

Enhancing the Security of Yioop Discussion Board

A Project

Presented to

The Faculty of the Department of Computer Science

San José State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

by

Prajna Gururaj Puranik

May 2023

© 2023

Prajna Gururaj Puranik

ALL RIGHTS RESERVED

The Designated Project Committee Approves the Project Titled

Enhancing the Security of Yioop Discussion Board

by

Prajna Gururaj Puranik

APPROVED FOR THE DEPARTMENT OF COMPUTER SCIENCE

SAN JOSÉ STATE UNIVERSITY

May 2023

Dr. Chris Pollett Department of Computer Science

Dr Nada Attar Department of Computer Science

Ms Namrata Bilurkar Vidyard

ABSTRACT

Enhancing the Security of Yioop Discussion Board

by Prajna Gururaj Puranik

Yioop is an open-source web portal that serves as a search engine and a discussion board, enabling users to create, join, and share content within groups. Data security is a critical concern for Yioop, as it involves storing and accessing user-generated data and generating statistical data. Yioop has an existing security mechanism in place, but continuous enhancements are needed to protect against potential vulnerabilities and cyber threats.

This project aims to strengthen the security of Yioop by implementing additional security measures that build upon the existing security mechanism. To prevent statistical attacks, this project extends differential privacy to mask the number of users in groups. Furthermore, a flag feature is added to allow users to flag posts that they find offensive, which is reviewed by a newly added moderator group. Secret sharing is employed to further fortify the encryption keys, ensuring that only authorized users with the required shares can potentially use the key. These security measures have been rigorously tested and evaluated to ensure that they effectively contribute to the overall security of Yioop, enabling users to enjoy secure interactions and content sharing within the portal while preserving their privacy and confidentiality.

This report provides an in-depth overview of these security measures, their implementation, testing procedures, and their impact on Yioop's overall security. The success of this project will contribute to a more secure environment for Yioop users, preserving user privacy and confidentiality while promoting secure interactions and content sharing within the portal.

ACKNOWLEDGMENTS

I would like to express my deepest gratitude to my project advisor, Dr. Chris Pollett, for his unwavering support, guidance, and invaluable insights throughout my master's journey. His mentorship and encouragement have been instrumental in shaping my growth. I would also like to thank my committee members, Dr. Nada Attar and Ms. Namrata Bilurkar, for their time and support. To my friends and family, thank you for being my pillars of support during this challenging but rewarding journey. Your love, encouragement, and understanding have been a source of motivation and inspiration.

TABLE OF CONTENTS

CHAPTER

1	Introduction	1
2	Background	5
2.1	Differential Privacy	5
2.2	Secret Sharing	7
2.3	Flagging	8
2.4	Moderation	9
2.5	Encrypted Groups in Yioop	10
3	Design	11
3.1	Yioop discussion board	11
3.2	Model View Controller	12
3.3	Subset of Yioop architecture	12
4	Implementation	14
4.1	Extending differential privacy	14
4.2	Flag functionality	15
4.3	Moderation functionality	17
4.4	Secret Sharing functionality	20
5	Testing	24
5.1	Differential Privacy Functional Testing	24
5.2	Flag and Moderation Group	25
5.2.1	Functional Testing	25

5.2.2	Data Verification Testing	27
5.3	Secret Sharing	28
5.3.1	Functional Testing	28
5.3.2	Data Verification Testing	28
5.4	System and Response Time Testing	29
6	Conclusion	32
	LIST OF REFERENCES	33
	APPENDIX	
	Testing images	35

CHAPTER 1

Introduction

Data security is of critical importance in the current age of information. It is important to protect data against threats like identity theft, data tampering, etc. Web security is crucial for preventing hackers from gaining access to data. Without a proactive security policy, web services run the risk of attacks and possible data leakage. This project focuses on enhancing the security measures of one such system – Yioop.

Yioop is an open-source web portal that functions as a search engine, a wiki, and a discussion board. One of the important features of the Yioop discussion board is the group feature provided to users which allows users to create and join groups. Users can customize their groups, restrict access, and allow members to read or start discussions with other users of their group. These capabilities do, however, have the potential to pose security problems, especially with regard to data privacy. Given the potential severity of data breaches and other malicious activities, Yioop places a high priority on preventing unauthorized access to user data.

To better ensure the privacy and security of Yioop users, it is important to add additional measures beyond the current security features of Yioop such as group data encryption and masked user statistics. This project implements these additional measures to provide an even higher level of protection and enhance the overall security of the platform.

In this project, we implement features such as group data encryption, masked user statistics, secret sharing, and content moderation to provide an even higher level of security and privacy for Yioop users. We also extend differential privacy to mask the number of users in groups, preventing potential statistical attacks. By incorporating these additional security measures, Yioop aims to provide a robust and

secure platform for users to interact, collaborate, and share content while preserving user privacy and confidentiality.

Pragya Rana's implementation of differential privacy in Yioop [1] has been an important step in securing user privacy. Currently, this technique is used in Yioop to mask group and query statistics such as the number of views in a group or the search queries entered by a user. This project further extends differential privacy to mask the number of users in groups, which will prevent potential statistical attacks.

Yioop allows the creation of encrypted groups that hide the title and description of the post. But the protection of the encryption keys is also crucial. This project uses secret sharing, a method of distributing a secret among multiple users to enhance the security of storing encryption keys. This ensures that only authorized users with the required shares can potentially use the encryption key, improving the overall security of the system.

In addition, full-stack features like flagging and the creation of a moderation group are implemented in this project. These features will provide users with the ability to flag harmful content and allow moderators to review such posts and take appropriate action.

The successful implementation of these security measures significantly enhances the security of Yioop, protecting user data from unauthorized access, preventing statistical attacks, and ensuring secure communication within groups. By incorporating these additional security measures, Yioop aims to provide a robust and secure platform for users to interact, collaborate, and share content while preserving user privacy and confidentiality.

The security features implemented in this project were decided based on observations of many websites and platforms that handle sensitive user data, including social media platforms. These systems implement techniques like content moderation,

encryption, differential privacy, and secret sharing to create a secure environment for users to interact, and share content while reducing potential security risks. Many websites and platforms that handle sensitive user data, including social media platforms, utilize security measures implemented in this project to protect user information and enhance overall security. For example, platforms like Facebook, Twitter, and Reddit implement content moderation to safeguard user data and prevent the spread of harmful content [2]

Implementing content moderation has been shown to have a significant impact on reducing harmful content and improving overall security in online platforms [2]. In this project, content moderation has been implemented in the form of flagging and adding a moderation group to approve or ban flagged posts. Lanius, Weber, and Mackenzie Jr, the authors of [3] conducted a survey to investigate the attitudes and behaviors of social media users towards using a flag feature to flag and remove misinformation. They found that the majority of respondents (over 60%) supported the use of flags to identify and remove misinformation and that these measures were seen as effective in limiting the spread of false information. As for content moderation, consider the example of Reddit banning several subreddits due to harassment policy violations in 2015 [4]. A study conducted on the banned subreddits [4] found that banning posts as a moderation approach saw an n 80% decrease in hate speech usage. In this project, when a post is banned, it will no longer be visible to other users on the platform, meaning that it will be removed from public view.

