Requirements List

Requirements in this color are meant for the full application, and will not be considered for the prototype

1. **Functionality**
    a. **User Authentication**
        i. **Users can authenticate with Login [MUST]**
        User must have a login username and password, with which they can access the platform, through a login form.
        Validation: User Login Use Case
        Verification: Log into the system.
        ii. **Users can register an account**
            1. **Users can register an account [MUST]**
            User can fill in a form, containing username, email and password as fields, in order to become a registered user.
            Validation:Use Registration Use Case
            Verification: Register an account into the system.
            2. **No accounts can have a duplicate username/email [MUST]**
            If a user enters into the registration form an existent username/email, they should not be able to register their account.
            Validation:Use Registration Use Case
            Verification:Request to create a new account with existent username/email is rejected.
            3. **User passwords must meet requirements [COULD]**
            User can not register, unless their password is at least 8 characters long.
            Validation: Use Registration Use Case
            Verification: Request to create an account with a password less than 8 characters long, should be rejected.
    b. **Create and Edit Posts/Topics by Authenticated Users**
        i. **Authenticated Users "New Post" button**
        Authenticated users see a "New Post" button in the main top navigation bar, which takes them to the post editor.
        Validation: This is shown by the *Create Post Use Case*
        Verification: As an authenticated user, you can see a "New Post" button in the main top navigation, which when clicked, will take you to the Post Editor.
        ii. **Posts Editor Page (Creation/Editing)**
            1. **Text Editor [MUST]**
            The post editor page has a text editor, including formatting tools (font size, font color, bold, italics, underline) and blocks (links,

code snippets).
Validation: This is referenced in the *Posts Editor Use Case*
Verification: As an authenticated user, on the post editor page, you can see a text editor with the aforementioned formatting and blocking tools.

2. **Discussion Group Selection [MUST]**
   The post editor page should allow the author to select a discussion group to target the post at
   Validation: This is referenced in the *Posts Editor Use Case*
   Verification: As an authenticated user, on the post editor page, you can select a discussion group
3. **Submission functionality [MUST]**
   The post editor page should allow the user to submit their post form, in order to publish their post on the platform
   Validation: This is mentioned in the *Posts Editor Use Case*
   Verification: As an authenticated user, I can publish my post from the post editor page
   a. **Empty posts can not be submitted [MUST]**
      If a user attempts to publish an empty post to the platform through the post editor page, their request should be rejected
      Validation: This is seen from the *Posts Editor Use Case*
      Verification: As an authenticated user on the post editor page, if I try to publish an empty post, my request does not go through.
   b. **User is taken to the published post upon successful submission [MUST]**
      Upon successful submission of the post, the user should be taken to the published post on the platform
      Validation: This is included in the *Posts Editor Use Case*
      Verification: After submitting a post (that meets submission requirements) I am taken to the published post
   c.

iii. **Post Viewing**

1. **Author of post sees an edit button [MUST]**
   On the post viewing page, the author of the post can access the functionality to edit the post, which will take the author to the Post Editor page
   Validation: Refer to the *Post Viewing Use Case*
   Verification: Open up a post that you published, and you should be able to edit the post
2. **Rating [COULD]**
   Non-authors can give the post an integer rating out of 5 points
   Validation: Refer to the *Post Viewing Use Case*

Verification: View a post that you did not write up yourself, and you should be able to give the post an integer rating out of 5 points

c. User Subscription to topics and subsequent new notifications

1. Users can subscribe to threads and posts [MUST]

Users are able to subscribe to a thread or post by clicking the "subscribe" button.
Validation: This is seen from the User Subscription Use Case
Verification: As an authenticated user, you can subscribe to threads and posts.

2. User can unsubscribe [MUST]

Users can unsubscribe to a thread or post should they decide that they are not interested or decide that it's not of use anymore.
Validation: This is seen from the User Subscription Use case.
Verification: As an authenticated user, you can unsubscribe a thread or a post.

3. Users are notified if posts are made on subscribed threads [MUST]

Users already subscribed to a subscription thread is automatically notified of a new post
Validation: This is shown from the User Subscription Use Case.
Verification: As an authenticated user, you will receive a notification inform you that the post is successfully posted on the subscription thread.

4. Users can't subscribe to private threads [SHOULD]

Users are not allowed to be subscribed to private threads since are meant to be private.
Validation: This is shown from the User Subscription Use case.
Verification: If you have created your own thread or post, you are not allow to unsubscribe your own thread or post.

