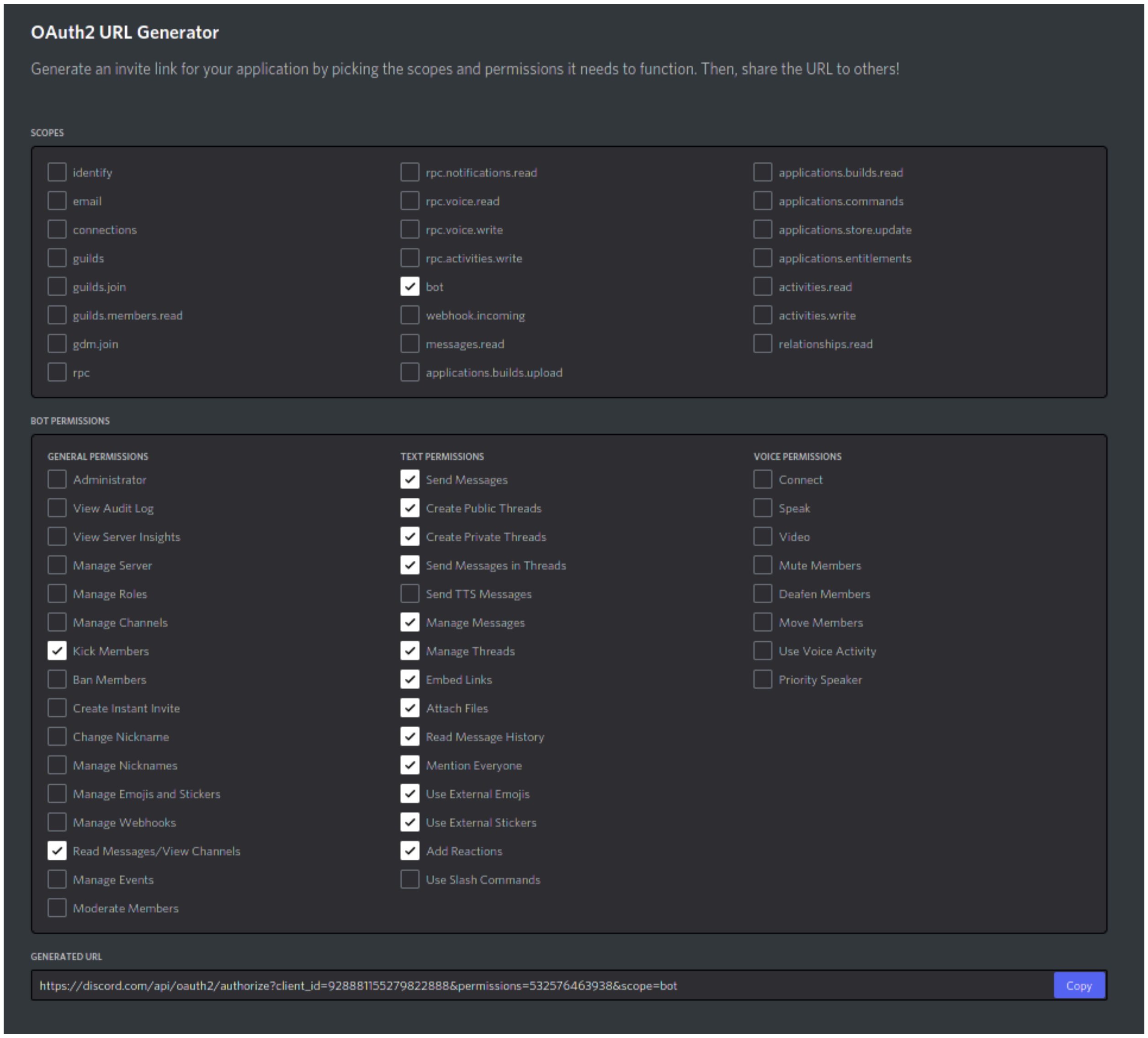
NOTE: After April 30, 2022, once your bot reaches 100 or more servers, this will require verification and approval. Read more here

An aside: It's unsafe to embed API keys in your code directly. If you put that code on GitHub, then anyone could find and use that key! (GitHub actually tries to detect code like this and forbids programmers from uploading it.) That's why we're storing them in a separate file which can be ignored by version control software.

Next, we'll add the bot to the 152 Discord server! You'll need to generate a link that the teaching team can use to invite your bot.

1. Click on the tab labeled "OAuth2" under "Settings"
2. Click the tab labeled "URL Generator" under "OAuth2".
3. Check the box labeled "bot". Once you do that, another area with a bunch of options should appear lower down on the page.

4. Check these permissions, then copy the link that's generated.



OAuth2 URL Generator

Generate an invite link for your application by picking the scopes and permissions it needs to function. Then, share the URL to others!