The California Consumer Privacy Act (CCPA) [5] and the General Data Protection Regulation (GDPR) [6] have had a significant impact on how data privacy is viewed online. These rules have compelled businesses to be more open about how they gather user data and to give users greater control over it, which has resulted in the adoption of better privacy measures on the internet [5] [6]. The development of

the security measures used in this project was significantly influenced by these laws.

According to [5] and [6], the GDPR and CCPA regulations both require companies to take measures to protect personal data collected from users and allow users to control their personal data. To comply with these regulations, this project has implemented features like differential privacy and secret sharing to keep sensitive user data safe and private. In addition, flagging and moderation features have been added to Yioop to ensure compliance with GDPR and CCPA. These features give users the ability to flag content containing their personal data, while the moderation group ensures that flagged content is handled properly in compliance with GDPR and CCPA regulations. Together, these features help Yioop maintain a secure and compliant platform for its users.

In summary, with the implementation of security measures like differential privacy, secret sharing, flagging, and moderator groups, this project seeks to enhance the security of Yioop. These methods are designed to protect user data from unauthorized access, secure group communication, and guard against statistical attacks. The overall objective is to create a stable and safe environment for Yioop users, offering a higher level of user privacy and data protection.

The remainder of the report is divided into four main sections. The next section covers the background information about the different privacy mechanisms implemented in this project. The preliminary work section covers the details of work done in the first half of the project to understand the Yioop codebase and the encryption techniques to be used. The design section gives an overview of Yioop and the architecture of Yioop system while the implementation section describes in detail the functionalities implemented to enhance the security. A concluding section wraps up the report.

CHAPTER 2

Background

This section provides an overview of the security enhancement implemented in this project, including an extension of differential privacy and the addition of content moderation, flagging, and secret sharing.

2.1 Differential Privacy

Differential privacy is a mathematical framework for protecting users' privacy in datasets [7]. By allowing data to be evaluated without disclosing private information about any of the dataset's individuals, it can offer a solid assurance of privacy. According to Dwork [8], differential privacy works by adding a tiny amount of random noise to the data in an effort to obscure any personal information. This ensures that sensitive information, such as medical records or financial data, is kept private and secure.

Yioop employs the concept of ϵ -differential privacy as a means of achieving differential privacy. As given by Dwork in [8] ϵ -differential privacy involves introducing random noise to a dataset while maintaining its statistical features and safeguarding user privacy. The existing differential privacy in Yioop has been developed with contributions from [1]. This technique is used to hide group and query statistics, such as the number of views in a group or the search queries entered by a user. An extension of differential privacy is covered in this project and it is used to conceal the number of users in a group, as detailed in the next section. This helps in preventing any potential privacy breaches.

To understand ϵ -differential privacy, consider a database with the ages of 10 individuals in it. We wish to determine the database's average age. The actual average age is 35, but since we want to protect the privacy of everyone in the database,

we're just interested in computing an approximate response that doesn't compromise anyone's privacy. We can employ ϵ -differential privacy to accomplish this. The privacy guarantee can be met by adding random noise to the genuine average age. The value of ϵ that we select affects how much noise we add. Now let's assume that we choose ϵ to be 1. This means that when a single person's age is added to the database or removed from it, we want to introduce enough noise into the system to ensure that the output of the query is not altered by more than a factor of e (about 2.718) in either direction. To calculate the noisy average age, we first generate a random number between -1 and 1. Let's say we generate a random number of 0.5. We then add this number to the true average age of 35 to get the noisy average age of 35.5. Now, suppose an attacker gains access to the noisy average age of 35.5. They are aware that the actual average age is close to 35, but they are unsure of the precise ages of the people in the database. The amount of noise we added to the query makes the output questionable, so even if they add or remove a person from the database, they won't be able to estimate their age with high certainty. This is how ϵ -differential privacy safeguards user privacy in a database while still enabling efficient statistical queries.

Mathematically, Dwork [8] gives the definition of ϵ -differential privacy as a randomized function K gives ϵ -differential privacy if for all data sets $D1$ and $D2$ differing on at most one element, and all $S \subseteq \text{Range}(K)$:

$$Pr[K(D1) \in S] \leq \exp(\epsilon) \times Pr[K(D2) \in S]$$

where ϵ is a positive real number and S is a subset of image K , which is a set of all output values that it might produce. [8] posits that any mechanism K addresses concerns that any participant may have regarding the leakage of their personal information. In other words, even if a participant were to remove their data

from the dataset, no outputs would become significantly more or less likely as a result.

According to [8], the mechanism operates by introducing carefully selected random noise to the answer $a = f(X)$, where f represents the query function and X is the database. The extent of the random noise is determined based on the most substantial alteration a solitary participant could make to the output of the query function. This is defined as the sensitivity of the function and can be represented as:

$$\Delta f = \max_{D, D'} \|f(D) - f(D')\|_1$$

where Δf represents the sensitivity of the function f , D and D' are two datasets that differ by one record.

2.2 Secret Sharing

Secret sharing is a method of sharing a secret among a group of participants in a way that no individual can deduce the secret by themselves. Shamir [9] introduced (k, n) threshold scheme of dividing the secret into n shares such that knowledge of any k or more shares makes the secret easily computable, while knowledge of any $k-1$ or fewer shares leaves the secret completely indeterminate.

As described in [9], the (k, n) threshold scheme divides the input data D into n pieces D_1, D_2, \dots, D_n such that:

- Knowledge of any k or more D_i pieces makes D easily computable
- Knowledge of any $k - 1$ or fewer D_i pieces leaves D completely undetermined, in the sense that all its possible values are equally likely

Shamir [9] defines secret sharing as particularly helpful for managing cryptographic keys. To protect encryption keys, the (n, k) threshold approach is the best option. In this project, secret sharing is used to restrict access to the encryption key.

Shamir [9] uses polynomial interpolation to implement secret sharing. A polyno-

mial of degree $k-1$ is chosen, and the secret is encoded as a coefficient of the polynomial. The polynomial is then evaluated to determine the shares of the secret, resulting in n shares of the data. Given any subset of k of these shares, the coefficients of the polynomial can be found by interpolation, and the secret can be reconstructed. The steps are defined in [9] as follows:

- Select a random $k-1$ degree polynomial such that $q(x) = a_0 + a_1(x) + a_2(x^2) + \dots + a_{k-1}(x_{k-1})$ in which $a_0 = D$
- Evaluate: $D_1 = q_1 \dots D_i = q_i \dots D_n = q_n$

Given any subset of k of these D_i values, we can find the coefficients of $q(x)$ by interpolation, and then evaluate $D = q(0)$. Knowledge of just $k - 1$ of these values does not suffice to calculate D .

Linear secret sharing is a special case of polynomial secret sharing where the polynomial has a degree of 1. Instead of a polynomial, a line is used to distribute the secret. This project uses a linear secret-sharing concept. Linear secret sharing has the advantage of being easier and quicker to construct than polynomial secret sharing because it just requires the most fundamental arithmetic operations.

2.3 Flagging

Flagging is a process of marking content that violates community guidelines as inappropriate or offensive. Users can report such content using the flagging feature, which is then reviewed by a moderator to determine if it violates the community standards or terms of service. If it does, appropriate action is taken.