4.1. Group Members can subscribe

Users who have access have the right to subscribe
Validation: This is seen from the User Subscription Use Case
Verification: As an authenticated user and as a group member of the team, you can subscribe the thread or the post.

4.2. Group Members can unsubscribe

Users who are already subscribed to the private threads can always unsubscribe to them as well.
Validation: This is seen from the User Subscription Use Case.
Verification: An an authenticated user and as a group member, you can unsubscribe the the thread or the post.

d. Creation of Discussion groups and optional restrictions

1. **Create groups based on interest/tags [COULD]**
   User is matched with another user based on tags used and can then decide to form a group with those users by adding them.
   Validation: This is shown from the Discussion groups Use Case.
   Verification: As an authenticated user, you can create groups based on interest/tags.
2. **Drop-down menu of all users [WONT]**
   User gets a drop-down menu with all users registered to the website
   Validation: This is shown from the Discussion groups Use Case.
   Verification: User attempts to add another user via the drop-down menu
3. **Search to add users [SHOULD]**
   User can use the search bar to find a particular user and add them into a group
   Validation: This is shown from the Discussion groups Use Case.
   Verification: User attempts to add another user to a group using the search function
4. **Users can leave the group [COULD]**
   Should a user decide to leave a group for a specific reason can do so by clicking the “Leave Group” button.
   Validation: This is shown from the Discussion groups Use Case.
   Verification: User attempts to leave a group and is no longer a group member
5. **Users can request access to groups [SHOULD]**
   Users who wish to be part of a group can request access to the group admin of the post
   Validation: This is shown from the Discussion groups Use Case.
   Verification: User attempts to request access to a group
6. **Group admin can add users after creation [SHOULD]**
   Group admins of a post are able to add users even after the creation of a discussion group
   Validation: This is shown from the Discussion groups Use Case.
   Verification: group admin attempts to add another user.
7. **Suggested users to add [COULD]**
   Group admins are given a suggested user drop-down menu of those who should be added to the group.
   Validation: This is shown from the Discussion groups Use Case.
   Verification: User attempts to create a thread and drop-down menu is shown.
8. **Group members are subscribed by default [COULD]**
   Users who are in a group are subscribed to a thread by default
   Validation: Discussion Use Case

Verification: User checks subscription settings and sees the group as subscribed.

9. **Group Admin can make members Group Admins[COULD]**

   Group Admin can decide on whether a user can be a group admin, gaining right such as adding or removing members
   Validation: Discussion Use Case
   Verification: User attempts to add another user to the group after being added and is successful in doing so.

10. **User can create group[MUST]**

    User can create group.
    Validation: Discussion Use Case
    Verification: As an authenticated user, user can create a group.

e. Creating a User Profile and Direct Messaging

1. **Clicking a user's profile picture/username shows profile [MUST]**

   A User can click on a targeted user's picture/user name through one of their post and be redirected to the targeted user's profile.
   Validation: User Profiles Use Case.
   Verification: Click on the target user's name or profile picture, available on one of the target user's posts and you should see their profile.

2. **User send messages to another user .[SHOULD]**

   Previewing user can fills the textbox with their desired Message.Previewing user can click on a direct message button to send a message to the target user.
   Validation: Direct Messaging Use case.
   Verification: Send a direct message and check if the target user can view it.

3. **The target user see the message if he is online[SHOULD]**

   Recipient user can see a message if he is online.
   Validation: Direct Messaging Use Case.
   Verification: User check if he has received a message.

4. **The target user receive a notification if he is offline.[SHOULD]**

   Recipient user can receive a notification if he is not online.
   Validation: Direct Messaging Use Case
   Verification: User check the notification if he has received a notification.

2. **Usability**
    a. **User Authentication**
        i. **Users see a notification when logging in/registering an account, and their attempt is not successful [MUST]**

        Users see a prompt in the login/registration form window, letting them know what issues resulted in the attempt not being successful

        Validation: User Registration Use Case, and User Login Use Case

        Verification: All failure extensions in either the User Registration Use Case or the User Login Use Case, should result in a prompt
    b. **Create and Edit Posts/Topics by Authenticated Users**
        i. **Post Editor Page (Creation/Editing)**
            1. **Discussion group dropdown menu [MUST]**

            On the post editor page, I can see a discussion group dropdown menu, under the text editor, defaulting to “None”.

