COMP353 / FALL 2024 (SECTION F) PROJECT REPORT

Prepared for:

Bipin C. Desai

Prepared by:

Chengharv Pen (40279890), Grecia Maria Olano O'Brien (25372232)

Course Group:

COMP 353_group_9

Gina School of Engineering and Computer Science

December 9, 2024

Table Of Contents

Project Overview

Project Description

Use MySQL Database Management System to develop the Private Online Community Social Network System - COSN. One of the objectives is to first flesh out the requirements of the system bearing in mind the minimum as sketched below. The application would include a collection of tables and services hosted by COSN. Hence COSN would enable members to access a local community based server to share information and ideas. It provides services for people sharing interests, activities, and backgrounds among themselves. The COSN system allows its members to create a profile, to create a list of other members with whom to share contents, and to view and add comments and contents - if enabled by the owner member of the web page. It also allows members to interact among each other via a self-contained messaging system. The objective is the sharing of news, pictures, posts, activities, events, interests with members in the community. Also, it allows members to create groups that share common interests or affiliations and hold discussions in forums.

You are required to develop a database system that will store at least the information about the following entities and relationships:

Details on members: ID, password, other personal information such as address, internal interaction pseudonym etc. Members have family, friends and colleagues, privilege and status. A member can specify what part of his/her personal information is public and what part is accessible to which members of his/her group or is private.

A new person in the community can become a member by entering his details and validate it by entering the required details such as the name and appropriate ID of an existing member or by being introduced to the system by an existing member. Only public information is visible to other non-members. Privilege can be either administrator, senior or junior. A member with an administrator privilege has the full power on all services such as creation, deletion, editing of all members and groups. The administrator could also post public items (accessible to the world). A member can post new items and specify which of his groups can access the post and who in each group can comment on it or add content to the post. An item could also be accessible to any other members.

A member with a senior privilege can create groups and manipulate groups created by him/her. The group is owned by the member who created it. Also, a member with a senior privilege can add new members to the COSN. A member can add a list of members and specify them as family members, friend members or colleague members. Status of a member can be either active, inactive or suspended. An active member can have access to all the functionalities of a member. An inactive member will not be visible to other members. A suspended member will not be able to login to the COSN system until his/her status is changed.

All new members start by default as junior members. Only a member with an administrator privilege can change the privilege of another member. The system by default has one member with username admin and password admin created initially(Both of these must be changed after the first login). Only members with administrator privileges can change the status of other members to suspended or reset it to active or inactive. A member with junior privilege can edit his/her profile, create a posting and communicate with other members. Also a junior member can post to groups that he/she is a member of only. A junior member can request to become a senior member. Each member can only have one profile including one email address.

When installed on an operational system with a functioning email server, the system could send out messages to indicate new contents to the members of the associated group. However, since there is a restriction of sending emails by AITS (No email messages are allowed to be sent out of the system), emails have to be simulated by a pop-up window and internal and sent email boxes.

The Applications Supported

The system should support at least the following functionalities through its interface:

1. Create/Delete/Edit/Display a member.
2. Create/Delete/Edit/Display a group.
3. Create/Delete/Edit/Display list of friends for a member.
4. Member request to be a friend of other member or join a group.
5. Member's ability to block another member or to be withdrawn from a group.
6. Member's ability to post texts, images or videos as well as to view posts by other members and comment on them.
7. Members can either post or view posts of only groups that they belong to.
8. Member's main page shows the best and latest posts from their groups and friends.
9. Members can send a private message to their friends.
10. Report of groups or members by specific category such as interest, age, profession, region, etc.
11. Ability to organize an event for the group by voting on date/time/place from a set posted and/ or alternates suggested by one of the group members
12. Registry and/or Gift exchange ideas among a family (secret Santa) or a group.

The Assumptions

It is expected that it would involve appropriate use of: CSS, HTML, Javascript, MariaDB-MySQL, and PHP.

The Limitations

Email Functionality: Due to restrictions imposed by AITS, the system cannot send real emails externally. Instead, email notifications will be simulated through pop-up windows and an internal email system within the platform. This limits the ability to send actual notifications or confirmations to members outside of the COSN platform.

Scalability: The system is designed to handle a moderate number of members and posts, but it may face performance issues when dealing with large-scale data such as thousands of members, groups, or extensive multimedia content (e.g., high-resolution images or videos). The database and PHP backend may require further optimization for larger-scale operations.

Limited Third-Party Integration: Due to the system's reliance on internal services and a private email simulation, integration with third-party services such as external social media platforms, external email systems, or advanced external APIs (e.g., for sending real-time notifications) is not supported.

Platform Compatibility: Although the system is designed to work on popular web browsers such as Firefox, its performance on less common or outdated browsers is not guaranteed. The application may experience compatibility issues with older browsers or non-standard devices.

Security Constraints: The system includes basic security measures such as password protection and role-based access control. However, more advanced security features like multi-factor authentication (MFA) or encrypted end-to-end communication for messages are not implemented due to time and resource constraints.

User Interface: The user interface is functional but may lack advanced features such as responsive design for mobile devices. While the system will work on most desktop browsers, the mobile experience may not be fully optimized.

Data Privacy: While the system allows members to set the visibility of their personal data, it does not implement advanced encryption for data storage, which could expose sensitive information if the system is compromised. Data privacy is limited to basic control over visibility within the platform.

Limited Group Management Features: While members can create and manage groups, some advanced group management features like moderating content, reporting inappropriate posts, or advanced member roles within groups (e.g., moderators or guests) are not available.

Event Organization Limitations: The event creation and voting process for selecting dates, times, and locations may be basic. It lacks integration with external calendar services or automatic reminders, which would enhance event management and user participation.

Multimedia Handling: The system supports posting texts, images, and videos, but the quality and size limitations of these media types may impact the user experience. The system may not support large files or high-quality video content effectively.

System Maintenance: Regular system updates and maintenance are required for smooth operation, but the system may not be able to handle extensive maintenance tasks or troubleshooting remotely without downtime, affecting user availability.

Architectural Design

The application is a two-tier system, which supports Firefox or any popular web browser at the client side and secure http server with PHP parser and a MySQL database at the server side. The system is expected to support all “representative” queries and operations required by a realistic COSN system

Missing Features from the Demo

Due to time constraints, we are unable to implement them in time. However, we will list them below:

1. Check if DOB is valid on Account Creation

2. GiftExchange: Giving Gifts instead of payments.

3. A Group Member should be able to attempt to publish a Group Post, which would send an approval request to a Group Admin. If a Group Admin approves the request, the Group Post will be published. Otherwise, if a Group Admin disapproves the request, the Group Post will NOT be published.

Contributions

Member Responsibility

(From the Group Leader, Chengharv Pen): I have completed the project by myself. My other group members have either dropped the course or are unresponsive.

Grecia worked with me for Assignments 0 to 3.

Therefore, I have given a Peer Review feedback of 100 to Grecia.

Git Logs

The commits below are shown from the newest to the oldest.

```
commit a9090d1af17e30c99cce2e26e6e4caefd4ef10ee (HEAD -> main, origin/main, origin/HEAD)
Author: cheng <chengharvp@gmail.com>
Date:   Sun Dec 8 16:10:31 2024 -0500

    query fix

commit 37a2d8bb6fa6133225b40bb664b5539361040c3a
Author: cheng <chengharvp@gmail.com>
Date:   Sat Dec 7 16:50:27 2024 -0500
```

```
    author comments

commit f053abd779a4cc3904b03dbe5ae0fe8cf6326b03
Author: cheng <chengharvp@gmail.com>
Date:   Sat Dec 7 15:47:47 2024 -0500

    changes while writing the report

commit 4527acb28578ec4f0c213b6d995d5d5256fa83d9
Author: cheng <chengharvp@gmail.com>
Date:   Fri Dec 6 01:50:41 2024 -0500

    last minute changes from testing on ENCS server

commit 340b8173b430ee4cc6e3325385b9ad4281c90f5d
Author: cheng <chengharvp@gmail.com>
Date:   Thu Dec 5 17:04:26 2024 -0500

    remove member from groups if Admin

commit 7c82b2e6ef8453bc893c3c75703574f7c77e48a3
Author: cheng <chengharvp@gmail.com>
Date:   Thu Dec 5 12:51:24 2024 -0500

    adding PaymentAmount Column for gift-exchange

commit ea1c88c36a42e41668499450deeb9f679a606f1c
Author: cheng <chengharvp@gmail.com>
Date:   Thu Dec 5 12:48:31 2024 -0500

    i lied, finishing gift-exchange

commit 659fb94283a4394dca49f71d9139cc90e3a1e1c9
Author: cheng <chengharvp@gmail.com>
Date:   Thu Dec 5 02:51:42 2024 -0500

    final changes implementation-wise, only styling left

commit f2ae84dda6c8eb14c0748efd076fa124e9b0f0ff
Author: cheng <chengharvp@gmail.com>
Date:   Thu Dec 5 01:52:33 2024 -0500

    senior promotion + general improvements

commit 5a94a747c457b336ced6efb2d29773f8486805c1
Author: cheng <chengharvp@gmail.com>
Date:   Wed Dec 4 22:56:19 2024 -0500
```

```
    updating project state, final updates will be for tomorrow

commit 95e40f88daaea2807bed3e74c3091bbeefed82d5
Author: cheng <chengharvp@gmail.com>
Date:   Wed Dec 4 22:41:40 2024 -0500

    login logic for suspended accounts, payment logic

commit aab7d1aa5094027199ded1c5e03ed1ab7adc0a86
Author: cheng <chengharvp@gmail.com>
Date:   Wed Dec 4 21:46:31 2024 -0500

    still implementing warnings system

commit 7adf4b3ed727bede4e0669abf25f0d902b15fcc1
Author: cheng <chengharvp@gmail.com>
Date:   Wed Dec 4 20:04:10 2024 -0500

    secret santa implementation

commit 5cd85ffcbdc1056fc12132df70da38a2373d9aa8
Author: cheng <chengharvp@gmail.com>
Date:   Wed Dec 4 19:09:28 2024 -0500

    edit posts and home page

commit a30c0000790dbd4b69d1542200664d6d8ec90ea5
Author: cheng <chengharvp@gmail.com>
Date:   Wed Dec 4 17:56:35 2024 -0500

    events (2/3)

commit 70d21291131c7f6508d74ea84b35abd945f4e0a3
Author: cheng <chengharvp@gmail.com>
Date:   Wed Dec 4 16:06:14 2024 -0500

    events (1/3)

commit 0e4fbef18112b21ba671296fe5cce2a01173d172
Author: cheng <chengharvp@gmail.com>
Date:   Wed Dec 4 15:28:46 2024 -0500

    Administrator >>> can change privilege/status of other members

commit a5d80776cd10b04b06d0b29813e16cc799f541c9
Author: cheng <chengharvp@gmail.com>
Date:   Wed Dec 4 15:08:24 2024 -0500
```

```
    obligatory admin user/pass change implemented

commit f378d2726859a32e479db69ff47df41d1efe3b83
Author: cheng <chengharvp@gmail.com>
Date:   Wed Dec 4 14:56:14 2024 -0500

    chat UI logic bug fix

commit 548c54e31c006f926d867e58ac1fcc4eada89130
Author: cheng <chengharvp@gmail.com>
Date:   Wed Dec 4 14:51:24 2024 -0500

    delete posts, setting up Members Inactive Status

commit ec61f049f559eebb4a0ea5996ccfaaecdb640671
Author: cheng <chengharvp@gmail.com>
Date:   Wed Dec 4 14:22:36 2024 -0500

    formatting inspect-single-post.php

commit 4705aabc4d1f47b63cd1595af2b9d1eb93c41e94
Author: cheng <chengharvp@gmail.com>
Date:   Wed Dec 4 14:17:47 2024 -0500

    implementing view/dispaly posts (3/3)
    adding filtered search for groups AND members

commit 7c100b03d35c8ff218441c959390700dab21ab5e
Author: cheng <chengharvp@gmail.com>
Date:   Wed Dec 4 02:10:27 2024 -0500

    improving chat UI

commit 7e7de4be96e7349005588d9c83ec6424e62f7b44
Author: cheng <chengharvp@gmail.com>
Date:   Wed Dec 4 01:10:25 2024 -0500

    improving email UI

commit c22e6ca2e1cb22efe8ae77ec9f7793fc023f91ad
Author: cheng <chengharvp@gmail.com>
Date:   Wed Dec 4 00:15:33 2024 -0500

    test

commit 2d28973c4eac2746ed5c0b30fa1b85312982bfdb
Author: cheng <chengharvp@gmail.com>
Date:   Tue Dec 3 22:52:23 2024 -0500
```

```
    bug fixing related to posts, refactoring files again

commit c429a78c2d43ceacdea2e7fdfbda97ef30633acb
Author: cheng <chengharvp@gmail.com>
Date:   Tue Dec 3 21:03:47 2024 -0500

    implementing publishing/viewing posts (2/3)

commit 9e826b4e2ee04f02f8b638fa0225622eb7486124
Author: cheng <chengharvp@gmail.com>
Date:   Tue Dec 3 19:38:48 2024 -0500

    refactoring

commit 458ffbb220ee6260bedc30f59f788cf321070b5c
Author: cheng <chengharvp@gmail.com>
Date:   Tue Dec 3 17:29:46 2024 -0500

    implementing publish/viewing posts (1/3)

commit 0e71fdeeac5aba72d26158e4f002eabe07647c3a
Author: cheng <chengharvp@gmail.com>
Date:   Tue Dec 3 15:12:03 2024 -0500

    fixing bugs in email

commit eaaac71a9d9cdc1ef18a6bde7b34de987ce1b516
Author: cheng <chengharvp@gmail.com>
Date:   Tue Dec 3 00:44:20 2024 -0500

    email implementation

commit decc55e4ecc5d688a744727112736b0abcd1bb10
Author: cheng <chengharvp@gmail.com>
Date:   Tue Dec 3 00:13:58 2024 -0500

    chat functionnalities, needs styling

commit 4a595dd4bf445da0547eb838fab281d7475dea61
Author: cheng <chengharvp@gmail.com>
Date:   Mon Dec 2 22:22:17 2024 -0500

    edit friends + bug fixing

commit ee86a50e2fea54aceada0a7372a9d1aa7809a69a
Author: cheng <chengharvp@gmail.com>
Date:   Mon Dec 2 21:45:51 2024 -0500
```

```
    create, display and delete friends

commit ceb4f760fb88d72b6d60519849fb3894e67ba693
Author: cheng <chengharvp@gmail.com>
Date:   Sat Nov 30 22:04:39 2024 -0500

    setting up friends for the next time i work on this

commit 9271cf796d81c5d0f8a40a15fb133cb36e36a883
Author: cheng <chengharvp@gmail.com>
Date:   Sat Nov 30 21:35:37 2024 -0500

    implemented groups

commit af59277443da7ff2b8b9e1cd1eb24c3baf1d9c62
Author: cheng <chengharvp@gmail.com>
Date:   Sat Nov 30 17:33:24 2024 -0500

    edit/delete groups (needs testing)

commit 150c46f9ccd6c97dc1804a9dc6114566911f3b4e
Author: cheng <chengharvp@gmail.com>
Date:   Sat Nov 30 14:39:58 2024 -0500

    display groups (needs testing)

commit f74ec64a6280273b5207a2fd07dcbc4ff3485c22
Author: cheng <chengharvp@gmail.com>
Date:   Sat Nov 30 14:13:30 2024 -0500

    edit members implemented

commit 125c51cf14dc7a219454f9c3bbfb4cc4fbcd0c78
Author: cheng <chengharvp@gmail.com>
Date:   Thu Nov 28 16:31:35 2024 -0500

    blocking members
    popup for delete members

    for display members(2/3), do display ALL members option
    for display members (3/3), do cleaner display

commit 13591d6a2b32076bddbabbea784bc54265e7424c
Author: cheng <chengharvp@gmail.com>
Date:   Fri Nov 22 17:18:07 2024 -0500

    search by username
```

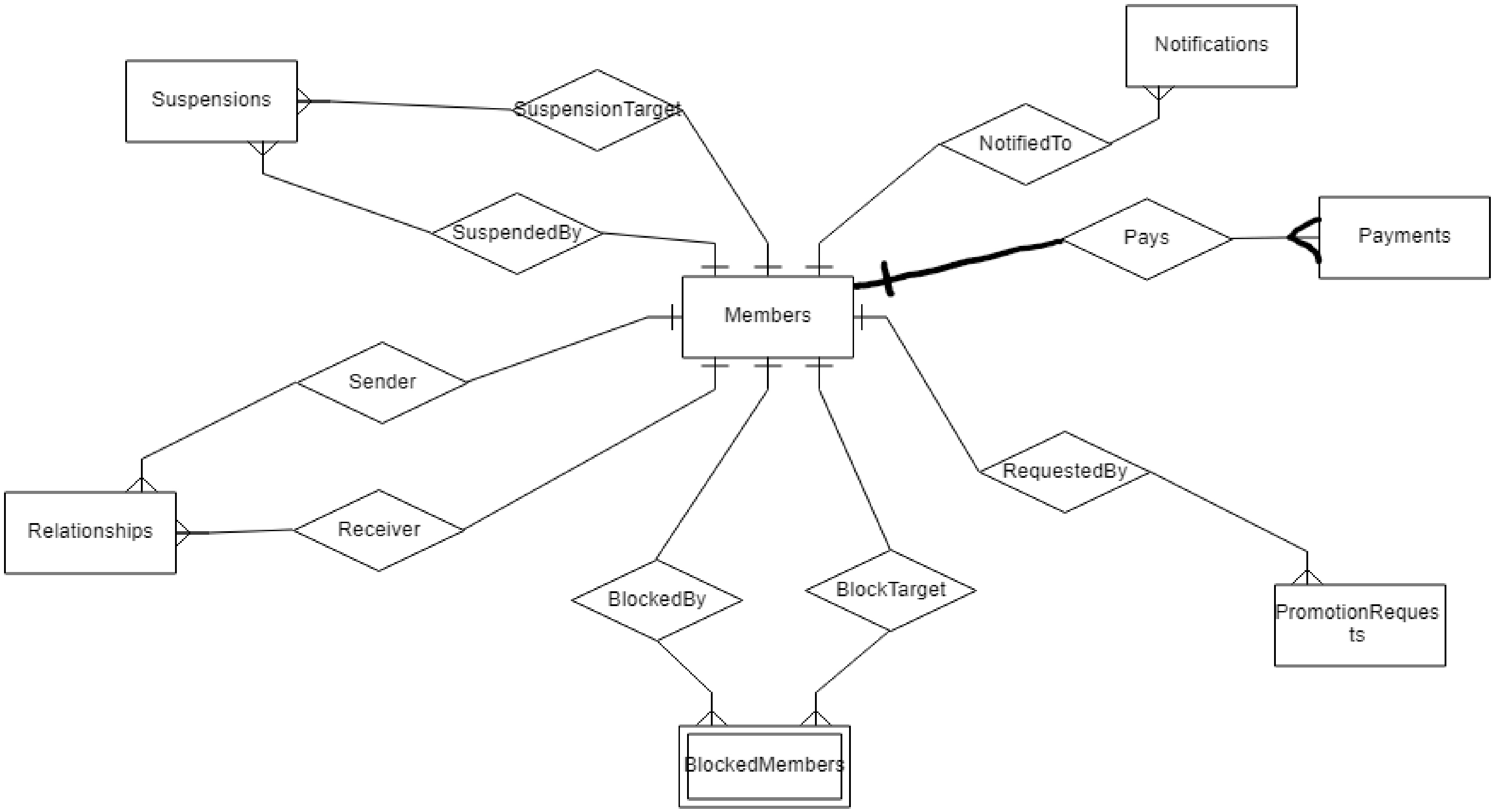
```
commit 7b75ce0aa2098dd89fe0cfc85a98986ede256c57
Author: cheng <chengharvp@gmail.com>
Date:   Fri Nov 22 16:27:02 2024 -0500

    delete members
    display members (1/3)

commit fa1cfbc773e95c0e5e82650c4ff218a3cdc70b73
Author: cheng <chengharvp@gmail.com>
Date:   Fri Nov 22 15:20:51 2024 -0500

    creating account (2/2)
    forgot password implemented

commit 52e1ff49d5d864b8bdeb541b3063dbfdae633e08
Author: cheng <chengharvp@gmail.com>
Date:   Thu Nov 21 16:43:34 2024 -0500

    creating accounts (1/2)

commit 609e695a4c9c19ade6e441966916658d78509e5a
Author: cheng <chengharvp@gmail.com>
Date:   Thu Nov 21 02:55:25 2024 -0500

    implementing logins (1)

commit 90616cb9e2ff12b79704e67d99faf660e8e3fe9a
Author: cheng <chengharvp@gmail.com>
Date:   Thu Nov 21 01:13:32 2024 -0500

    correcting README

commit 7158de926b27e7802154bb472af2d7a9aedbfc95
Author: cheng <chengharvp@gmail.com>
Date:   Thu Nov 21 01:12:50 2024 -0500

    pushing README.md

commit 90cca9c4f9a03e1c8d80743983bd859f71f576f8
Author: cheng <chengharvp@gmail.com>
Date:   Thu Nov 21 01:11:13 2024 -0500

    Setting up UI

commit 595c4f05e7686479a12ba9d1fb777bf786eeefe7
Author: cheng <chengharvp@gmail.com>
Date:   Thu Nov 21 00:11:50 2024 -0500
```

```
    x

commit 8002d724b73d71aba202f39ca94d4dbe005599ca
Author: cheng <chengharvp@gmail.com>
Date:   Thu Nov 21 00:07:34 2024 -0500

    initial files
```

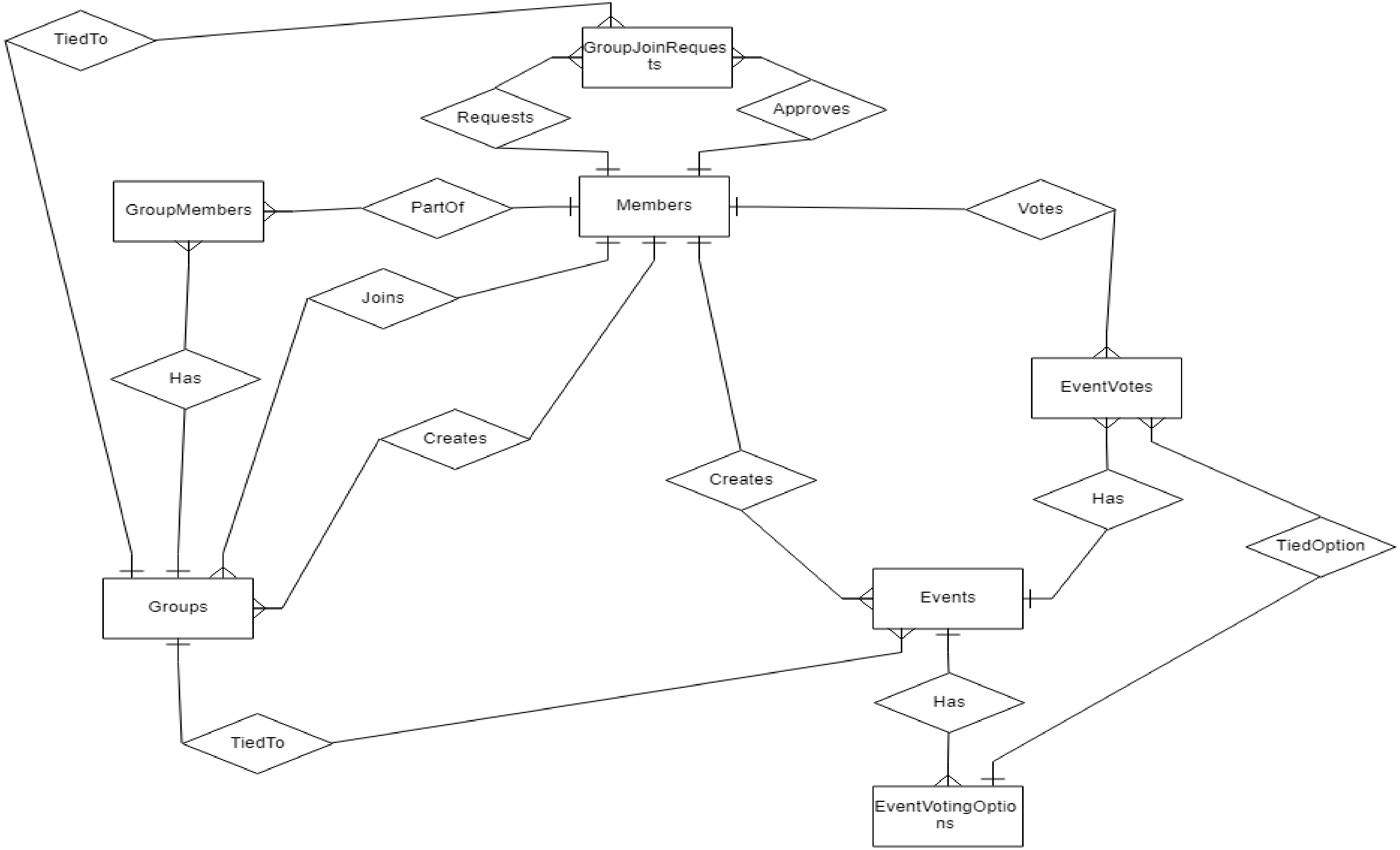
Design

<u>E-R Diagrams (WE WILL NOT SHOW ATTRIBUTES FOR THE SAKE OF REDUCING CLUTTER. PLEASE HAVE A LOOK AT THE RELATIONAL SCHEMA FOR THEIR DETAILS.)</u>

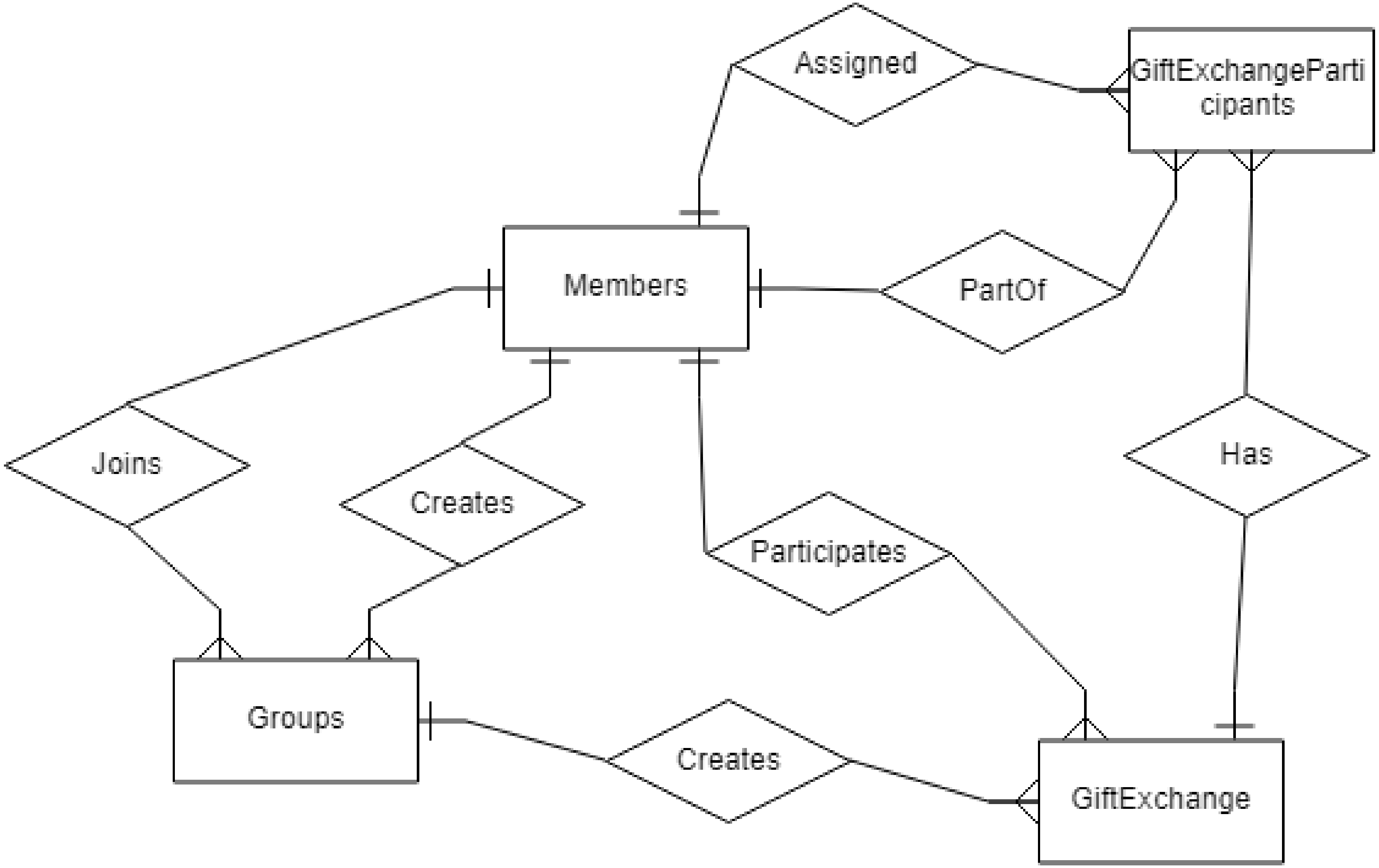
LAYER 1: ENTITIES THAT ONLY HAVE MEMBERS TO MEMBERS RELATIONSHIPS



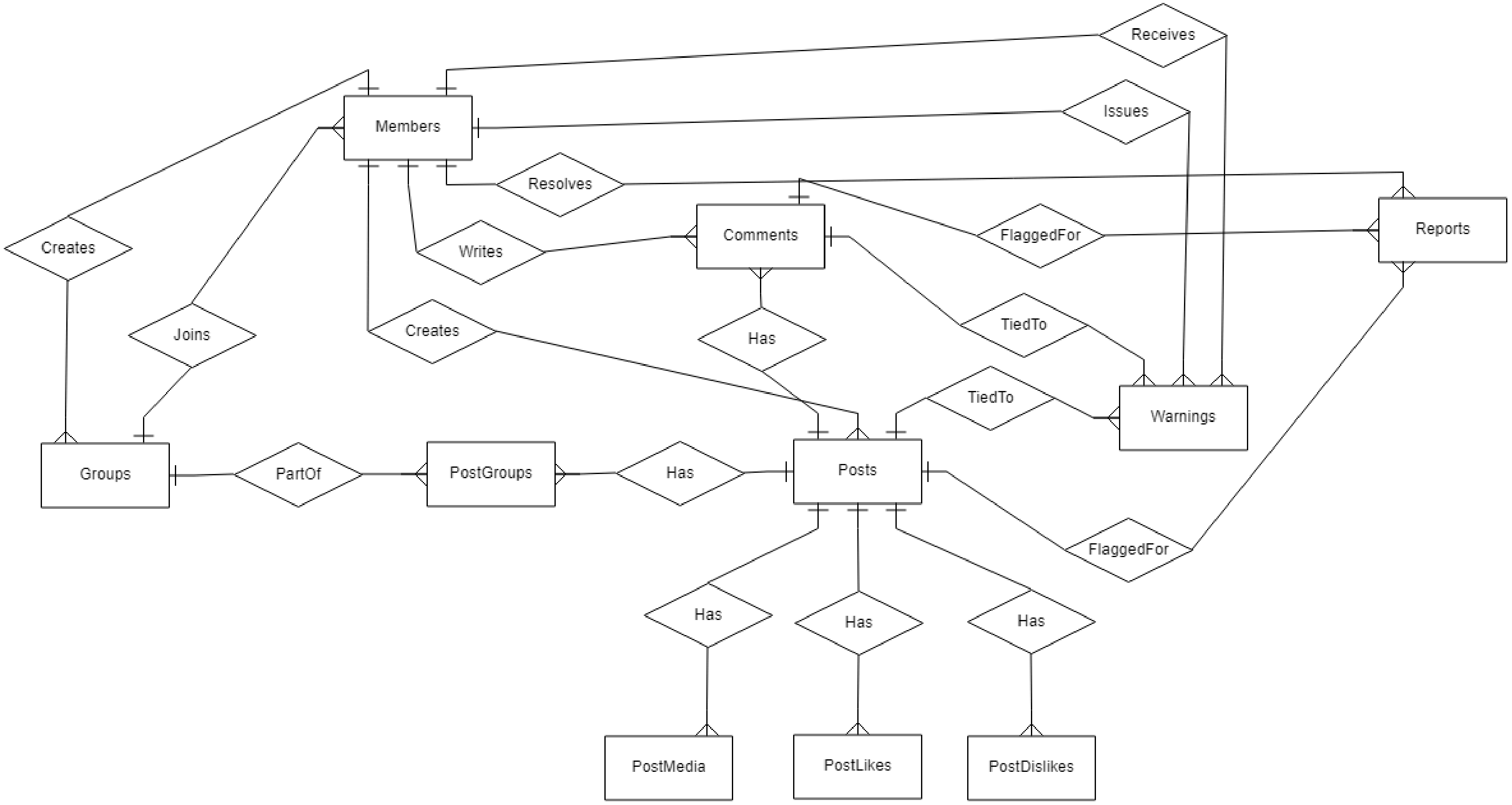
LAYER 2: GROUPS, GROUPMEMBERS, GROUPJOINREQUESTS AND EVENTS



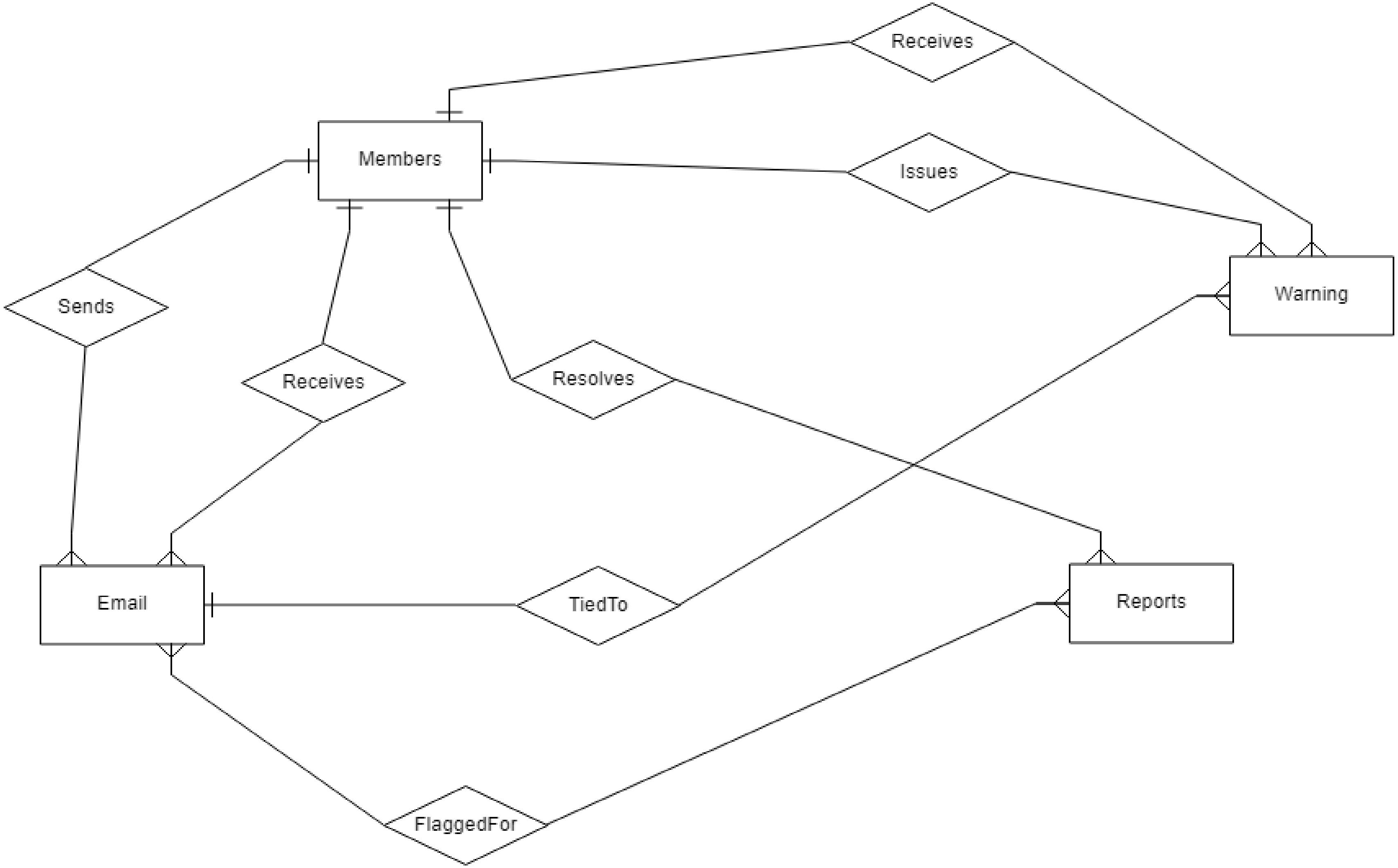
LAYER 3: GROUPS AND GIFT EXCHANGE



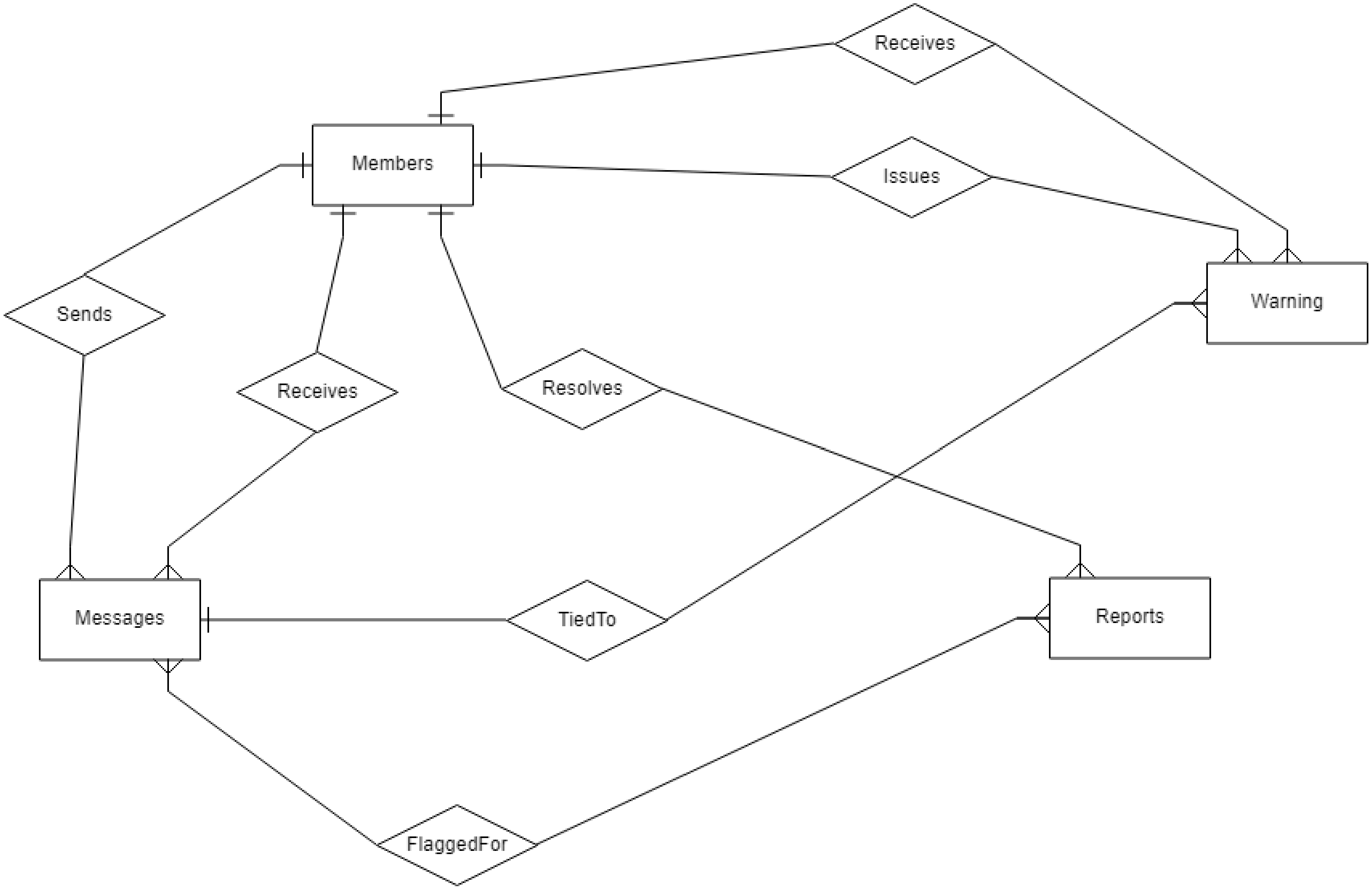
LAYER 4: GROUPS, POSTS, WARNINGS AND REPORTS