Incorporating a flagging feature in Yioop can enhance security in several ways. It enables early detection of harmful or inappropriate content, making it possible to take prompt action against it. Additionally, proactive monitoring and moderation of flagged posts by designated moderators or administrators can help maintain compliance with

policies and deter potential malicious activities.

According to YouTube [10], flagging has helped in detecting unsavory videos, with 92 million videos being removed in 2015 alone. This demonstrates the effectiveness of flagging in promoting a safe online environment. By implementing a flagging feature, Yioop can increase user trust and confidence in the platform's commitment to maintaining a secure online space.

2.4 Moderation

The moderation group functionality refers to the ability of online platforms and applications to establish a group of moderators who are authorized to monitor and moderate user-generated content. This feature is frequently used to promote a safe atmosphere for members in online communities, forums, and social media platforms. A moderator group offers a structured method of content moderation and hence boosts security. The online platform can use moderator groups to make sure that information is constantly and properly examined for potential security threats [11]

According to [12], content moderation has been around since the early days of social media, with early adopters such as MySpace recognizing the need for a professional moderation staff. As sites like Facebook grew in popularity, the idea that user-generated content could fuel engagement also took off. However, this led to a wide range of content being uploaded, including inappropriate, disturbing, or illegal material that could pose problems for companies concerned about liability or brand management. Initially, companies used a patchwork approach to moderation, relying on a combination of in-house staff and third-party contractors. At present, every social media platform has its distinct and intricate set of guidelines, which are continuously evolving and improving, along with a designated team of moderators who scrutinize the content. The need for effective content moderation has become

increasingly important, as online communities have become a critical part of modern communication and information sharing.

In this project, a moderation group acts as a centralized platform to display flagged content to moderators for necessary action. Moderators, who are added to the group by the administrator, can approve or delete flagged content. Each flagged item appears as a separate thread within the group, enabling moderators to comment on threads, discuss flagged content, and share insights with one another. This collaborative approach can aid in identifying patterns, trends, or potential issues that may necessitate further attention or action. Such a collaborative effort can lead to swift and effective handling of flagged content, boosting the overall security posture of the platform. The moderation group also enables moderators to refer to the original thread for context regarding the flagged content, allowing them to make informed decisions.

2.5 Encrypted Groups in Yioop

Another security mechanism implemented in Yioop as given in [1] is the creation of encrypted groups. This feature ensures that all posts within an encrypted group are encrypted before being stored in the database. When displaying posts from an encrypted group, a key stored in an external database is accessed first to decrypt the data before it is displayed on the discussion board. But the key can be accessed by anyone with the ID of the group and hence it may be vulnerable to attacks. To address this security issue, this project utilizes secret sharing to enhance the security of the encryption key.

CHAPTER 3

Design

The purpose of this section is to provide an overview of the Yioop discussion board's operation and the architecture of the Yioop codebase. The aim is to offer a better understanding of the data flow within the Yioop codebase, which will aid in the understanding of the implementation details outlined in the subsequent section.

3.1 Yioop discussion board

The discussion board is a key component of Yioop, giving users a place to converse, exchange knowledge, and work together on a range of interesting topics. Users can create groups, invite other users to their groups, control the visibility of the content in groups, and so on. Users can publish messages, start new threads, and communicate with one another by leaving comments on the discussion board. Users can edit and delete their own posts, delete their groups, and ask to join other groups.

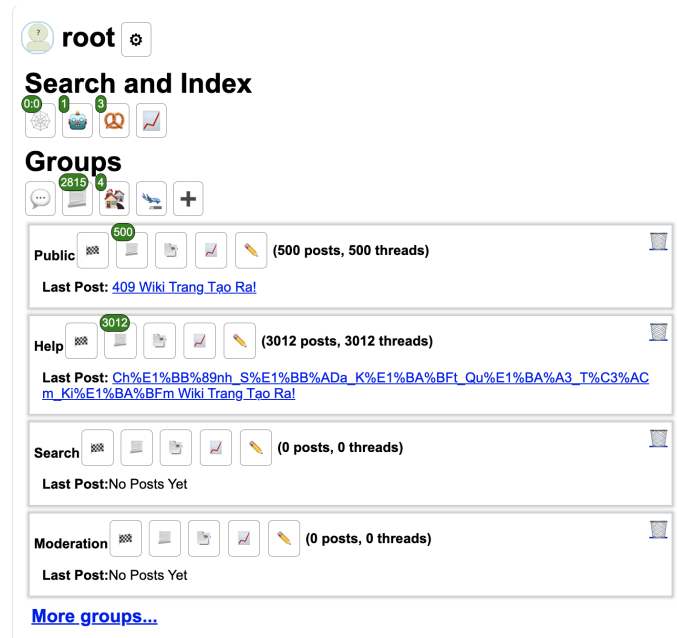


Figure 1: User homepage

3.2 Model View Controller

Yioop makes use of the Model-View-Controller software design paradigm. In this architecture, as described in [13], an application is divided into three main components: the Model, which manages data and business logic, the View, which serves as the user interface, and the Controller, which processes user input and modifies the Model and View. By promoting the division of labor and enabling independent changes to various components, the MVC architecture enhances the maintainability, modularity, and flexibility of software development. Data management, presentation logic, and user input processing are well separated, which promotes code reuse, scalability, and testability for software applications.

3.3 Subset of Yioop architecture

This project deals with the discussion board component of Yioop and hence involves a set of specific components. The system architecture for this project can be illustrated in Figure 2:

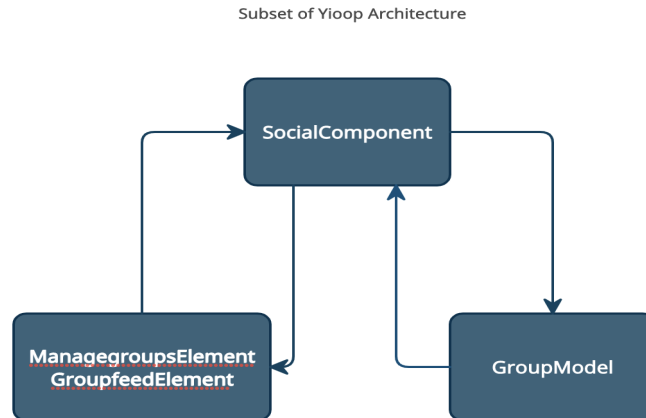


Figure 2: Project’s subset of Yioop components

Apart from configuration, setup and helper files, the major classes modified in this project are represented in a diagram in Figure 3.