            Validation: Refer to the *Post Editor Use Case*

            Verification: As an authenticated user, if you access the post editor page (either by editing an existing post you wrote, or creating a new one) you can see the discussion group dropdown menu under the text editor.
        ii. **Post Viewing**
            1. **Author of post sees an edit button [MUST]**

            Author can see an edit button at the top right corner of their post when they are viewing the post

            Validation: Refer to the *Post Viewing Use Case*

            Verification: View a post that you published, and you should see an edit button in the top right corner of the post
    c. **Creating a User Profile and Direct Messaging**
            1. **The User profile shows user details [MUST]**

            Previewing user can see the profile picture, username, email and registration date of the target user

            Validation: User Profiles Use case.

            Verification: Open up a user profile and ensure that the aforementioned details are present.

3. **Reliability**
4. **Performance**
5. **Supportability**
6. **Design constraints**
    a. **The Flask framework is used.**

        Use of the flask web development framework imposes design restrictions regarding how web services are structured and how mobile devices are supported. Much of the project architecture is pre-determined by the flask framework.

        Validation: Imposed by the COMP2005 context of the project

Verification: Examine the implementation code.

b. Python will be used as an implementation language.

Use of the python language imposes design restrictions regarding how the whole project are implemented. The implementation includes algorithm design, database design, user interface design.

Validation: Imposed by the Group-L context of the project.

Verification: Examine the implementation code.

7. **Implementation requirements**
8. **Interface requirements**
9. **Physical requirements**

Title: User Profiles
Primary Actor: Previewing User
Stakeholders and Interests:

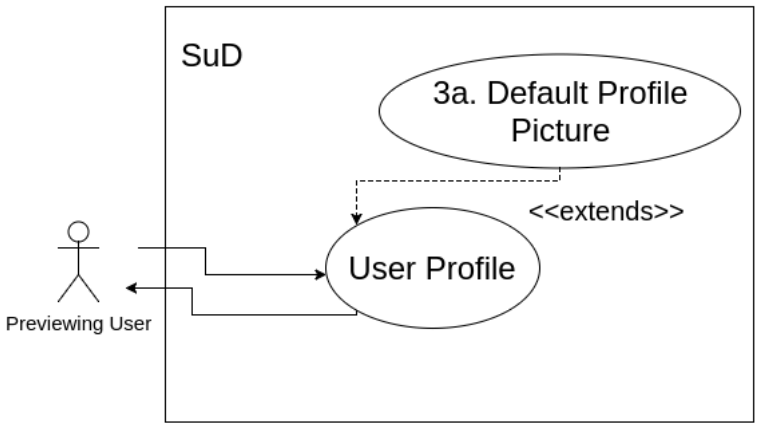
Previewing User - wants to view the profile of another user

Precondition: Previewing user is logged into the platform
Trigger: Previewing user clicks on the target user's name or profile picture, available on one of the target user's posts
Main success scenario:

1. Previewing user requests to see the profile of the target user
2. Target user id is passed to the backend, and the user profile view is retrieved
3. The user profile shows the profile picture, username, email and registration date of the target user
4. Previewing user can click on a direct messaging button to chat with the user
5. Upon such a click, launch the direct messaging use case

Extensions:

3a. Target user does not have a profile picture
3a1. Display the default user profile picture instead



Title: Direct Messaging
Primary Actor: Typing User
Stakeholders and Interests:

Typing User - wants to send a direct message to the recipient user

Recipient User - wants to receive any message sent to them from other users

Precondition: Typing User is logged into the platform.
Trigger:

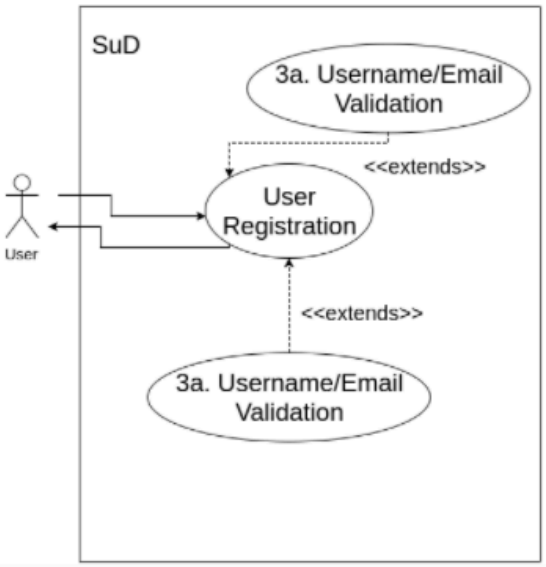
- User clicks the direct messaging button on the recipient's user profile, OR
- User clicks on inbound direct messaging notification