LAYER 5: EMAIL, WARNINGS AND REPORTS

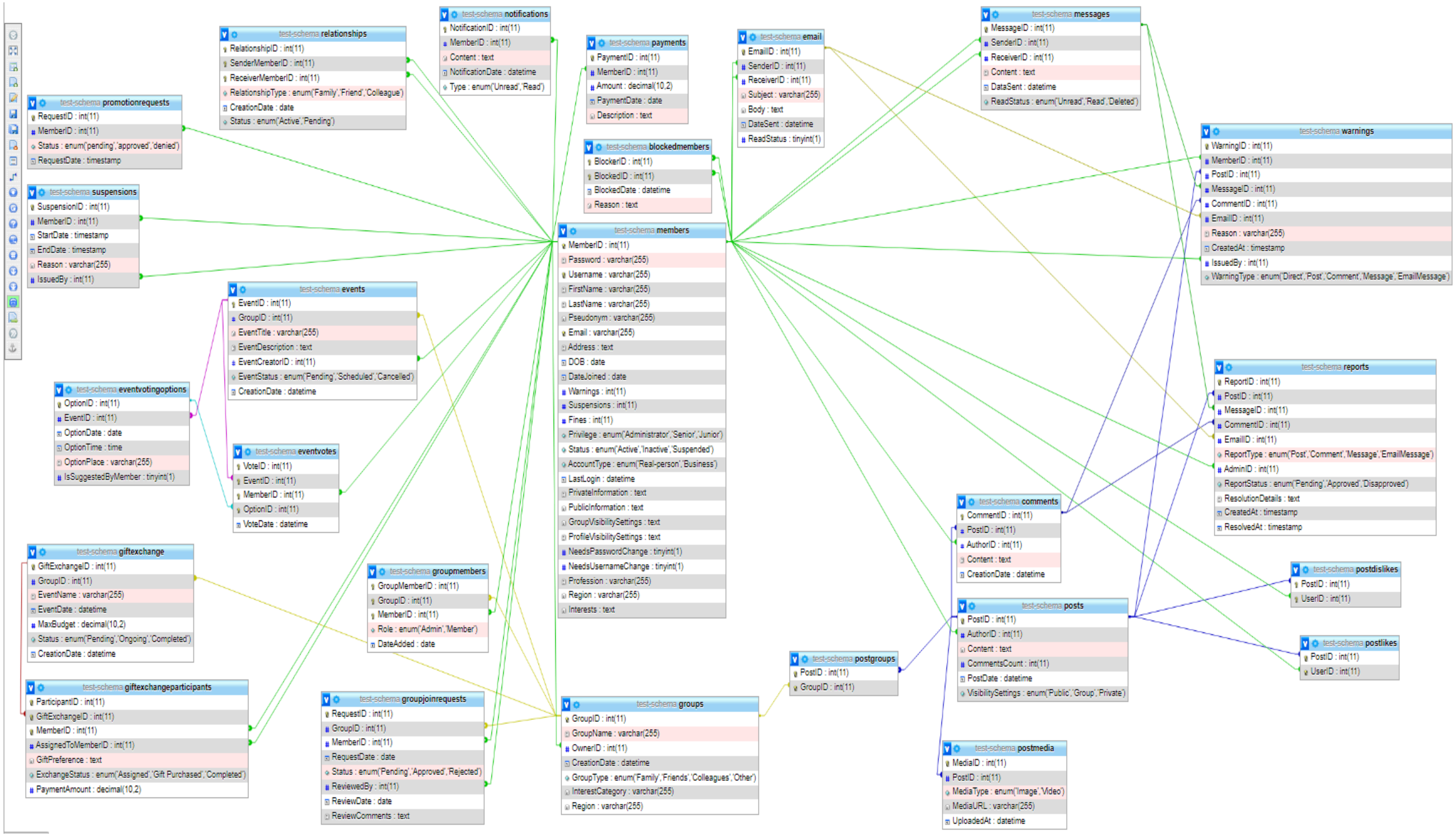


LAYER 6: MESSAGES, WARNINGS AND REPORTS



Relational Schema

(PLEASE USE 200-300% ZOOM)



test-schema promotionrequests
- RequestID : int(11)
- MemberID : int(11)
- Status : enum('pending','approved','denied')
- RequestDate : timestamp

test-schema suspensions
- SuspensionID : int(11)
- MemberID : int(11)
- StartDate : timestamp
- EndDate : timestamp
- Reason : varchar(255)
- IssuedBy : int(11)

test-schema relationships
- RelationshipID : int(11)
- SenderMemberID : int(11)
- ReceiverMemberID : int(11)
- RelationshipType : enum('Family','Friend','Colleague')
- CreationDate : date
- Status : enum('Active','Pending')

test-schema notifications
- NotificationID : int(11)
- MemberID : int(11)
- Content : text
- NotificationDate : datetime
- Type : enum('Unread','Read')

test-schema payments
- PaymentID : int(11)
- MemberID : int(11)
- Amount : decimal(10,2)
- PaymentDate : date
- Description : text

test-schema blockedmembers
- BlockerID : int(11)
- BlockedID : int(11)
- BlockedDate : datetime
- Reason : text

test-schema email
- EmailID : int(11)
- SenderID : int(11)
- ReceiverID : int(11)
- Subject : varchar(255)
- Body : text
- DateSent : datetime
- ReadStatus : tinyint(1)

test-schema messages
- MessageID : int(11)
- SenderID : int(11)
- ReceiverID : int(11)
- Content : text
- DataSent : datetime
- ReadStatus : enum('Unread','Read','Deleted')

test-schema warnings
- WarningID : int(11)
- MemberID : int(11)
- PostID : int(11)
- MessageID : int(11)
- CommentID : int(11)
- EmailID : int(11)
- Reason : varchar(255)
- CreatedAt : timestamp
- IssuedBy : int(11)
- WarningType : enum('Direct','Post','Comment','Message','EmailMessage')

test-schema events
- EventID : int(11)
- GroupID : int(11)
- EventTitle : varchar(255)
- EventDescription : text
- EventCreatorID : int(11)
- EventStatus : enum('Pending','Scheduled','Cancelled')
- CreationDate : datetime

test-schema members
- MemberID : int(11)
- Password : varchar(255)
- Username : varchar(255)
- FirstName : varchar(255)
- LastName : varchar(255)
- Pseudonym : varchar(255)
- Email : varchar(255)
- Address : text
- DOB : date
- DateJoined : date
- Warnings : int(11)
- Suspensions : int(11)
- Fines : int(11)
- Privilege : enum('Administrator','Senior','Junior')
- Status : enum('Active','Inactive','Suspended')
- AccountType : enum('Real-person','Business')
- LastLogin : datetime
- PrivateInformation : text
- PublicInformation : text
- GroupVisibilitySettings : text
- ProfileVisibilitySettings : text
- NeedsPasswordChange : tinyint(1)
- NeedsUsernameChange : tinyint(1)
- Profession : varchar(255)
- Region : varchar(255)
- Interests : text

test-schema eventvotingoptions
- OptionID : int(11)
- EventID : int(11)
- OptionDate : date
- OptionTime : time
- OptionPlace : varchar(255)
- IsSuggestedByMember : tinyint(1)

test-schema eventvotes
- VoteID : int(11)
- EventID : int(11)
- MemberID : int(11)
- OptionID : int(11)
- VoteDate : datetime

test-schema reports
- ReportID : int(11)
- PostID : int(11)
- MessageID : int(11)
- CommentID : int(11)
- EmailID : int(11)
- ReportType : enum('Post','Comment','Message','EmailMessage')
- AdminID : int(11)
- ReportStatus : enum('Pending','Approved','Disapproved')
- ResolutionDetails : text
- CreatedAt : timestamp
- ResolvedAt : timestamp

test-schema comments
- CommentID : int(11)
- PostID : int(11)
- AuthorID : int(11)
- Content : text
- CreationDate : datetime

test-schema giftexchange
- GiftExchangeID : int(11)
- GroupID : int(11)
- EventName : varchar(255)
- EventDate : datetime
- MaxBudget : decimal(10,2)
- Status : enum('Pending','Ongoing','Completed')
- CreationDate : datetime

test-schema groupmembers
- GroupMemberID : int(11)
- GroupID : int(11)
- MemberID : int(11)
- Role : enum('Admin','Member')
- DateAdded : date

test-schema postdislikes
- PostID : int(11)
- UserID : int(11)

test-schema posts
- PostID : int(11)
- AuthorID : int(11)
- Content : text
- CommentsCount : int(11)
- PostDate : datetime
- VisibilitySettings : enum('Public','Group','Private')

test-schema postlikes
- PostID : int(11)
- UserID : int(11)

test-schema postgroups
- PostID : int(11)
- GroupID : int(11)

test-schema giftexchangeparticipants
- ParticipantID : int(11)
- GiftExchangeID : int(11)
- MemberID : int(11)
- AssignedToMemberID : int(11)
- GiftPreference : text
- ExchangeStatus : enum('Assigned','Gift Purchased','Completed')
- PaymentAmount : decimal(10,2)

test-schema groupjoinrequests
- RequestID : int(11)
- GroupID : int(11)
- MemberID : int(11)
- RequestDate : date
- Status : enum('Pending','Approved','Rejected')
- ReviewedBy : int(11)
- ReviewDate : date
- ReviewComments : text

test-schema groups
- GroupID : int(11)
- GroupName : varchar(255)
- OwnerID : int(11)
- CreationDate : datetime
- GroupType : enum('Family','Friends','Colleagues','Other')
- InterestCategory : varchar(255)
- Region : varchar(255)

test-schema postmedia
- MediaID : int(11)
- PostID : int(11)
- MediaType : enum('Image','Video')
- MediaURL : varchar(255)
- UploadedAt : datetime

ER to Relation Conversions, Keys and Functional Dependencies

COSN = {Members, BlockedMembers, PromotionRequests, Groups, GroupMembers, GroupJoinRequests, Relationships, Posts, PostGroups, PostLikes, PostDislikes, Messages, Email, Events, EventVotingOptions, EventVotes, GiftExchange, GiftExchangeParticipants, Warnings, Payments, Suspensions, Reports, Notifications}

Members

Members(MemberID, Password, Username, FirstName, LastName, Pseudonym, Email, Address, DOB, DateJoined, Warnings, Suspensions, Fines, Privilege, Status, AccountType, LastLogin, PrivateInformation, PublicInformation, GroupVisibilitySettings, ProfileVisibilitySettings, NeedsPasswordChange, NeedsUsernameChange, Profession, Region, Interests)

Primary Key: MemberID

Foreign Keys: None

F = {MemberID ➔ Password, Username, FirstName, LastName, Pseudonym, Email, Address, DOB, DateJoined, Warnings, Suspensions, Fines, Privilege, Status, AccountType, LastLogin, PrivateInformation, PublicInformation, GroupVisibilitySettings, ProfileVisibilitySettings, NeedsPasswordChange, NeedsUsernameChange, Profession, Region, Interests}

BlockedMembers

BlockedMembers(BlockerID, BlockedID, BlockedDate, Reason)

Primary Key: (BlockerID, BlockedID)

Foreign Keys: BlockerID ➔ Members(MemberID)
BlockedID ➔ Members(MemberID)

F = {BlockerID, BlockedID ➔ BlockedDate, Reason}

PromotionRequests

PromotionRequests(RequestID, MemberID, Status, RequestDate)

Primary Key: RequestID

Foreign Keys: MemberID ➔ Members(MemberID)

F = {RequestID ➔ MemberID, Status, RequestDate}

Groups

Groups(GroupID, GroupName, OwnerID, CreationDate, GroupType, InterestCategory, Region)

Primary Key: GroupID

Foreign Keys: OwnerID ➔ Members(MemberID)

F = {GroupID ➔ GroupName, OwnerID, CreationDate, GroupType, InterestCategory, Region}

GroupMembers

GroupMembers(GroupMemberID, GroupID, MemberID, Role, DateAdded)

Primary Key: GroupMemberID

Foreign Keys: MemberID ➔ Members(MemberID)
GroupID ➔ Groups(GroupID)

F = {GroupMemberID ➔ GroupID, MemberID, Role, DateAdded}

GroupJoinRequests

GroupJoinRequests(RequestID, GroupID, MemberID, RequestDate, Status, ReviewedBy, ReviewDate, ReviewComments)

Primary Key: RequestID

Foreign Keys: GroupID ➔ Groups(GroupID)
MemberID ➔ Members(MemberID)
ReviewedBy ➔ Members(MemberID)

F = {RequestID ➔ GroupID, MemberID, RequestDate, Status, ReviewedBy, ReviewDate, ReviewComments}

Relationships

Relationships(RelationshipID, SenderMemberID, ReceiverMemberID, RelationshipType, CreationDate, Status)

Primary Key: RelationshipID

Foreign Keys: SenderMemberID ➔ Members(MemberID)
ReceiverMemberID ➔ Members(MemberID)

F = {RelationshipID ➔ SenderMemberID, ReceiverMemberID, RelationshipType, CreationDate, Status}

Posts

Posts(PostID, AuthorID, Content, CommentsCount, PostDate, VisibilitySettings)

Primary Key: PostID

Foreign Keys: AuthorID ➔ Members(MemberID)

F = {PostID ➔ AuthorID, Content, CommentsCount, PostDate, VisibilitySettings}

PostGroups

PostGroups(PostID, GroupID)

Primary Key: (PostID, GroupID)

Foreign Keys: PostID ➔ Posts(PostID)
GroupID ➔ Groups(GroupID)

F = {}

PostMedia

PostMedia(MediaID, PostID, MediaType, MediaURL, UploadedAt)

Primary Key: MediaID

Foreign Keys: PostID ➔ Posts(PostID)

F = {MediaID ➔ PostID, MediaType, MediaURL, UploadedAt}

PostLikes

PostLikes(PostID, UserID)

Primary Key: (PostID, UserID)

Foreign Keys: PostID ➔ Posts(PostID)
UserID ➔ Members(MemberID)

F = {}

PostDislikes

PostDislikes(PostID, UserID)

Primary Key: (PostID, UserID)

Foreign Keys: PostID ➔ Posts(PostID)
UserID ➔ Members(MemberID)

F = {}

Comments

Comments(CommentID, PostID, AuthorID, Content, CreationDate)

Primary Key: CommentID

Foreign Keys: PostID ➔ Posts(PostID)
AuthorID ➔ Members(MemberID)

F = {CommentID ➔ PostID, AuthorID, Content, CreationDate}

Messages

// There is a typo here, DataSent is supposed to be DateSent. This would need some fixing in the code after changing it (./chat/chat.php)

Messages(MessageID, SenderID, ReceiverID, Content, DataSent, ReadStatus)

Primary Key: MessageID

Foreign Keys: SenderID ➔ Members(MemberID)
ReceiverID ➔ Members(MemberID)

F = {MessageID ➔ SenderID, ReceiverID, Content, DataSent, ReadStatus}

Email

Email(EmailID, SenderID, ReceiverID, Subject, Body, DateSent, ReadStatus)

Primary Key: EmailID

Foreign Keys: SenderID ➔ Members(MemberID)
ReceiverID ➔ Members(MemberID)

F = {EmailID ➔ SenderID, ReceiverID, Subject, Body, DateSent, ReadStatus}

Events

Events(EventID, GroupID, EventTitle, EventDescription, EventCreatorID, EventStatus, CreationDate)

Primary Key: EventID

Foreign Keys: GroupID ➔ Groups(GroupID)
EventCreatorID ➔ Members(MemberID)

F = {EventID ➔ GroupID, EventTitle, EventDescription, EventCreatorID, EventStatus, CreationDate}

EventVotingOptions

EventVotingOptions(OptionID, EventID, OptionDate, OptionPlace, IsSuggestedByMember)

Primary Key: OptionID

Foreign Keys: EventID ➔ Events(EventID)

F = {OptionID ➔ EventID, OptionDate, OptionPlace, IsSuggestedByMember}

EventVotes

EventVotes(VoteID, EventID, MemberID, OptionID, VoteDate)

Primary Key: VoteID

Foreign Keys: EventID ➔ Events(EventID)
MemberID ➔ Members(MemberID)
OptionID ➔ EventVotingOptions(OptionID)

F = {VoteID ➔ EventID, MemberID, OptionID, VoteDate}

GiftExchange

GiftExchange(GiftExchangeID, GroupID, EventName, EventDate, MaxBudget, Status, CreationDate)

Primary Key: GiftExchangeID

Foreign Keys: GroupID ➔ Groups(GroupID)

F = {GiftExchangeID ➔ GroupID, EventName, EventDate, MaxBudget, Status, CreationDate}

GiftExchangeParticipants

GiftExchangeParticipants(ParticipantID, GiftExchangeID, MemberID, AssignedToMemberID, GiftPreference, ExchangeStatus, PaymentAmount)

Primary Key: ParticipantID

Foreign Keys: GiftExchangeID ➔ GiftExchange(GiftExchangeID)
MemberID ➔ Members(MemberID)
AssignedToMemberID ➔ Members(MemberID)

F = {ParticipantID ➔ GiftExchangeID, MemberID, AssignedToMemberID, GiftPreference, ExchangeStatus, PaymentAmount}

Warnings

Warnings(WarningID, MemberID, PostID, MessageID, CommentID, EmailID, Reason, CreatedAt, IssuedBy, WarningType)

Primary Key: WarningID

Foreign Keys: MemberID ➔ Members(MemberID)
PostID ➔ Posts(PostID)
MessageID ➔ Messages(MessageID)
CommentID ➔ Comments(CommentsID)
EmailID ➔ Email(EmailID)
IssuedBy ➔ Members(MemberID)

F = {WarningID ➔ MemberID, PostID, MessageID, CommentID, EmailID, Reason, CreatedAt, IssuedBy, WarningType}

Payments

Payments(PaymentID, MemberID, Amount, PaymentDate, Description)

Primary Key: PaymentID

Foreign Keys: MemberID ➔ Members(MemberID)

F = {Payment ➔ MemberID, Amount, PaymentDate, Description}

Suspensions

Suspensions(SuspensionID, MemberID, StartDate, EndDate, Reason, IssuedBy)

Primary Key: SuspensionID

Foreign Keys: MemberID ➔ Members(MemberID)
IssuedBy ➔ Members(MemberID)

F = {SuspensionID ➔ MemberID, StartDate, EndDate, Reason, IssuedBy}

Reports (NOT USED)

Reports(ReportID, PostID, MessageID, CommentID, EmailID, ReportType, AdminID, ReportStatus, ResolutionDetails, CreatedAt, ResolvedAt)

Primary Key: ReportID

Foreign Keys: AdminID ➔ Members(MemberID)
PostID ➔ Posts(PostID)
MessageID ➔ Messages(MessageID)
CommentID ➔ Comments(CommentsID)
EmailID ➔ Email(EmailID)

F = {ReportID ➔ PostID, MessageID, CommentID, EmailID, ReportType, AdminID, ReportStatus, ResolutionDetails, CreatedAt, ResolvedAt}

Notifications (NOT USED)

Notifications(NotificationID, MemberID, Content, NotificationDate, Type)

Primary Key: NotificationID

Foreign Keys: MemberID ➔ Members(MemberID)

F = {NotificationID ➔ MemberID, Content, NotificationDate, Type}

3NF Normalization

Yes, all the relations in the database are in 3NF with respect to their functional dependencies. This condition is true for every relation in the database.

Indeed, we designed our database system with respect to the third normal form. We did our best to eliminate any attributes that do not depend on the key and create separate tables for them.

Implementation

Local Testing Setup

I used XAMPP v.3.3.0 to test the project's PHP code on localhost using its provided Apache web server. This version of XAMPP has some importance, since it matches the ENCS Server's PHP and MySQL versions, which are 7.4.27 and 8.0.22 respectively.

The code was written using the text editor Visual Studio Code, because it has Git already integrated into it. Therefore, we were able to easily push changes to our code to a GitHub repository within the editor.

Locally, my details to connect to my database schema were:

```
// Database connection
$host = "localhost"; // Change if using a different host
$dbname = "db-schema";
$username = "root";
$password = "";
```

ENCS Server Setup

As mentioned previously, the ENCS Server uses PHP v7.4.27 and MySQL v8.0.22. I have ensured that the code's queries are correct, by putting backticks around the word Groups. MySQL considers Groups as a keyword, so if we try to run the code without the backticks, there will be an error.

We will now assume that a local computer is connected to Concordia's VPN using FortiClient.

Within Visual Studio Code, I used an extension named "Remote-SSH", which allows me to SSH to the ENCS Server. I have self-documented this process below:

1. Ctrl+Shift+P >>> "Remote-SSH: Open SSH Configuration File"

   then type this in the config file...

   ```
   Host ENCS-SERVER
   ```

```
HostName login.encs.concordia.ca
User <YOUR ENCS USERNAME>
StrictHostKeyChecking no
```

2. Ctrl+Shift+P >>> "Remote-SSH: Connect to Host" >>> ENCS-SERVER

3. Enter your username and password... (you might have to retry multiple times)

4. Once the SSH is successful, Ctrl+Shift+E

5. Open Folder with your directory location

   For us, we would have to delete everything in that bar, then type "/www/groups/n/np_comp353_2/"

6. Enter your username and password... (you might have to retry multiple times) and you're in!

PSA: If you want to access it again, you can open a new Visual Studio Code window and click on the [SSH: ENCS-SERVER] link to jump straight to 6.

Accessing the ENCS Server remotely as such allows me to directly copy and paste my project's files into the ENCS Server's group directory. It also gives me access to the terminal, where I modified the file permissions.

The commands that I used to modify them were:

1. Set the sgid bit on all subdirectories:

   ```
   chmod g+s ./*
   ```

2. Show permissions in current directory:

   ```
   ls -l ./
   ```

3. For folders:

   ```
   chmod 755 ./<folder_name>
   ```

4. For files:

   ```
   chmod 644 ./<file_name>
   ```

files should have rw-r--r-- permissions
folders should have drwxr-sr-x permissions

For the ENCS Server, the details to connect to the database schema were:

```
// Database connection
$host = "npc353.encs.concordia.ca"; // Change if using a different host
$dbname = "npc353_2";
$username = "npc353_2";
$password = "WrestFrugallyErrant43";
```

Connecting to a Database

If there is a need to change the details to connect to the database schema, here are the locations in the code where changes are needed.

Lines 12 to 16: ./db-connect.php

Lines 17 to 21: ./login/needs-userorpass-change.php

Lines 6 to 10: ./login/logout.php

Lines 12 to 16: ./login/login.php

Lines 5 to 9: ./login/create-account.php

Lines 5 to 9: ./login/change-login.php

General Procedure for Coding the Website

Most PHP files are separated such that there is a php section at the top half, and the bottom half has an html section. This ensures some separation of the front end and the back end code.

If a PHP file is not separated as such, there is a good chance that it is purely back end code.

In the subdirectories other than login, the general procedure looks like this:

1. Top Half

    1.1 Include db-connect.php to connect to the database
    1.2 Conditions to handle form submissions (if any)
    1.3 Executing queries related to the database (this will be discussed after the list)
    1.4 If needed, redirect based on success or failure
    1.5 Error Handling

2. Bottom Half

    2.1 Include the necessary stylesheets in the HTML head
    2.2 Write the body code. There may be some PHP to implement the front end display logic.

2.3 If needed, some JavaScript is used at the bottom of the body, surrounded by script tags.

For 1.3 in particular, we will now describe the process of executing queries. We have used PDO as an interface to connect with the database, because it is flexible (as in, it lets us easily switch to other databases such as SQLite).

A good example of executing a query would be in ./login/create-account.php . We first fetch the data from the POST form submission. Then, if the email input is valid, we hash the password input, save the SQL query in a variable called $sql.

Here comes the important part: In the SQL query, you may notice placeholder values that start with “:” such as “:password”. We will use PDO to prepare the statement in $sql, then bind parameters to these placeholder values.

This prevents SQL injection attacks, because the operation of binding a parameter sanitizes the POST form’s values. Sanitizing means that we check if the value is of a certain type, before binding the parameter to the placeholder value.

In this case, by default, bindParam(...) will make sure that every POST form value is a String. We could also add a third parameter to bindParam(...) to specify what value type we should sanitize for.

Code Example for 1.3:

```
// Handle form submission
if ($_SERVER['REQUEST_METHOD'] == 'POST') {
    // Get input from the form and sanitize inputs
    $usernameInput = $_POST['username'] ?? '';
    $emailInput = $_POST['email'] ?? '';
    $firstNameInput = $_POST['firstName'] ?? '';
    $lastNameInput = $_POST['lastName'] ?? '';
    $DOBInput = $_POST['DOB'] ?? '';
    $passwordInput = $_POST['password'] ?? '';
    $accountTypeInput = $_POST['accountType'] ?? '';

    // Validate email to ensure it's a ProtonMail email address
    if (substr($emailInput, -15) !== '@protonmail.com') {
        $feedback = "Email must be a ProtonMail address (e.g., user@protonmail.com).";
    } else {
        // Hash the password securely
        $hashedPassword = password_hash($passwordInput, PASSWORD_DEFAULT);

        // SQL query with placeholders
        $sql = "INSERT INTO Members (
                    Password,
                    Username,
                    FirstName,
                    LastName,
                    Pseudonym,
                    Email,
                    Address,
                    DOB,
                    DateJoined,
                    Privilege,
                    AccountType,
                    Status,
                    NeedsPasswordChange,
                    NeedsUsernameChange
                ) VALUES (
                    :password,
                    :username,
                    :firstName,
                    :lastName,
                    NULL,
                    :email,
                    NULL,
                    :dob,
                    CURDATE(),
                    'Junior',
                    :accountType,
                    'Inactive',
                    FALSE,
                    FALSE
                )";

        try {
            // Prepare the statement
            $statement = $pdo->prepare($sql);

            // Bind values to the placeholders
            $statement->bindParam(':password', $hashedPassword);
            $statement->bindParam(':username', $usernameInput);
            $statement->bindParam(':firstName', $firstNameInput);
            $statement->bindParam(':lastName', $lastNameInput);
            $statement->bindParam(':email', $emailInput);
            $statement->bindParam(':dob', $DOBInput);
            $statement->bindParam(':accountType', $accountTypeInput);

            // Execute the query and check success
            if ($statement->execute()) {
                $feedback = "User successfully added!";
            } else {
                $feedback = "Failed to add the user.";
            }
        } catch (PDOException $e) {
```

```
            echo "Error: " . $e->getMessage();
        }
    }
}
```

SQL Queries

SQL for Database Creation

```
--  Create Members Relation
CREATE TABLE Members (
        MemberID INT PRIMARY KEY AUTO_INCREMENT,
        Password VARCHAR(255) NOT NULL,
        Username VARCHAR(255) UNIQUE NOT NULL,
        FirstName VARCHAR(255) NOT NULL,
        LastName VARCHAR(255) NOT NULL,
        Pseudonym VARCHAR(255),
        Email VARCHAR(255) UNIQUE NOT NULL,
        Address TEXT,
        DOB DATE NOT NULL,
        DateJoined DATE NOT NULL,
        Warnings INT DEFAULT 0,
        Suspensions INT DEFAULT 0,
        Fines INT DEFAULT 0,
        Privilege ENUM('Administrator', 'Senior', 'Junior') DEFAULT 'Junior',
        Status ENUM('Active', 'Inactive', 'Suspended') DEFAULT 'Active',
        AccountType ENUM('Real-person', 'Business'),
        LastLogin DATETIME,
        PrivateInformation TEXT,
        PublicInformation TEXT,
        GroupVisibilitySettings TEXT,
        ProfileVisibilitySettings TEXT,
        NeedsPasswordChange BOOLEAN DEFAULT TRUE,
        NeedsUsernameChange BOOLEAN DEFAULT TRUE,
        Profession VARCHAR(255), -- New field for profession
        Region VARCHAR(255), -- New field for region
        Interests TEXT -- New field for interests
);

--  Create BlockedMembers Relation
CREATE TABLE BlockedMembers (
        BlockerID INT NOT NULL,
        BlockedID INT NOT NULL,
        BlockedDate DATETIME DEFAULT CURRENT_TIMESTAMP,
        Reason TEXT,
        PRIMARY KEY (BlockerID, BlockedID),
        FOREIGN KEY (BlockerID) REFERENCES Members(MemberID) ON DELETE
CASCADE,
        FOREIGN KEY (BlockedID) REFERENCES Members(MemberID) ON DELETE
CASCADE
);

--  Create PromotionRequests Relation
CREATE TABLE PromotionRequests (
   RequestID INT AUTO_INCREMENT PRIMARY KEY,
   MemberID INT NOT NULL,
   Status ENUM('pending', 'approved', 'denied') DEFAULT 'pending',
   RequestDate TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
   FOREIGN KEY (MemberID) REFERENCES Members(MemberID)
);

--  Create Groups Relation
CREATE TABLE `Groups` (
        GroupID INT PRIMARY KEY AUTO_INCREMENT,
        GroupName VARCHAR(255) NOT NULL,
        OwnerID INT NOT NULL,
        CreationDate DATETIME NOT NULL,
        GroupType ENUM('Family', 'Friends', 'Colleagues', 'Other'),
        InterestCategory VARCHAR(255), -- New field for group interest category
        Region VARCHAR(255), -- New field for group region
        FOREIGN KEY (OwnerID) REFERENCES Members(MemberID) ON DELETE
CASCADE
);

--  Create GroupMembers Relation
CREATE TABLE GroupMembers (
        GroupMemberID INT PRIMARY KEY AUTO_INCREMENT,
        GroupID INT NOT NULL,
        MemberID INT NOT NULL,
        Role ENUM('Admin', 'Member') DEFAULT 'Member',
        DateAdded DATE NOT NULL,
        FOREIGN KEY (GroupID) REFERENCES `Groups`(GroupID) ON DELETE
CASCADE,
        FOREIGN KEY (MemberID) REFERENCES Members(MemberID) ON DELETE
CASCADE,
        UNIQUE (GroupID, MemberID)
);
```

```
--  Create GroupJoinRequests Relation
CREATE TABLE GroupJoinRequests (
        RequestID INT PRIMARY KEY AUTO_INCREMENT,
        GroupID INT NOT NULL,
        MemberID INT NOT NULL,
        RequestDate DATE NOT NULL DEFAULT (CURDATE()),
        Status ENUM('Pending', 'Approved', 'Rejected') DEFAULT 'Pending',
        ReviewedBy INT,
        ReviewDate DATE,
        ReviewComments TEXT,
        FOREIGN KEY (GroupID) REFERENCES `Groups`(GroupID) ON DELETE
CASCADE,
        FOREIGN KEY (MemberID) REFERENCES Members(MemberID) ON DELETE
CASCADE,
        FOREIGN KEY (ReviewedBy) REFERENCES Members(MemberID) ON DELETE
SET NULL
);

--  Create Posts Relation
CREATE TABLE Posts (
        PostID INT PRIMARY KEY AUTO_INCREMENT,
        AuthorID INT NOT NULL,
        Content TEXT,
        CommentsCount INT DEFAULT 0,
        PostDate DATETIME NOT NULL,
        VisibilitySettings ENUM('Public', 'Group', 'Private') DEFAULT 'Group',
        FOREIGN KEY (AuthorID) REFERENCES Members(MemberID) ON DELETE
CASCADE
);

--  Create PostGroups Relation
CREATE TABLE PostGroups (
   PostID INT,
   GroupID INT,
   PRIMARY KEY (PostID, GroupID),
   FOREIGN KEY (PostID) REFERENCES Posts(PostID) ON DELETE CASCADE,
   FOREIGN KEY (GroupID) REFERENCES `Groups`(GroupID) ON DELETE CASCADE
);

--  Create PostMedia Relation
CREATE TABLE PostMedia (
        MediaID INT PRIMARY KEY AUTO_INCREMENT,
        PostID INT NOT NULL,
        MediaType ENUM('Image', 'Video') NOT NULL,
        MediaURL VARCHAR(255) NOT NULL,
        UploadedAt DATETIME NOT NULL,
        FOREIGN KEY (PostID) REFERENCES Posts(PostID) ON DELETE CASCADE
);
```

```
-- Table to track likes
CREATE TABLE PostLikes (
    PostID INT,
    UserID INT,
    PRIMARY KEY (PostID, UserID),
    FOREIGN KEY (PostID) REFERENCES Posts(PostID) ON DELETE CASCADE,
    FOREIGN KEY (UserID) REFERENCES Members(MemberID) ON DELETE CASCADE
);

-- Table to track dislikes
CREATE TABLE PostDislikes (
    PostID INT,
    UserID INT,
    PRIMARY KEY (PostID, UserID),
    FOREIGN KEY (PostID) REFERENCES Posts(PostID) ON DELETE CASCADE,
    FOREIGN KEY (UserID) REFERENCES Members(MemberID) ON DELETE CASCADE
);

--  Create Comments Relation
CREATE TABLE Comments (
        CommentID INT PRIMARY KEY AUTO_INCREMENT,
        PostID INT NOT NULL,
        AuthorID INT NOT NULL,
        Content TEXT NOT NULL,
        CreationDate DATETIME NOT NULL,
        FOREIGN KEY (PostID) REFERENCES Posts(PostID) ON DELETE CASCADE,
        FOREIGN KEY (AuthorID) REFERENCES Members(MemberID) ON DELETE
CASCADE
);

--  Create Relationships Relation
CREATE TABLE Relationships (
        RelationshipID INT PRIMARY KEY AUTO_INCREMENT,
        SenderMemberID INT NOT NULL,
        ReceiverMemberID INT NOT NULL,
        RelationshipType ENUM('Family', 'Friend', 'Colleague') NOT NULL,
        CreationDate DATE NOT NULL,
        Status ENUM('Active', 'Pending') DEFAULT 'Active',
        FOREIGN KEY (SenderMemberID) REFERENCES Members(MemberID) ON
DELETE CASCADE,
        FOREIGN KEY (ReceiverMemberID) REFERENCES Members(MemberID) ON
DELETE CASCADE,
        UNIQUE (SenderMemberID, ReceiverMemberID)
);
```

```
--  Create Messages Relation
CREATE TABLE Messages (
    MessageID INT PRIMARY KEY AUTO_INCREMENT,
    SenderID INT NOT NULL,
    ReceiverID INT NOT NULL,
    Content TEXT NOT NULL,
    DataSent DATETIME NOT NULL,
    ReadStatus ENUM('Unread', 'Read', 'Deleted') DEFAULT 'Unread',
    FOREIGN KEY (SenderID) REFERENCES Members(MemberID) ON DELETE CASCADE,
    FOREIGN KEY (ReceiverID) REFERENCES Members(MemberID) ON DELETE CASCADE
);
```

```
--  Create Email Relation
CREATE TABLE Email (
    EmailID INT PRIMARY KEY AUTO_INCREMENT,
    SenderID INT NOT NULL,
    ReceiverID INT NOT NULL,
    Subject VARCHAR(255),
    Body TEXT,
    DateSent DATETIME,
    ReadStatus TINYINT(1) DEFAULT 0 NOT NULL,
    FOREIGN KEY (SenderID) REFERENCES Members(MemberID),
    FOREIGN KEY (ReceiverID) REFERENCES Members(MemberID)
);
```

```
-- Create Events relation
CREATE TABLE Events (
    EventID INT PRIMARY KEY AUTO_INCREMENT,
    GroupID INT NOT NULL,
    EventTitle VARCHAR(255) NOT NULL,
    EventDescription TEXT,
    EventCreatorID INT NOT NULL,
    EventStatus ENUM('Pending', 'Scheduled', 'Cancelled') DEFAULT 'Pending', -- Event status
    CreationDate DATETIME DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (GroupID) REFERENCES `Groups`(GroupID) ON DELETE CASCADE,
    FOREIGN KEY (EventCreatorID) REFERENCES Members(MemberID) ON DELETE CASCADE
);
```

```
-- Create EventVotingOptions relation
CREATE TABLE EventVotingOptions (
	OptionID INT PRIMARY KEY AUTO_INCREMENT,
	EventID INT NOT NULL,
	OptionDate DATE,  -- Date of the event
	OptionTime TIME,  -- Time of the event
	 OptionPlace VARCHAR(255),  -- Location of the event
	IsSuggestedByMember BOOLEAN DEFAULT FALSE,
	FOREIGN KEY (EventID) REFERENCES Events(EventID) ON DELETE CASCADE
);



-- Create EventVotes relation
CREATE TABLE EventVotes (
	VoteID INT PRIMARY KEY AUTO_INCREMENT,
	EventID INT NOT NULL,
	MemberID INT NOT NULL,
	OptionID INT NOT NULL,
	VoteDate DATETIME DEFAULT CURRENT_TIMESTAMP,  -- Timestamp of vote
	FOREIGN KEY (EventID) REFERENCES Events(EventID) ON DELETE CASCADE,
	FOREIGN KEY (MemberID) REFERENCES Members(MemberID) ON DELETE CASCADE,
	FOREIGN KEY (OptionID) REFERENCES EventVotingOptions(OptionID) ON DELETE CASCADE,
	UNIQUE (EventID, MemberID, OptionID)  -- Prevents multiple votes by the same member for the same option
);

-- Create GiftExchange relation
CREATE TABLE GiftExchange (
	GiftExchangeID INT PRIMARY KEY AUTO_INCREMENT,
	GroupID INT NOT NULL, -- To track which group or family is organizing this
	EventName VARCHAR(255) NOT NULL, -- e.g., "Christmas Secret Santa"
	EventDate DATETIME NOT NULL, -- The date of the exchange event
	MaxBudget DECIMAL(10, 2), -- Max budget for the gift
	Status ENUM('Pending', 'Ongoing', 'Completed') DEFAULT 'Pending', -- Status of the gift exchange
	CreationDate DATETIME DEFAULT CURRENT_TIMESTAMP,
	FOREIGN KEY (GroupID) REFERENCES `Groups`(GroupID) ON DELETE CASCADE
);
```

```
-- Create GiftExchangeParticipants relation
CREATE TABLE GiftExchangeParticipants (
	ParticipantID INT PRIMARY KEY AUTO_INCREMENT,
	GiftExchangeID INT NOT NULL,
	MemberID INT NOT NULL,
	AssignedToMemberID INT, -- The person the member will buy a gift for (Secret Santa pair)
	GiftPreference TEXT, -- Optional: Members can specify gift preferences
	ExchangeStatus ENUM('Assigned', 'Gift Purchased', 'Gift Given', 'Completed') DEFAULT 'Assigned', -- Status of the exchange for each participant
	PaymentAmount DECIMAL(10, 2) DEFAULT 0,
	FOREIGN KEY (GiftExchangeID) REFERENCES GiftExchange(GiftExchangeID) ON DELETE CASCADE,
	FOREIGN KEY (MemberID) REFERENCES Members(MemberID) ON DELETE CASCADE,
	FOREIGN KEY (AssignedToMemberID) REFERENCES Members(MemberID) ON DELETE SET NULL, -- If gift assignment is not yet made
	UNIQUE (GiftExchangeID, MemberID)
);

-- Create Warnings Relation
CREATE TABLE Warnings (
	WarningID INT AUTO_INCREMENT PRIMARY KEY,
	MemberID INT,
	PostID INT,
	MessageID INT,
	CommentID INT,
	EmailID INT,
	Reason VARCHAR(255) NOT NULL,
	CreatedAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
	IssuedBy INT,
	WarningType ENUM('Direct', 'Post', 'Comment', 'Message', 'EmailMessage') NOT NULL,
	FOREIGN KEY (MemberID) REFERENCES Members(MemberID) ON DELETE CASCADE,
	FOREIGN KEY (PostID) REFERENCES Posts(PostID) ON DELETE CASCADE,
	FOREIGN KEY (MessageID) REFERENCES Messages(MessageID) ON DELETE CASCADE,
	FOREIGN KEY (CommentID) REFERENCES Comments(CommentID) ON DELETE CASCADE,
	FOREIGN KEY (EmailID) REFERENCES Email(EmailID) ON DELETE CASCADE,
	FOREIGN KEY (IssuedBy) REFERENCES Members(MemberID) ON DELETE SET NULL
);
```

```
--  Create Payments Relation
CREATE TABLE Payments (
        PaymentID INT PRIMARY KEY AUTO_INCREMENT,
        MemberID INT NOT NULL,
        Amount DECIMAL(10, 2) NOT NULL,
        PaymentDate DATE NOT NULL,
        Description TEXT,
        FOREIGN KEY (MemberID) REFERENCES Members(MemberID) ON DELETE
CASCADE
);

--  Create Suspensions Relation
CREATE TABLE Suspensions (
   SuspensionID INT PRIMARY KEY AUTO_INCREMENT,
   MemberID INT,
   StartDate TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
   EndDate TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
   Reason VARCHAR(255),
   IssuedBy INT,  -- The member who issued the suspension, probably an admin
   FOREIGN KEY (MemberID) REFERENCES Members(MemberID) ON DELETE
CASCADE,
   FOREIGN KEY (IssuedBy) REFERENCES Members(MemberID) ON DELETE SET NULL
);

-- Create Reports Relation
CREATE TABLE Reports (
        ReportID INT AUTO_INCREMENT PRIMARY KEY,
        PostID INT,
        MessageID INT,
        CommentID INT,
        EmailID INT,
        ReportType ENUM('Post', 'Comment', 'Message', 'EmailMessage') NOT NULL,
        AdminID INT NOT NULL,  -- Administrator handling the report
        ReportStatus ENUM('Pending', 'Approved', 'Disapproved') DEFAULT 'Pending',
        ResolutionDetails TEXT,  -- Details on how the report was resolved
        CreatedAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
        ResolvedAt TIMESTAMP NULL DEFAULT NULL,
        FOREIGN KEY (AdminID) REFERENCES Members(MemberID) ON DELETE
CASCADE,
        FOREIGN KEY (PostID) REFERENCES Posts(PostID) ON DELETE CASCADE,
        FOREIGN KEY (MessageID) REFERENCES Messages(MessageID) ON DELETE
CASCADE,
        FOREIGN KEY (CommentID) REFERENCES Comments(CommentID) ON DELETE
CASCADE,
        FOREIGN KEY (EmailID) REFERENCES Email(EmailID) ON DELETE CASCADE
);
```

```
--  Create Notifications Relation
CREATE TABLE Notifications (
        NotificationID INT PRIMARY KEY AUTO_INCREMENT,
        MemberID INT NOT NULL,
        Content TEXT NOT NULL,
        NotificationDate DATETIME NOT NULL,
        Type ENUM('Unread', 'Read') DEFAULT 'Unread',
        FOREIGN KEY (MemberID) REFERENCES Members(MemberID) ON DELETE
CASCADE
);

--  Insert a default Member in Members with Username ‘admin’ and Password ‘admin’
INSERT INTO Members (
        Password,
        Username,
        FirstName,
        LastName,
        Pseudonym,
        Email,
        Address,
        DOB,
        DateJoined,
        Privilege,
        AccountType,
        Status
)
VALUES (
        '$2y$10$AjecbbHv73ewhzCL0tbKueen1iEa8GG7CF0C86WWORP7ymAM1AEZe',
        'admin',
        'Admin',
        'User',
        NULL,
        'admin@protonmail.com',
        NULL,
        '1970-01-01',
        CURDATE(),
        'Administrator',
        'Real-person',
        'Inactive'
);
```