ManagegroupsElement and GroupfeedElement are View classes that render the

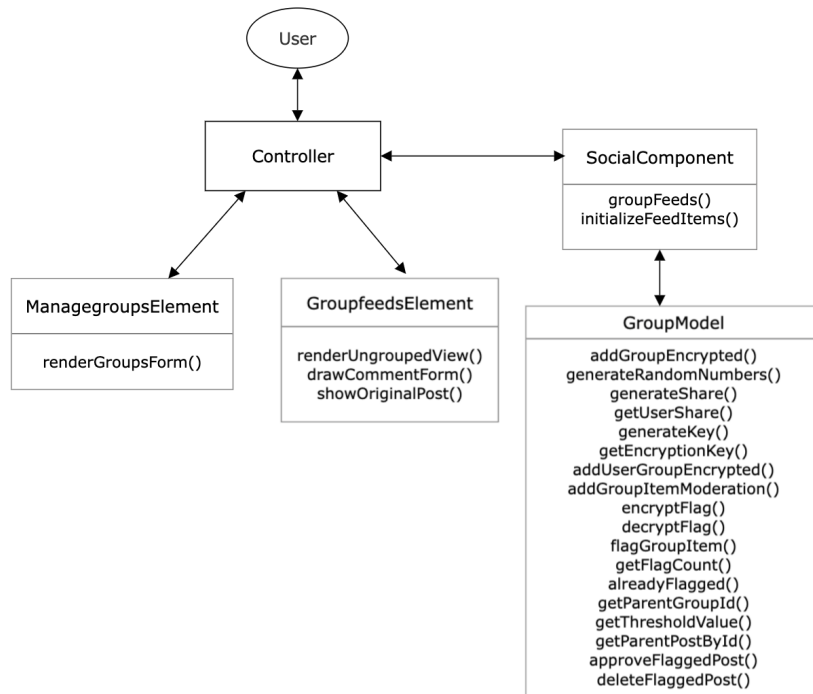


Figure 3: Diagram of the subset of Yioop components

display seen by the users. All the user interface changes implemented in this project belong to either of the two files: GroupfeedElement for flagging and moderation feature implementation while ManagegroupsElement is tweaked to support the secret sharing implementation. SocialComponent contains all the controller code changes of the project. It contains the method groupFeeds() which supports activities like posting, editing, modifying, and deleting group feed items. that while initializeFeedItems() is used to compute a list of feed items to be displayed when a user logs in. The functions added and modified in GroupModel are numerous and are used to implement the features detailed in the next section.

CHAPTER 4

Implementation

This section details the implementation details of features added to increase the security of Yioop.

4.1 Extending differential privacy

The existing implementation of Differential Privacy in Yioop provides the functionality to hide user statistics and query statistics. This project enhances the privacy of user data by extending the use of Differential Privacy to mask the number of users in a group. By utilizing the appropriate function to apply Differential Privacy, the project ensures that the number of users in a group remains private and secure.

There are three instances in the user interface of Yioop where the group user count is displayed:

- The root user can see the group count in the Edit Group section
- The root user can see the group count in the edit members section of Edit group
- A general user can see the group user count when accessing information about the group



Edit Group [?](#)

Name:

Owner:

Register:

Access:

Voting:

Post Lifetime:

Encryption:

Feed:

Members:

Figure 4: Root user: Edit Group

Exposing the total number of users in a group to all users can lead to the disclosure of sensitive information about the group and its members. For instance, if the user base is small, the group may become more susceptible to harassment or attacks. Therefore, hiding the user count can be beneficial in maintaining the security and privacy of the group and its members.

4.2 Flag functionality

Users are provided with an option to flag posts of a group that they are a part of. A user can flag all posts except the post on the first page of a group which displays the titles of the threads. The flag option is displayed to all users who are part of the group and have access to comment.

Each user is only permitted to flag a post once in order to ensure fairness. We stop a user from flagging a post repeatedly to manipulate the system. The system will not permit a user to flag a post again after they have previously done so and an error message will appear. This also helps to maintain the integrity of the flagging system and ensures that flagged posts are reviewed by moderators only when they have been flagged by multiple users, indicating a potential issue with the post.

To enable users to flag inappropriate content on the discussion board, this project implements a simple yet effective flagging mechanism. Once a user flags a post, a dialog box appears to confirm their choice, and they can only flag each post once. The post is only sent to the moderator for approval when it reaches a threshold number of flags.

The threshold for flagging a post is a crucial variable that can influence the balance between enabling user-generated content and preserving the quality of the discussion board. In this project, the threshold value was chosen depending on the size of the group, and the required level of moderation, among other things. This threshold



Figure 5: Flag Button

value can be changed by the developer to strike the right balance and guarantee a satisfying user experience.

To implement the flagging feature, a button is added to the view, and a new column is added to the database table to store the number of flags associated with each post. When a post is flagged, the flag count is incremented and compared to the threshold value. If the threshold is met, the post is added to the moderation group, which is explained in detail in the following section.

To ensure the privacy of flagged posts in encrypted groups, the flag values are encrypted using Advanced Encryption Standard (AES) in ECB mode. Initially, the flag count value is encrypted with AES to hide an initial flag value of 0. When a user flags a post, the encrypted flag value is fetched from the database and AES is applied to it again using the same key associated with the group. This process is repeated until the encrypted flag count value becomes equal to the encrypted threshold value. Once the flag count equals the threshold, the post is sent to the moderation group for

review as described in the following section.

As described in [14], the Electronic Codebook (ECB) mode of AES is a simple block cipher mode of operation that divides the input data into fixed-size blocks and independently encrypts each block with the same key. This approach is straightforward and computationally efficient, making it an attractive option. Using the Advanced Encryption Standard (AES) in ECB mode to encrypt the flag count value in Yioop provides a secure way to protect the flag values of encrypted groups. By applying AES encryption repeatedly until the encrypted flag value equals the threshold value, Yioop ensures that flagged posts in encrypted groups are treated with the same level of scrutiny as those in unencrypted groups.

4.3 Moderation functionality

The moderation group serves as a dedicated space where moderators can review and take action on flagged posts. The root user has the authority to designate any valid user as a moderator by adding them as members to the moderation group. Once a post is flagged, it appears as a separate thread within the moderation group, allowing moderators to easily identify and review the post in question.

Within the moderation group, moderators have the ability to approve or delete flagged posts, as well as leave comments and engage in discussion with other moderators. Additionally, moderators can browse the thread containing the flagged post for context to help inform their decision-making process. This streamlined approach to post moderation helps maintain the quality and integrity of user-generated content, promoting a positive and productive user experience.

To enable moderation of flagged posts, the database structure needs to be modified. All group posts are stored in the `GROUP_ITEM` table, which contains information such as the post's `TITLE`, `DESCRIPTION`, `USER_ID` of the creator of the post, and

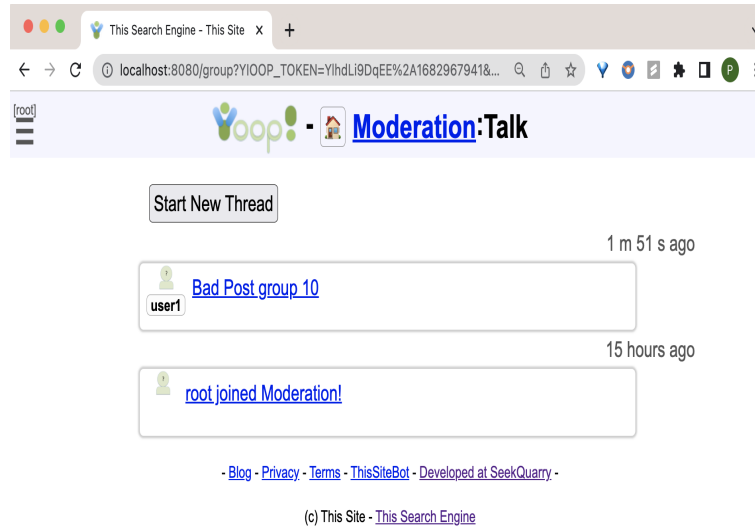


Figure 6: Moderation Group Home

GROUP_ID of the group that the post is made in. When a post is flagged and the flag count reaches the predetermined threshold value, it is added to the GROUP_ITEM table with GROUP_ID = 5, which represents the moderation group. If the flagged post is from an encrypted thread, it is decrypted and added to the table for moderators to review.