Main success scenario:

1. Typing user sees a window in the recipient user's profile showing the chat history, with a textbox under it
2. Typing user fills the textbox with their desired message
3. Typing user presses the `Enter` key or clicks the "Send" button
4. The direct message is sent to the backend, so that it can be dispatched through a web socket to the recipient
5. Recipient user sees the message if he has the chat open, or receives a notification otherwise

Extensions:

3a. Could not connect to backend or push direct message to websocket
3a1. Show a client-side notice to the typing user, indicating the message was not delivered successfully



Title: Login Authentication
Primary Actor: User
Stakeholders and Interests:

User - wants to access the platform via a login form.

Precondition: User should not be already logged in.
Trigger: User click "login" button on login form.
Main success scenario:

1. User must have a distinct username and password.
2. User click the "login" button.
3. The users will be able to login with acceptable login information.
4. User is redirected to the Post viewing page or thier User Profile if login is successful.

Extensions:

3a. User enters an unregistered unique/distinct username and password:
    3a1. SuD does not recognize users information.
    3a2. SuD rejects users information.
    3a3. By means of a notice displayed on the login form, the user is notified about unsuccessful login.
    3a4. User clicks the "not registered" button to register new account.
    3a5. "Not Registered" button initiates *Registration* use case.

Title: User Registration
Primary Actor: User
Stakeholders and Interests:

User - wants to register an account on the platform

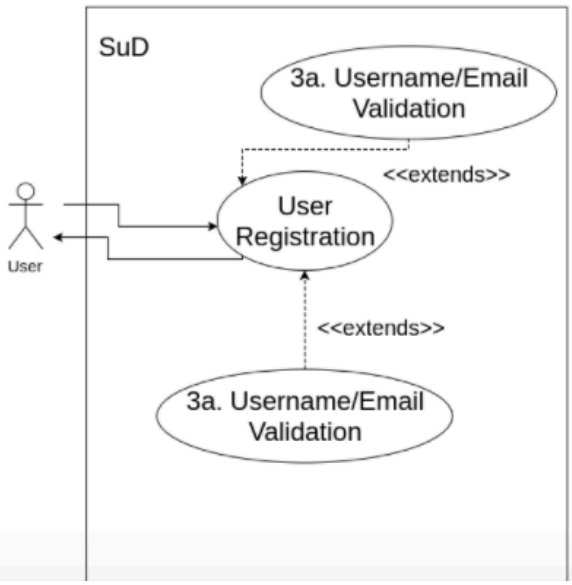
Precondition: User is not logged in.
Trigger: User clicks "Not registered" on the login form.
Main success scenario:

1. User fills in the registration form with username, email and password
2. User clicks the registration button
3. Form data is passed to the backend, and a user account is generated
4. User is redirected to his profile page

Extensions:

3a. User entered an existing username/account into the form
3a1. Do not generate a new user account
3a2. Return the registration page view with a note about the username/email being already taken.

3b. User entered an insecure password (< 8 characters long)
3b1. Refuse to generate new user account.
3b2. Return the registration page view with a note about the required minimum length of password



Title: Create Post
Primary Actor: User
Stakeholders and Interests:

User - wants to create a new post/topic on the platform.

Precondition: User must be authenticated and logged in.
Trigger: User clicks the "New Post" button from the main top navigation bar.
Main success scenario:

1. Verified users click the "New Post" button in the main top navigation bar.
2. The user is directed to the "Post Editor" when "New Post" button is click.

Extensions:

There are no Extensions for this Use case

Title: Post Viewing
Primary Actor: User
Stakeholders and Interests:

User - wants to see and access a post to edit.

Precondition: User must be logged into the platform.
Trigger: User must click the "edit post" button.
Main success scenario:

1. User must be on the post viewing page to see and access post.
2. Author of the post can only access the functionality to edit post via the Post viewing page.
3. Non-Authors can rate a post on the "Post viewing page" out of 5 points.