SQL for Database Population

```
--  Insert Members (EVERY PASSWORD IS JUST ‘admin’ BUT ENCRYPTED)
INSERT INTO Members (MemberID, Password, Username, FirstName, LastName,
Pseudonym, Email, Address, DOB, DateJoined, Warnings, Suspensions, Fines, Privilege,
Status, AccountType, LastLogin, PrivateInformation, PublicInformation,
GroupVisibilitySettings, ProfileVisibilitySettings, NeedsPasswordChange,
NeedsUsernameChange, Profession, Region, Interests)
VALUES
(2, '$2y$10$qPiIbWr2DUQyQAWapi9Wqu94xX92lsCFvG4psuoY4N1qyLQLC8spa',
'admin123', 'Admin', 'User', '', 'admin123@protonmail.com', '', '1970-01-01', '2024-12-05', 0,
0, 0, 'Administrator', 'Inactive', 'Real-person', '2024-12-06 15:16:06', '', '', 'Public', 'Public', 0,
0, '', '', ''),
(3, '$2y$10$I6YhXpjFVYmPUva4WGA5DeB61q0Wo4yb9H/2uTZNvDkzwg1q5be6y',
'admin12', 'Admin', 'User', '', 'admin12@protonmail.com', '', '1970-01-01', '2024-12-05', 0, 0,
0, 'Administrator', 'Inactive', 'Real-person', '2024-12-06 15:17:03', '', '', '', '', 0, 0, '', '', ''),
(4, '$2y$10$ANaZZOTsMOkpE3DM/u7wOeGyAgQI0U0bknYou4arGQUR/CRg53nAC',
'admin1', 'Admin', 'User', '', 'admin1@protonmail.com', '', '1970-01-01', '2024-12-05', 0, 0, 0,
'Administrator', 'Inactive', 'Real-person', '2024-12-06 14:59:29', '', '', '', '', 0, 0, '', '', ''),
(5, '$2y$10$tVW4GdZZrNkSsVRisBhFIem4LaO9voPoKFMyW8/kEsTWTI3NIqEAm', 'j1', 'j',
'1', '', 'j1@protonmail.com', '', '1997-04-23', '2024-12-05', 0, 0, 0, 'Junior', 'Inactive',
'Real-person', '2024-12-06 15:08:15', '', '', '', '', 0, 0, '', '', ''),
(6, '$2y$10$e1IX3NrmDoAMcd/zIi72LuR4c58zHSTLjM7m0AvKRWpSMnaipCirq', 'j2', 'j', '2',
'', 'j2@protonmail.com', '', '1923-09-02', '2024-12-05', 0, 0, 0, 'Junior', 'Inactive', 'Real-person',
'2024-12-06 00:09:30', '', '', '', '', 0, 0, '', '', ''),
(7, '$2y$10$wHo93MQhMgdHKaQUYv9Tv.f4IEBNuSpFQZ.gmOECgTqgTkTpE.UWa', 's1',
's', '1', '', 's1@protonmail.com', '', '1923-03-08', '2024-12-05', 0, 0, 0, 'Senior', 'Inactive',
'Real-person', '2024-12-06 00:26:28', '', '', '', '', 0, 0, '', '', ''),
(8, '$2y$10$fSC9RXPNqMB0/G067Se0PuhzWQzTg48Sf42ezVMWAhYca.Ps4QcB.', 's2', 's',
'2', '', 's2@protonmail.com', '', '1958-02-01', '2024-12-05', 0, 0, 0, 'Senior', 'Inactive',
'Real-person', '2024-12-06 15:07:56', '', '', '', '', 0, 0, '', '', ''),
(9, '$2y$10$YyczqZsdcR/zX61pUXSMruU3pLxp8XP5Rd0No46ozS.HeIlc7fQvq', 's3', 's', '3',
'', 's3@protonmail.com', '', '1923-03-23', '2024-12-05', 0, 0, 0, 'Senior', 'Inactive',
'Real-person', '2024-12-06 00:11:07', '', '', '', '', 0, 0, '', '', ''),
(10, '$2y$10$AcMRB2HMa66dmB/82WpZmOsQ7YyQZWYUvdsI1U8F63h6bQ2G17Yra',
'j3', 'j', '3', '', 'j3@protonmail.com', '', '1932-02-10', '2024-12-05', 3, 1, 0, 'Junior', 'Suspended',
'Real-person', '2024-12-06 00:09:45', '', '', '', '', 0, 0, '', '', ''),
(11, '$2y$10$T3MqZ47C5V.KqiKOPzIvA.oT8caUR2RsHmIiO8wNqLpo6gY6KKXIi',
'test-fraud', 'test', 'fraud', '', 'test-fraud@protonmail.com', '', '2004-03-31', '2024-12-06', 3, 1, 0,
'Junior', 'Suspended', 'Real-person', NULL, '', '', '', '', 0, 0, '', '', ''),
(12, '$2y$10$7Knfo3iSP6.0en2D.rJ/NOp0Q2e4Lik/Gx2BGARY6F8QTmJ/R.jha',
'test-fraud-business', 'test', 'fraud-business', '', 'test-fraud-business@protonmail.com', '',
'2004-03-01', '2024-12-06', 6, 2, 4, 'Junior', 'Suspended', 'Business', '2024-12-06 01:28:10', '',
'', '', '', 0, 0, '', '', ''),
(13, '$2y$10$ni3lhNx/Epz5c/6yCTUwgudr86FJLPOfJAqVBRtZk16NhXtL.IisG', 'new-user',
'new', 'user', '', 'new-user@protonmail.com', '', '2004-12-23', '2024-12-06', 0, 0, 0, 'Junior',
'Inactive', 'Real-person', '2024-12-06 15:03:01', '', '', '', '', 0, 0, '', '', '');

--  Insert BlockedMembers (NOTHING)


--  Insert PromotionRequests
INSERT INTO PromotionRequests (RequestID, MemberID, Status, RequestDate) VALUES
(1, 7, 'denied', '2024-12-05 22:48:33');
```

```
--  Insert Groups
INSERT INTO `Groups` (GroupID, GroupName, OwnerID, CreationDate, GroupType,
InterestCategory, Region)
VALUES
(3, 'Admin Group', 2, '2024-12-05 22:56:58', 'Family', 'admin', 'Admin'),
(5, 'Senior Group', 8, '2024-12-05 23:23:36', 'Friends', 'Senior', 'Senior'),
(6, 'Junior Group', 2, '2024-12-05 23:36:34', 'Friends', 'Junior', 'Junior'),
(9, 'Test Group', 8, '2024-12-06 15:03:48', 'Family', '', '');

--  Insert GroupMembers
INSERT INTO GroupMembers (GroupMemberID, GroupID, MemberID, Role, DateAdded)
VALUES
(3, 3, 2, 'Admin', '2024-12-05'),
(5, 3, 3, 'Member', '2024-12-05'),
(7, 5, 8, 'Admin', '2024-12-05'),
(8, 5, 9, 'Member', '2024-12-05'),
(9, 6, 2, 'Admin', '2024-12-05'),
(10, 6, 5, 'Admin', '2024-12-05'),
(11, 6, 6, 'Member', '2024-12-05'),
(12, 6, 10, 'Member', '2024-12-05'),
(13, 5, 2, 'Admin', '2024-12-05'),
(14, 5, 7, 'Member', '2024-12-05'),
(17, 9, 8, 'Admin', '2024-12-06'),
(19, 9, 5, 'Member', '2024-12-06');

--  Insert GroupJoinRequests
INSERT INTO GroupJoinRequests (RequestID, GroupID, MemberID, RequestDate, Status,
ReviewedBy, ReviewDate, ReviewComments)
VALUES
(1, 3, 5, '2024-12-05', 'Approved', 2, '2024-12-06', 'Sample Text'),
(2, 3, 3, '2024-12-05', 'Approved', 2, '2024-12-06', 'Sample Text'),
(3, 5, 9, '2024-12-05', 'Approved', 8, '2024-12-06', 'Sample Text'),
(4, 6, 5, '2024-12-05', 'Approved', 2, '2024-12-06', 'Sample Text'),
(5, 6, 6, '2024-12-05', 'Approved', 5, '2024-12-06', 'Sample Text'),
(6, 6, 10, '2024-12-05', 'Approved', 5, '2024-12-06', 'Sample Text'),
(7, 5, 2, '2024-12-05', 'Approved', 8, '2024-12-06', 'Sample Text'),
(8, 5, 7, '2024-12-05', 'Approved', 8, '2024-12-06', 'Sample Text'),
(9, 9, 5, '2024-12-06', 'Approved', 2, '2024-12-06', 'Sample Text'),
(10, 9, 5, '2024-12-06', 'Approved', 2, '2024-12-06', 'Sample Text');

--  Insert Relationships
INSERT INTO Relationships (RelationshipID, SenderMemberID, ReceiverMemberID,
RelationshipType, CreationDate, Status)
VALUES
(1, 8, 7, 'Friend', '2024-12-05', 'Active'),
(2, 8, 9, 'Friend', '2024-12-05', 'Active'),
(3, 8, 2, 'Friend', '2024-12-05', 'Active'),
```

```sql
(4, 8, 3, 'Friend', '2024-12-05', 'Active'),
(5, 5, 6, 'Friend', '2024-12-05', 'Active'),
(6, 5, 10, 'Friend', '2024-12-05', 'Active'),
(7, 5, 2, 'Friend', '2024-12-05', 'Active'),
(8, 5, 3, 'Friend', '2024-12-05', 'Active'),
(9, 9, 7, 'Friend', '2024-12-05', 'Active'),
(11, 6, 10, 'Friend', '2024-12-05', 'Active'),
(13, 2, 3, 'Friend', '2024-12-06', 'Active');

--  Insert Posts
INSERT INTO Posts (PostID, AuthorID, Content, CommentsCount, PostDate,
VisibilitySettings)
VALUES
(1, 2, 'Admin Group Only Post', 0, '2024-12-05 23:52:44', 'Group'),
(2, 2, 'Junior Group Only Post', 0, '2024-12-05 23:52:54', 'Group'),
(3, 2, 'Senior Group Only Post', 0, '2024-12-05 23:53:03', 'Group'),
(8, 5, 'j1 Post (Private)', 0, '2024-12-06 00:04:53', 'Private'),
(10, 2, 'admin123 Post (Private)', 0, '2024-12-06 00:09:02', 'Private'),
(11, 3, 'admin12 Post (Private)', 0, '2024-12-06 00:09:24', 'Private'),
(12, 6, 'j2 Post (Private)', 0, '2024-12-06 00:09:39', 'Private'),
(13, 10, 'j3 Post (Private)', 0, '2024-12-06 00:09:56', 'Private'),
(14, 7, 's1 Post (Private)', 0, '2024-12-06 00:10:25', 'Private'),
(15, 8, 's2 Post (Private)', 0, '2024-12-06 00:10:53', 'Private'),
(16, 9, 's3 Post (Private)', 0, '2024-12-06 00:11:20', 'Private'),
(17, 2, 'Sample Text (Public)', 0, '2024-12-06 00:15:34', 'Public'),
(18, 2, 'Image Post (Public)', 2, '2024-12-06 00:19:01', 'Public'),
(19, 2, 'Video Post (Public)', 0, '2024-12-06 00:19:41', 'Public'),
(23, 8, 'public post from s2', 0, '2024-12-06 15:05:32', 'Public'),
(24, 8, 'image post', 0, '2024-12-06 15:05:59', 'Public'),
(25, 8, 'video post', 0, '2024-12-06 15:06:11', 'Public'),
(26, 8, 'group post', 0, '2024-12-06 15:07:08', 'Group');



--  Insert PostGroups
INSERT INTO PostGroups (PostID, GroupID)
VALUES
(1, 3),
(3, 5),
(2, 6),
(26, 9);

--  Insert PostMedia
INSERT INTO PostMedia (MediaID, PostID, MediaType, MediaURL, UploadedAt)
VALUES
(1, 18, 'Image', '../uploads/media_67528945cedb33.62488687.png', '2024-12-06 00:19:01'),
(2, 19, 'Video', '../uploads/media_6752896d8b88b7.13569071.mkv', '2024-12-06 00:19:41'),
(5, 24, 'Image', '../uploads/media_67535927acfe02.45653758.jpg', '2024-12-06 15:05:59'),
(6, 25, 'Video', '../uploads/media_67535933a5acb0.21770493.mkv', '2024-12-06 15:06:11');

--  Insert PostLikes
INSERT INTO PostLikes (PostID, UserID)
VALUES
(17, 5),
(18, 5),
(17, 7),
(15, 9),
(17, 12);

--  Insert PostDislikes (NOTHING)


--  Insert Comments
INSERT INTO Comments (CommentID, PostID, AuthorID, Content, CreationDate)
VALUES
(2, 18, 7, 'hello from s1', '2024-12-06 00:20:59'),
(4, 18, 8, 'hello from j1', '2024-12-06 00:25:47');

--  Insert Messages
INSERT INTO Messages (MessageID, SenderID, ReceiverID, Content, DataSent,
ReadStatus)
VALUES
(1, 2, 3, 'hello admin12', '2024-12-06 00:14:56', 'Unread'),
(2, 3, 2, 'hello admin123', '2024-12-06 00:15:12', 'Unread'),
(3, 2, 3, 'h', '2024-12-06 00:28:19', 'Unread'),
(4, 2, 3, 'e', '2024-12-06 00:28:21', 'Unread'),
(5, 2, 3, 'l', '2024-12-06 00:28:22', 'Unread'),
(6, 2, 3, 'l', '2024-12-06 00:28:23', 'Unread'),
(7, 2, 3, 'o', '2024-12-06 00:28:24', 'Unread'),
(8, 2, 3, 'test chat', '2024-12-06 15:10:17', 'Unread'),
(9, 3, 2, 'test chat 2\r\n', '2024-12-06 15:10:32', 'Unread');

--  Insert Email
INSERT INTO Email (EmailID, SenderID, ReceiverID, Subject, Body, DateSent, ReadStatus)
VALUES
(1, 2, 3, 'TEST SEND EMAIL', 'TEST SEND EMAIL\r\n\r\n(This should be in admin12''s
Inbox.)', '2024-12-06 00:29:54', 1),
(2, 3, 2, 'TEST SEND EMAIL', 'TEST SEND EMAIL\r\n\r\n(This should be in admin123''s
Inbox.)', '2024-12-06 00:30:39', 1),
(3, 2, 3, 'popup test', 'popup test', '2024-12-06 13:52:51', 1),
(4, 2, 3, 'demo sent email', 'a', '2024-12-06 15:14:01', 1);

--  Insert Events
INSERT INTO Events (EventID, GroupID, EventTitle, EventDescription, EventCreatorID,
EventStatus, CreationDate)
VALUES
```

```
(1, 3, 'Admin Event (Expired)', 'Admin Event (Expired)', 2, 'Scheduled', '2024-12-06
00:33:28'),
(2, 3, 'Admin Event (Upcoming)', 'Admin Event (Upcoming)', 2, 'Scheduled', '2024-12-06
00:46:37'),
(3, 3, 'Admin Event (Upcoming 2)', 'Admin Event (Upcoming 2)', 2, 'Scheduled', '2024-12-06
01:16:33'),
(4, 3, 'Test Event', 'sample text', 2, 'Scheduled', '2024-12-06 15:11:20');

--  Insert EventVotingOptions
INSERT INTO EventVotingOptions (OptionID, EventID, OptionDate, OptionTime,
OptionPlace, IsSuggestedByMember)
VALUES
(1, 1, '2024-12-01', '00:00:00', 'Test Street 12345', 0),
(2, 1, '2024-12-01', '00:15:00', 'Test Street 1234', 1),
(3, 2, '3000-01-01', '00:48:00', 'Upcoming Street 111', 1),
(4, 2, '3000-02-28', '00:49:00', 'Upcoming Street 111', 0),
(5, 3, '2050-10-02', '01:18:00', 'Upcoming Street 222', 0),
(6, 4, '2999-12-31', '15:11:00', 'Test Street', 0),
(7, 4, '2999-12-30', '15:11:00', 'Test 12345', 1);

--  Insert EventVotes
INSERT INTO EventVotes (VoteID, EventID, MemberID, OptionID, VoteDate)
VALUES
(1, 1, 3, 2, '2024-12-06 00:35:11'),
(2, 1, 2, 2, '2024-12-06 00:35:34'),
(3, 2, 3, 3, '2024-12-06 00:48:02'),
(4, 3, 2, 5, '2024-12-06 01:16:54'),
(5, 4, 3, 7, '2024-12-06 15:12:40'),
(6, 4, 2, 7, '2024-12-06 15:13:00');



--  Insert GiftExchange
INSERT INTO GiftExchange (GiftExchangeID, GroupID, EventName, EventDate,
MaxBudget, Status, CreationDate)
VALUES
(1, 3, 'Test Gift Exchange', '2024-12-14 01:20:00', 0.0, 'Completed', '2024-12-06 01:18:37'),
(2, 3, 'Gift Exchange', '2024-12-06 15:14:00', 4000.0, 'Completed', '2024-12-06 15:15:10');

--  Insert GiftExchangeParticipants
INSERT INTO GiftExchangeParticipants (ParticipantID, GiftExchangeID, MemberID,
AssignedToMemberID, GiftPreference, ExchangeStatus, PaymentAmount)
VALUES
(1, 1, 2, 3, 'Tie', 'Completed', 5000.0),
(2, 1, 3, 2, 'Ring', 'Completed', 5000.0),
(4, 2, 2, 3, 'Hello', 'Completed', 5000.0),
(5, 2, 3, 2, 'World', 'Completed', 1000.0);
```

```
--  Insert Warnings
INSERT INTO Warnings (WarningID, MemberID, PostID, MessageID, CommentID, EmailID,
Reason, CreatedAt, IssuedBy, WarningType)
VALUES
(1, 11, NULL, NULL, NULL, NULL, 'test-fraud', '2024-12-06 01:23:55', 2, 'Direct'),
(2, 11, NULL, NULL, NULL, NULL, 'test-fraud', '2024-12-06 01:23:59', 2, 'Direct'),
(3, 11, NULL, NULL, NULL, NULL, 'test-fraud', '2024-12-06 01:24:07', 2, 'Direct'),
(4, 12, NULL, NULL, NULL, NULL, 'business fraud', '2024-12-06 01:25:41', 2, 'Direct'),
(5, 12, NULL, NULL, NULL, NULL, 'business fraud', '2024-12-06 01:25:43', 2, 'Direct'),
(6, 12, NULL, NULL, NULL, NULL, 'business fraud', '2024-12-06 01:25:54', 2, 'Direct'),
(7, 12, NULL, NULL, NULL, NULL, 'business fraud', '2024-12-06 01:26:01', 2, 'Direct'),
(8, 12, NULL, NULL, NULL, NULL, 'business fraud', '2024-12-06 01:26:08', 2, 'Direct'),
(9, 12, NULL, NULL, NULL, NULL, 'business fraud', '2024-12-06 01:32:16', 2, 'Direct'),
(10, 10, NULL, NULL, NULL, NULL, 'warning', '2024-12-06 15:00:56', 2, 'Direct'),
(11, 10, NULL, NULL, NULL, NULL, 'warning', '2024-12-06 15:01:00', 2, 'Direct'),
(12, 10, NULL, NULL, NULL, NULL, 'warning', '2024-12-06 15:01:04', 2, 'Direct');

--  Insert Payments
INSERT INTO Payments (PaymentID, MemberID, Amount, PaymentDate, Description)
VALUES
(4, 12, 200.0, '2024-12-06', 'Fine for excessive warnings (Fine #4)');

--  Insert Suspensions
INSERT INTO Suspensions (SuspensionID, MemberID, StartDate, EndDate, Reason,
IssuedBy)
VALUES
(1, 11, '2024-12-06 01:24:07', '2024-12-14 01:24:07', 'Excessive warnings', 2),
(2, 12, '2024-12-06 01:26:08', '2024-12-06 01:27:51', 'Excessive fines (more than 2)', 2),
(3, 12, '2024-12-06 01:32:16', '2025-01-05 01:32:16', 'Excessive fines (more than 2)', 2),
(4, 10, '2024-12-06 15:01:04', '2024-12-14 15:01:04', 'Excessive warnings', 2);
```

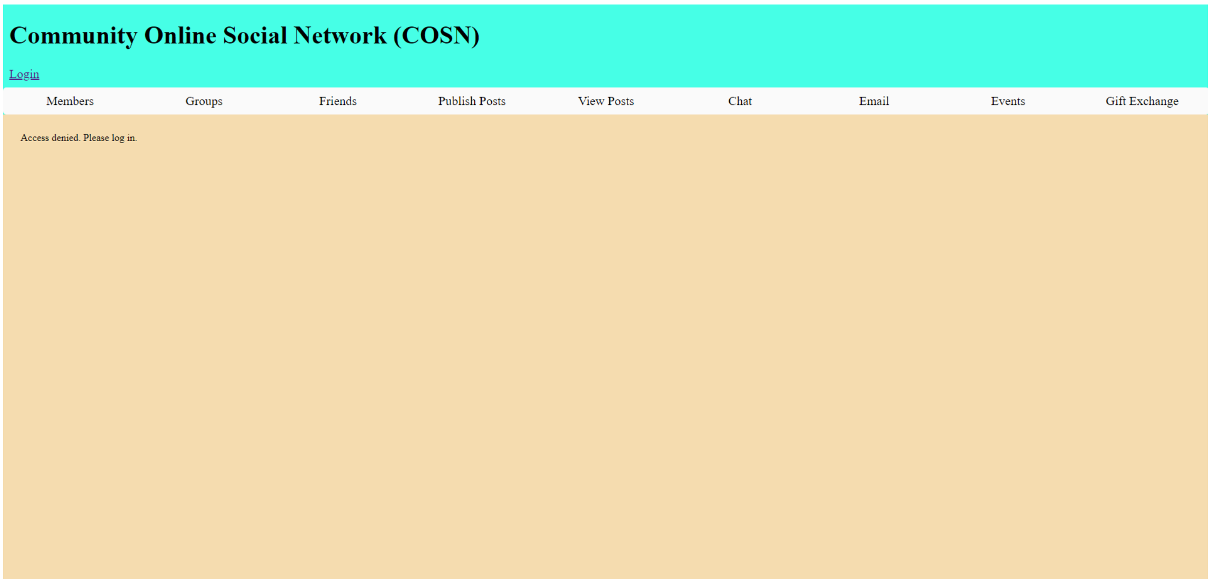
User Interface Snapshots and User Manual

Home Page

This is the Home Page of the Community Online Social Network. At the moment of this screenshot, access to the website's contents is denied due to the user not being logged in. To log into the website, we must click the "Login" button on the top left of the screen capture.

See the User Manual - Section 1: Accounts to learn the process of logging in.

NOTE: CLICKING ON "Community Online Social Network (COSN)" WILL REDIRECT YOU TO THE HOME PAGE.



Community Online Social Network (COSN)

Login

Access denied. Please log in.

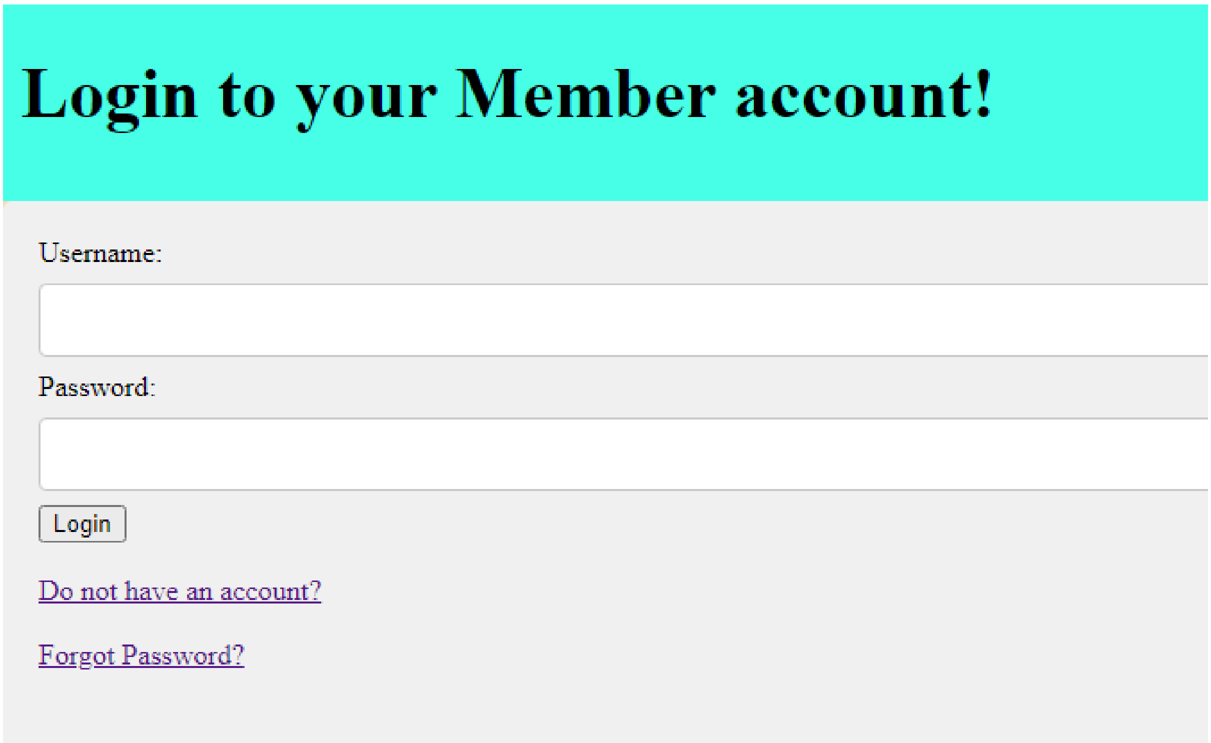
User Manual

Section 1: Accounts

I. Creating Account

From the Home Page, click the “Login” link on the top left.

You should be redirected to the login page, which looks like this:

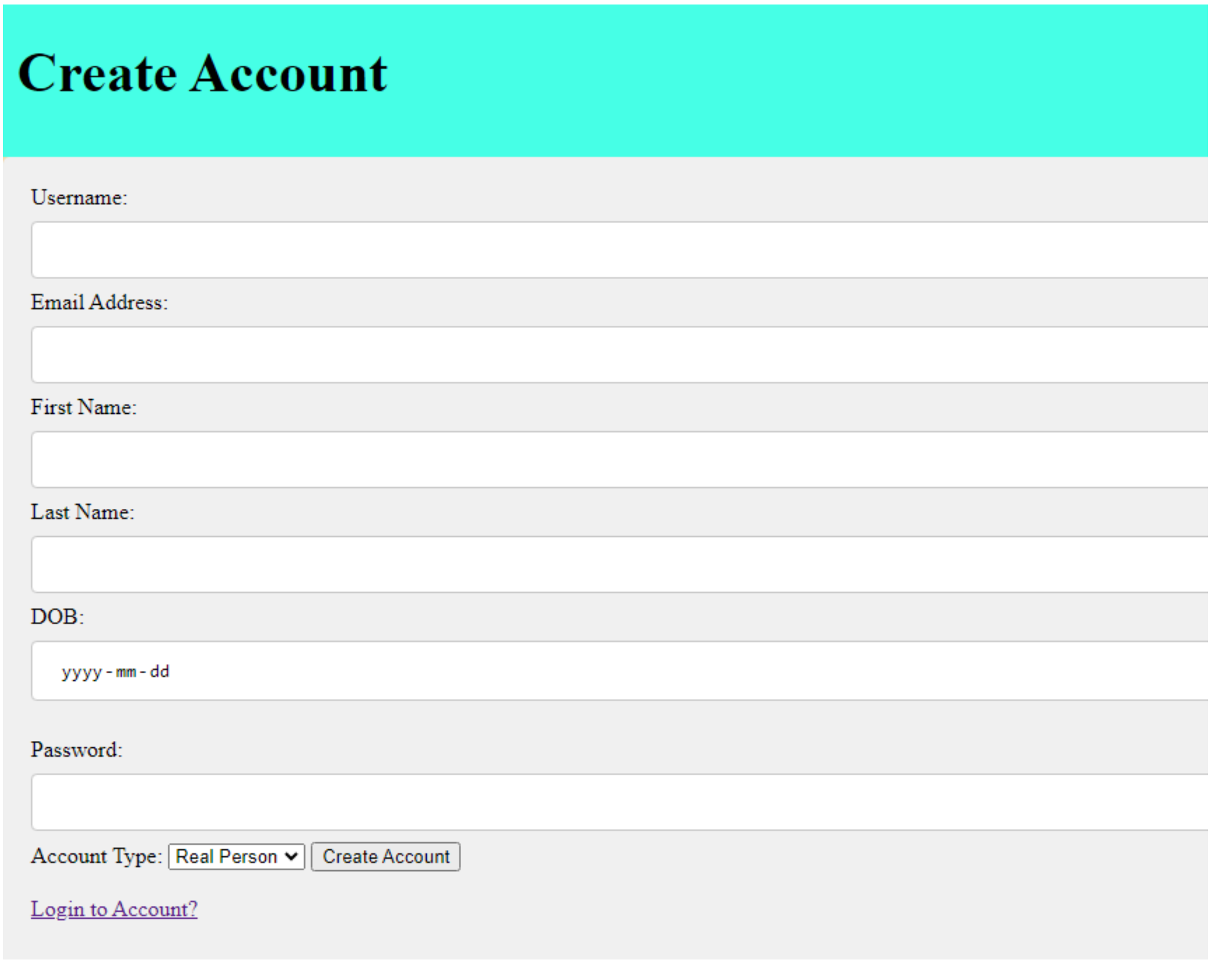


Login to your Member account!

Username:

Password:

Then, click “Do not have an account?” and proceed with account creation



Create Account

Username:

Email Address:

First Name:

Last Name:

DOB:

yyyy - mm - dd

Password:

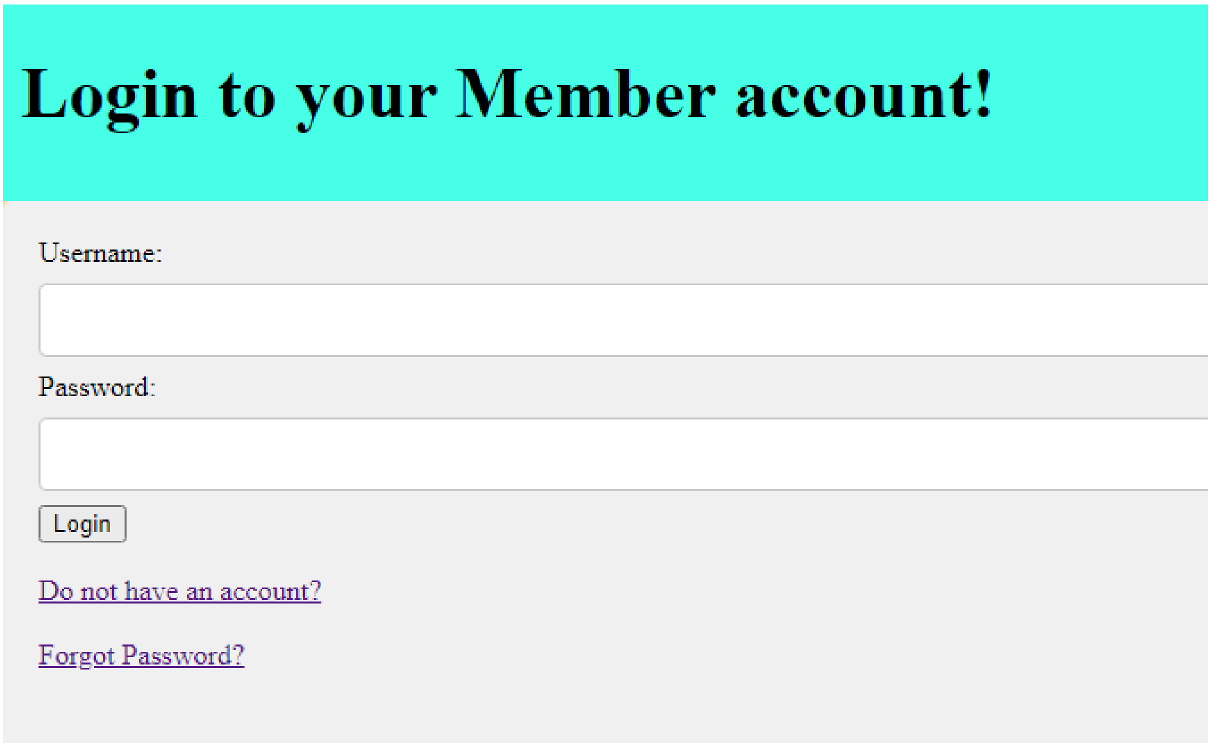
NOTE: Username is unique, Email Address is unique. Email Address will only accept emails ending with “@protonmail.com”.