The implementation of the flag feature requires adding a FLAG column to this table to keep track of the number of times a post has been flagged. A new column called PARENT_ITEM_ID is also added to this table to keep track of the original thread that has been flagged. This helps in implementing moderator actions like approve and delete.

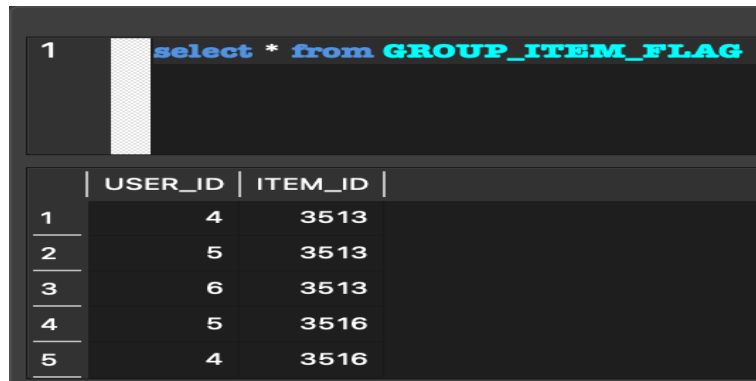
```
1 select * from GROUP_ITEM
```

ID	PARENT_ID	GROUP_ID	USER_ID	URL	TITLE	DESCRIPTION	PUBDATE	EDIT_DATE	UPS	DOWNS	FLAG	PARENT_ITEM_ID	TYPE
3519	35...	3520	5	3	-- New	flag this post	1682967830	1682967830	0	0	3	3516	0

Figure 7: FLAG and PARENT_ITEM_ID columns added to the table

When a moderator approves a flagged post, the flag count in the original post's database column is set to 0 to indicate that the post has been reviewed and no action is needed. However, in the moderation group, the post is not deleted but marked as resolved by setting its flag column value in the database to -1. This helps moderators to keep track of resolved posts and focus on unresolved flagged posts that still require review. Similarly, the flag column value of the flagged post is also set to -1 in the moderation group database to indicate that the post has been deleted. Additionally, the original post's description is modified to "This post has been flagged", replacing the original flagged content. This helps users understand why the post was deleted and encourages them to follow the community guidelines to avoid posting inappropriate content in the future. These functionalities require the use of the original thread ID stored in the PARENT_ITEM_ID column.

To ensure that users are unable to flag the same post multiple times, a new table called GROUP_ITEM_FLAG is created and maintained. This table keeps a record of all users who have flagged a particular post. If a user tries to flag the same post again, the system checks the GROUP_ITEM_FLAG table to confirm that the post has already been flagged by the user.



The image shows a terminal window with a SQL query and its results. The query is `select * from GROUP_ITEM_FLAG`. The results are displayed in a table with three columns: an index, USER_ID, and ITEM_ID.

	USER_ID	ITEM_ID
1	4	3513
2	5	3513
3	6	3513
4	5	3516
5	4	3516

Figure 8: GROUP_ITEM_FLAG table

To help moderators easily identify which flagged posts have been resolved, the

titles of these posts are displayed in green. This way, moderators can quickly see which posts have already been addressed and focus their attention on unresolved issues.



Figure 9: Resolved Threads in Moderation group

When the number of flags on a post reaches the threshold value and the post is sent to the moderation group for review, the system checks if the post has already been flagged and sent for review.

4.4 Secret Sharing functionality

Yioop allows the creation of encrypted groups that encrypt the title and description of posts using the AES encryption technique. A key is generated for each group created and stored in the private database. This key is accessed by supplying the `GROUP_ID`, which is matched to the corresponding key in the private database.

This approach may pose security risks, as providing direct access to the encryption key using just the `GROUP_ID` could compromise the confidentiality and integrity of the data if an attacker gains access to the `GROUP_ID` and retrieves the encryption key. To mitigate this risk, this project ensures that the encryption key is not stored in the database. Instead, linear secret sharing is used to compute the key.

This project uses a group owner's ID to generate the key. Any user who wants to create an encrypted group and add users to their group needs to enter their password.

This password is then used to compute the key, which is shared among the group members using secret sharing. This approach ensures that the encryption key is not stored in the database and provides an extra layer of security for encrypted groups in Yioop.

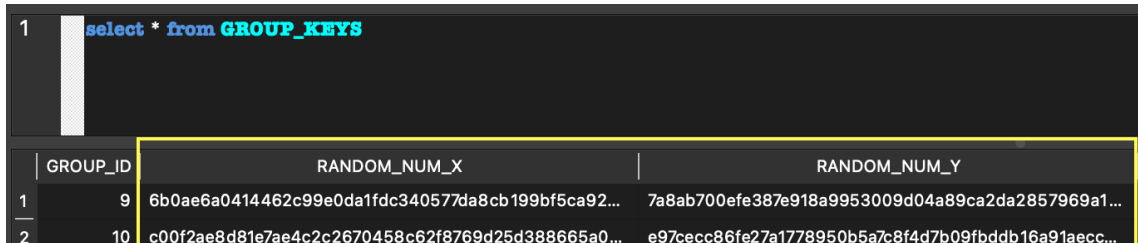
When a user creates an encrypted group by providing their password, a 32-byte random number is assigned to the group and stored in the SOCIAL_GROUP table.



GROUP_ID	GROUP_NAME	CREATED_TIME	OWNER_ID	REGISTER_TYPE	MEMBER_ACCESS	VOTE_ACCESS	POST_LIFETIME	ENCRYPTION	RANDOM_NUM
10	17 FinalTest	1684180444.855903	3	1	4	0	-2	1	d08201c8a8a849475ebfa8e9db5cdd740529d68c25fa6e...

Figure 10: Modified SOCIAL_GROUP table

The USER_GROUPS table is modified to include a column for a 32-byte random number associated with the owner of the group. The private database is modified to add a table GROUP_KEYS which stores the GROUP_ID of the created encrypted group and two other 32-byte random numbers.



GROUP_ID	RANDOM_NUM_X	RANDOM_NUM_Y
9	6b0ae6a0414462c99e0da1fdc340577da8cb199bf5ca92...	7a8ab700efe387e918a9953009d04a89ca2da2857969a1...
10	c00f2ae8d81e7ae4c2c2670458c62f8769d25d388665a0...	e97c6cc86fe27a1778950b5a7c8f4d7b09fbddb16a91aecc...

Figure 11: GROUP_KEYS table

When the owner creates an encrypted group, they provide their password which is hashed with the GROUP_ID and stored in the USER_GROUPS table.

When a new user is added to this group, a line is created using two points: Point 1 from PRIVATE_DB which contains two random numbers for each group, and Point 2 is retrieved from the USER_GROUPS with x-coordinate as the created hash of the owner's password with GROUP_ID and y-coordinate as the random number

```
select * from USER_GROUP
```

USER_ID	GROUP_ID	STATUS	JOIN_DATE	USER_HASH	SHARE
6	17	1	1684277836	85ea151b8c5b5ab0d3349100e441bd4b8dc20740d429c...	61628557933094781165493810943645093584565102...
7	17	1	1684277836	d536a8c1664fec0bc85615cf3cb2645871e8b2935c964...	61628557933094781165493810943645093584565102...

