CS4530 Final Project: Safe Learning Environment
Group 414: Neil, Kevin, Ann, and Anusha

Our Feature: Moderation and Chat Enforcement

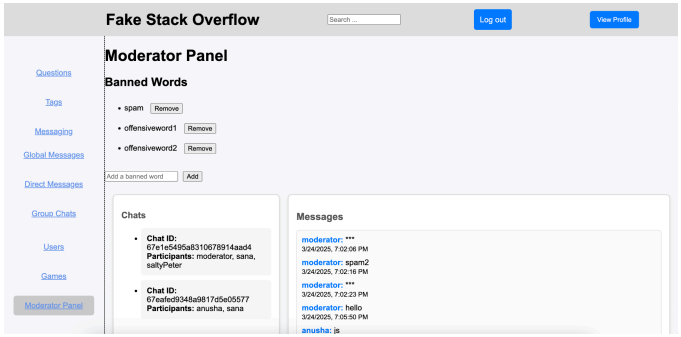
We wanted to focus on creating a safe environment for our users to ask questions and focus on their learning. We began by extending chat functionality beyond just direct messages and allowing our users to be able to create group chats and message multiple users at once. We also allowed users to assign names for these chats so that it would be easy to differentiate a “chemistry” group chat from a “biology” group chat.

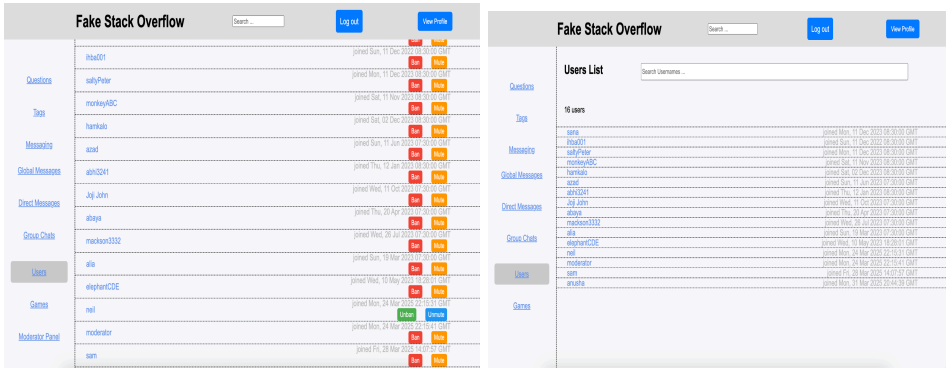
We also wanted to make sure that the content of our site remained appropriate for students and kept the conversation focused on academic topics. In order to do so we focused on creating functionality for moderators to be able to make sure that they had abilities that standard users did not. In the image on the right you can see some of the capabilities that users had including muting and banning users from the users list.

We also added a moderator panel that only logged in moderators can view. Through this panel moderators are able to add and remove banned words as well as view all chat logs in order to keep an eye on chats and figure out which users are potentially violating Fake Stack Overflow rules.

Demo and Source

Here are the links to our client site, backend server, and source code on GitHub: [Client](), [Server](), [Source Code]()





Logged in as a moderator vs logged in as a standard user

Technology Stack & Design

We were able to take advantage of the existing starter code given to us when it came to building certain features in our project. We utilized the existing chat service that was used for direct messages and used it for group chats as well. There was already an existing navigation bar so creating a new page for the moderator panel wasn’t too difficult. A lot of the work required thoroughly testing our backend routes and controllers to ensure that they were fetching the correct information as intended. A key part of the design was also determining whether or not to show the user the moderator aspects and we solved this problem by utilizing localStorage to determine who the current logged in user was.

We tested our frontend work very thoroughly utilizing the Google Chrome Developer Tools Window to ensure that there was no unexpected behavior occurring regarding communication between our client and server. Our pipeline included an automated testing suite as well to help us be even more confident in our code.

Future Work

In the future we would love to dive into more features expanding on what a user can achieve within a chat. Whether that involves sending code blocks, highlighting messages, etc, there are plenty of possibilities that we would explore given more time!