II. Logging In

Once the account has been successfully created, you should be redirected back to the Home Page.

Click the “Login” link on the top left again.

Enter your account's credentials in the login page…



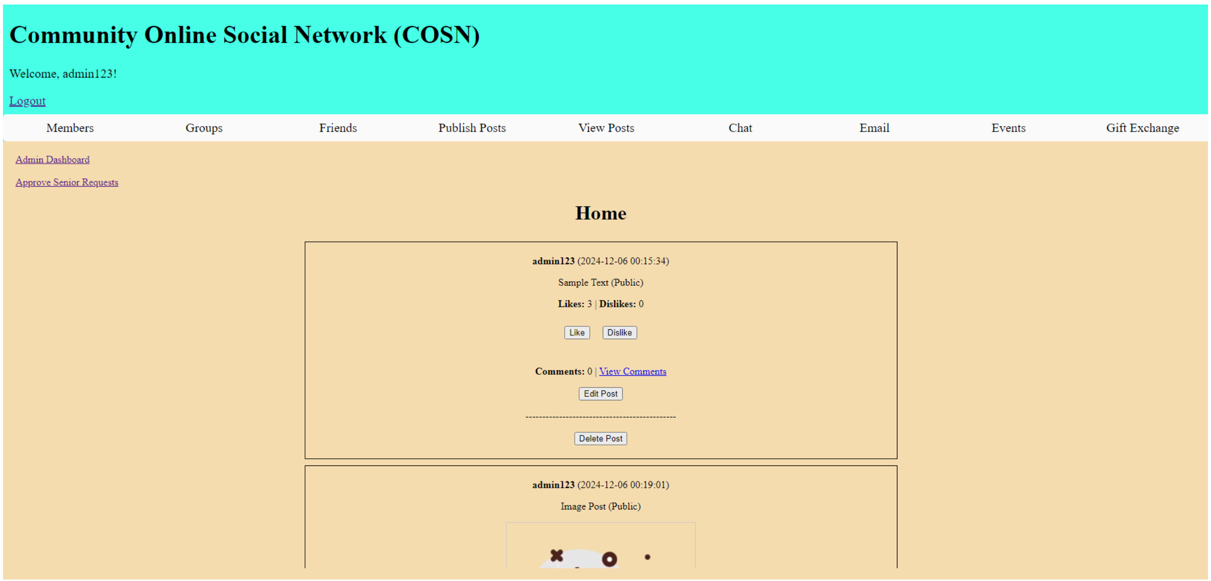
Login to your Member account!

Username:

Password:

Once you have successfully logged into COSN, you will see the Home Page with your created Posts.

An example would be this:



Community Online Social Network (COSN)

Welcome, admin123!

Home

admin123 (2024-12-06 00:15:34)

Sample Text (Public)

Likes: 3 | **Dislikes:** 0

Comments: 0 | View Comments

----

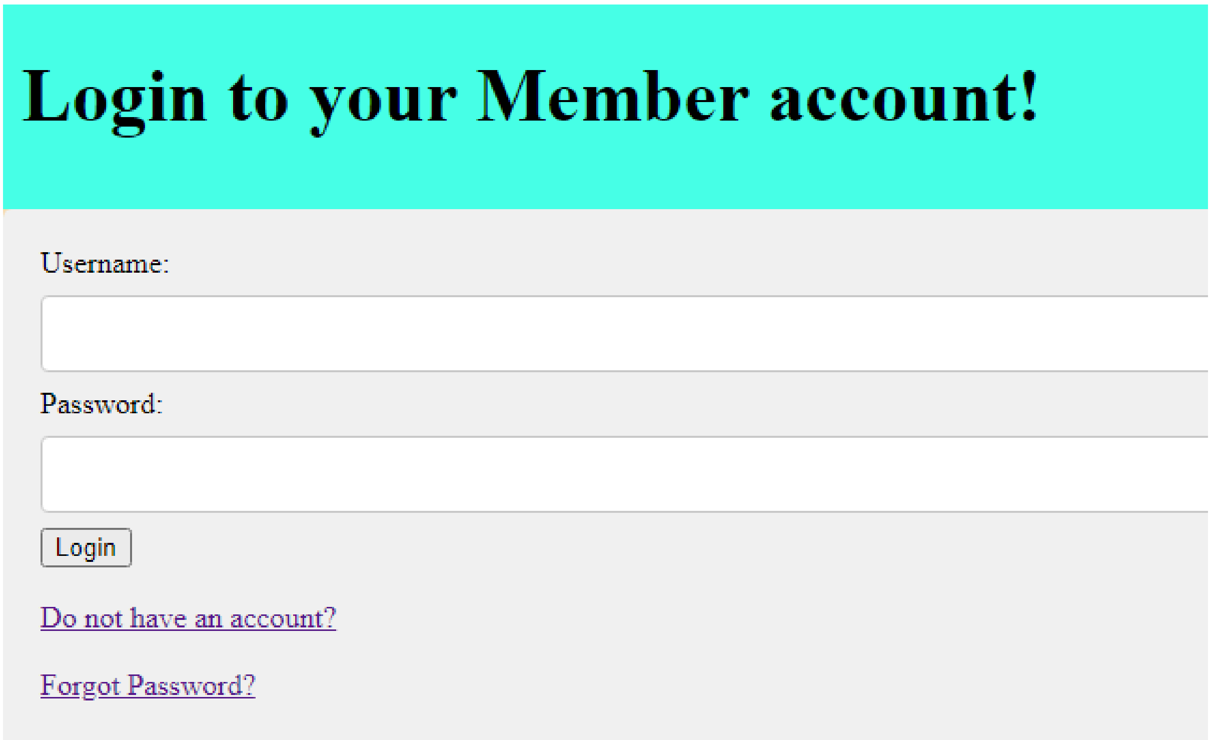
admin123 (2024-12-06 00:19:01)

Image Post (Public)

NOTE: "Admin Dashboard" and "Approve Senior Requests" only show for Members with Administrator Privilege. More on this later.

III. Changing Password

Assuming that you are logged into COSN, the first thing you must do before changing your password is to click the "Logout" link on the top left of the Home Page.
Then, click the "Login" link on the top left of the Home Page.

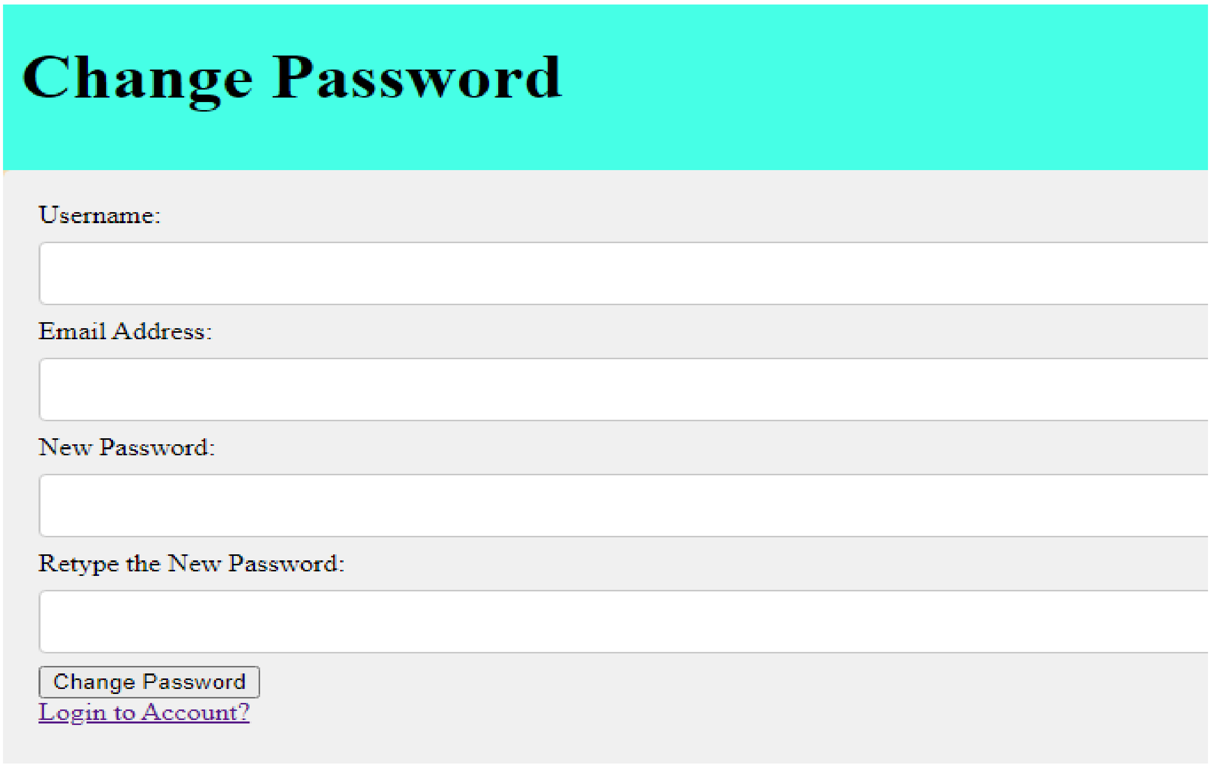


Login to your Member account!

Username:

Password:

Then, click "Forgot Password?" to redirect yourself to a password change page.



Change Password

Username:

Email Address:

New Password:

Retype the New Password:

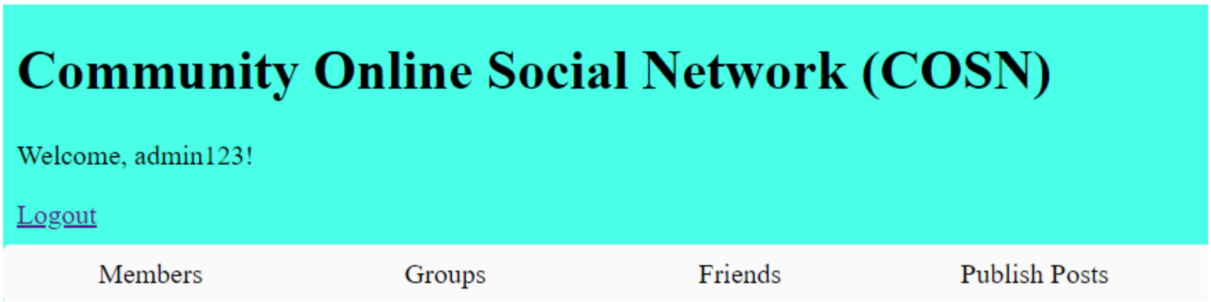
Enter your credentials and the new password. You should see a confirmation message below the form if successful.

Click “Login to Account?” to go back to the Login page.

IV. Changing Username / Editing Profile Details

To change your Username, first Login into your Member account.

Then, click the “Members” tab in the header of “Community Online Social Network (COSN)”.

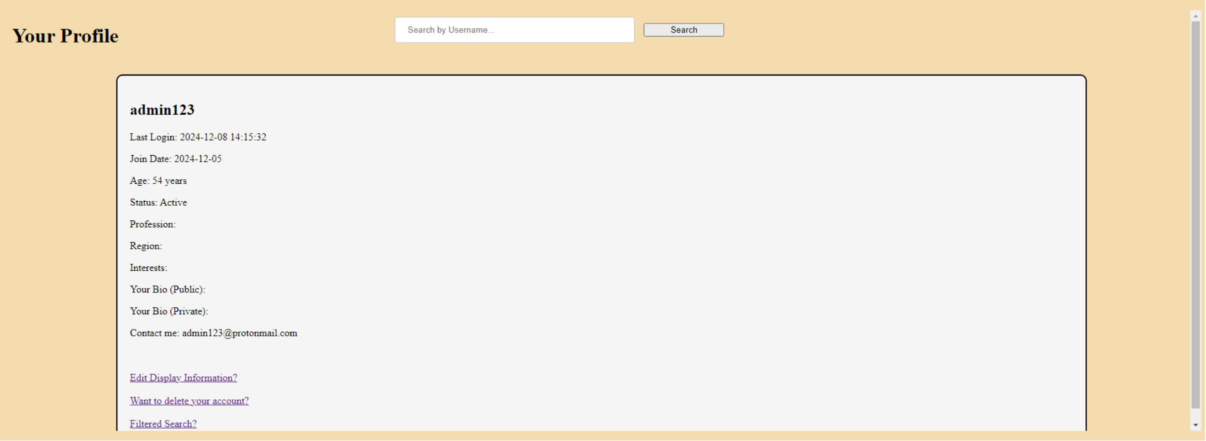


Community Online Social Network (COSN)

Welcome, admin123!

Members should turn dark blue to signify that you are displaying that tab.

You will now see your Profile’s details as such:



Your Profile

admin123

Last Login: 2024-12-08 14:15:32

Join Date: 2024-12-05

Age: 54 years

Status: Active

Profession:

Region:

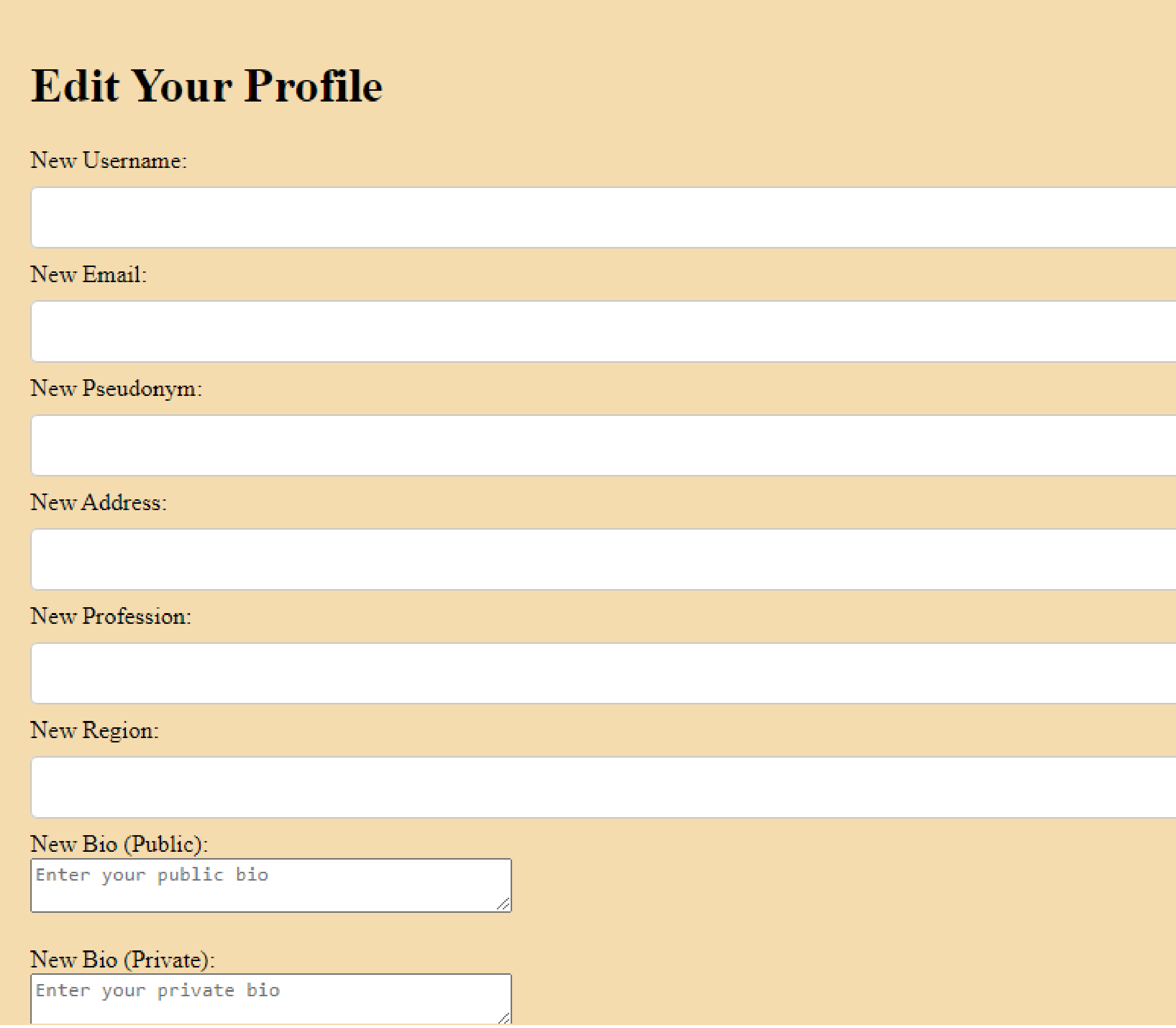
Interests:

Your Bio (Public):

Your Bio (Private):

Contact me: admin123@protonmail.com

Click on “Edit Display Information?”. This should redirect you to a form where you can change your profile’s details, including your Username.



Edit Your Profile

New Username:

New Email:

New Pseudonym:

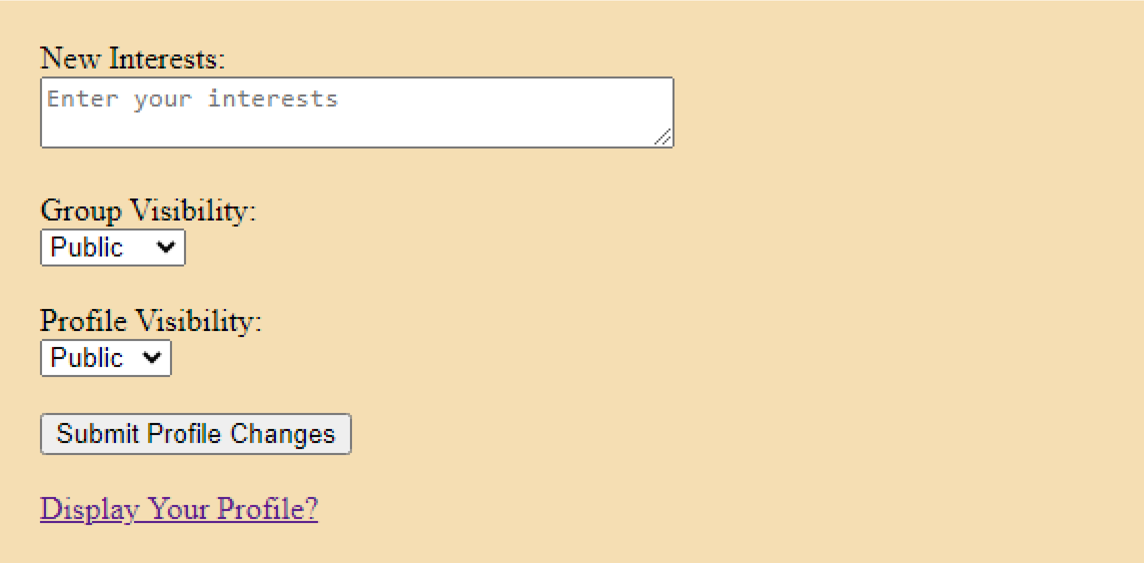
New Address:

New Profession:

New Region:

New Bio (Public):
Enter your public bio

New Bio (Private):
Enter your private bio



New Interests:
Enter your interests

Group Visibility:
Public

Profile Visibility:
Public

Fill out the form, and click “Display Your Profile?” to see the changes. An empty field in the form means that you do not want to change the old value.

NOTE: Username is unique. If you try to change your Profile to an already existing Username, it will not work.

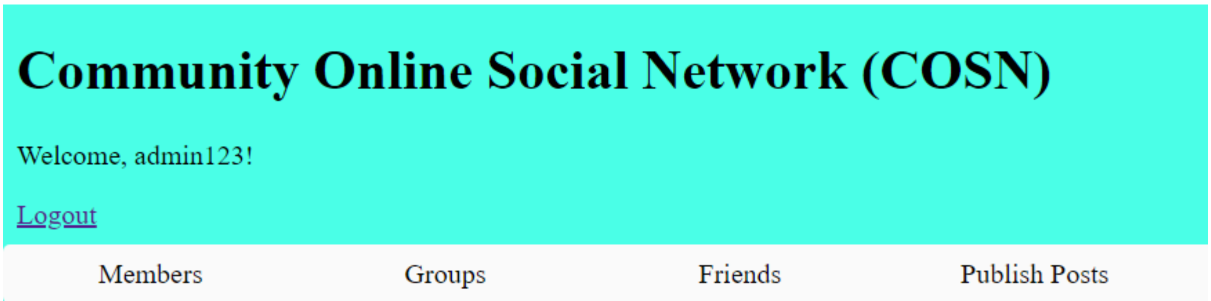
V. Logging out

Assuming that you are logged in, simply click the “Logout” link in the top left of the Home Page.

VI. Deleting Account

To change your Username, first Login into your Member account.

Then, click the “Members” tab in the header of “Community Online Social Network (COSN)”.



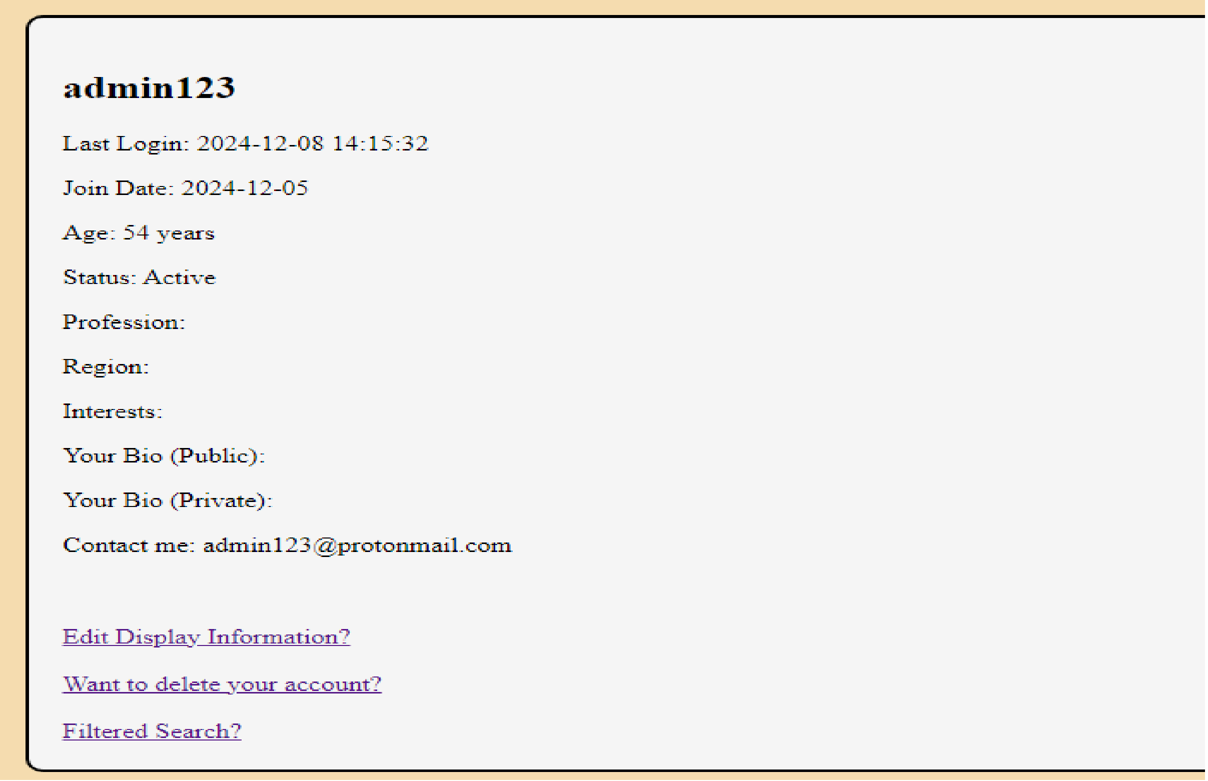
Community Online Social Network (COSN)

Welcome, admin123!

Logout

Members should turn dark blue to signify that you are displaying that tab.

You will now see your Profile’s details as such:



admin123

Last Login: 2024-12-08 14:15:32

Join Date: 2024-12-05

Age: 54 years

Status: Active

Profession:

Region:

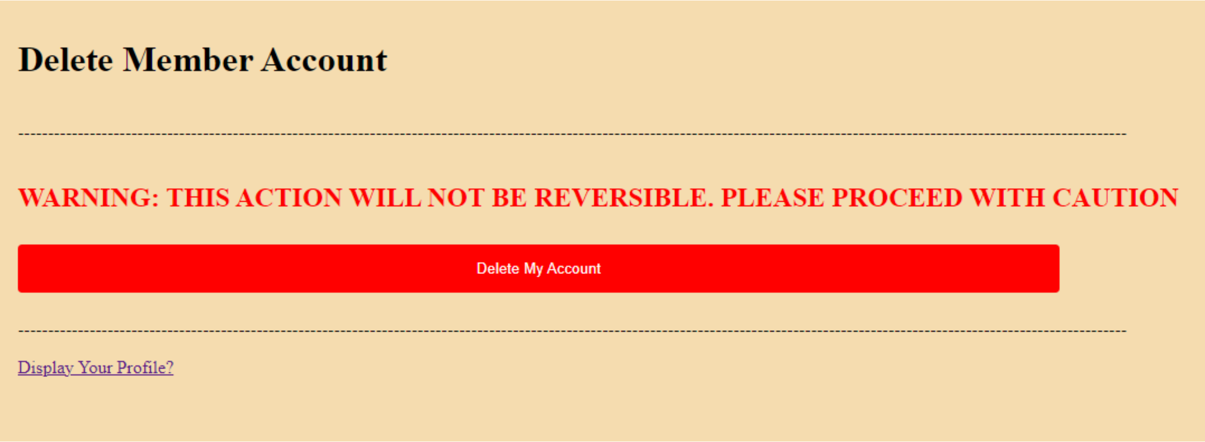
Interests:

Your Bio (Public):

Your Bio (Private):

Contact me: admin123@protonmail.com

To delete your account, click “Want to delete your account?”. This will redirect you to a page that looks like this:



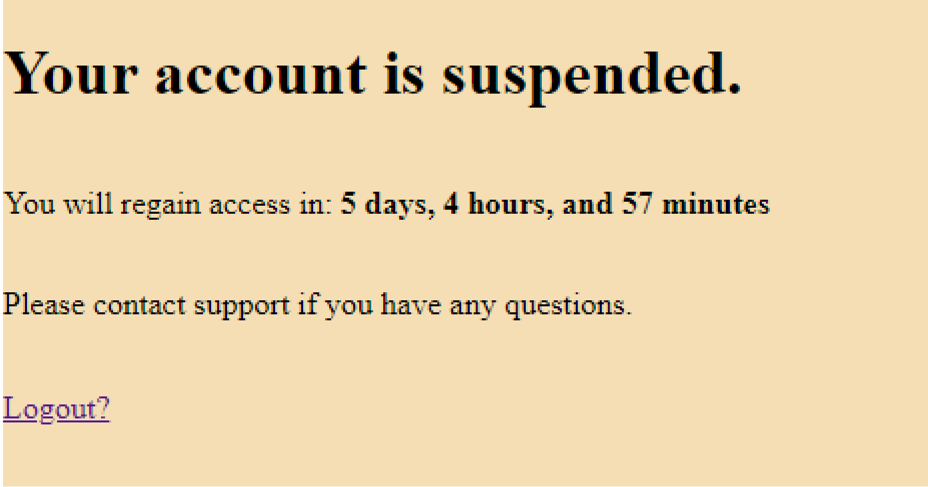
Delete Member Account

WARNING: THIS ACTION WILL NOT BE REVERSIBLE. PLEASE PROCEED WITH CAUTION

Clicking the “Delete My Account” button will make a confirmation pop-up appear. If you click “Ok”, the Member account will be deleted from the database.

VII. Suspended Account

If your account is suspended and you try to login, you will see a page like this:



Your account is suspended.

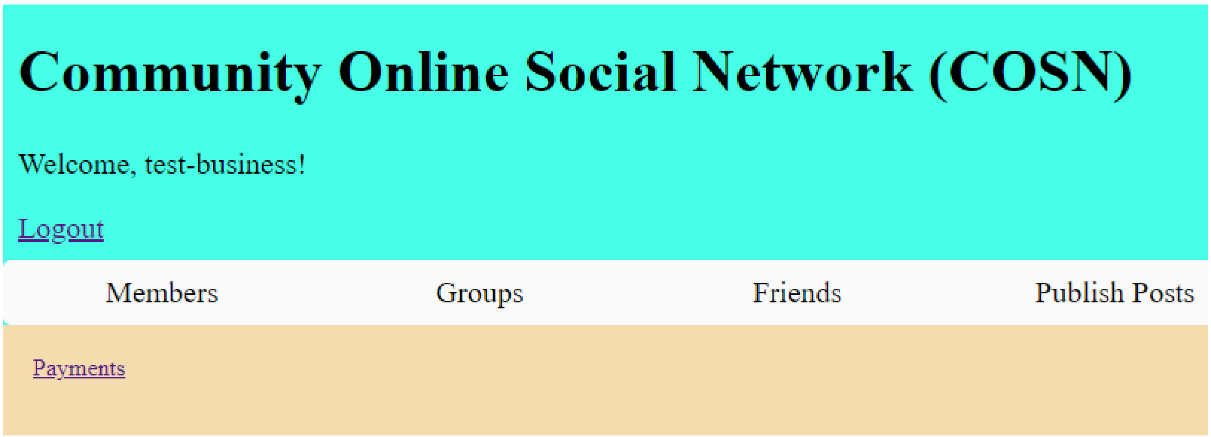
You will regain access in: **5 days, 4 hours, and 57 minutes**

Please contact support if you have any questions.

The only options are to either Logout or simply wait.

VIII. Paying Fines (Business Accounts)

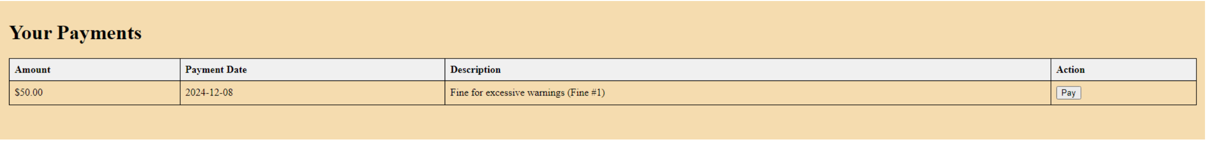
A Business Account will have a link to “Payments” in the Home Page.



Community Online Social Network (COSN)

Welcome, test-business!

Clicking on it will redirect to a page where it can pay its Fines.



Your Payments

Amount	Payment Date	Description	Action
$50.00	2024-12-08	Fine for excessive warnings (Fine #1)	Pay

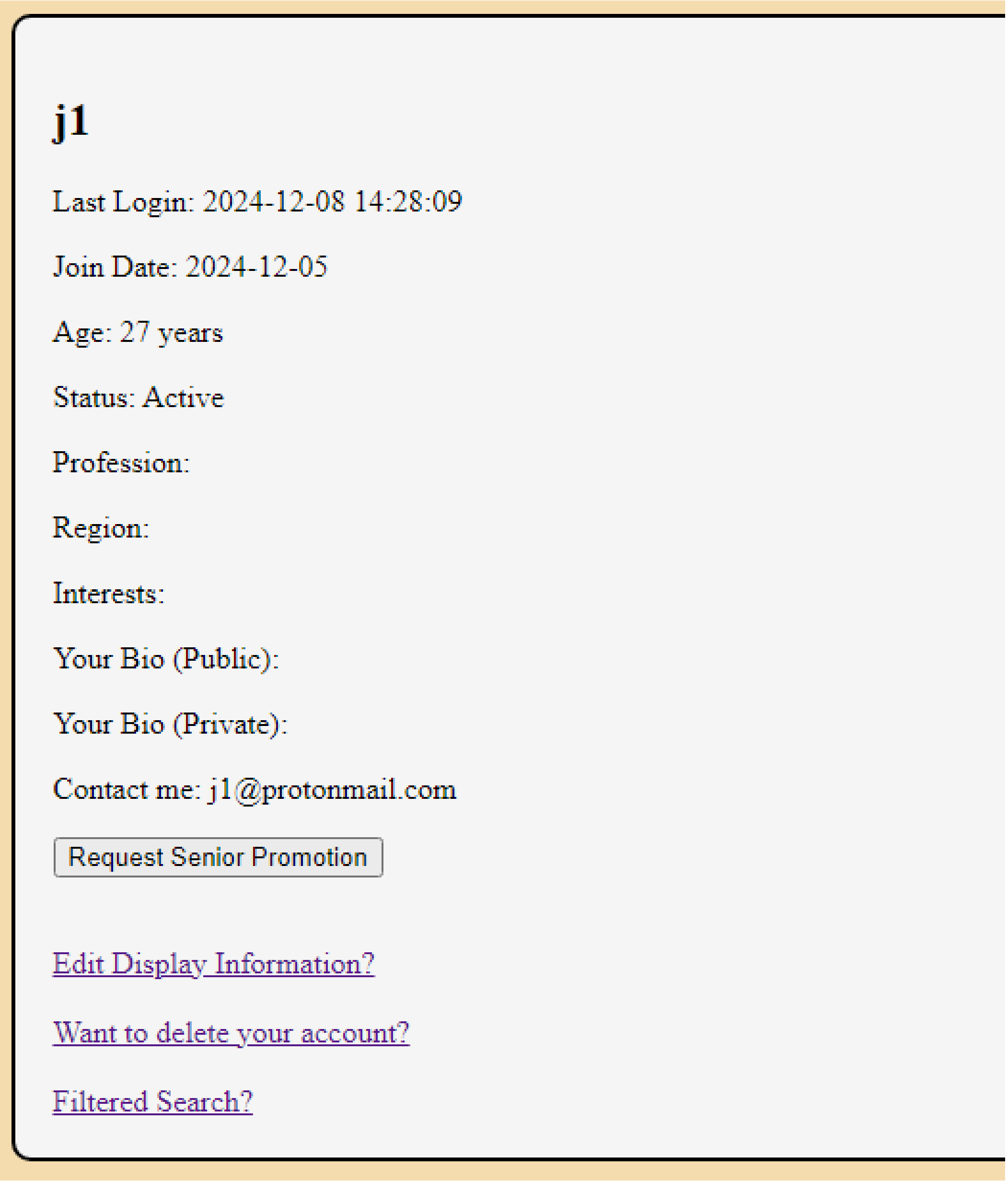
Section 2: Member Permissions

NOTE: ALL PERMISSIONS FROM LESSER ROMAN NUMBERS CARRY OVER TO EQUAL OR UPPER ROMAN NUMBERS.

I. Junior Members

A Junior Member can publish Posts, edit his Posts, write comments on a Post, delete his comments on a Post (more on this in Section 5 and 6).

It can also request to be a Senior by pressing the “Request Senior Promotion” button, in its profile:



j1

Last Login: 2024-12-08 14:28:09

Join Date: 2024-12-05

Age: 27 years

Status: Active

Profession:

Region:

Interests:

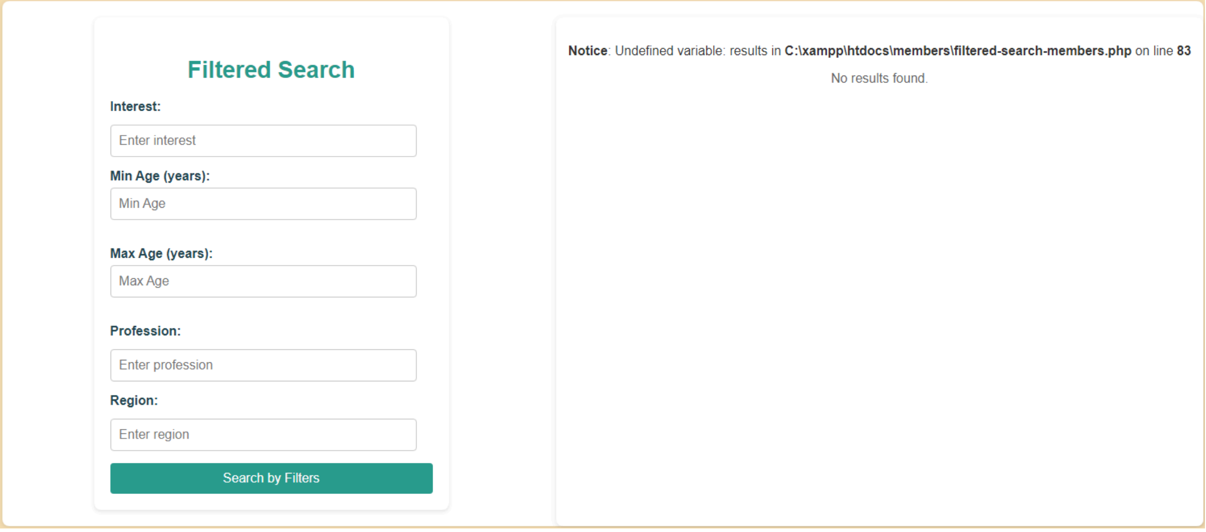
Your Bio (Public):

Your Bio (Private):

Contact me: j1@protonmail.com

A Junior Member can also send Friend Requests, Chat with Friends and Email to any user (more on this in Section 3, 7 and 8).

A Junior Member may use the search functions in the “Members” tab, to find other profiles. It can click the “Filtered Search?” link in its profile, for a Filtered Search.