Figure 12: USER_GROUP table

associated with each owner of the group. With the line created and an x-coordinate value as the hash of the added user's USER_ID with the GROUP_ID, a y-coordinate value is generated which is the user's share. This is stored in the USER_GROUPS table.

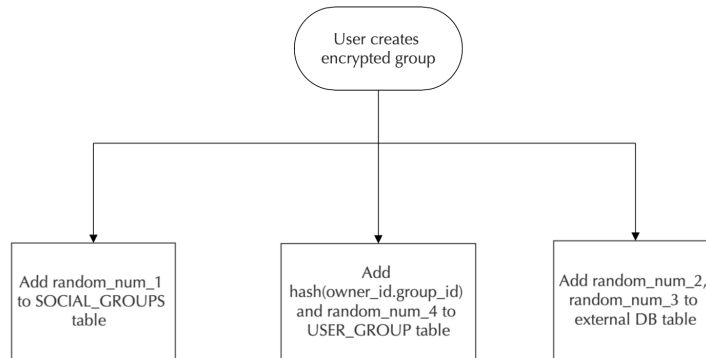


Figure 13: Encrypted group creation

When a user wants to decrypt the encrypted posts and in this regard, wants to fetch the encryption key from the database, their share is fetched from the database and verified by computing $\text{hash}(\text{USER_ID}.\text{GROUP_ID})$. If this user does belong to the group, their share is fetched from the USER_GROUPS table and serves as a point to create a line along with the two random numbers stored in the GROUP_KEYS table. With the line created and using the x-coordinate as the random number stored in the SOCIAL_GROUPS, a Y value is generated, which serves as the encryption key.

This approach provides additional security compared to storing the encryption key directly in the database. In the event of a database breach, an attacker would not be able to retrieve the encryption key and access the encrypted data without also knowing the user's password.

CHAPTER 5

Testing

This section presents the test results of the implemented features as detailed in the previous section.

5.1 Differential Privacy Functional Testing

This project extended the implemented differential privacy in order to hide the number of users in a group. To test the working of the implemented feature, a new group was created with three users including the root user. After implementing differential privacy, the same group was accessed again and the total number of users, three in this case, was hidden and replaced instead by fuzzified values.



Figure 14: Before and after adding differential privacy

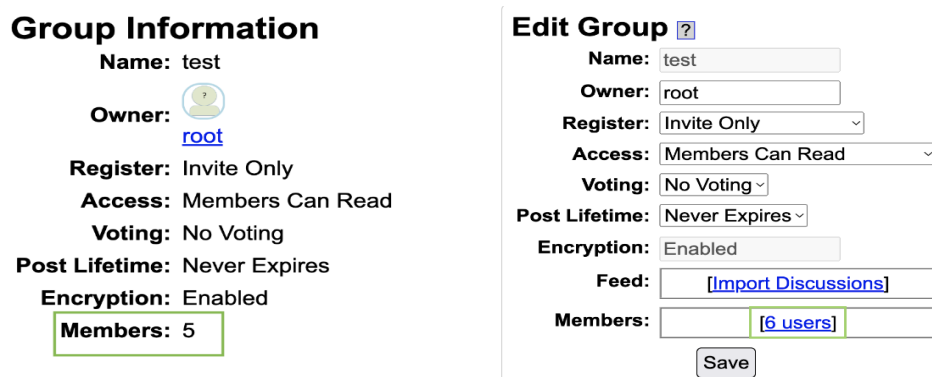


Figure 15: Differential Privacy in action

5.2 Flag and Moderation Group

The implemented flagging feature allows users to flag inappropriate posts made by other members of the group. When a certain threshold of flags is reached, the post is automatically sent for moderation. To aid in this process, a new group was created specifically for moderators. Moderators can review flagged posts, provide comments, and decide whether to approve or delete them after reviewing the context of the original thread.

5.2.1 Functional Testing

To test the flag feature, a new group called *TestModeration* was created with two users. A post was then created in the group by one of the members, let's call them Alice. Another user Bob was made to flag this post using the added flag button. The functionality of this button was tested to ensure it operates correctly, and the appearance of the corresponding dialog box was validated to confirm it meets design specifications. When Bob confirmed the decision to flag the post, an appropriate message was displayed.

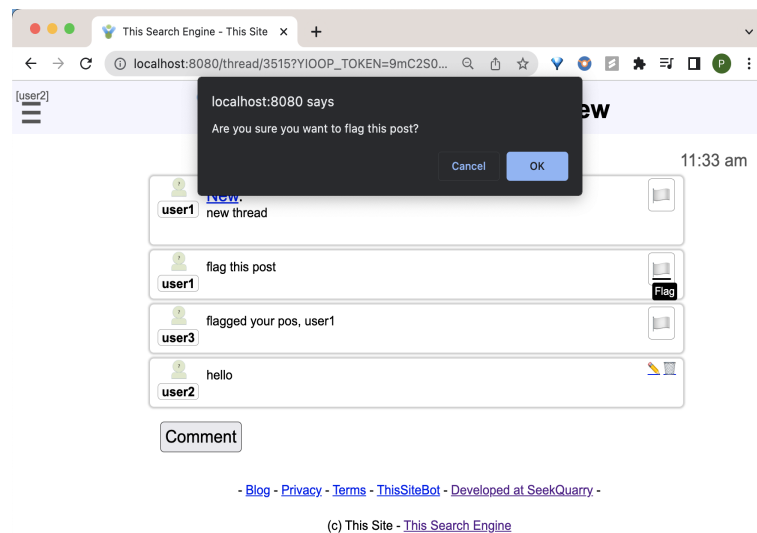


Figure 16: Flag confirmation

Bob then attempted to flag the post again. Bob was prevented from flagging the post again, as he had already flagged it previously. This behavior was verified to ensure that users are not able to flag the same post multiple times.

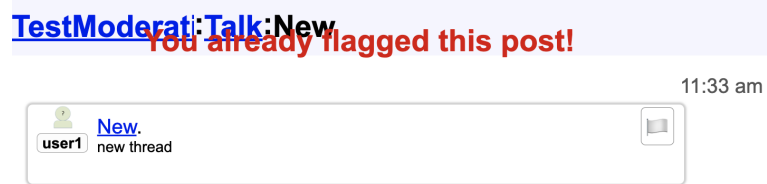


Figure 17: Flag confirmation

To test the moderation group feature, two more users were added to the group *TestModeration* and they were made to flag Alice's post. The threshold value for the group was set to 3, so when the flag count reached 3, the post was sent to the moderation group. In the moderation group, the post appeared as a separate thread as shown in Figure 18.