SCOPES

☐ identify	☐ rpc.notifications.read	☐ applications.builds.read
☐ email	☐ rpc.voice.read	☐ applications.commands
☐ connections	☐ rpc.voice.write	☐ applications.store.update
☐ guilds	☐ rpc.activities.write	☐ applications.entitlements
☐ guilds.join	✓ bot	☐ activities.read
☐ guilds.members.read	☐ webhook.incoming	☐ activities.write
☐ gdm.join	☐ messages.read	☐ relationships.read
☐ rpc	☐ applications.builds.upload	

BOT PERMISSIONS

GENERAL PERMISSIONS	TEXT PERMISSIONS	VOICE PERMISSIONS
☐ Administrator	✓ Send Messages	☐ Connect
☐ View Audit Log	✓ Create Public Threads	☐ Speak
☐ View Server Insights	✓ Create Private Threads	☐ Video
☐ Manage Server	✓ Send Messages in Threads	☐ Mute Members
☐ Manage Roles	☐ Send TTS Messages	☐ Deafen Members
☐ Manage Channels	✓ Manage Messages	☐ Move Members
✓ Kick Members	✓ Manage Threads	☐ Use Voice Activity
☐ Ban Members	✓ Embed Links	☐ Priority Speaker
☐ Create Instant Invite	✓ Attach Files	
☐ Change Nickname	✓ Read Message History	
☐ Manage Nicknames	✓ Mention Everyone	
☐ Manage Emojis and Stickers	✓ Use External Emojis	
☐ Manage Webhooks	✓ Use External Stickers	
✓ Read Messages/View Channels	✓ Add Reactions	
☐ Manage Events	☐ Use Slash Commands	
☐ Moderate Members		

GENERATED URL

https://discord.com/api/oauth2/authorize?client_id=928881155279822888&permissions=532576463938&scope=bot

5. Send that link to any of the TAs via Discord (or by email) - they will use it to add your bot to the server. Once they do, your bot will appear in the `#general` channel and will be a part of the server!

Note that these permissions are just a starting point for your bot. We think they'll cover most cases, but it's entirely possible you'll run into cases where you want to be able to do more. If you do, you're welcome to send updated links to the teaching team to re-invite your bot with new permissions.

Setting up the starter code

First things first, the starter code is written in Python. You'll want to make sure that you have Python 3 installed on your machine. Alternatively, you can use a text editor of your choice.

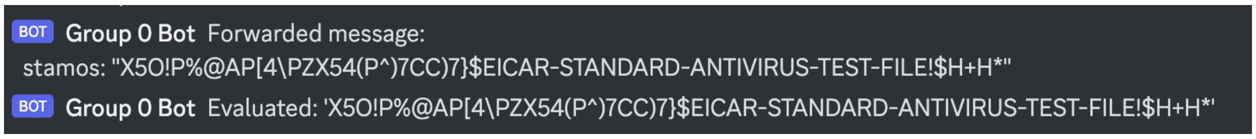
Once you’ve done that, open a terminal in the same folder as your bot.py file. (If you haven’t used your terminal before, check out this guide!)

You’ll need to install some libraries if you don’t have them already, namely:

```
# python3 -m pip install requests
# python3 -m pip install discord.py
```

Guide to the starter code

Next up, let's take a look at what bot.py already does. To do this, run bot.py and leave it running in your terminal. Next, go into your team's private `group-#` channel and try typing any message. You should see something like this pop up in the `group-#-mod` channel:



```
BOT Group 0 Bot Forwarded message:
stamos: "X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*"
BOT Group 0 Bot Evaluated: 'X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*'
```

The default behavior of the bot is, any time it sees a message (from a user), it sends that message to the moderator channel with no possible actions. This is obviously not the final behavior you'll want for your bot - you should update this to match your report flow. However, the infrastructure is there for your bot to automatically flag messages and (potentially) moderate them somehow.

Next up, click on your app in the right sidebar under "Online" to begin direct messaging it (or click on its name). First of all, try sending "help".

Try following its instructions from there by reporting a message from one of the channels to get a sense for the reporting flow that's already built out for you. (Make sure to only report messages from channels that the bot is also in.)

If you look through the starter code, you'll see the beginnings of the reporting flow that are already there. It will be up to you to build that out in whatever way your group decides is best. You're welcome to edit any part of the starter code you'd like if you want to change what's already there - we encourage it! This is just meant to be a starting point that you can pattern match off of.

If you're not familiar with Python and asynchronous programming, **please come to a section** for an introduction. The TAs are happy to walk you through the starter code and explain anything that's unclear.

Troubleshooting

Exception: tokens.json not found!

If you're seeing this error, it probably means that your terminal is not open in the right folder. Make sure that it is open inside the folder that contains bot.py and tokens.json. You can check this by typing in `ls` and verifying that the output looks something like this:

```
# ls
bot.py     tokens.json
```

SSL: CERTIFICATE_VERIFY_FAILED error

Discord has a slight incompatibility with Python3 on Mac. To solve this, navigate to your `/Applications/Python 3.6/` folder and double click the `Install Certificates.command`. Try running the bot again; it should be able to connect now.