Filtered Search

Interest: Enter interest

Min Age (years): Min Age

Max Age (years): Max Age

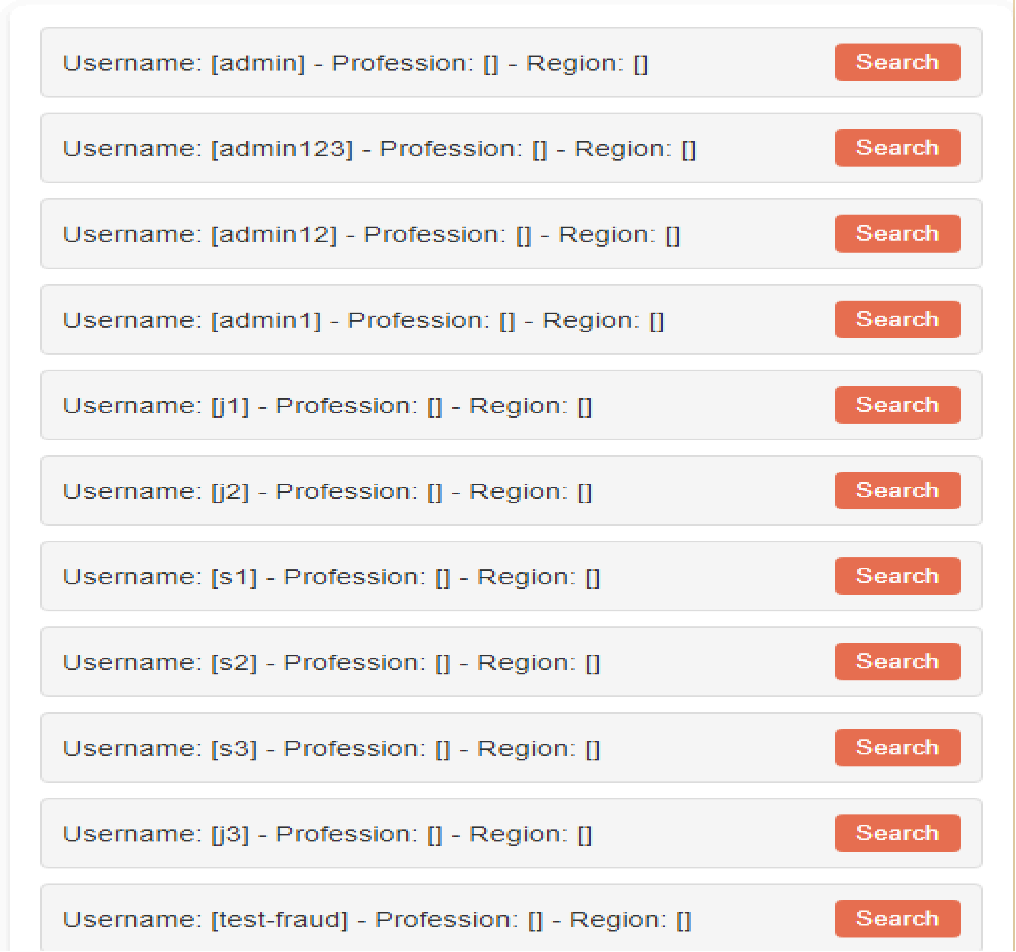
Profession: Enter profession

Region: Enter region

Notice: Undefined variable: results in **C:\xampp\htdocs\members\filtered-search-members.php** on line **83**

No results found.

If it clicks the “Search for Filters” without any parameters, it will return a list of ALL the other Members in the box on the right.



Username: [admin] - Profession: [] - Region: []

Username: [admin123] - Profession: [] - Region: []

Username: [admin12] - Profession: [] - Region: []

Username: [admin1] - Profession: [] - Region: []

Username: [j1] - Profession: [] - Region: []

Username: [j2] - Profession: [] - Region: []

Username: [s1] - Profession: [] - Region: []

Username: [s2] - Profession: [] - Region: []

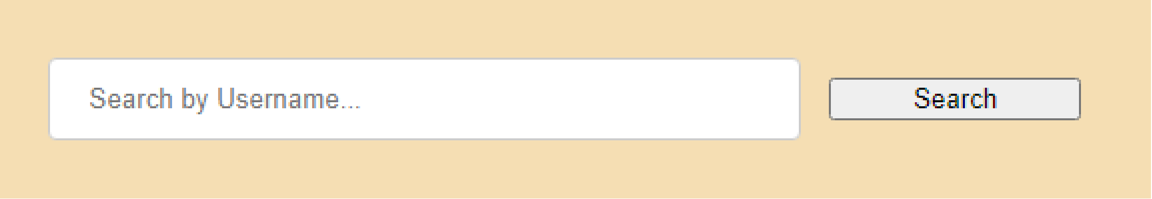
Username: [s3] - Profession: [] - Region: []

Username: [j3] - Profession: [] - Region: []

Username: [test-fraud] - Profession: [] - Region: []

Whenever “Search” is clicked, it will display that Member’s profile.

Another way to Search is to simply type the Username above the Member’s personal profile.



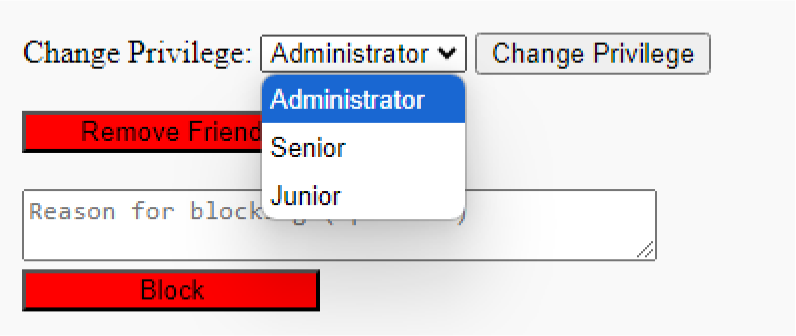
Clicking the “Search” button will provide you with the profile of the matching Username, if any.

II. Senior Members

A Senior Member has the permission to create Groups and add people to Groups (more on this in Section 4).

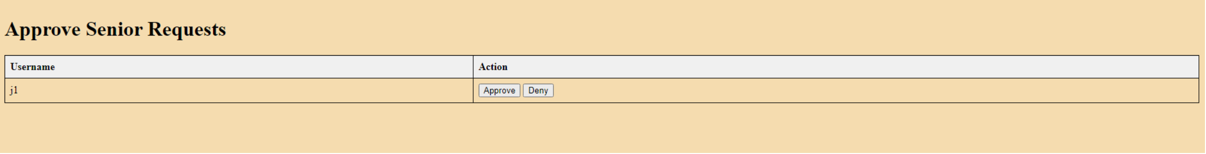
III. Administrator Members

An Administrator Member can change the Privilege of all other members, excluding himself via a searched Member profile.



It also has access to 2 links in the Home Page: Admin Dashboard and Approve Senior Requests.

Approve Senior Request looks like this, given a single sent Senior Promotion Request from a Junior Member:

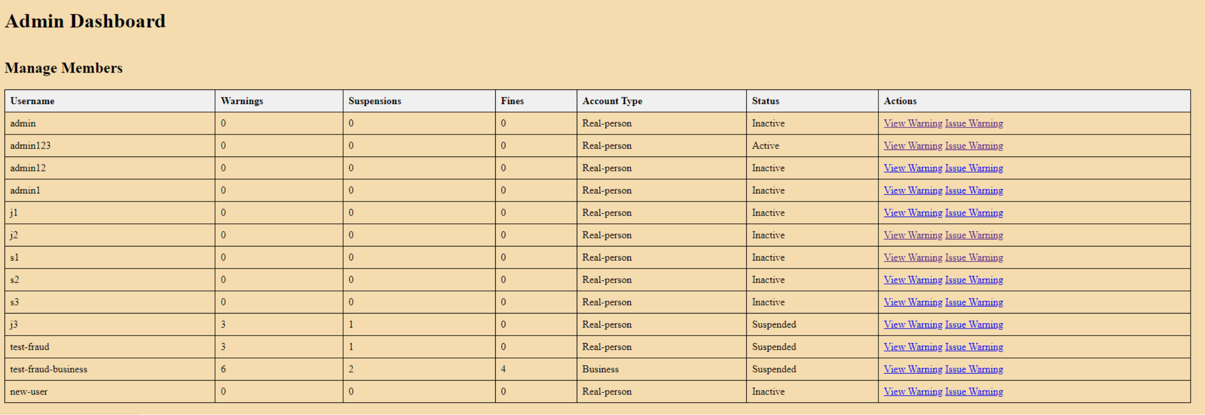


Approve Senior Requests

Username	Action
j1	Approve Deny

Clicking “Approve” will promote the Junior to Senior, while clicking “Deny” will not promote the Junior to Senior.

Admin Dashboard looks like this:



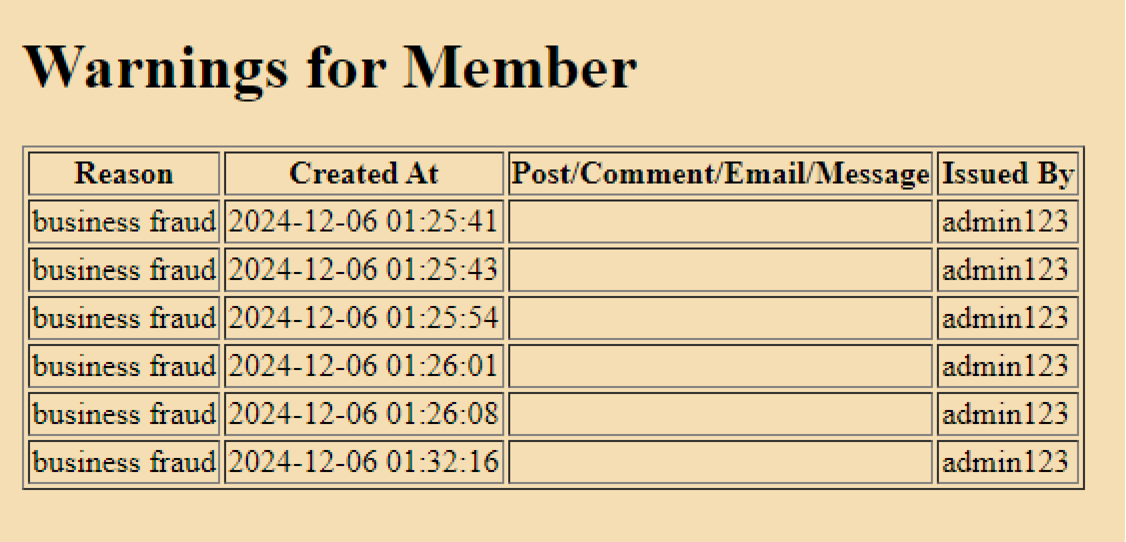
Admin Dashboard

Manage Members

Username	Warnings	Suspensions	Fines	Account Type	Status	Actions
admin	0	0	0	Real-person	Inactive	View Warning Issue Warning
admin123	0	0	0	Real-person	Active	View Warning Issue Warning
admin12	0	0	0	Real-person	Inactive	View Warning Issue Warning
admin1	0	0	0	Real-person	Inactive	View Warning Issue Warning
j1	0	0	0	Real-person	Inactive	View Warning Issue Warning
j2	0	0	0	Real-person	Inactive	View Warning Issue Warning
s1	0	0	0	Real-person	Inactive	View Warning Issue Warning
s2	0	0	0	Real-person	Inactive	View Warning Issue Warning
s3	0	0	0	Real-person	Inactive	View Warning Issue Warning
j3	3	1	0	Real-person	Suspended	View Warning Issue Warning
test-fraud	3	1	0	Real-person	Suspended	View Warning Issue Warning
test-fraud-business	6	2	4	Business	Suspended	View Warning Issue Warning
new-user	0	0	0	Real-person	Inactive	View Warning Issue Warning

We can see a list of ALL the created Members in the database, as well as the "View Warning" and "Issue Warning" actions.

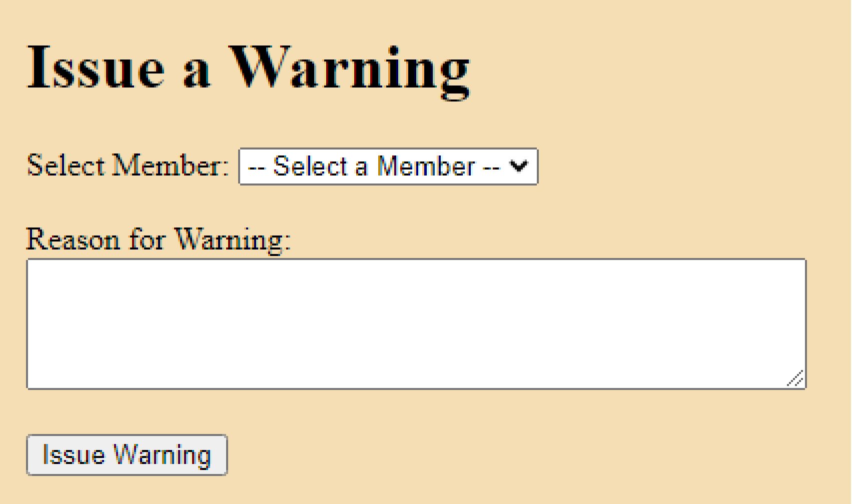
View Warning simply displays a Member's warnings. In the previous image's case, we will have a look at test-fraud's warnings.



Warnings for Member

Reason	Created At	Post/Comment/Email/Message	Issued By
business fraud	2024-12-06 01:25:41		admin123
business fraud	2024-12-06 01:25:43		admin123
business fraud	2024-12-06 01:25:54		admin123
business fraud	2024-12-06 01:26:01		admin123
business fraud	2024-12-06 01:26:08		admin123
business fraud	2024-12-06 01:32:16		admin123

Issue Warning will redirect you to a form, where you can directly issue a warning to a Member.



Issue a Warning

NOTE: WARNINGS TIED TO A POST/COMMENT/EMAIL/MESSAGE HAS NOT BEEN IMPLEMENTED. ONLY DIRECT WARNING HAS BEEN IMPLEMENTED.

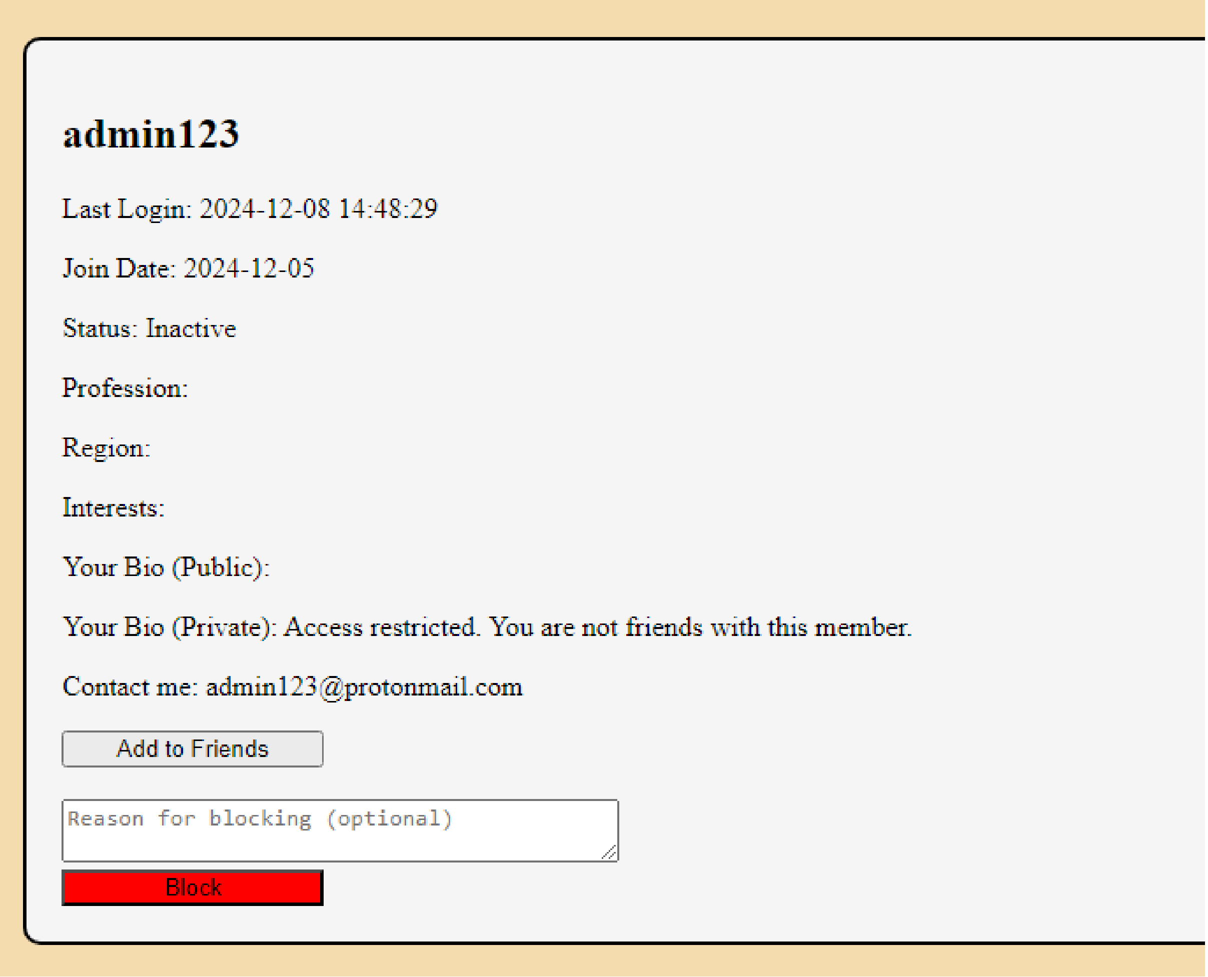
A Real-Person Member will get a Suspension every 3 warnings. The first Suspension lasts 7 days, the second 30 days and the third 365 days.

A Business Member will receive a Fine for every warning after the second one. If a Business Member has more than 2 fines, it will receive a Suspension. The first Suspension lasts 7 days, the second 30 days and the third 365 days.

Section 3: Friends

We will assume that you have searched for a Member as described in Section 2 (I. Junior Members).

The searcher can click the “Add to Friends” or “Block” buttons.



admin123

Last Login: 2024-12-08 14:48:29

Join Date: 2024-12-05

Status: Inactive

Profession:

Region:

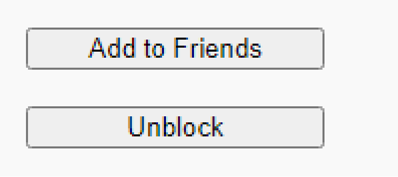
Interests:

Your Bio (Public):

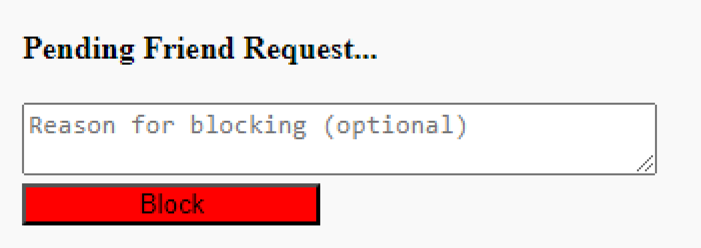
Your Bio (Private): Access restricted. You are not friends with this member.

Contact me: admin123@protonmail.com

Blocking a Member will remove a member as a Friend. It will also make the blocked Member's Posts invisible. If you click "Block", an "Unblock" button will appear.

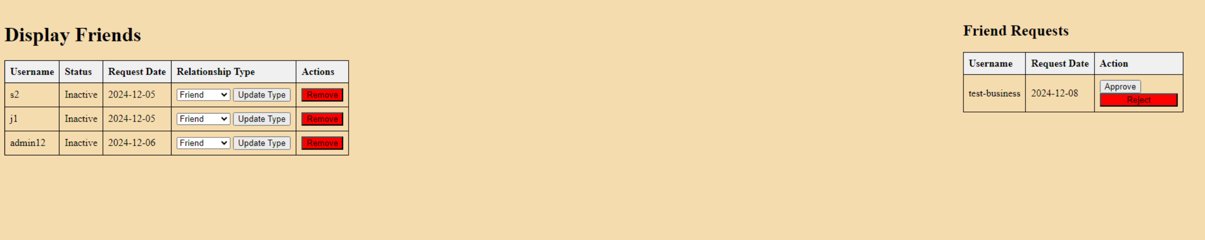


If "Add to Friends" is clicked, a Friend Request is sent for the receiving Member to handle.



Pending Friend Request...

The receiving Member can login to COSN, click the "Friends" tab in the header and see the Friends page:



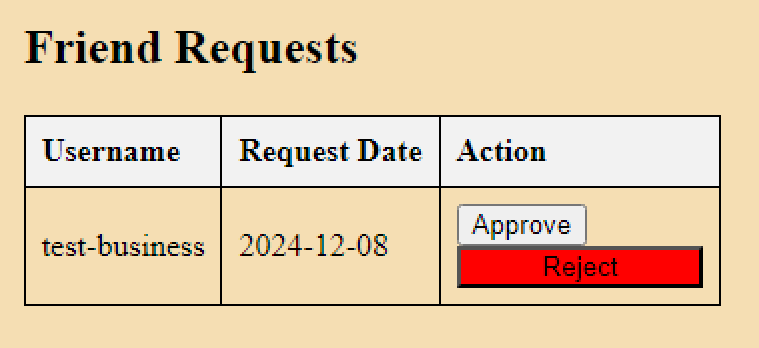
Display Friends

Username	Status	Request Date	Relationship Type	Actions
s2	Inactive	2024-12-05	Friend Update Type	Remove
j1	Inactive	2024-12-05	Friend Update Type	Remove
admin12	Inactive	2024-12-06	Friend Update Type	Remove

Friend Requests

Username	Request Date	Action
test-business	2024-12-08	Approve Reject

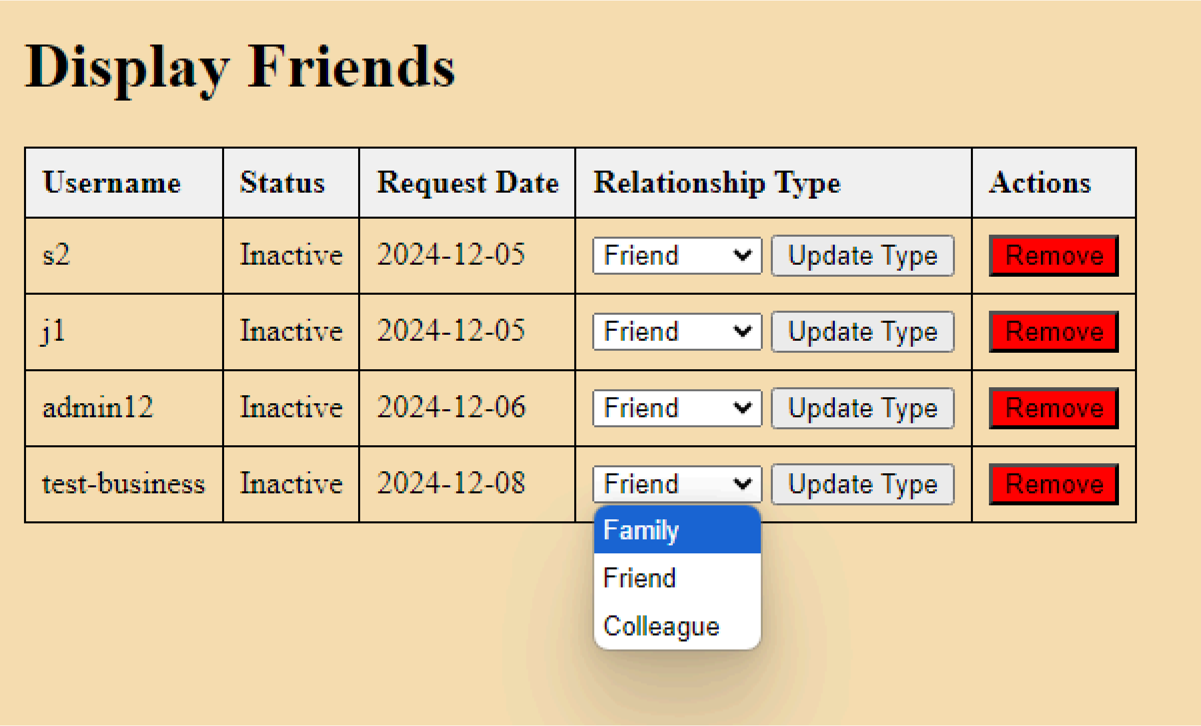
In particular, the receiving Member has the options to approve or reject the Friend Request:



Friend Requests

Username	Request Date	Action
test-business	2024-12-08	Approve Reject

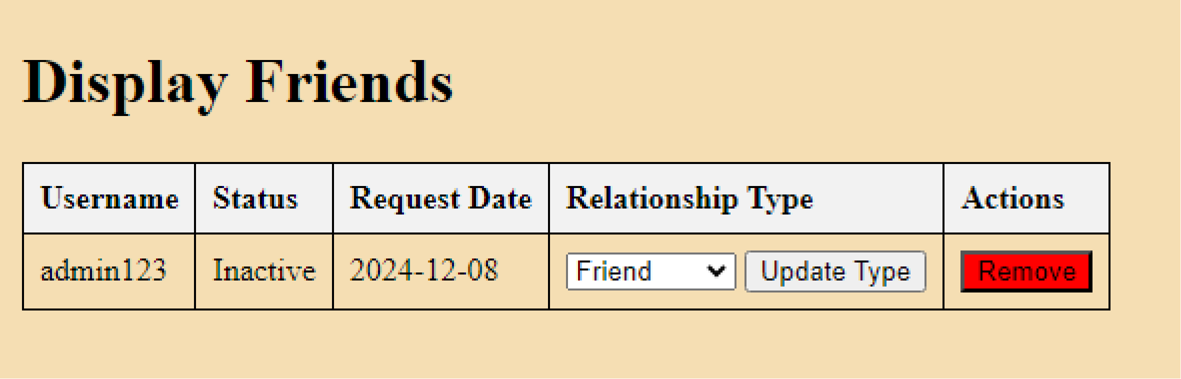
If it is approved, the receiving Member can update the Relationship Type or Remove from the friends list.



Display Friends

Username	Status	Request Date	Relationship Type	Actions
s2	Inactive	2024-12-05	Friend ▾ Update Type	Remove
j1	Inactive	2024-12-05	Friend ▾ Update Type	Remove
admin12	Inactive	2024-12-06	Friend ▾ Update Type	Remove
test-business	Inactive	2024-12-08	Friend ▾ Update Type	Remove

This is how it looks on the sending Member's end:



Display Friends

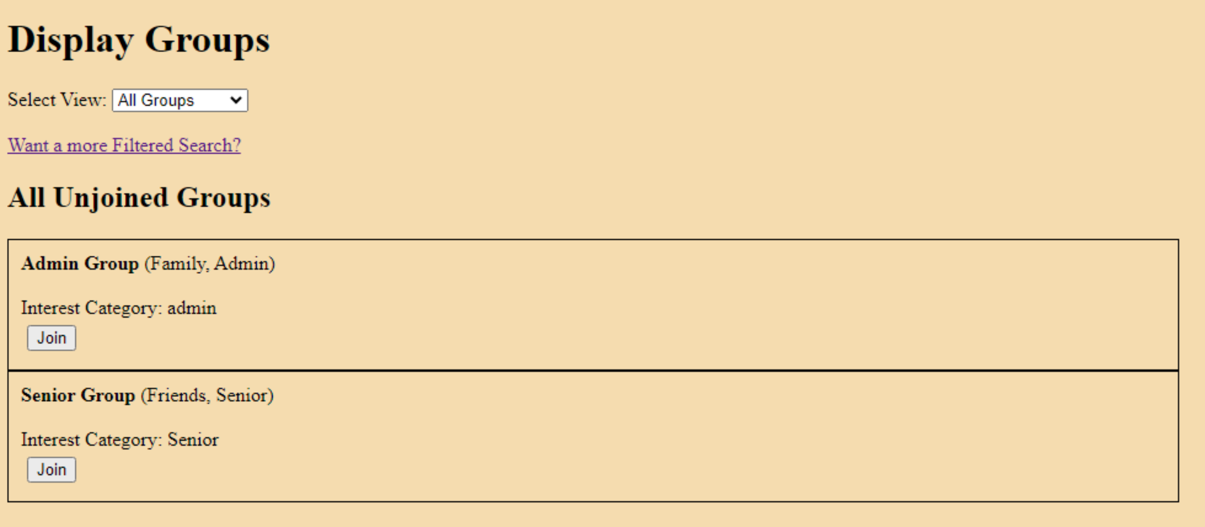
Username	Status	Request Date	Relationship Type	Actions
admin123	Inactive	2024-12-08	Friend ▾ Update Type	Remove

Section 4: Groups

NOTE: Group Admin is treated like an "Owner" of the Group. There can be many Admins in a Group. The last Admin of a Group cannot demote himself. It must first promote another Group Member to Admin, then let the new Admin demote it.

I. Searching for Groups

On the COSN Header, you can click the "Groups" tab. It will redirect you to a page with the available Groups to join:



Display Groups

Select View: All Groups

Want a more Filtered Search?

All Unjoined Groups

Admin Group (Family, Admin)

Interest Category: admin

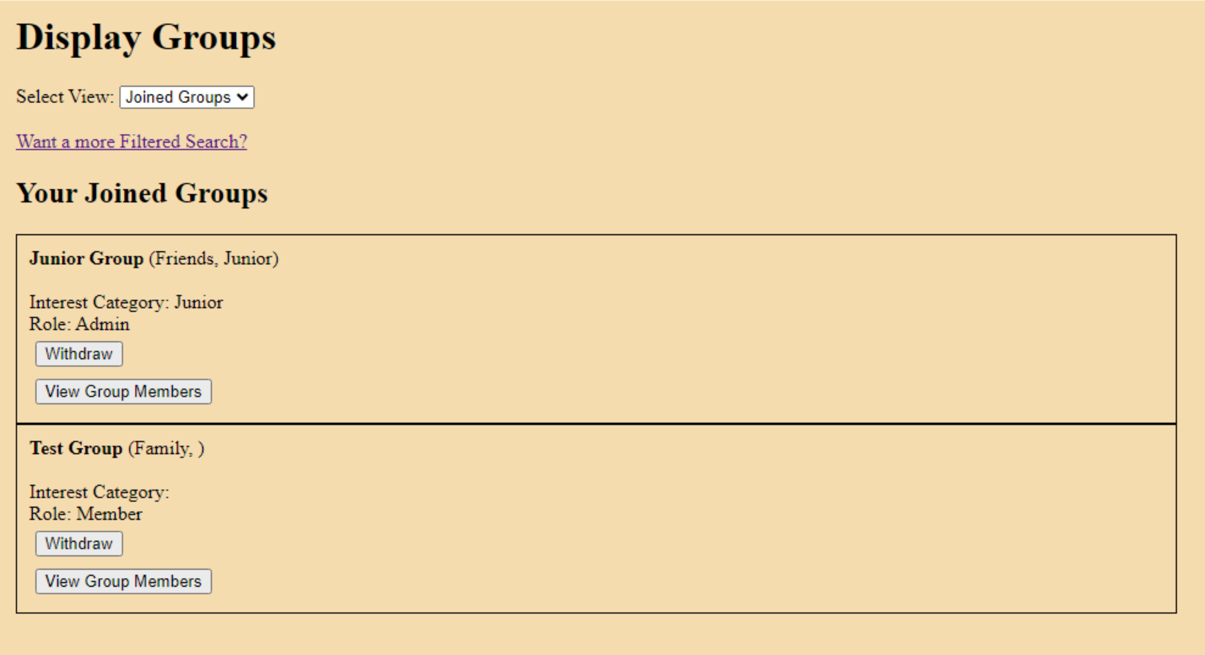
Join

Senior Group (Friends, Senior)

Interest Category: Senior

Join

To see your Joined Groups, change "Select View: " to "Joined Groups":



Display Groups

Select View: Joined Groups

Want a more Filtered Search?

Your Joined Groups

Junior Group (Friends, Junior)

Interest Category: Junior
Role: Admin

Withdraw

View Group Members

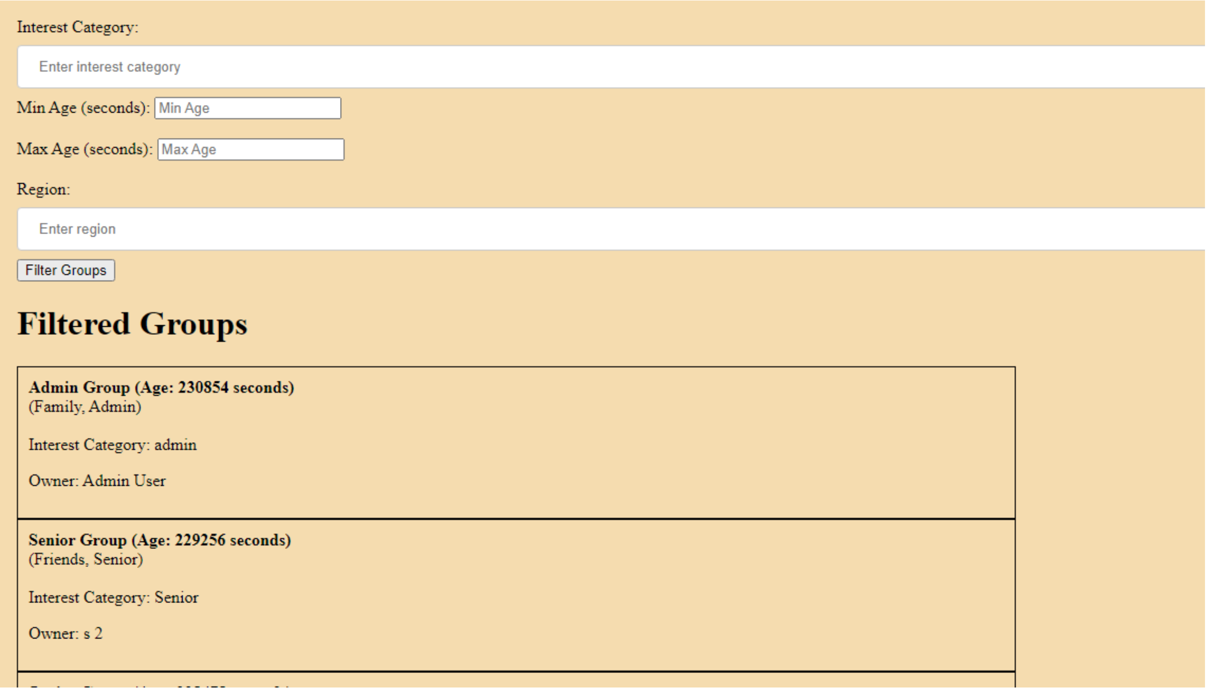
Test Group (Family,)

Interest Category:
Role: Member

Withdraw

View Group Members

Clicking on the “Want a more Filtered Search?” link will redirect you to a Filtered Search page:



Filtered Groups

Admin Group (Age: 230854 seconds)
(Family, Admin)

Interest Category: admin

Owner: Admin User

Senior Group (Age: 229256 seconds)
(Friends, Senior)

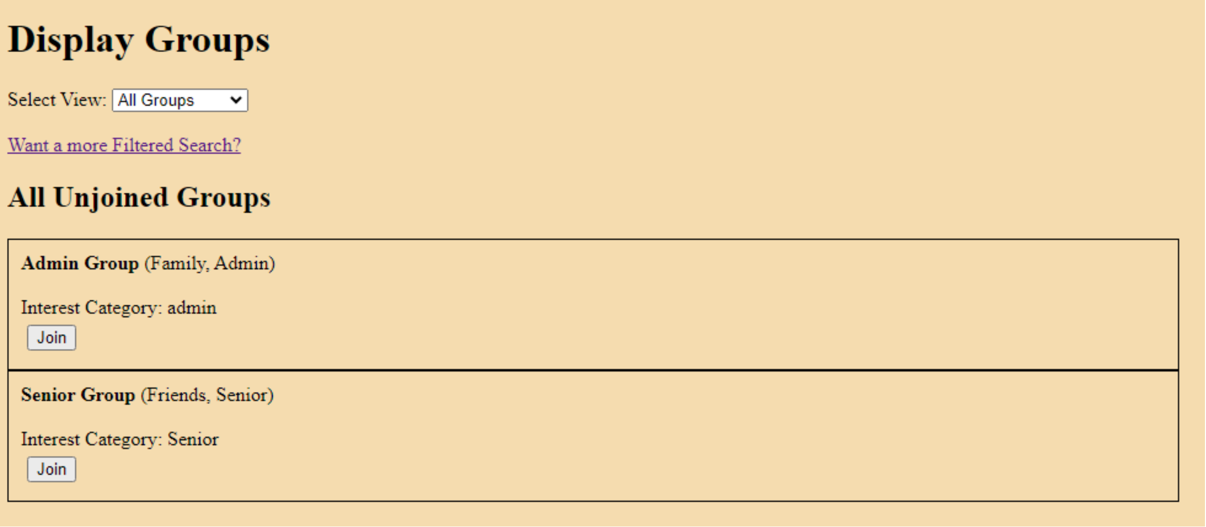
Interest Category: Senior

Owner: s 2

NOTE: THE OWNER IS FORMATTED AS THE GROUP CREATOR’S “<First Name> <Last Name>”.

II. Joining a Group

On the COSN Header, you can click the “Groups” tab. It will redirect you to a page with the available Groups to join:



Display Groups

Select View: All Groups

All Unjoined Groups

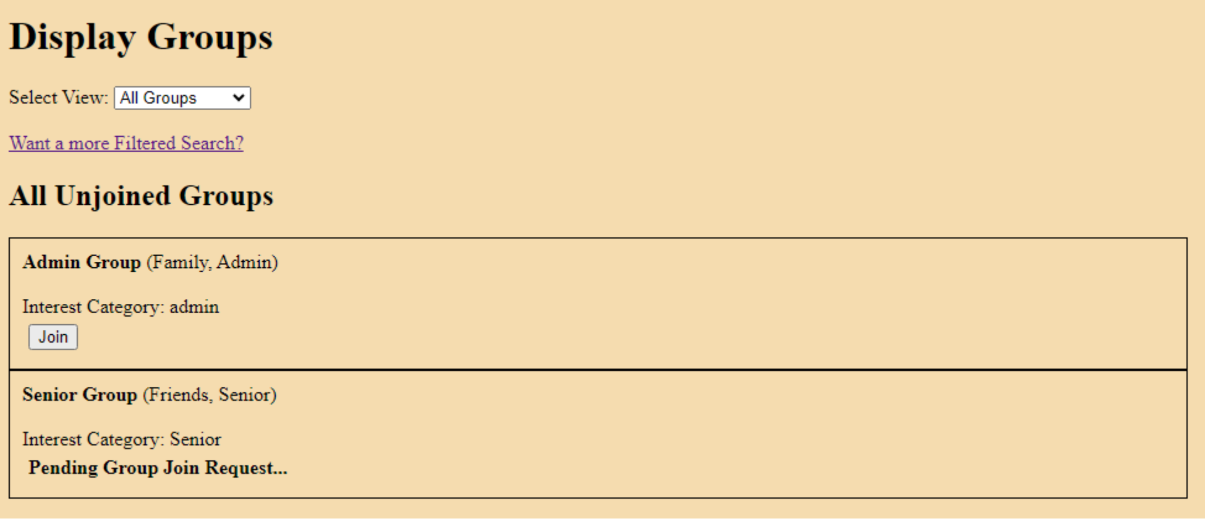
Admin Group (Family, Admin)

Interest Category: admin

Senior Group (Friends, Senior)

Interest Category: Senior

Clicking Join on a Group should display this:



Display Groups

Select View: All Groups

Want a more Filtered Search?