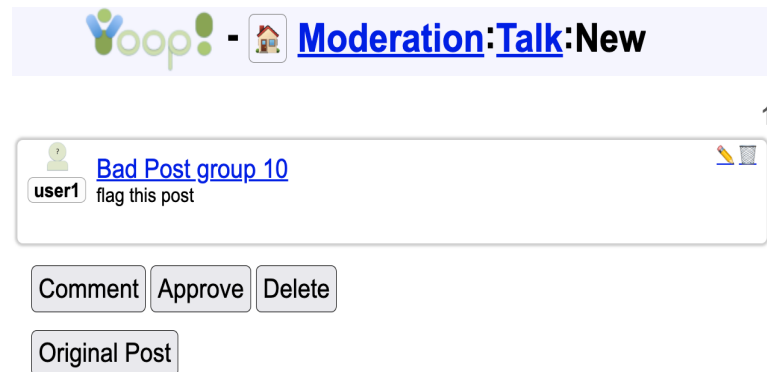


Figure 18: Flagged Post in Moderation Group

If the moderator chooses to delete a flagged post, the flagged post is removed from the database, and the original post's description is modified to say "This post has been flagged" to replace the original flagged content.



Figure 19: Flagged Post deleted by Moderators

5.2.2 Data Verification Testing

This action creates a new entry in the GROUP_ITEMS table with a GROUP_ID of 5, indicating that it belongs to the moderation group. The flag column value is set to 3, indicating that the post has been flagged three times, and the PARENT_ITEM_ID is set to the ID of the original post created by user1. This is illustrated in Figure 20.

```
1 select * from GROUP_ITEM
```

ID	PARENT_ID	GROUP_ID	USER_ID	URL	TITLE	DESCRIPTION	PUBDATE	EDIT_DATE	UPS	DOWN	FLAG	PARENT_ITEM_ID	TYPE
3515	3516	3515	10	3	-- New	flag this post	16829660...	16829660...	0	0	0	0	0
3516	3517	3515	10	5	-- New	flagged your post,user1	1682966061	1682971264	0	0	1	0	0
3517	3518	3515	10	4	-- New	hello	1682966148	1682966148	0	0	0	0	0
3518	3519	3515	10	6	-- New	okay	1682967721	1682967721	0	0	0	0	0
3519	3520	3520	5	3	-- New	flag this post	1682967830	1682967830	0	0	3	3516	0

Figure 20: Flagged Post in Moderation Group

When a flagged post is approved by a moderator, the corresponding database entry in the moderation group is marked as resolved by setting the flag column value to -1. Figure 21 shows the change in FLAG column value when a post is either approved or deleted by a moderator.

```
1 select * from GROUP_ITEM where GROUP_ID = 5
```

ID	PARENT_ID	GROUP_ID	USER_ID	URL	TITLE	DESCRIPTION	PUBDATE	EDIT_DATE	UPS	DOWN	FLAG	PARENT_ITEM_ID	TYPE
1	3520	3520	5	3	-- New	flag this post	1682967830	1682967830	0	0	-1	3516	0
2	3522	3522	5	3	-- New	bad post	1682968792	1682968792	0	0	3	3521	0

Figure 21: Database column value for resolved post

5.3 Secret Sharing

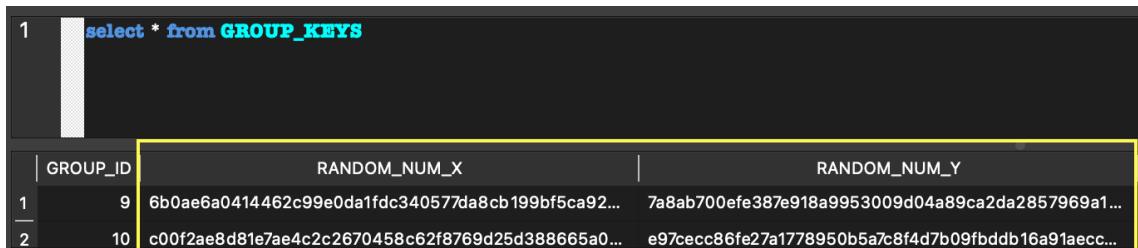
The aim of utilizing secret sharing is to eliminate the need to store the encryption key in the database and to ensure that possession of only the GROUP_ID is insufficient for retrieving the key from the private database.

5.3.1 Functional Testing

To test the implemented secret sharing, an encrypted group was created. The next step was to test if the encryption key generated by the implemented key generation process effectively allowed users added to the group by the owner to view posts in the encrypted group. It was confirmed that users were able to view the content as usual when browsing the encrypted groups they were part of. The verification process included ensuring that the added user could see all the content within the encrypted group.

5.3.2 Data Verification Testing

This kind of testing involves looking at values in the concerned database tables to ensure that the values are populated as expected. We check USER_GROUPS, SOCIAL_GROUPS, and GROUP_KEYS to check if values are populated for the created encrypted group.



```
1 | select * from GROUP_KEYS
```

	GROUP_ID	RANDOM_NUM_X	RANDOM_NUM_Y
1	9	6b0ae6a0414462c99e0da1fdc340577da8cb199bf5ca92...	7a8ab700efe387e918a9953009d04a89ca2da2857969a1...
2	10	c00f2ae8d81e7ae4c2c2670458c62f8769d25d388665a0...	e97cecc86fe27a1778950b5a7c8f4d7b09fbddb16a91aecc...

Figure 22: GROUP_KEYS table

Testing reveals that the expected columns of the mentioned table are populated with the right values and the user is able to encrypt and decrypt posts in their encrypted groups.

```
select * from USER_GROUP
```

USER_ID	GROUP_ID	STATUS	JOIN_DATE	USER_HASH	SHARE
6	17	1	1684277836	85ea151b8c5b5ab0d3349100e441bd4b8dc20740d429c...	61628557933094781165493810943645093584565102...
7	17	1	1684277836	d536a8c1664fec0bc85615cf3cb2645871e8b2935c964...	61628557933094781165493810943645093584565102...

Figure 23: USER_GROUPS table

```
select * from SOCIAL_GROUPS
```

GROUP_ID	GROUP_NAME	CREATED_TIME	OWNER_ID	REGISTER_TYPE	MEMBER_ACCESS	VOTE_ACCESS	POST_LIFETIME	ENCRYPTION	RANDOM_NUM
17	FinalTest	1684180444.855903	3	1	4	0	-2	1	d08201c8a8a849475ebfa8e9db5cdd740529d68c25fae...

Figure 24: SOCIAL_GROUPS table

5.4 System and Response Time Testing

Performance testing for web applications includes testing for system load time, page load time, and response time. Measuring the time it takes for the system to start up and load the web application is system load time testing. This is important as it can affect the overall performance of the application. Page load time testing involves measuring the time it takes for a particular web page to fully load in the user’s browser. It includes the time taken for all resources, including images, scripts, and stylesheets, to load. Response time is measured by timing how long it takes for a certain request or action to be handled by the web application, such as submitting a form or clicking a button

The details of the specifications of the system used to run the tests are represented in Table 1.

OS	macOS Monterey
Processor	2 GHz Quad-Core Intel Core i5
Memory	16 GB 3733 MHz
Browser	Google Chrome
Network Speed	648.12 Mbps

Table 1: Hardware Specifications

In the page load and system testing, consistent starting conditions were ensured by using the same hardware and software configuration for each trial. The starting conditions included using the latest version of the Chrome browser with cleared cache, and a stable internet connection. About 3 trials were conducted for each test scenario to gather sufficient data and the average is represented in the tables for each implemented feature.