Extensions:

2a. The functionality to edit a post is only made available on the post viewing page:

Title: Post Editor
Primary Actor: Author
Stakeholders and Interests:

Author - wants to write up a post / edit a previous post

Precondition: Author is logged into the platform.
Trigger:

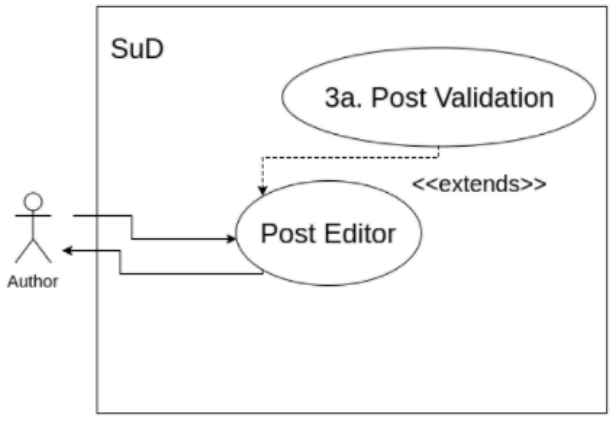
- Author clicks on the new post button from the top navigation bar, OR
- Author clicks on the edit post button on an existent post

Main success scenario:

1. Author fills in the post title
2. Author fills in the post body, with access to a set of formatting tools for fonts and blocks
3. Author selects their preferred discussion group for the post, with the discussion group defaulting to "None"
4. Author publishes/submits the post
5. Data is passed to the backend, where the post is created or modified if it already exists
6. Author is redirected to the published post

Extensions:

4a. Author submits an empty post
4a1. Reject publishing the post
4a2. Return to the post editor page with a note stating that the post can not be empty.



Title: Subscription and notification
Primary Actor: User
Stakeholders and Interests:

User - wants to subscribe or unsubscribe to threads and posts and wants to know whether a post is successfully posted on the subscription thread or not

Precondition: User is logged into the platform.
Trigger:

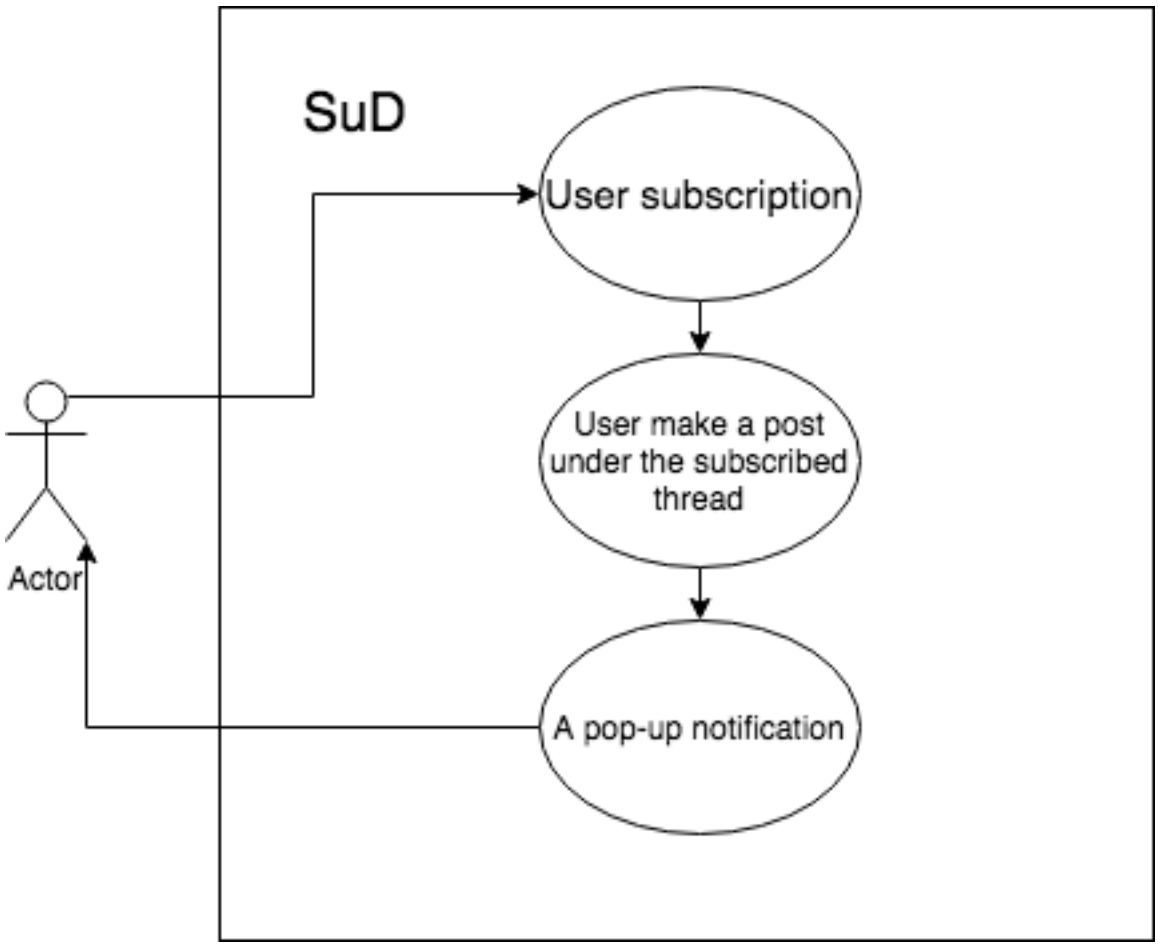
- User clicks on the "subscribe" button under the thread or the post, OR
- User clicks on the "unsubscribe" button under the thread or the post.
- User makes a post on the subscribed thread.