All Unjoined Groups

Admin Group (Family, Admin)

Interest Category: admin

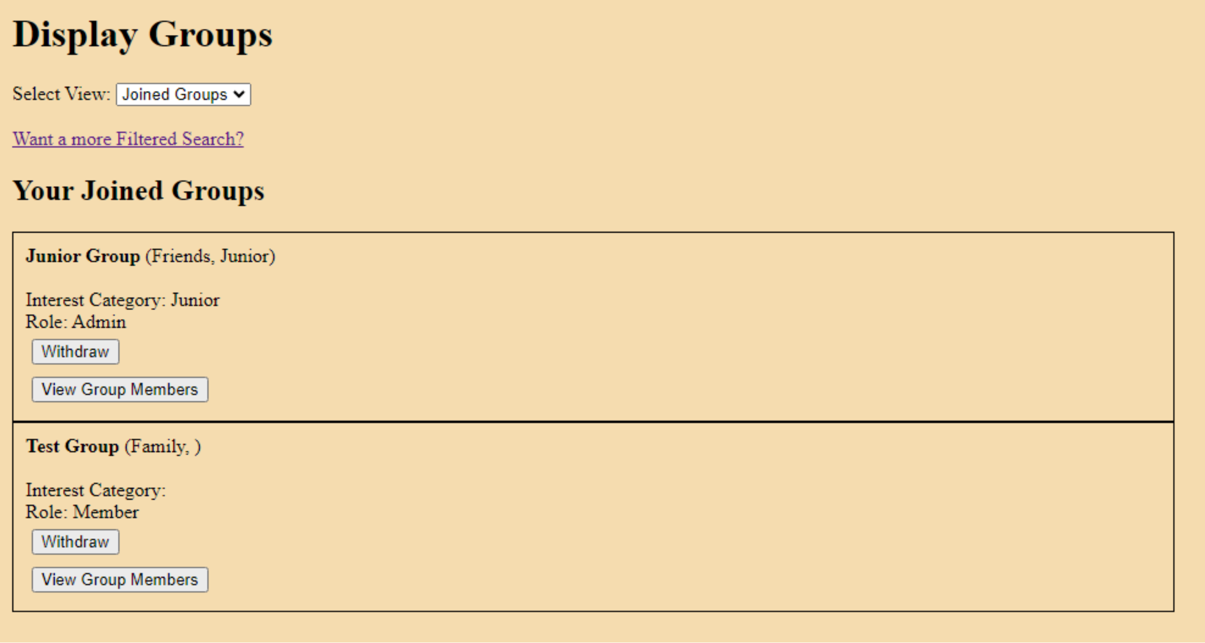
Senior Group (Friends, Senior)

Interest Category: Senior

Pending Group Join Request...

III. Withdrawing from a Group

To withdraw from a Group, simply Select View to Joined Groups:



Display Groups

Select View: Joined Groups

Want a more Filtered Search?

Your Joined Groups

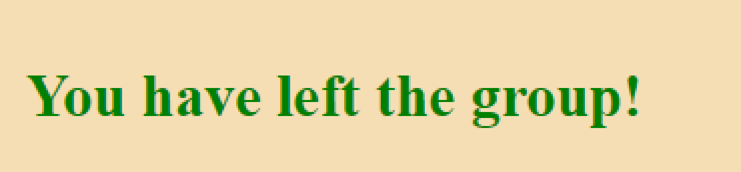
Junior Group (Friends, Junior)

Interest Category: Junior
Role: Admin

Test Group (Family,)

Interest Category:
Role: Member

Then, press the "Withdraw" button on the desired Group.
There should be a confirmation message:

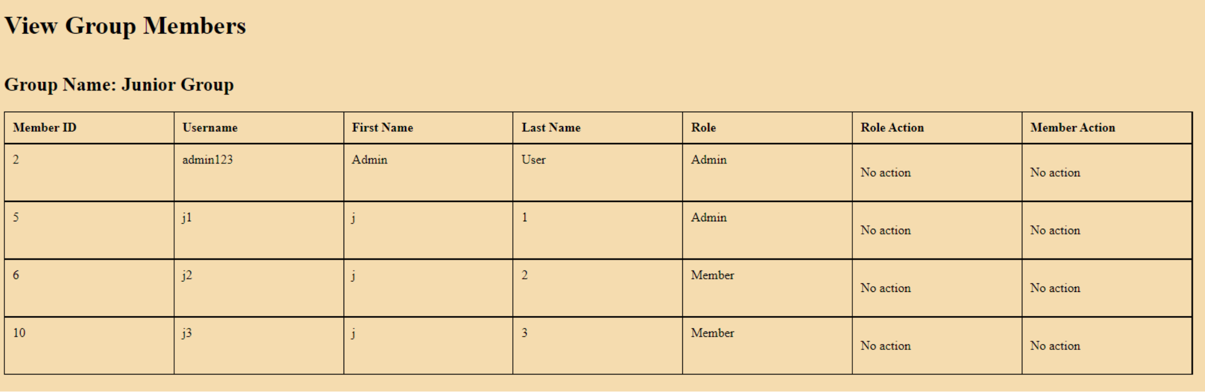


You have left the group!

IV. Viewing Group Members

We use a similar procedure to III. but we click “View Group Members” instead of “Withdraw”.

For a Group Member, the page looks like this:

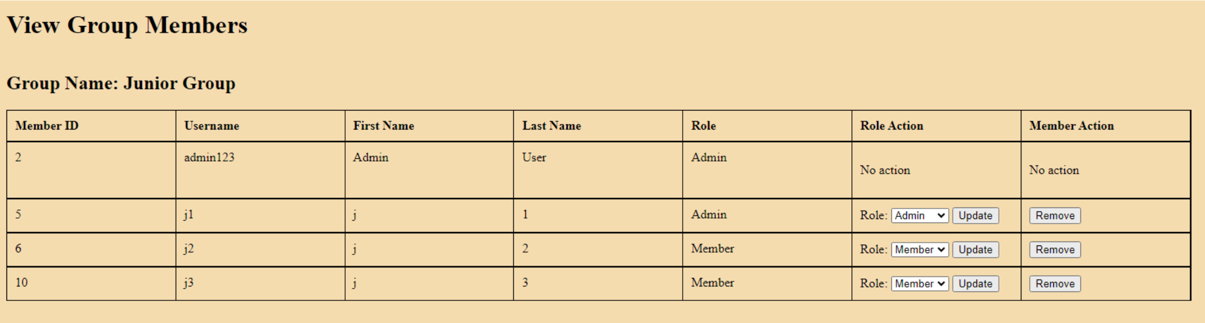


View Group Members

Group Name: Junior Group

Member ID	Username	First Name	Last Name	Role	Role Action	Member Action
2	admin123	Admin	User	Admin	No action	No action
5	j1	j	1	Admin	No action	No action
6	j2	j	2	Member	No action	No action
10	j3	j	3	Member	No action	No action

For a Group Admin, the page looks like this:



View Group Members

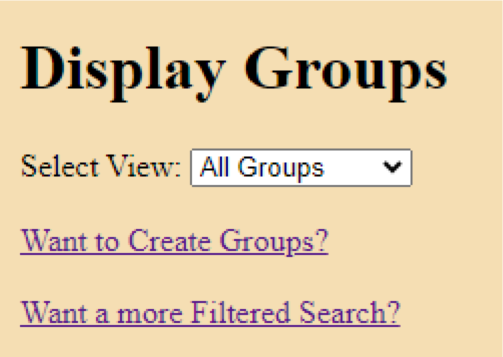
Group Name: Junior Group

Member ID	Username	First Name	Last Name	Role	Role Action	Member Action
2	admin123	Admin	User	Admin	No action	No action
5	j1	j	1	Admin	Role: Admin ⌄ Update	Remove
6	j2	j	2	Member	Role: Member ⌄ Update	Remove
10	j3	j	3	Member	Role: Member ⌄ Update	Remove

V. Creating a Group (Senior+)

On the COSN Header, you can click the “Groups” tab.

A Senior Member should see a link to “Want to Create Groups?”.



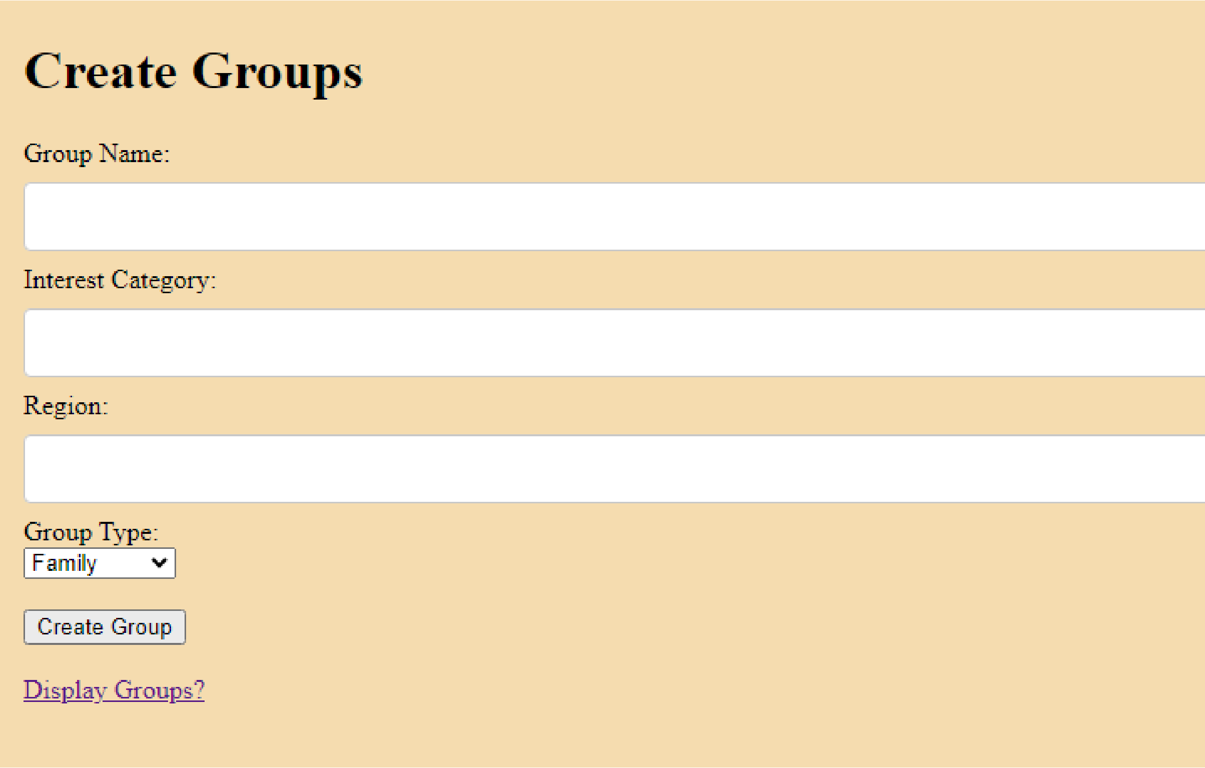
Display Groups

Select View: All Groups ⌄

Want to Create Groups?

Want a more Filtered Search?

Once clicked, the Senior Member can fill this form to create a Group. Submitting the form will redirect to “Display Groups”.



Create Groups

Group Name:

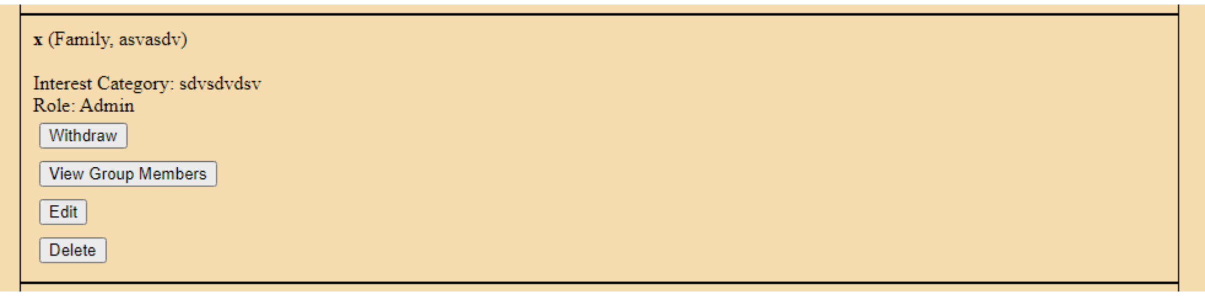
Interest Category:

Region:

Group Type:
Family

VI. Editing a Group (Senior+, Group Admin)

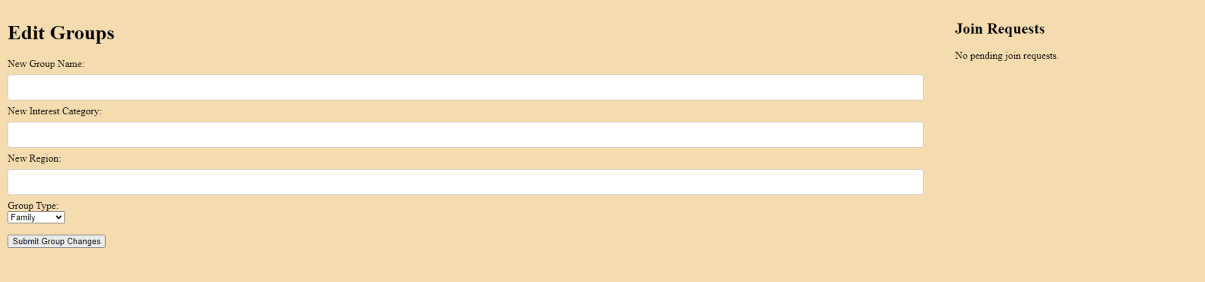
Assuming that the user is a Senior/Administrator Member that is also a Group Admin, it can see the Edit option in its Joined Groups:



x (Family, asvasdv)

Interest Category: sdvsdvdsv
Role: Admin

Clicking it redirects to a page like this:



Edit Groups

New Group Name:

New Interest Category:

New Region:

Group Type:
Family

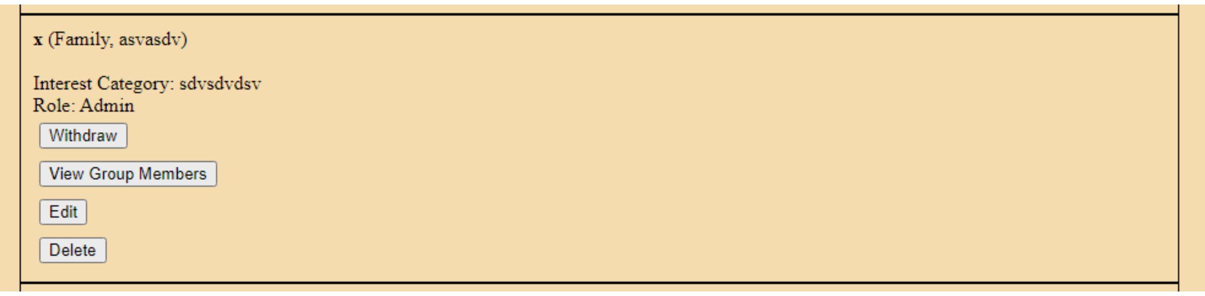
Join Requests

No pending join requests.

Submitting the form will change the Group’s details.

VII. Approving Group Join Requests (Senior+, Group Admin)

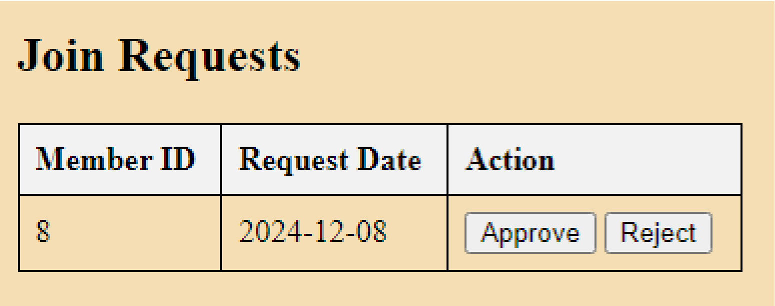
Assuming that the user is a Senior/Administrator Member that is also a Group Admin, it can see the Edit option in its Joined Groups:



x (Family, asvasdv)

Interest Category: sdvsdvdsv
Role: Admin

The same Edit page is used to approve Join Requests. A Join Request looks like this:



Join Requests

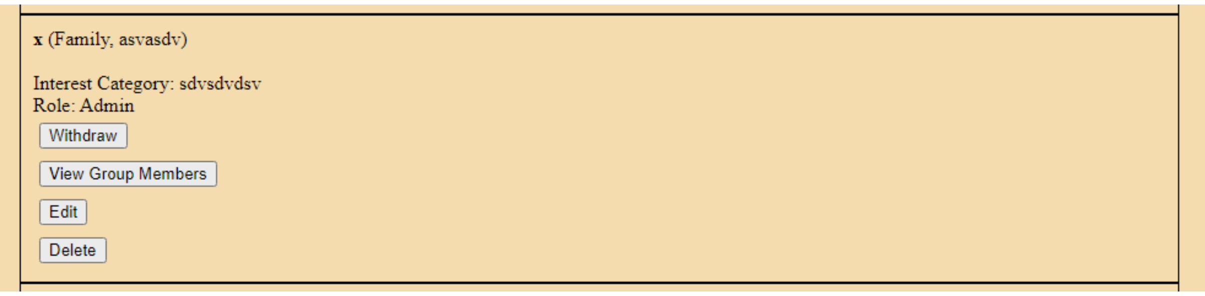
Member ID	Request Date	Action
8	2024-12-08	Approve Reject

Approving will let the requesting Member join the Group, while rejecting will not let the requesting Member join the Group.

NOTE: AN APPROVED MEMBER WILL BE A GROUP MEMBER. IT WILL NOT BE A GROUP ADMIN.

VIII. Deleting a Group (Senior+, Group Admin)

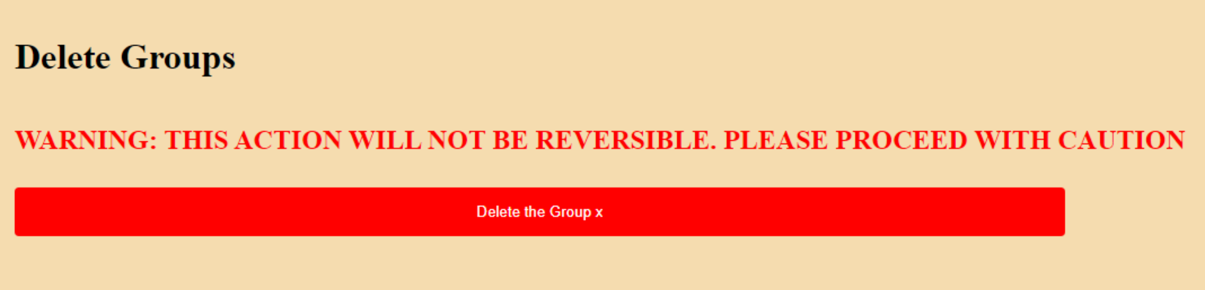
Assuming that the user is a Senior/Administrator Member that is also a Group Admin, it can see the Delete option in its Joined Groups:



x (Family, asvasdv)

Interest Category: sdvsdvdsv
Role: Admin

Clicking the “Delete” button will redirect to the Delete Groups page:



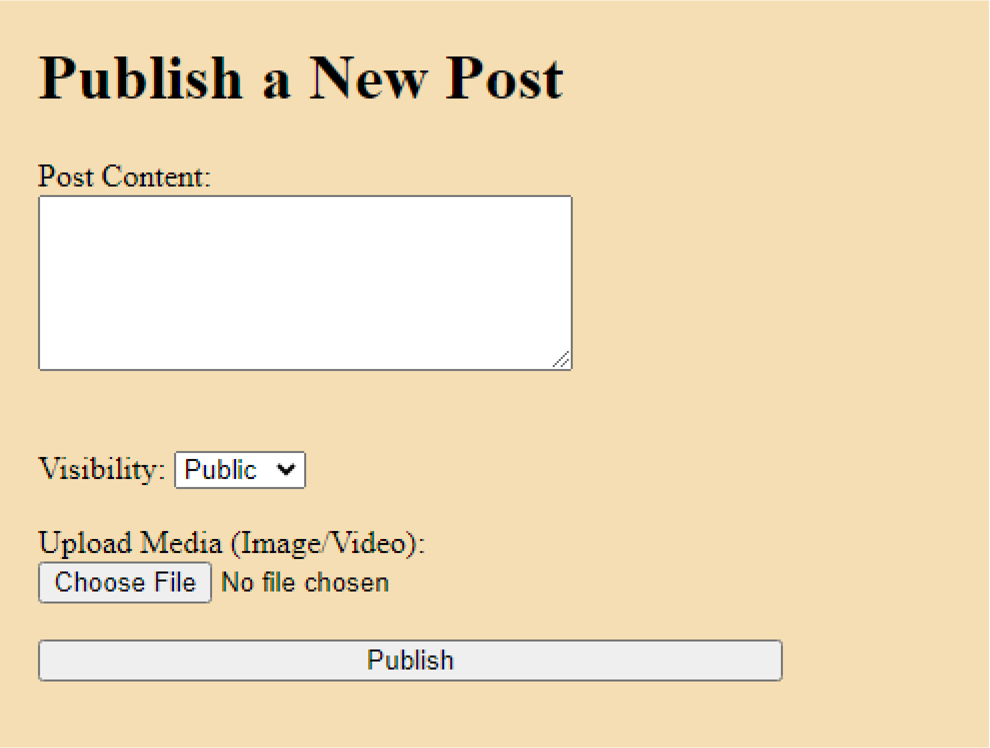
Delete Groups

WARNING: THIS ACTION WILL NOT BE REVERSIBLE. PLEASE PROCEED WITH CAUTION

Clicking the “Delete the Group <GroupName>” button will make a confirmation pop-up appear. If you click “Ok”, the Group <GroupName> will be deleted from the database.

Section 5: Publishing Posts

On the COSN header, you can click the “Publish Posts” tab. This will redirect you to a page that lets you publish Posts:



Publish a New Post

Post Content:

Visibility: Public

Upload Media (Image/Video):

No file chosen

Post Content is simply a text box.

The Visibility select has 3 options: Public, Groups and Private.
Public Posts are visible to ALL Members.
Group Posts are visible to ALL Group Members/Admins. This implementation of COSN only allows a Group Admin to make Group Posts.
Private Posts are visible to ALL Friends.

Upload Media supports some common image and video formats, such as PNG, JPG, MP4, MKV, etc.

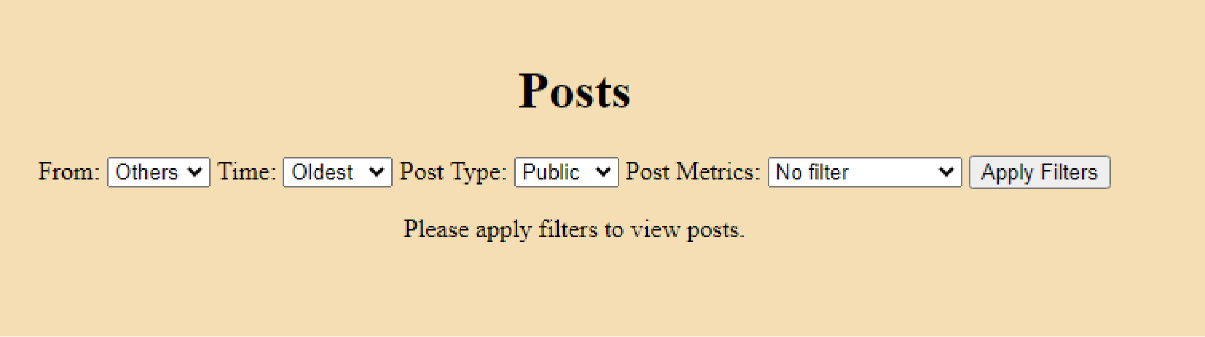
Once done filling out the form, you can publish the Post.

Section 6: Viewing Posts

NOTE: THE HOME PAGE WILL DISPLAY ALL THE POSTS POSTED BY THE LOGGED IN MEMBER.

I. Filtering

On the COSN header, you can click the "View Posts" tab. This will redirect you to a page that lets you view Posts:



Posts

From: Others Time: Oldest Post Type: Public Post Metrics: No filter

Please apply filters to view posts.

The user must apply the filters before being able to see Posts. The filters will be listed below:

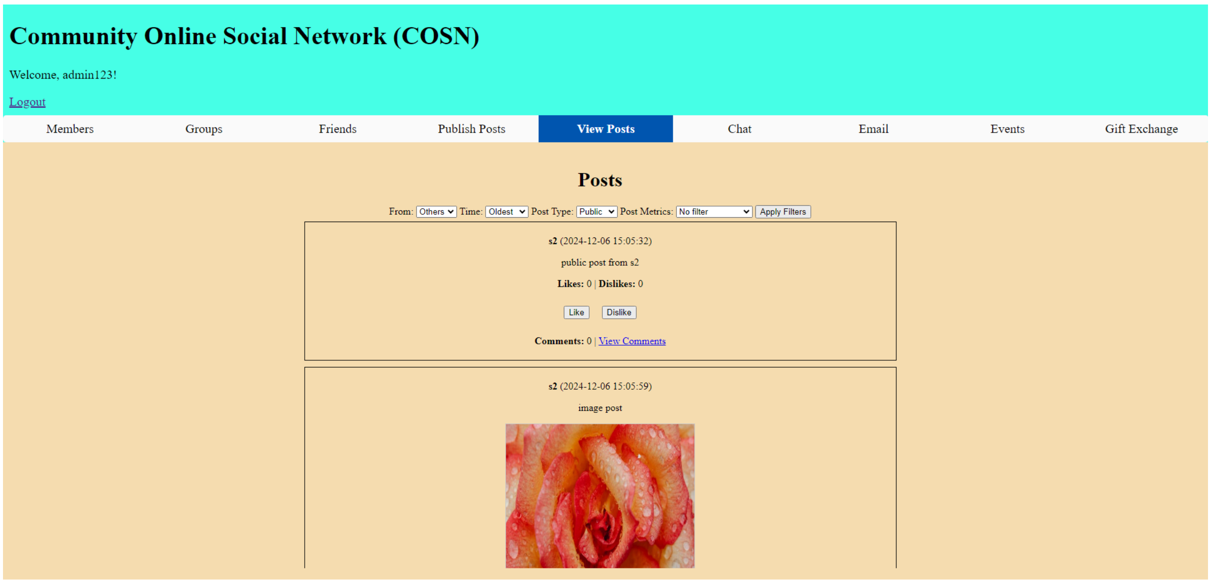
From: Others / You

Time: Oldest / Newest

Post Type: Public / Group / Private

Post Metrics: No filter / Most Likes / Most Comments

Once filtered, we can see the Posts:



Community Online Social Network (COSN)

Welcome, admin123!

Posts

From: Others Time: Oldest Post Type: Public Post Metrics: No filter

s2 (2024-12-06 15:05:32)

public post from s2

Likes: 0 | **Dislikes:** 0

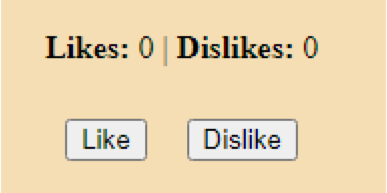
Comments: 0 | View Comments

s2 (2024-12-06 15:05:59)

image post

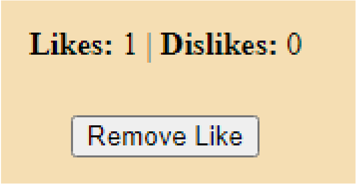
II. Likes/Dislikes

In a Post, there are buttons to like and dislike a Post.



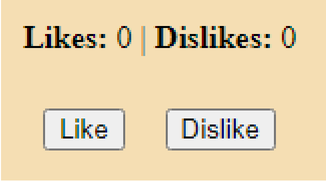
Likes: 0 | **Dislikes:** 0

Clicking “Like” will result in this:



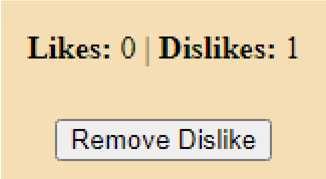
Likes: 1 | **Dislikes:** 0

Clicking “Remove Like” will result in this:



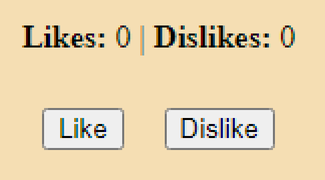
Likes: 0 | **Dislikes:** 0

Clicking “Dislike” will result in this:



Likes: 0 | **Dislikes:** 1

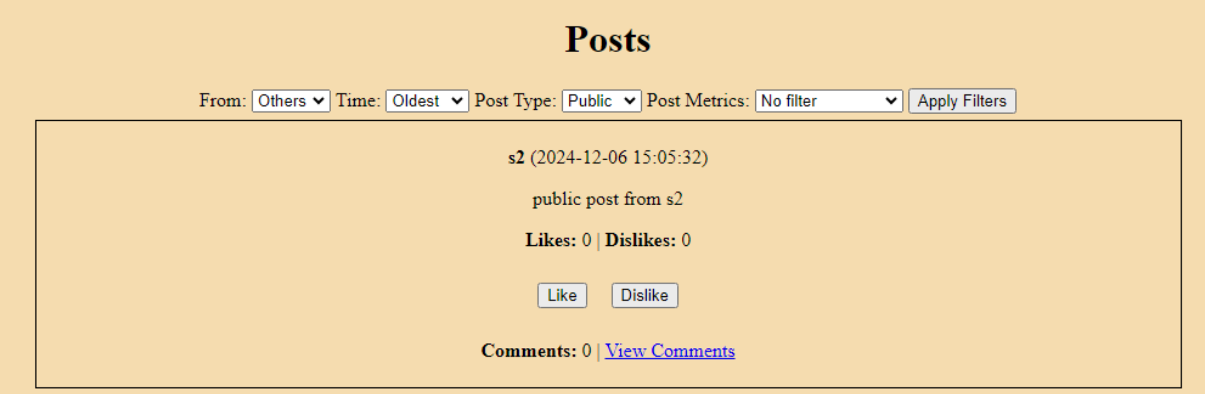
Clicking “Remove Dislike” will result in this:



Likes: 0 | **Dislikes:** 0

III. Comments

On a Post, there is a link named “View Comments” that is clickable.



Posts

From: Others Time: Oldest Post Type: Public Post Metrics: No filter

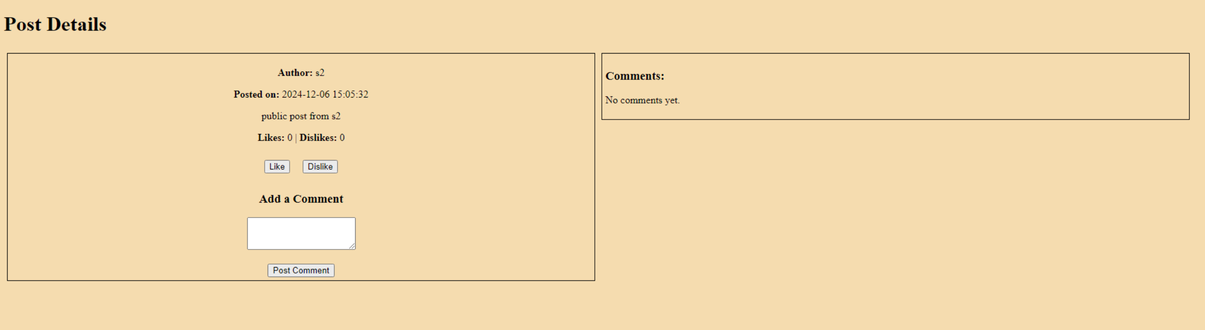
s2 (2024-12-06 15:05:32)

public post from s2

Likes: 0 | **Dislikes:** 0

Comments: 0 | View Comments

Once clicked, the related Post will be focused, and a Member can send comments:



Post Details

Author: s2

Posted on: 2024-12-06 15:05:32

public post from s2

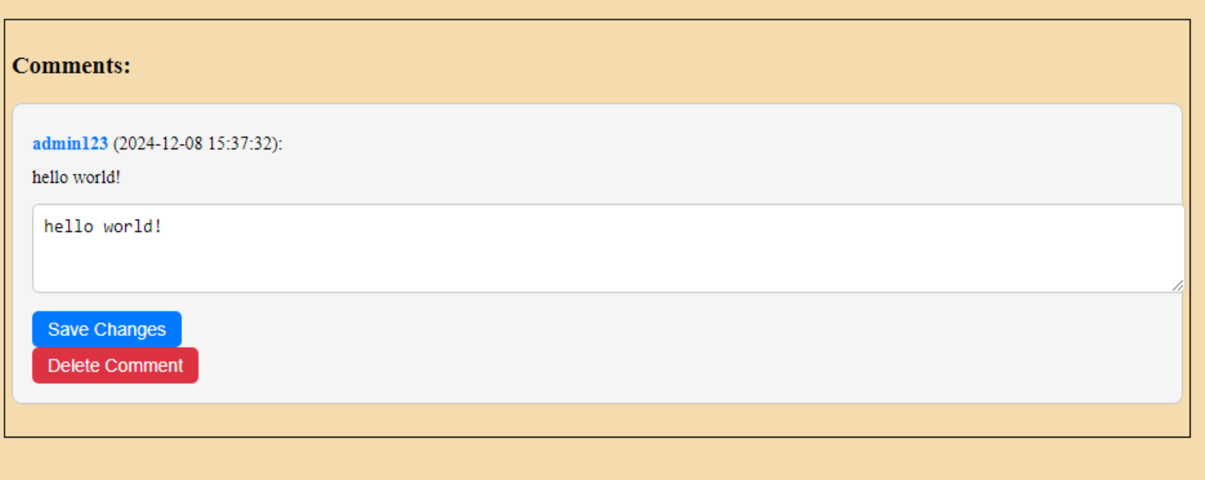
Likes: 0 | **Dislikes:** 0

Add a Comment

Comments:

No comments yet.

After writing something in the text box and clicking “Post Comment”:



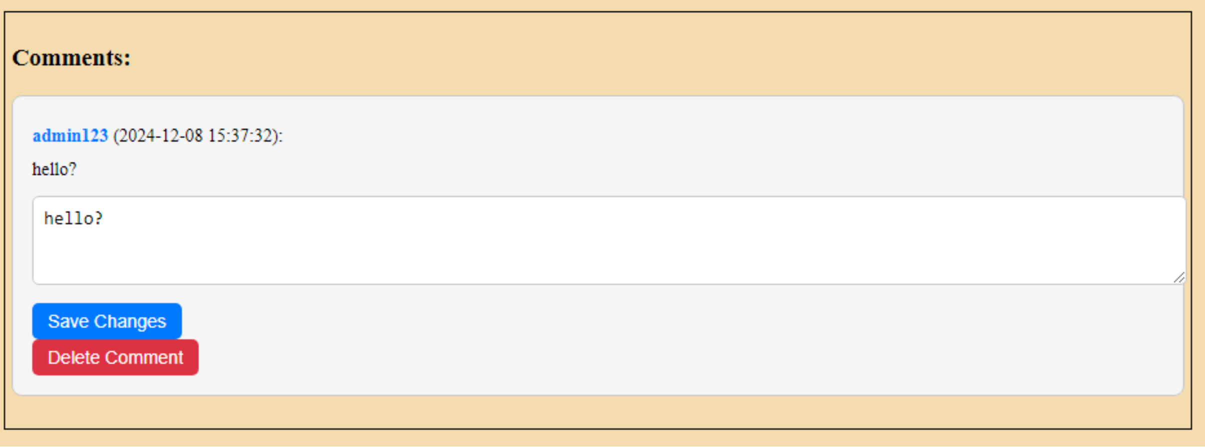
Comments:

admin123 (2024-12-08 15:37:32):

hello world!

```
hello world!
```

Editing the comment to “hello?”, then clicking the “Save Changes” button:



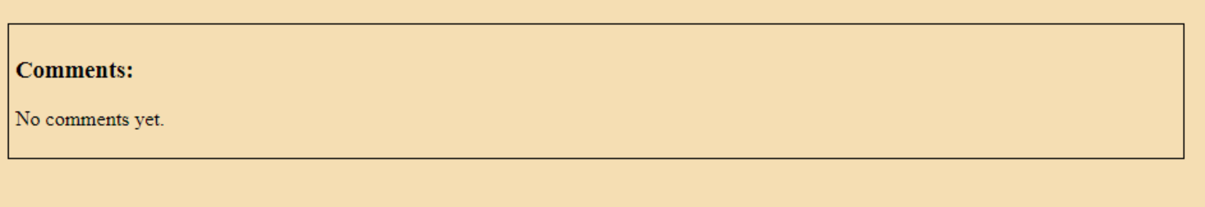
Comments:

admin123 (2024-12-08 15:37:32):

hello?

```
hello?
```

Clicking the “Delete Comment” button:

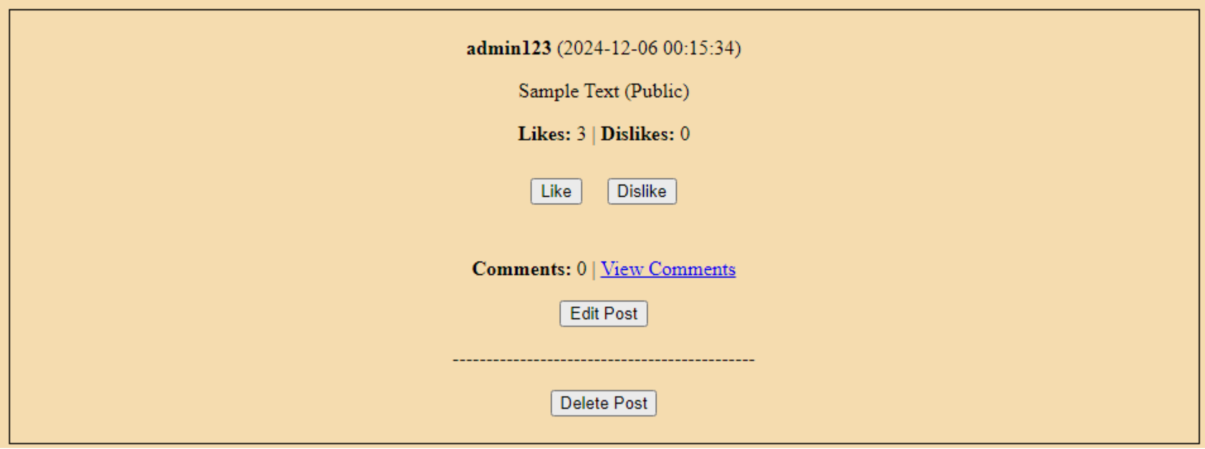


Comments:

No comments yet.