Test Type	Baseline	Post Implementation
System Load Time	0.091s	0.091s
Page load time - Edit Group Page	0.19s	0.19s
Page load time - View Group Page	0.15s	0.16s
Page load time - Manage Group Page	0.1s	0.11s

Table 2: Testing Differential Privacy

Test Type	Baseline	Post Implementation
System Load Time	0.091s	0.092s
Page load time - Group Thread page	0.29s	0.32s
Response Time - Flag post	1.6s	0.92s

Table 3: Testing Flagging Feature

Test Type	Baseline	Post Implementation
System Load Time	0.091s	0.093s
Page load time - Root login and load	0.513s	0.515s
Time taken to approve/delete	-	0.3s
Time taken to view original thread	0.14s	0.12s
Time taken to comment	0.13s	0.13s
Time to add new users	0.16s	0.18s

Table 4: Testing Moderation Feature

The tables for each implemented feature list the particular pages whose load times are tested, since these are the pages impacted by this project. From the results, it can be observed that none of the implemented features have a significant impact

Test Type	Baseline	Post Implementation
System Load Time	0.091s	0.096s
Page load time - Group Creation	0.24s	0.25s
Page load time - View Group Page	-	0.72s

Table 5: Testing Secret Sharing Feature

on the load times, which helps maintain the system's load times. This is crucial as it ensures that the system remains efficient and responsive even after the addition of new features. The testing page load times and response times are essential as it allows for the identification of any bottlenecks or performance issues that could affect user experience. Additional images of testing are included in the appendix.

CHAPTER 6

Conclusion

In conclusion, this project has implemented several security mechanisms to enhance the security of Yioop. By extending differential privacy to mask the number of users in a group and using secret sharing to protect the encryption key, the project has increased the protection of user information and prevented potential malicious activities. The addition of a flagging feature has also allowed for early detection of inappropriate content and contributed to maintaining a safe online environment. Furthermore, the project has provided a detailed overview of the Yioop architecture and modifications made to the codebase, contributing to a better understanding of the design of the system. Overall, these enhancements have improved the overall security and usability of Yioop, and can serve as a model for implementing security measures in other online communities.

There are several avenues for future work that can enhance the security features of Yioop. Exploring other cryptographic techniques such as homomorphic encryption and secure multi-party computation can provide new ways to protect user data and maintain user privacy. As Yioop continues to evolve, future work can focus on improving its security features and making it a safer platform for users to engage in online discussions.

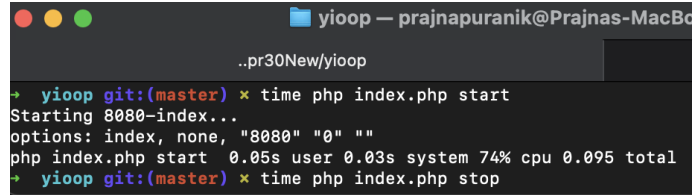
LIST OF REFERENCES

- [1] P. Rana, “Adding differential privacy in an open board discussion board system,” Master’s thesis, San Jose State University, San Jose, CA, Spring 2016.
- [2] B. Ahmad. “What is the importance of content moderation?” Sept. 2021. [Online]. Available: <https://www.techmaish.com/what-is-the-importance-of-content-moderation/>
- [3] C. Lanius, R. Weber, and W. I. MacKenzie Jr., “Use of bot and content flags to limit the spread of misinformation among social networks: a behavior and attitude survey,” *Social Network Analysis and Mining*, vol. 11, no. 32, Mar 2021.
- [4] E. Chandrasekharan, U. Pavalanathan, A. Srinivasan, A. Glynn, J. Eisenstein, and E. Gilbert. “You can’t stay here: The efficacy of reddit’s 2015 ban examined through hate speech.” 2017. [Online]. Available: <https://doi.org/10.1145/3134666>
- [5] R. Bonta, “California consumer privacy act (ccpa),” *California Department of Justice*, 2021. [Online]. Available: <https://oag.ca.gov/privacy/ccpa>
- [6] B. Wolford. “What is gdpr, the eu’s new data protection law?” 2018. [Online]. Available: <https://gdpr.eu/what-is-gdpr/>
- [7] E. Devaux, “What is differential privacy: definition, mechanisms, and examples,” *Stattice.ai*, December 2022. [Online]. Available: <https://www.stattice.ai/post/what-is-differential-privacy-definition-mechanisms-examples#:~:text=Differential%20privacy%20is%20a%20mathematical,any%20individual%20in%20the%20dataset>.
- [8] C. Dwork, “Differential privacy,” in *33rd International Colloquium on Automata, Languages and Programming, part II (ICALP 2006)*, ser. Lecture Notes in Computer Science, vol. 4052, July 2006, pp. 1--12. [Online]. Available: <https://www.microsoft.com/en-us/research/publication/differential-privacy/>
- [9] A. Shamir, “How to share a secret,” *Communications of the ACM*, pp. 612--613, 1979.
- [10] J. Downs. “Why flagging matters.” [Online]. Available: <https://youtube.googleblog.com/2016/09/why-flagging-matters.html>
- [11] M. Aggarwal, A. Dasgupta, and A. Jaiswal. “Safeguarding social media: How effective content moderation can help clean up

- the internet.” Accessed on 2022-05-06. 11 2021. [Online]. Available: <https://www.everestgrp.com/safeguarding-social-media-how-effective-content-moderation-can-help-clean-up-the-internet-blog.html>
- [12] T. Stackpole. Harvard Business Review. “Content moderation is terrible by design. a conversation about how to fix the front lines of the internet.” Accessed on 2022-05-06. November 09 2022. [Online]. Available: <https://hbr.org/2022/11/content-moderation-is-terrible-by-design>
- [13] G. Krasner and S. Pope, “A description of the model-view-controller user interface paradigm in the smalltalk80 system,” *Journal of Object-oriented Programming - JOOP*, vol. 1, 01 1988.
- [14] S. Almuhammadi and I. Al-Hejri, “A comparative analysis of aes common modes of operation,” in *2017 IEEE 30th Canadian conference on electrical and computer engineering (CCECE)*. IEEE, 2017, pp. 1--4.

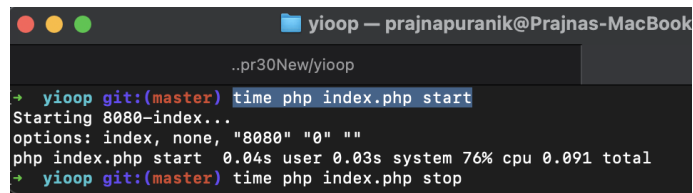
APPENDIX

Testing images



```
yioop — prajnapuranik@Prajnas-MacBo
..pr30New/yioop
+ yioop git:(master) * time php index.php start
Starting 8080-index...
options: index, none, "8080" "0" ""
php index.php start 0.05s user 0.03s system 74% cpu 0.095 total
+ yioop git:(master) * time php index.php stop
```

Figure A.25: Baseline Yioop



```
yioop — prajnapuranik@Prajnas-MacBook
..pr30New/yioop
+ yioop git:(master) time php index.php start
Starting 8080-index...
options: index, none, "8080" "0" ""
php index.php start 0.04s user 0.03s system 76% cpu 0.091 total
+ yioop git:(master) time php index.php stop
```

Figure A.26: Yioop with project features implemented