Main success scenario:

1. User clicks the "subscribe" button or the "unsubscribe" button under the thread or the post.
2. Subscription data is passed to the backend, where the user profile is stored.
3. User makes a post on the subscribed thread.
4. A notification appears to show that user has successfully posted a post on the subscribed thread.
5.
6. Author is redirected to the published post

1a. User clicks the "subscribe" button under a thread or a post, which is already subscribed by user.
    1a1. A pop-up notification notifies user that he/she has already subscribed the thread or the post.
    1a2. Refuse to subscribe the thread or the post again.

1b. User clicks the "unsubscribe" button under a thread or a post, which hasn't subscribed by the user yet.
    1b1. A pop-up notification notifies user that he/she has not subscribed the thread or the post.
    1b2. Refuse to unsubscribe the thread or the post.


SuD
User subscription
User make a post under the subscribed thread
A pop-up notification
Actor

Title: Create Discussion Groups
Primary Actor: User
Stakeholders and Interests:

User - wants to create a Discussion Group for a particular topic

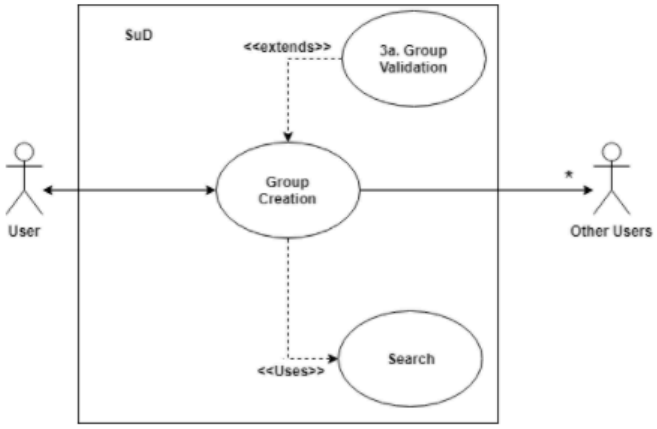
Precondition: User is not in a group
Trigger: User clicks "Create Group" on main page
Main success scenario:

1. User uses search bar to find users by username or tags used
2. User clicks those who they wish to add
3. Group is formed and invitations are sent to the invited users
4. User becomes the Group Admin

Extensions:

1a. User adds no one
    1a1. Group is formed with the User as the only member
    1a2. User becomes the Group Admin

3a. User has the same Discussion Group name as another group
    3a1. Refuse to generate new Discussion Group
    3a2. Notify User that they cannot have the same group name as another group



Title: Request Access to Group
Primary Actor: User
Stakeholders and Interests:

User - Requests access to group via group admin

Group Admin - Decides to grant access to invitation made by user

Precondition: User is signed in
Trigger: User clicks on "Join Group" for a particular group they wish to join
Main success scenario:

1. User searches for a group relating to a particular topic that the group focuses on
2. User requests access to group via "Join Group" button
3. User invitation data is passed to the Group Admin
4. Group admin clicks on "Add User"
5. User is notified that they have been added to the group

Extensions:

3a. User is already a member of the group
    3a1. Notify user that they are already part of the group

4a. User is rejected by Group Admin
    4a1. Group Admin clicks on "Deny User" and gives optional reason
    5a1. Notify user that they have been denied access by Group Admin, with optional reason