IV. Editing a Post

If a Post is sent by the logged-in Member, there will be “Edit Post” and “Delete Post” buttons.



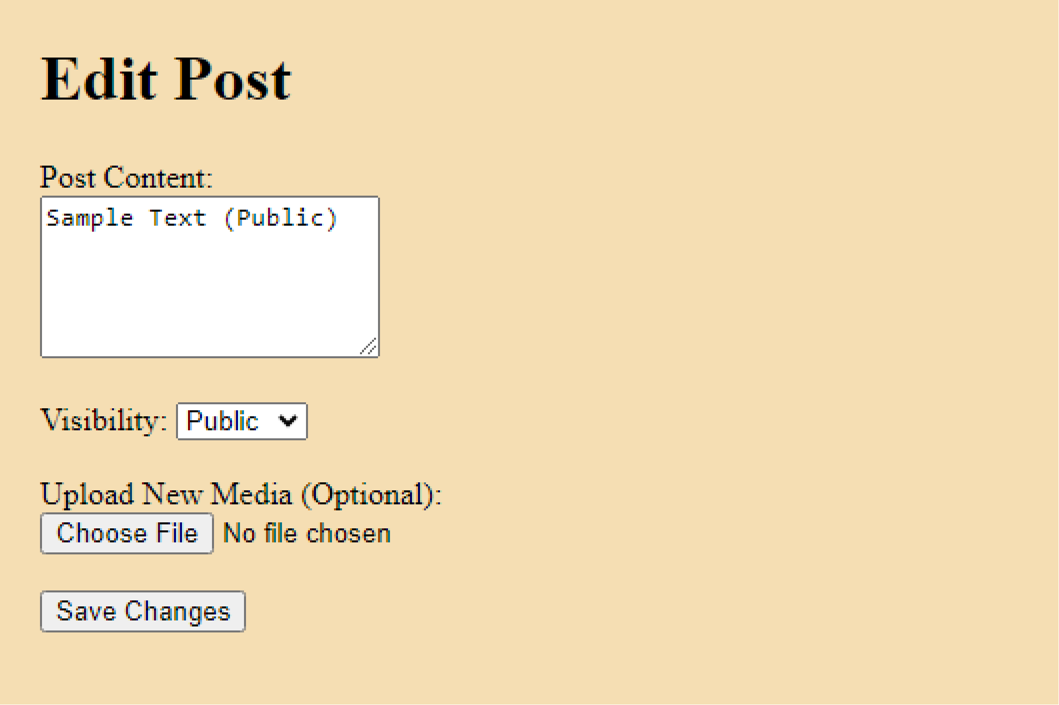
admin123 (2024-12-06 00:15:34)

Sample Text (Public)

Likes: 3 | **Dislikes:** 0

Comments: 0 | View Comments

Clicking on the “Edit Post” button will redirect the user to a “Edit Post” page:



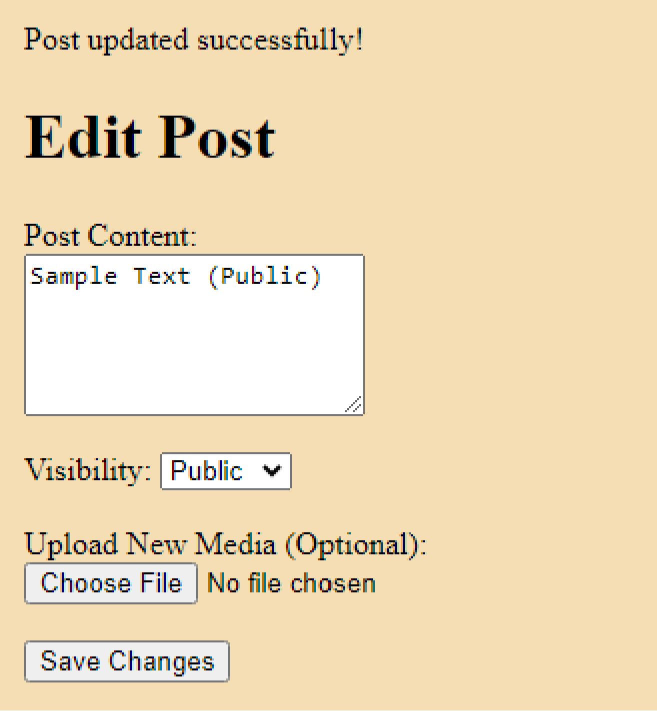
Edit Post

Post Content: Sample Text (Public)

Visibility: Public

Upload New Media (Optional): No file chosen

If we change “Sample Text (Public)” to “Sample Text (Public Report)”, then click “Save Changes”, we should see “Post updated successfully!” at the top of the page:



Post updated successfully!

Edit Post

Post Content: Sample Text (Public)

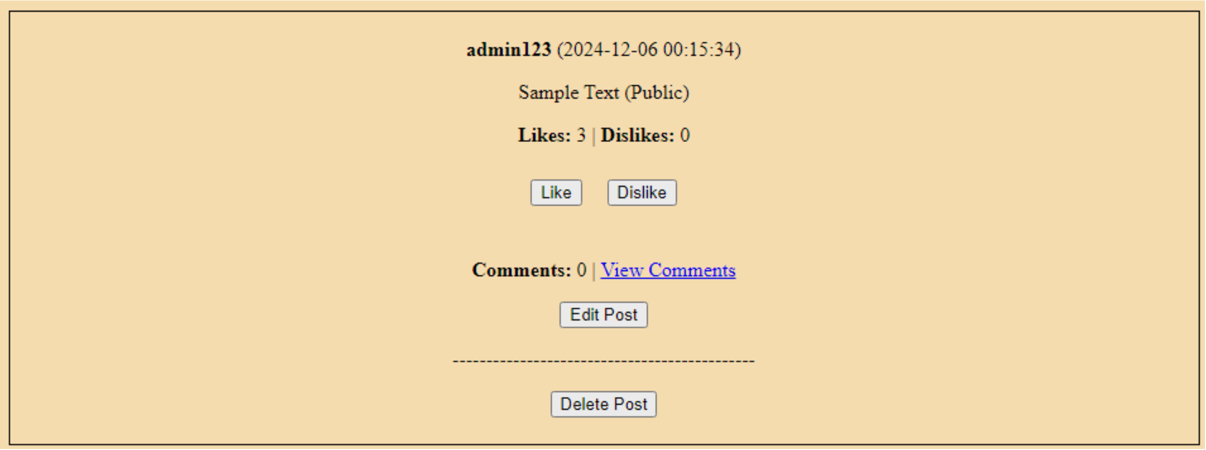
Visibility: Public

Upload New Media (Optional): No file chosen

NOTE: IT MAY LOOK LIKE THE CONTENT HAS NOT CHANGED, DUE TO POST CONTENT REFRESHING WITH THE PREVIOUS VALUE, BUT IF YOU CHECK THE POST, THE CHANGES ARE REFLECTED.

V. Deleting a Post

If a Post is sent by the logged-in Member, there will be “Edit Post” and “Delete Post” buttons.



admin123 (2024-12-06 00:15:34)

Sample Text (Public)

Likes: 3 | **Dislikes:** 0

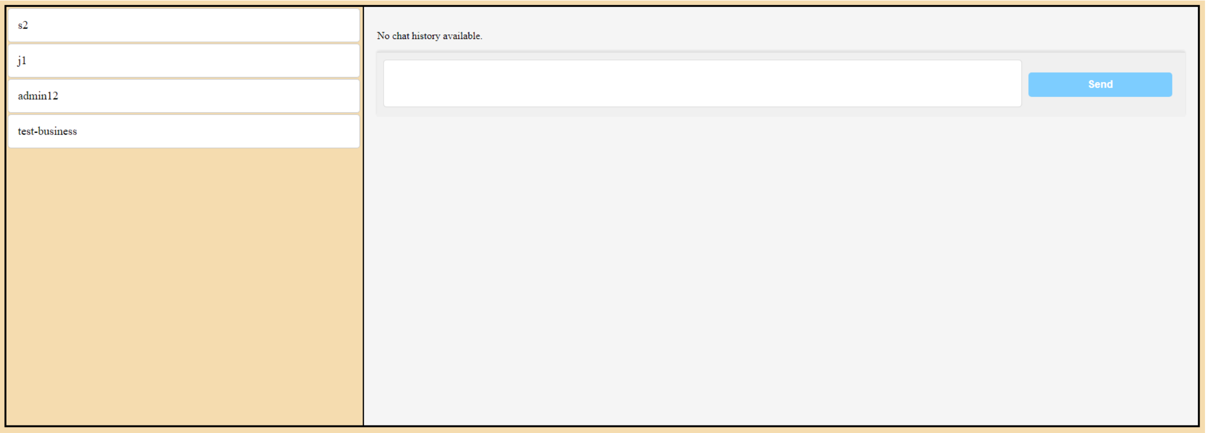
Comments: 0 | View Comments

This implementation of COSN was designed such that only the Member that published a Post can delete it.

Clicking the “Delete Post” button will simply remove the Post, without any confirmation pop-up.

Section 7: Chat

On the COSN header, you can click the “Chat” tab. This will redirect you to a page that lets you chat with Friends:



s2

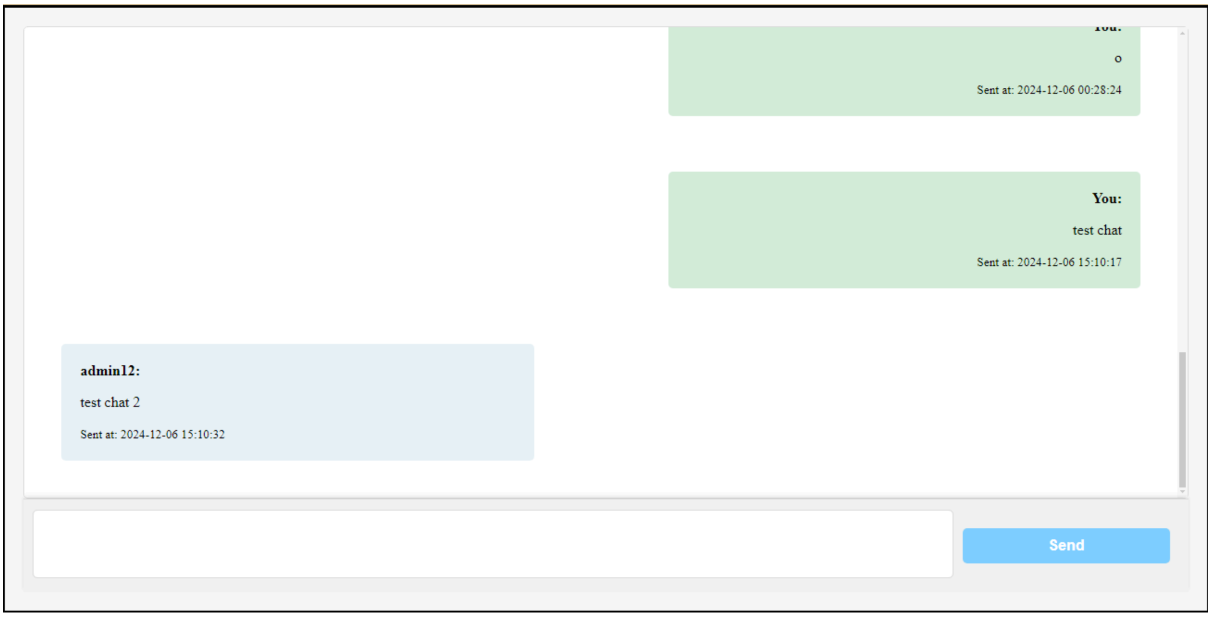
j1

admin12

test-business

No chat history available.

On the left, you may select a Friend Member, which will display the private chat logs between you and the Friend Member.



You:

o

Sent at: 2024-12-06 00:28:24

You:

test chat

Sent at: 2024-12-06 15:10:17

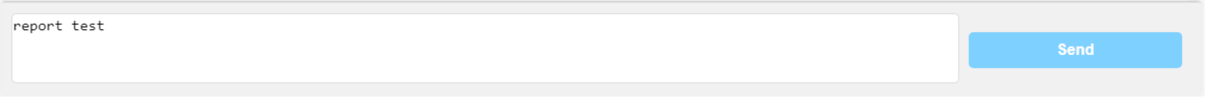
admin12:

test chat 2

Sent at: 2024-12-06 15:10:32

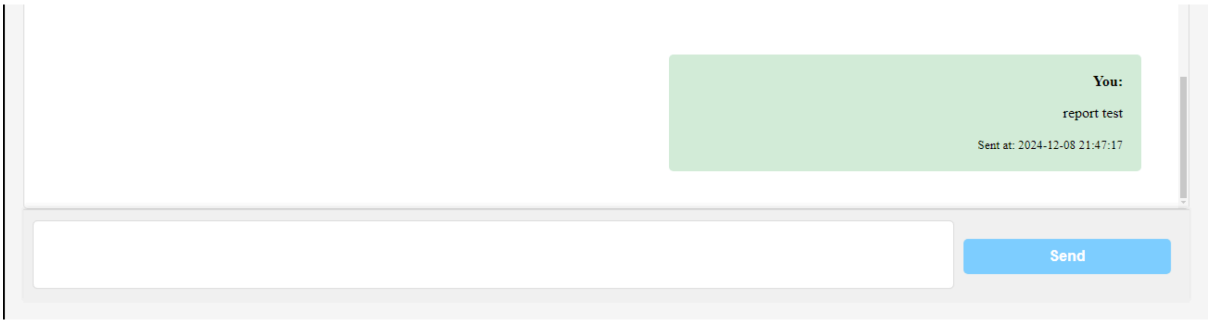
The chat logs are sorted from newest to oldest, bottom to top.
To send a message, simply type in the text box, and click the “Send” button.

[BEFORE SENDING]



report test

[AFTER SENDING]



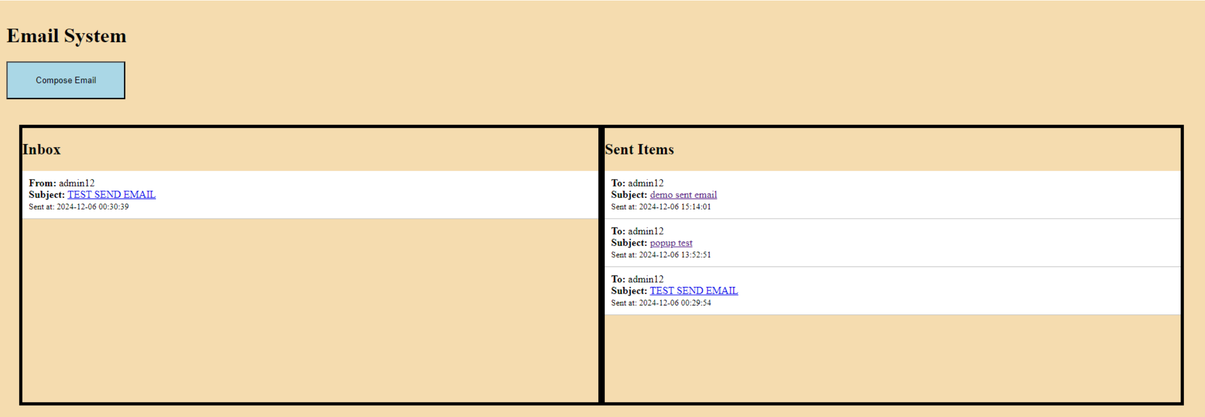
You:

report test

Sent at: 2024-12-08 21:47:17

Section 8: Email

On the COSN header, you can click the “Email” tab. This will redirect you to a page that lets you send emails to any Members:



Inbox

From: admin12
Subject: TEST SEND EMAIL
Sent at: 2024-12-06 00:30:39

Sent Items

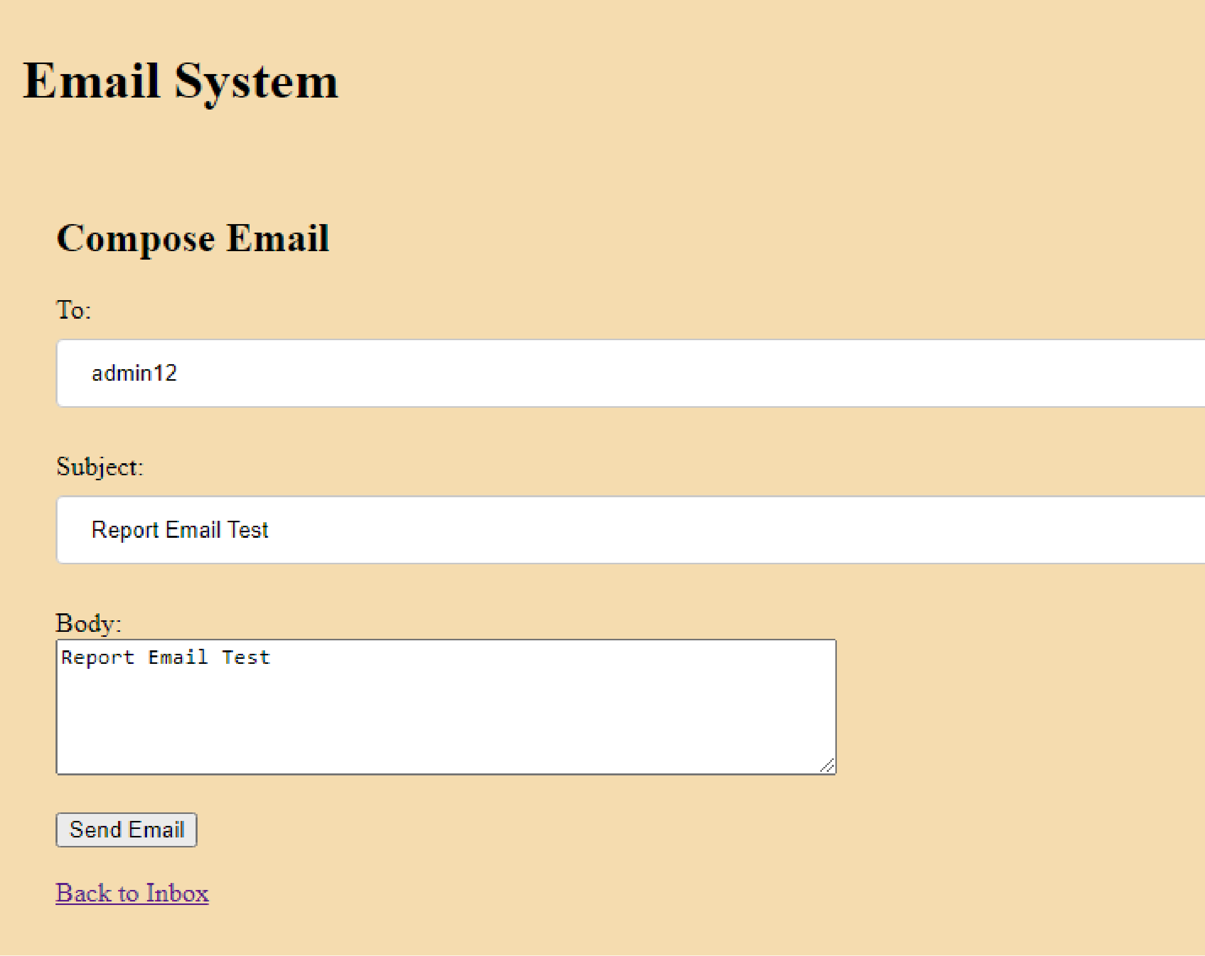
To: admin12
Subject: demo sent email
Sent at: 2024-12-06 15:14:01

To: admin12
Subject: popup test
Sent at: 2024-12-06 13:52:51

To: admin12
Subject: TEST SEND EMAIL
Sent at: 2024-12-06 00:29:54

Clicking on the “Compose Email” button will redirect to a page where the user can compose a mail to another user.

For example, we are logged in as “admin123” and we want to send an email to “admin12”:

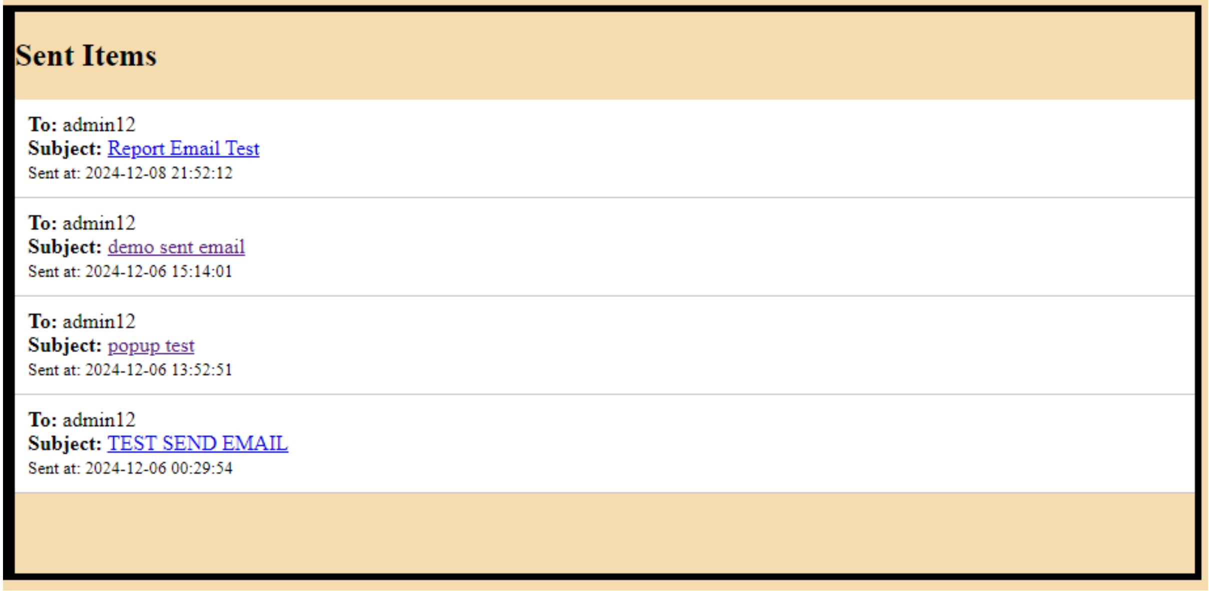


To: admin12

Subject: Report Email Test

Body: Report Email Test

Once “Send Email” is clicked, on admin123’s end, the sent email should be reflected in “Sent Items”:



Sent Items

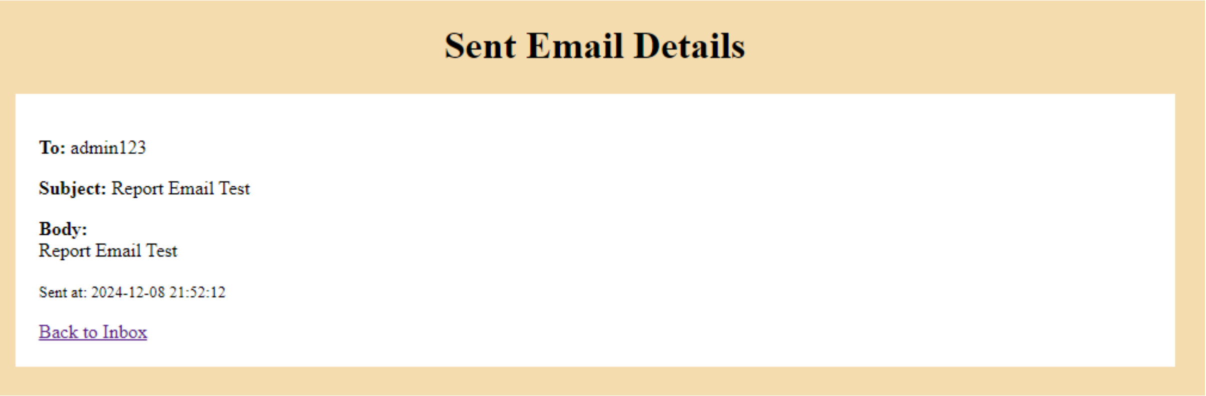
To: admin12
Subject: Report Email Test
Sent at: 2024-12-08 21:52:12

To: admin12
Subject: demo sent email
Sent at: 2024-12-06 15:14:01

To: admin12
Subject: popup test
Sent at: 2024-12-06 13:52:51

To: admin12
Subject: TEST SEND EMAIL
Sent at: 2024-12-06 00:29:54

Clicking on “Report Email Test” will show the sent email:



Sent Email Details

To: admin123

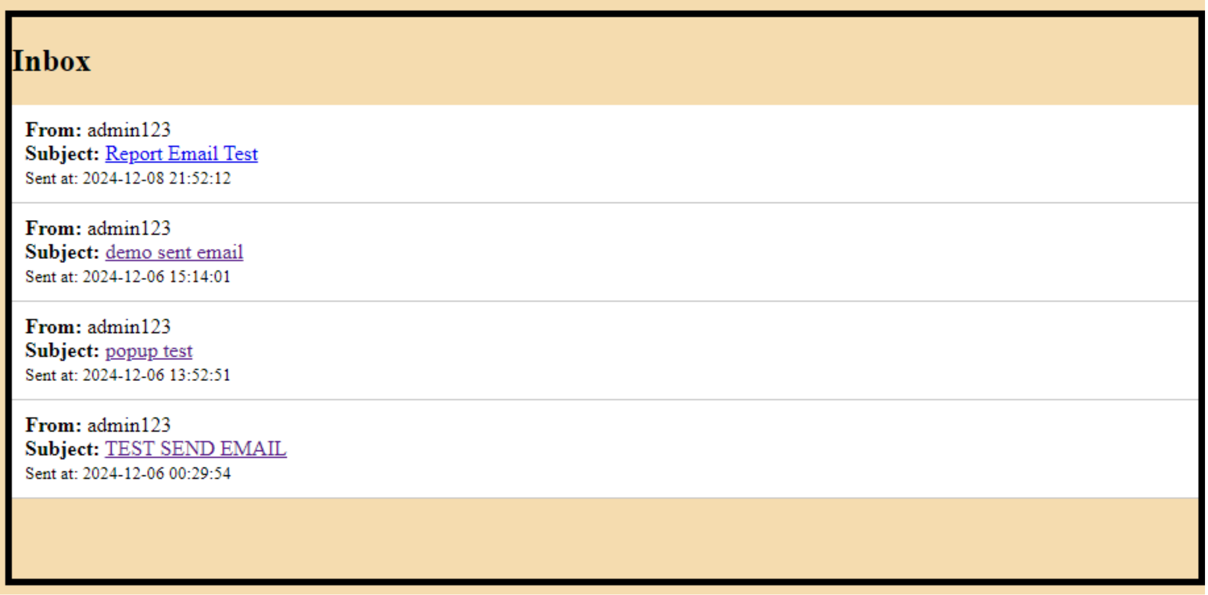
Subject: Report Email Test

Body:
Report Email Test

Sent at: 2024-12-08 21:52:12

Back to Inbox

On admin12’s end, we should see the sent email in the “Inbox”:



Inbox

From: admin123
Subject: Report Email Test
Sent at: 2024-12-08 21:52:12

From: admin123
Subject: demo sent email
Sent at: 2024-12-06 15:14:01

From: admin123
Subject: popup test
Sent at: 2024-12-06 13:52:51

From: admin123
Subject: TEST SEND EMAIL
Sent at: 2024-12-06 00:29:54

Whenever either the sender or the receiver does not view the email, the Inbox item looks like this:

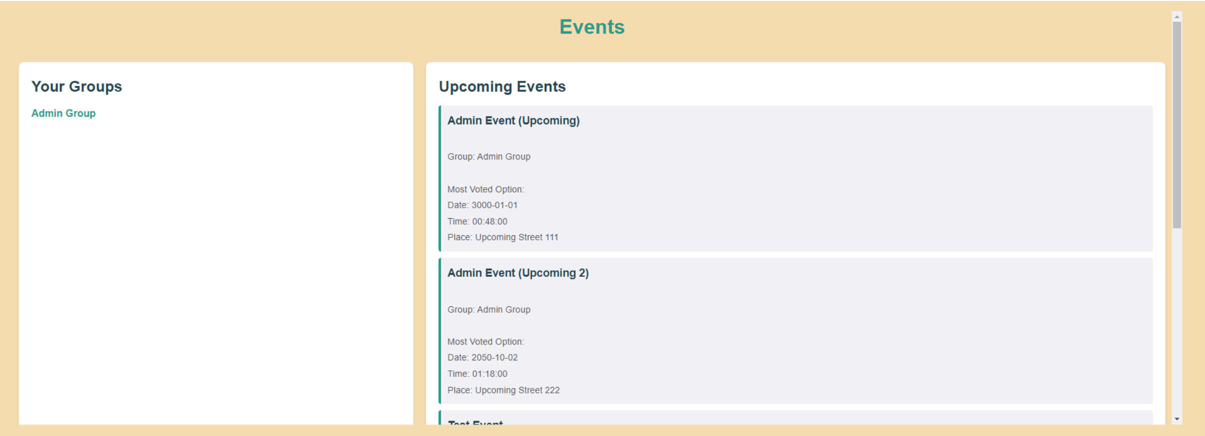
From: <text>
Subject: < … >
Sent at: <text>

This signifies an unviewed email. If that email remains unviewed, there will be a pop-up that appears every 5 seconds telling the user that there is a new mail. Once either the sender or receiver views the email, the pop-up will stop appearing.

Section 9: Events

I. General

On the COSN header, you can click the “Events” tab. This will redirect you to a page that lets you see a Group’s Events:



Events

Your Groups

Admin Group

Upcoming Events

Admin Event (Upcoming)

Group: Admin Group

Most Voted Option:
Date: 3000-01-01
Time: 00:48:00
Place: Upcoming Street 111

Admin Event (Upcoming 2)

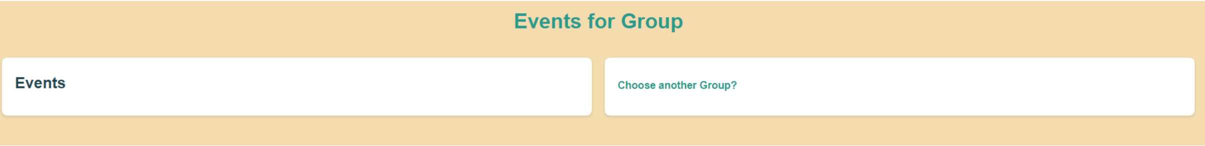
Group: Admin Group

Most Voted Option:
Date: 2050-10-02
Time: 01:18:00
Place: Upcoming Street 222

Test Event

Scrolling down will also display Previous Events.

Whenever a Group Member clicks on a group in “Your Groups”, it will see this:

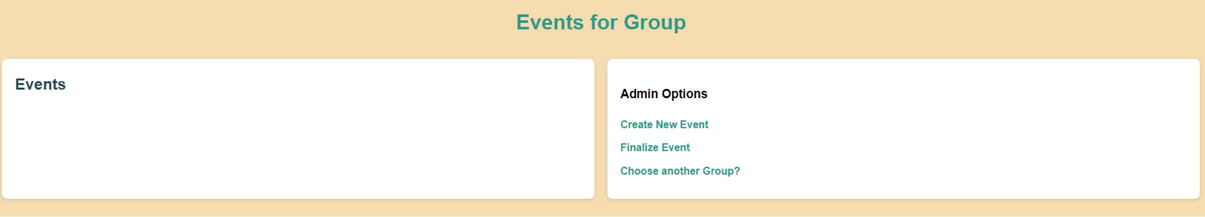


Events for Group

Events

Choose another Group?

Whenever a Group Admin clicks on a group in “Your Groups”, it will see this:



Events for Group

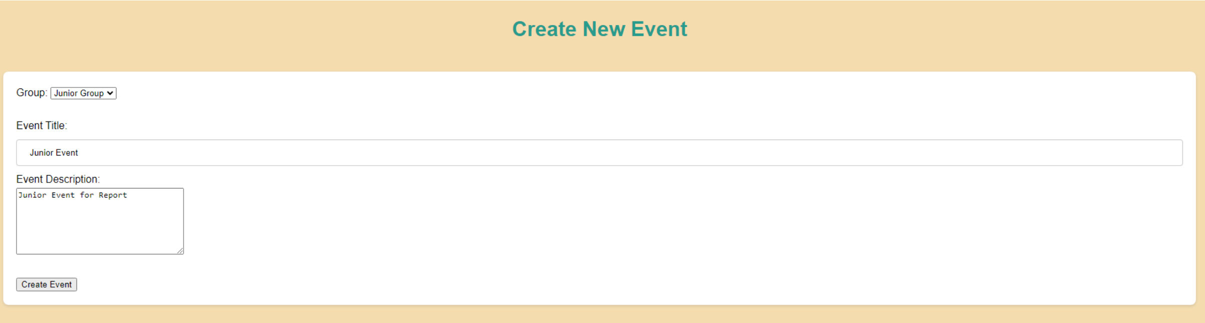
Events

Admin Options

Create New Event
Finalize Event
Choose another Group?

II. Create New Event (Group Admin)

Assuming that the Group Admin has already clicked on a group in “Your Groups”, then it can click on “Create New Event” to redirect to an event creation page:



Create New Event

Group: Junior Group

Event Title:

Junior Event

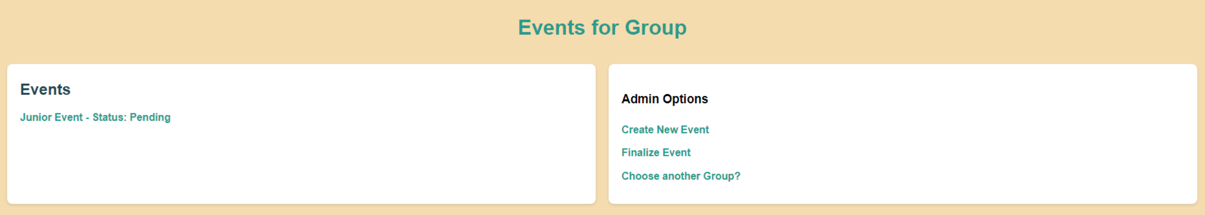
Event Description:

Junior Event for Report

We will now create an Event for Junior Group, where its title is “Junior Event” and its event description is “Junior Event for Report”.

Click on the “Create Event” button to create the event.

We should now see a Pending Event in the selected Event Group. This is visible to ALL Group Members:



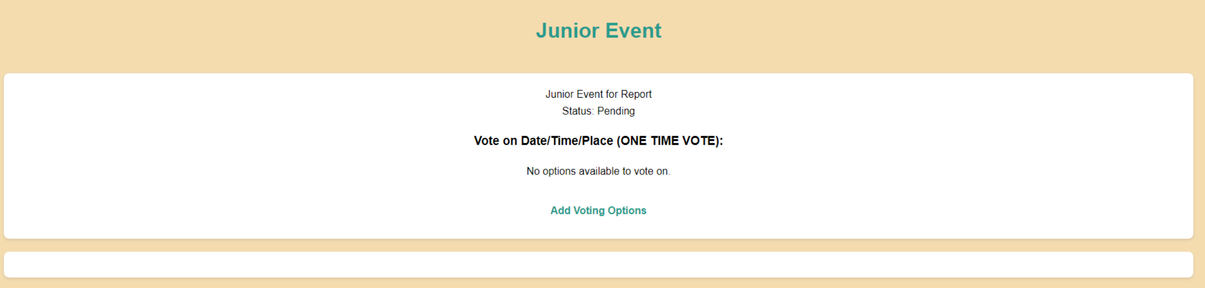
Events for Group

Events

Junior Event - Status: Pending

Admin Options

Now clicking on Junior Event…



Junior Event

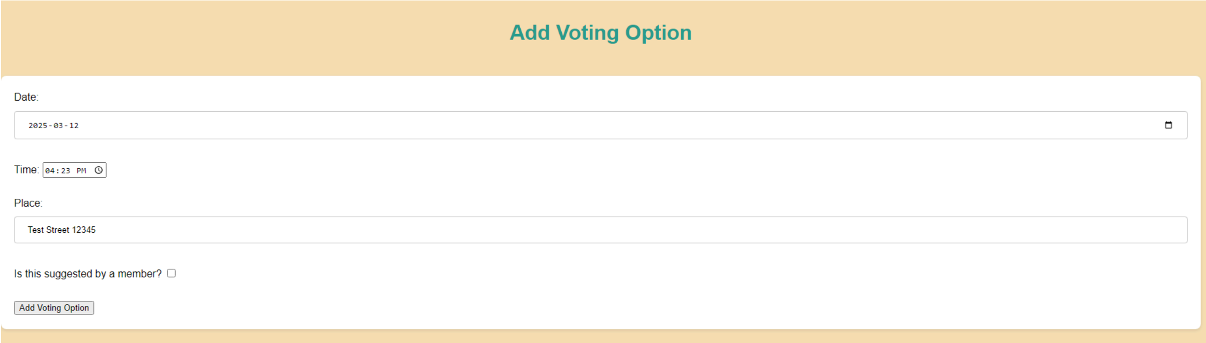
Junior Event for Report

Status: Pending

Vote on Date/Time/Place (ONE TIME VOTE):

No options available to vote on.