If you're still having trouble, try running a different version of Python (i.e. use the command python3.7 or python3.8) instead. If that doesn't work, come to section and we'll be happy to help!

intents has no attribute message_content error

This is an issue with the version of Discord API that is installed. Try the following steps:

1. running `pip install --upgrade discord` in the terminal in your folder in the project that contains this file
2. IF that does not work, try changing the line in bot.py that says `intents.message_content = True` to `intents.messages = True`

Milestone 3: Moderation System Implementation and Evaluation and Poster

Percent of Final Grade: 30%

Deliverables and due date:

- Poster, presented in person, on Wednesday, June 7 from 6-8pm. Your poster should be completely set up by 5:30pm. You can arrive as early as 5:00pm. You should have a video showing your bot's functionality that you can play for judges. We recommend having this video on a tablet, as the space lacks tables and outlets.
- A PDF of your final poster and all code files are due on Tuesday, June 6 at 11:59pm PT.

Submission*:* Please have **one** group member upload the poster PDF and video to the CS 152 Canvas site. Your code files should be in the Github repository submitted for Milestone 2.

Please fill out the work distribution survey to assess how equally different members of each group contributed to the project. We reserve the right to change grades based on the results.

Part 1: Building Automated Abuse Detection

Your bot will be responsible for handling both manual reports from users (which you implemented in Milestone 2) as well as automatically detecting and flagging abusive content (the primary goal for this Milestone). This will include finding or collecting a dataset of examples to use to evaluate the efficacy of your solutions.

Here are some examples of what we envision you accomplishing for this milestone. Please note that this is not a checklist of things you must accomplish, just ideas.

- Training a classifier
- Using your collected dataset to test a few publicly available packages or APIs and noting their pros/cons
    - We have provided instructions on how to use a few different APIs at the bottom of this document
- Designing and building a robust backend for logging and maintaining per-user statistics
- Building a framework by which communities can specify their own regex-like rules for content they don't want to see
- Creating a tool that automatically detects and visits all links in a piece of text to see if they host undesirable content
- Building a system by which users can outsource unwanted content to their friends for review

To handle multiple languages, you can use packages like google_trans_new to automatically translate everything to English or langdetect for language detection (make sure to rate limit yourself where necessary).

Part 2: Evaluation

You should develop and implement a strategy for evaluating the efficacy of your back-end solution. If you trained a classifier or utilized external APIs, this might look like utilizing your dataset to generate a confusion matrix and figuring out whether your model is over or under sensitive and what kinds of problems this might cause at scale. If your solution is more user or design focused, you could conduct further user studies; for instance, you could invite friends to interact with the bot to assess the additional functionality and identify cases in which your design is clunky or might not scale well. What scenarios does your bot handle effectively? What scenarios does your bot not handle as effectively? What explains your bot's strengths and weaknesses? With more time and resources, what would be some of your next steps?

Part 3: Poster

A key final deliverable for this milestone will be a poster that you'll display and discuss with your platform's "executives" (the teaching team as well as guest industry judges who will stop by your poster during the poster session). You will likely want to explain how your platform currently handles your specific type of abuse (your back-end solution), and address its strengths and shortcomings, leading to clear and specific recommendations for how the platform should move forward. You will also want to answer questions from the "executives." We encourage you all to think creatively about how to communicate your work! We encourage your group to have a prepared 5 minute pitch, and to make sure all group members who are present have the chance to participate in the pitch.

Your poster should include:

- Problem Description
- Policy Language
- Technical Back-end
- Evaluation
- Looking Forward

More details on these components are below.

Problem Description

Give a short description of your group's abuse type and victim profile. You can assume that people viewing your poster have a general awareness of your abuse type.

Policy Language

Create a written policy in the kind of language you have seen from the community standards and terms of service you have seen that is specifically targeted at your abuse type. The policy should be less than 400 words and understandable by a normal user.

Technical Back-end

Discuss the original goals and final state of your back-end technology in more detail, explaining the work you did to build it and what its current capabilities are. If there are things you tried which didn't make it into the final product, be sure to mention them here, along with the reasoning behind not including them.

Evaluation

Provide a clear analysis of how well your group's back-end technology accomplishes what it originally set out to do. Make sure to address both the successes and the shortcomings of your current solution. Discuss any negative unintended consequences you foresee and which users may be more affected by them. Try to think about what critics/stakeholders would say about your technology.

Looking Forward

Discuss what impact you believe implementation would have on platform safety. Discuss other engineering approaches that your group didn't pursue but that you'd want to propose going forward.

Practical Details

You can print your poster however you'd like. The poster session will have corkboards where you can pin up your poster without a rigid backing. We recommend that your poster be 24"x36". It should not be larger than that, as we have limited space.

Stanford Undergraduate Research has tips for creating good posters here. **We encourage you to avoid having a lot of text on your poster; hit the points you need to make, but keep the poster readable.**

Pre-recorded Demo

You are going to want to show off your excellent reporting system to our judges, friends and family! Please record a short demo video that you can show to judges and narrate.

Division of Labor

We encourage the political science student to provide strong support for the evaluation component of this milestone, for example by leading a rigorous qualitative evaluation of the bot to supplement a quantitative evaluation. The political science student could generate a typology of abuse manifestations, and evaluate how the bot performs on them. The political science student could recruit testers and do a deep dive into why the bot did or did not respond appropriately to the testers. These are just ideas, think creatively about this! We also encourage the political science student to lead the "policy language" section. The political science student could also assist with dataset creation.