Clicking on “Add Voting Options” leads to a page where you can add voting options using a form:



Add Voting Option

Date:

2025-03-12

Time: 04:23 PM

Place:

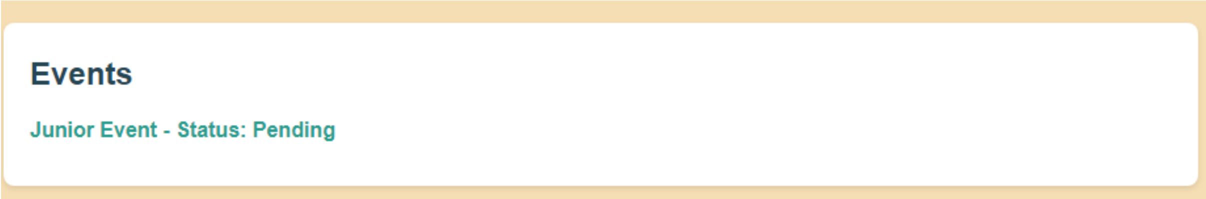
Test Street 12345

Is this suggested by a member? □

Add Voting Option

III. Voting on a Date/Time/Place for Event (Group Members)

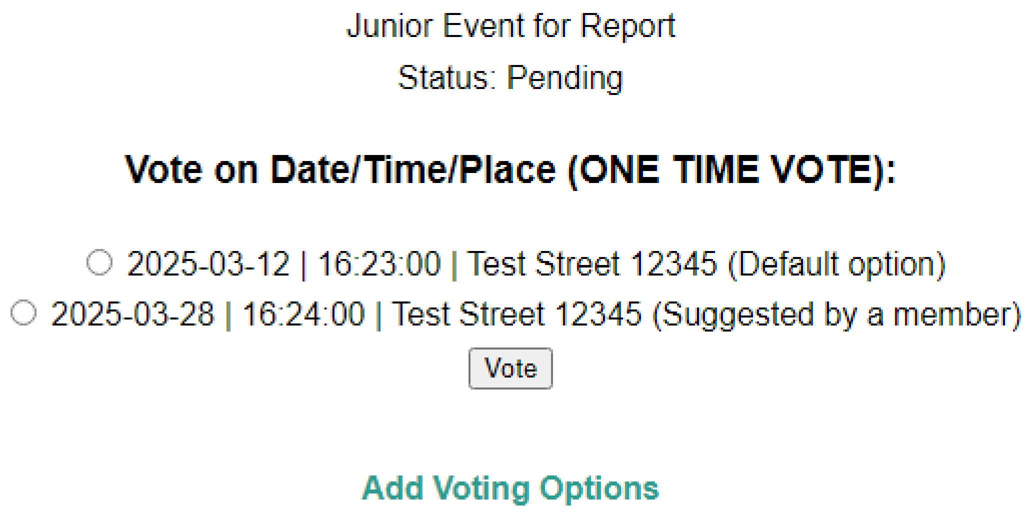
Assuming that the Group Member has already clicked on a group in “Your Groups”, then it can click on the Pending Event to vote on it:



Events

Junior Event - Status: Pending

After a Group Admin has added 2 Event Voting Options to Junior Group, a Group Member see this:



Junior Event for Report

Status: Pending

Vote on Date/Time/Place (ONE TIME VOTE):

○ 2025-03-12 | 16:23:00 | Test Street 12345 (Default option)

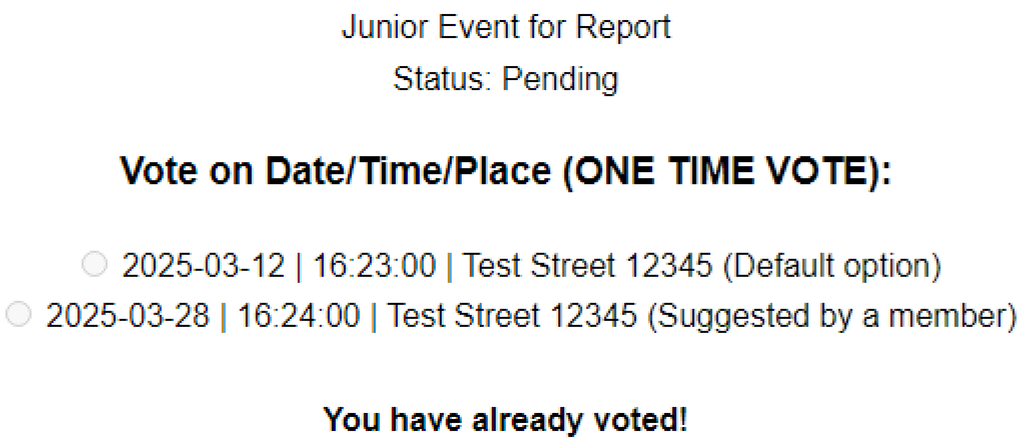
○ 2025-03-28 | 16:24:00 | Test Street 12345 (Suggested by a member)

Vote

Add Voting Options

NOTE: VOTING IS A ONE-TIME OPERATION. ONCE THE VOTE IS IN, YOU CANNOT CHANGE IT.

We will now vote on the Default option:



Junior Event for Report

Status: Pending

Vote on Date/Time/Place (ONE TIME VOTE):

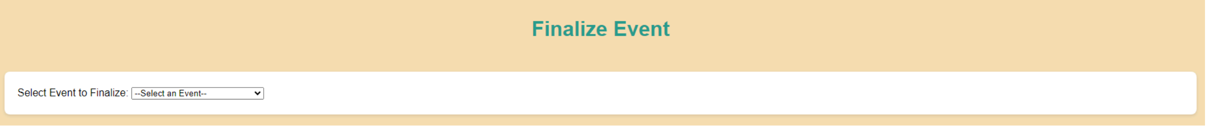
○ 2025-03-12 | 16:23:00 | Test Street 12345 (Default option)

○ 2025-03-28 | 16:24:00 | Test Street 12345 (Suggested by a member)

You have already voted!

IV. Finalize Event (Group Admin)

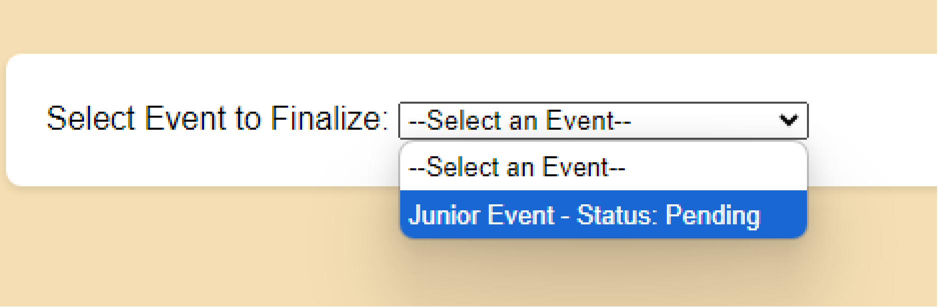
Assuming that the Group Admin has already clicked on a group in “Your Groups”, then it can click on “Finalize Event” to redirect to an event creation page:



Finalize Event

Select Event to Finalize: --Select an Event--

Select the Event in the Group:

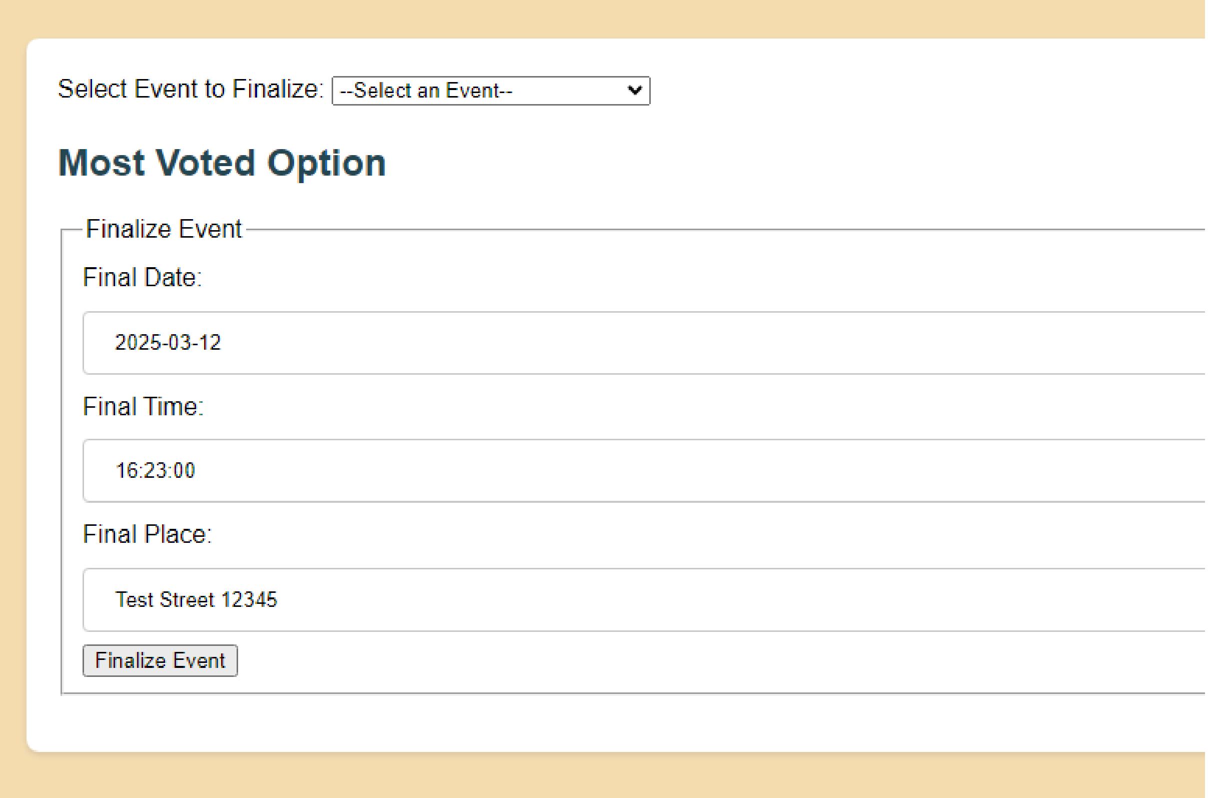


Select Event to Finalize: --Select an Event--

--Select an Event--

Junior Event - Status: Pending

Finalize the event by clicking on “Finalize Event”:



Select Event to Finalize: --Select an Event--

Most Voted Option

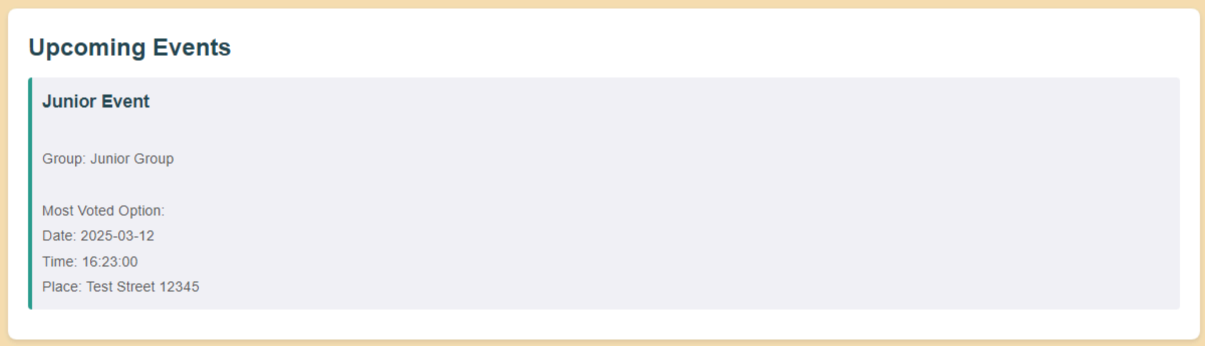
Finalize Event

Final Date: 2025-03-12

Final Time: 16:23:00

Final Place: Test Street 12345

The Group Members should now see this in “Upcoming Events”, if our current date is before 2025-03-12 16:23:00 :



Upcoming Events

Junior Event

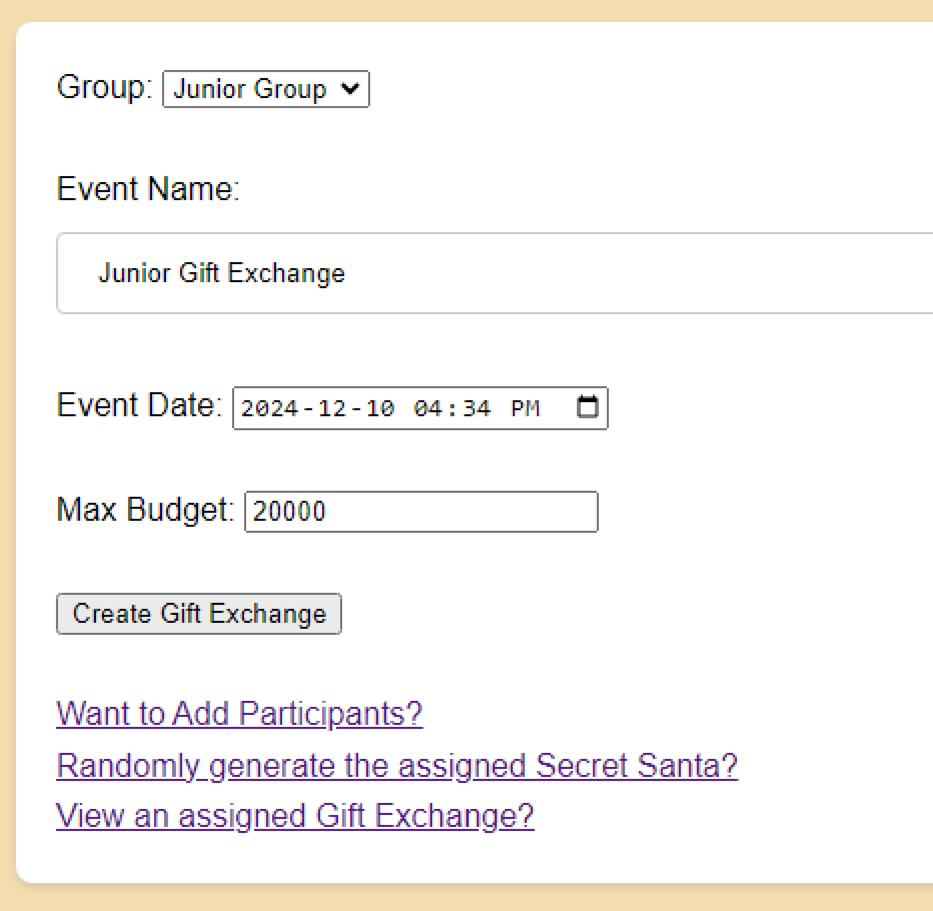
Group: Junior Group

Most Voted Option:
Date: 2025-03-12
Time: 16:23:00
Place: Test Street 12345

Section 10: Gift Exchange

I. Creating a Gift Exchange

On the COSN header, you can click the “Gift Exchange” tab. This will redirect us to a page that lets us create a Gift Exchange for a Group:



Group: Junior Group

Event Name: Junior Gift Exchange

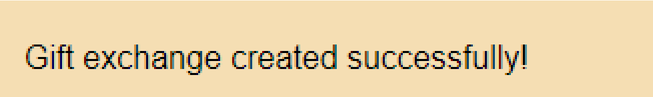
Event Date: 2024-12-10 04:34 PM

Max Budget: 20000

Create Gift Exchange
Want to Add Participants?
Randomly generate the assigned Secret Santa?
View an assigned Gift Exchange?

Any Group Members can initiate a Gift Exchange within the Group. We can submit the form by clicking the “Create Gift Exchange” button.

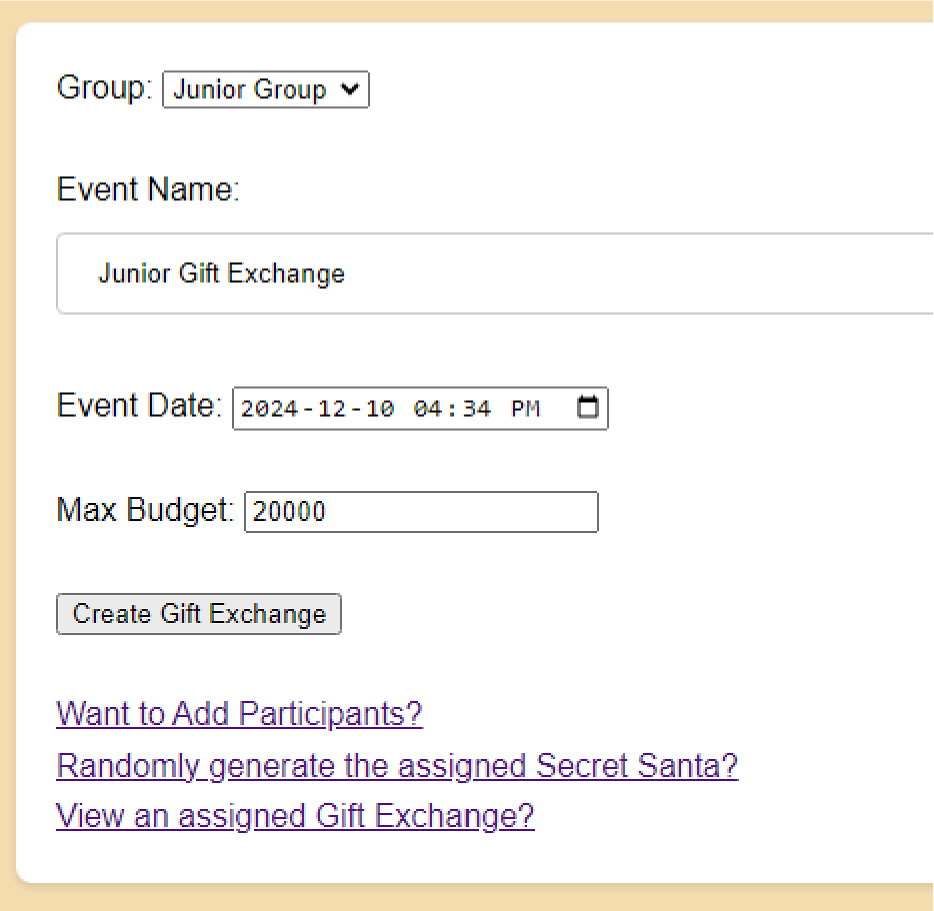
There should be a confirmation message:



Gift exchange created successfully!

II. Adding Participants to a Gift Exchange

On the COSN header, you can click the “Gift Exchange” tab. This will redirect us to a page that lets us create a Gift Exchange for a Group:



Group: Junior Group

Event Name:

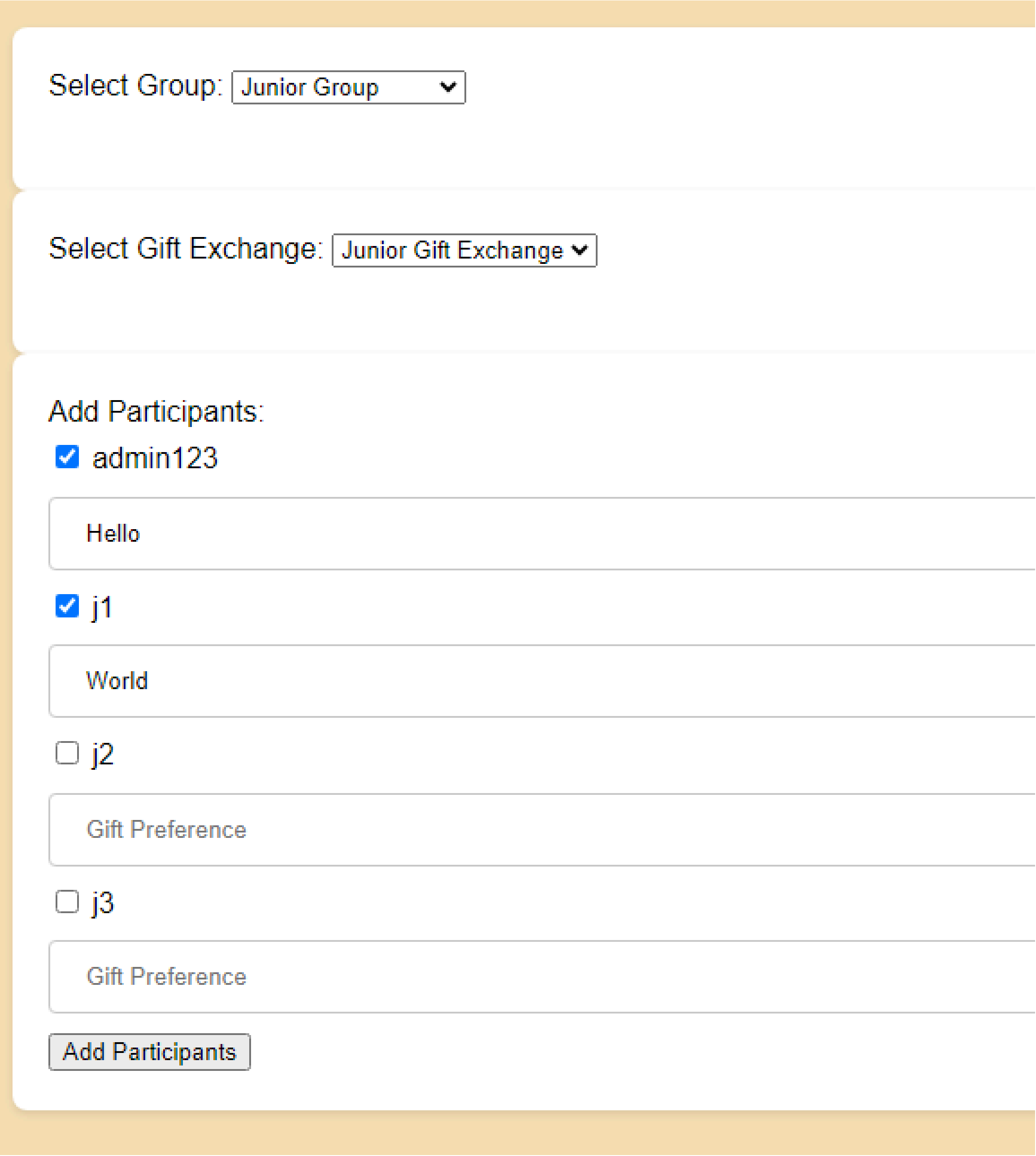
Junior Gift Exchange

Event Date: 2024-12-10 04:34 PM

Max Budget: 20000

Create Gift Exchange
Want to Add Participants?
Randomly generate the assigned Secret Santa?
View an assigned Gift Exchange?

Clicking on “Want to Add Participants?” leads us to a page where we can select the Group and the Gift Exchange:



Select Group: Junior Group

Select Gift Exchange: Junior Gift Exchange

Add Participants:

☑ admin123

Hello

☑ j1

World

□ j2

Gift Preference

□ j3

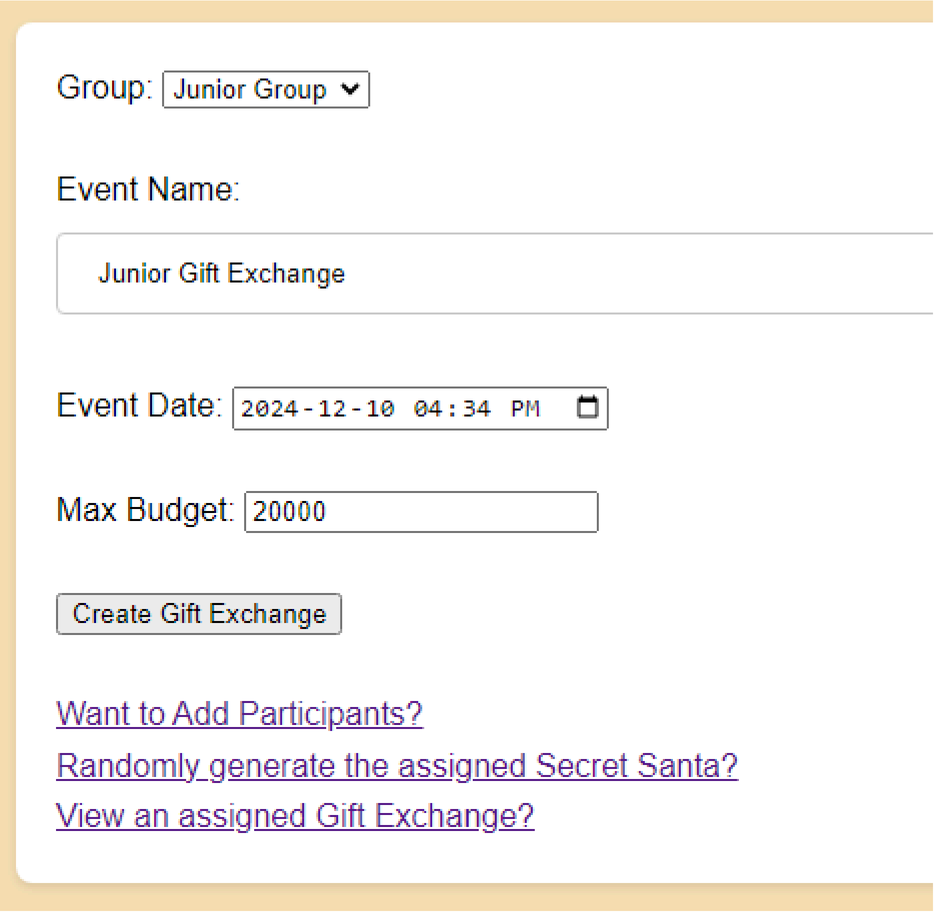
Gift Preference

We can type the Gift Preference of a Participant if known, and check the checkboxes of the participants that we want to add to the Gift Exchange.

To submit this form, click the “Add Participants” button.

III. Assign the Secret Santa

On the COSN header, you can click the “Gift Exchange” tab. This will redirect us to a page that lets us create a Gift Exchange for a Group:



Group: Junior Group

Event Name:

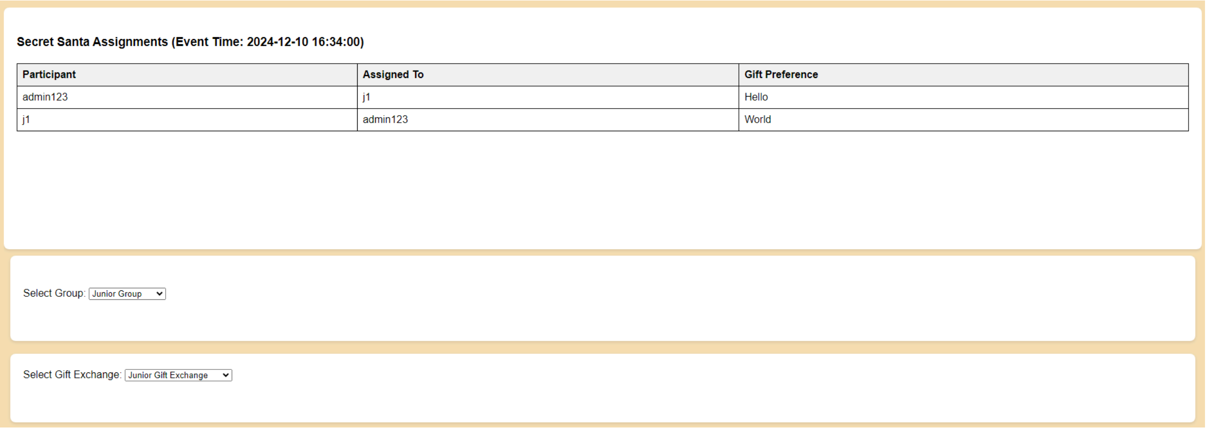
Junior Gift Exchange

Event Date: 2024-12-10 04:34 PM

Max Budget: 20000

Want to Add Participants?
Randomly generate the assigned Secret Santa?
View an assigned Gift Exchange?

Clicking on “Randomly generate the assigned Secret Santa?” leads us to a page where we can select the Group and the Gift Exchange for random assignment of Secret Santa:



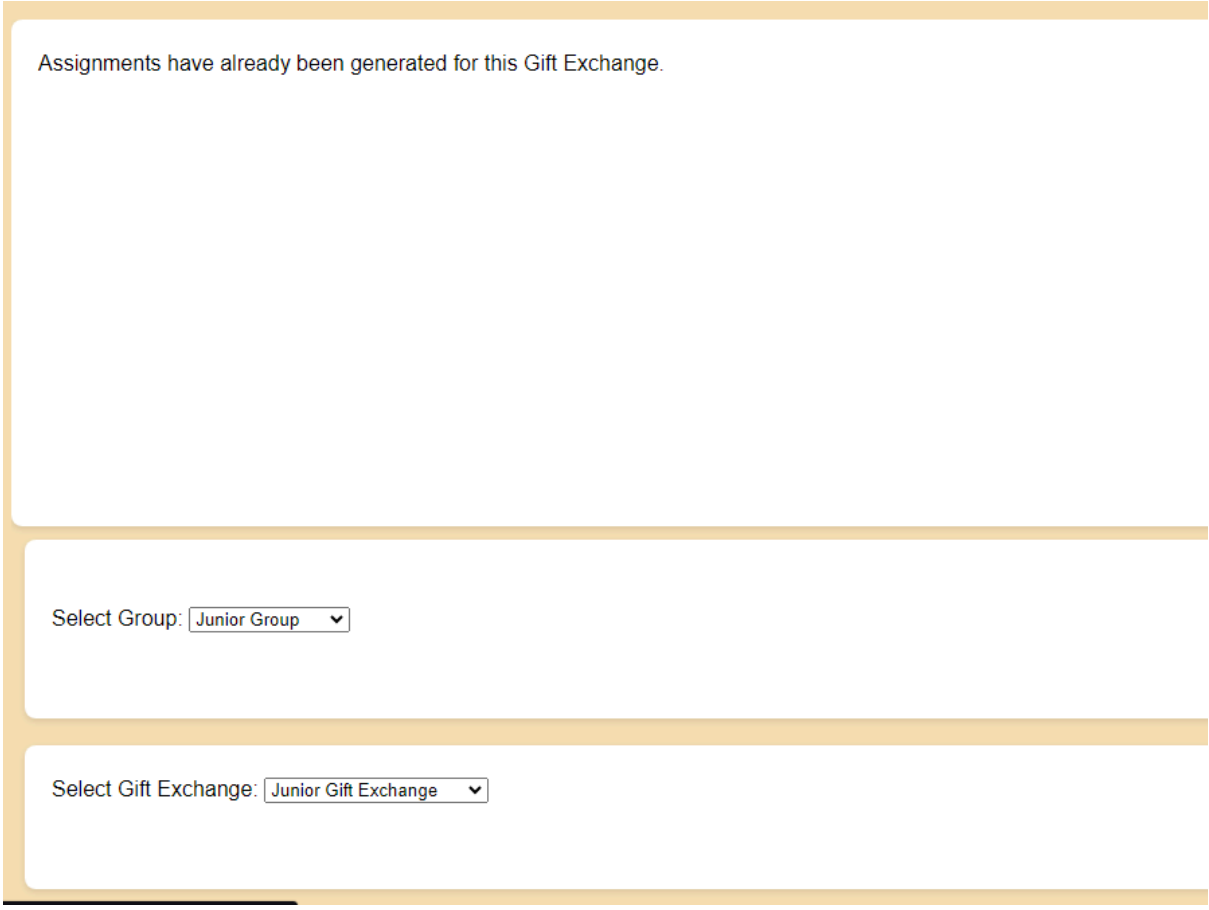
Secret Santa Assignments (Event Time: 2024-12-10 16:34:00)

Participant	Assigned To	Gift Preference
admin123	j1	Hello
j1	admin123	World

Select Group: Junior Group

Select Gift Exchange: Junior Gift Exchange

If we try to reassign the Gift Exchange, it will not be allowed:



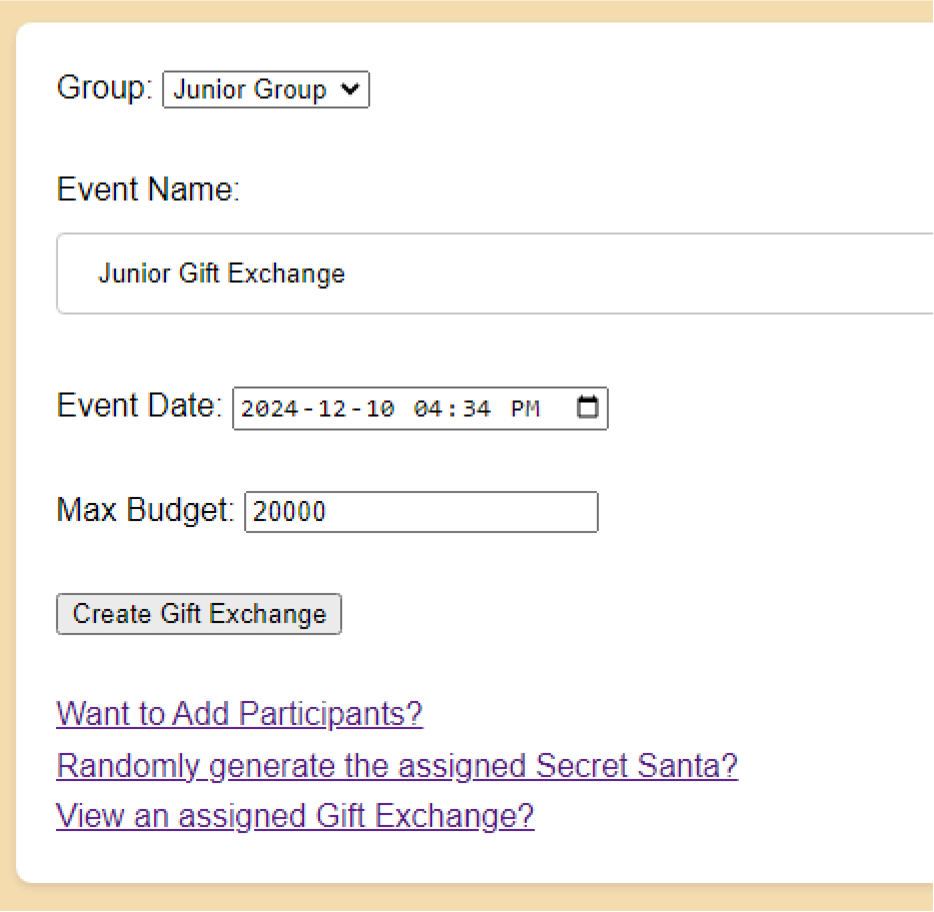
Assignments have already been generated for this Gift Exchange.

Select Group: Junior Group

Select Gift Exchange: Junior Gift Exchange

IV. View an assigned Gift Exchange

On the COSN header, you can click the “Gift Exchange” tab. This will redirect us to a page that lets us create a Gift Exchange for a Group:



Group: Junior Group

Event Name:

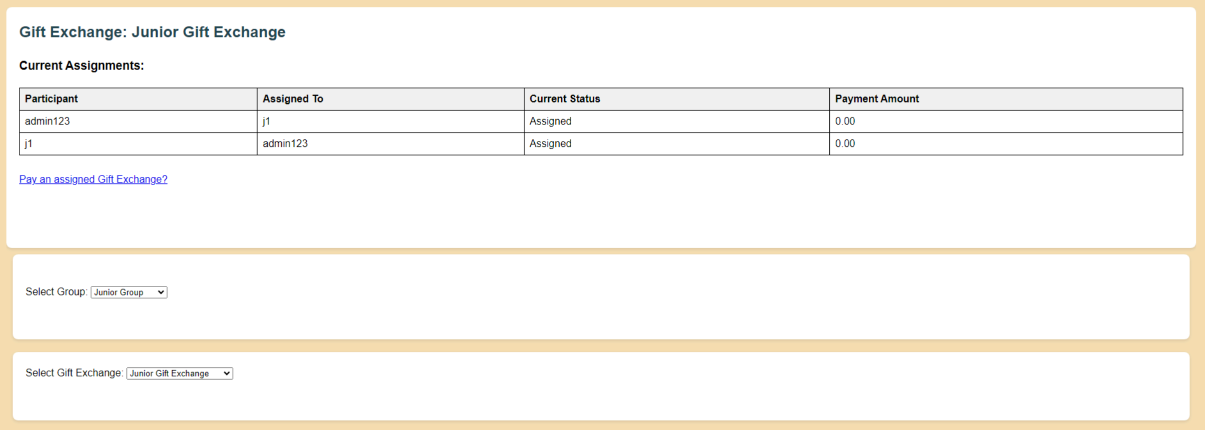
Junior Gift Exchange

Event Date: 2024-12-10 04:34 PM

Max Budget: 20000

Create Gift Exchange
Want to Add Participants?
Randomly generate the assigned Secret Santa?
View an assigned Gift Exchange?

To view an assigned Gift Exchange, click on “View an assigned Gift Exchange?”:



Gift Exchange: Junior Gift Exchange

Current Assignments:

Participant	Assigned To	Current Status	Payment Amount
admin123	j1	Assigned	0.00
j1	admin123	Assigned	0.00

Select Group: Junior Group

Select Gift Exchange: Junior Gift Exchange

V. Exchanging Gifts

NOTE: THIS FEATURE WAS NOT IMPLEMENTED PROPERLY

Potential Improvements

Database

I have noticed a simple optimization for the database relations after coding the project. Notice these tables:

```
-- Table to track likes
CREATE TABLE PostLikes (
   PostID INT,
   UserID INT,
   PRIMARY KEY (PostID, UserID),
   FOREIGN KEY (PostID) REFERENCES Posts(PostID) ON DELETE CASCADE,
   FOREIGN KEY (UserID) REFERENCES Members(MemberID) ON DELETE CASCADE
);

-- Table to track dislikes
CREATE TABLE PostDislikes (
   PostID INT,
   UserID INT,
   PRIMARY KEY (PostID, UserID),
   FOREIGN KEY (PostID) REFERENCES Posts(PostID) ON DELETE CASCADE,
   FOREIGN KEY (UserID) REFERENCES Members(MemberID) ON DELETE CASCADE
);
```

It is possible to merge these 2 tables together to reduce the number of database calls in the code. We propose this SQL script to create the merged table:

```
-- Table to track likes AND dislikes
CREATE TABLE PostLikesAndDislikes (
   PostID INT,
   UserID INT,
   UserLiked BOOLEAN DEFAULT FALSE,
   UserDisliked BOOLEAN DEFAULT FALSE,
   PRIMARY KEY (PostID, UserID),
   FOREIGN KEY (PostID) REFERENCES Posts(PostID) ON DELETE CASCADE,
   FOREIGN KEY (UserID) REFERENCES Members(MemberID) ON DELETE CASCADE
);
```

The use of this table in the code would be to fetch its entries ONLY ONE TIME to see if a user liked or disliked a post. This eliminates the need to fetch the entries from PostLikes and PostDislikes, which are 2 operations.

Features

Due to time constraints, we were unable to implement the features from the demo. To reiterate, here are the features:

1. Check if DOB is valid on Account Creation

2. GiftExchange: Giving Gifts instead of payments.

3. A Group Member should be able to attempt to publish a Group Post, which would send an approval request to a Group Admin. If a Group Admin approves the request, the Group Post will be published. Otherwise, if a Group Admin disapproves the request, the Group Post will NOT be published.

We propose more features that could make the user experience better:

1. Notification System for better feedback to the user

2. Integrating an Email API so that sending/receiving email is reflected on the email address's scale instead of local scale.

3. Overhauling the UI to make it look more modern.

4. Implementing a proper Reporting System for Chat, Email, Posts and Comments so that Administrator Members may review the reports from Senior/Junior Members.