BOGAZICI UNIVERSITY
Swe573 Software Development Practice, Spring 2024
Project Final Deliverable

ONUR ALTINKURT
2022719198
19.05.2024

Project Name : Comflex
Deployment URL: http://54.228.20.218:8001/home
Git URL: https://github.com/ctrlaltkurt/SWE573-onuraltinkurt
Git tag version URL: https://github.com/ctrlaltkurt/SWE573-onuraltinkurt/tags

HONOR CODE

Related to the submission of all the project deliverables for the Swe573 2024 Spring semester project reported in this report, I, ONUR ALTINKURT, declare that:
- I am a student in the Software Engineering MS program at Bogazici University and am registered for Swe573 course during the Spring 2024 semester.
- All the material that I am submitting related to my project (including but not limited to the project repository, the final project report, and supplementary documents) have been exclusively prepared by myself.
- I have prepared this material individually without the assistance of anyone else with the exception of permitted peer assistance which I have explicitly disclosed in this report.

ONUR ALTINKURT

[illegible]

Table of Contents

1. Introduction

This project contains a deployment url and 5 github links. Github links are for git repository, git tag version and git project board.

Final Deliverables: https://github.com/ctrlaltkurt/SWE573-onuraltinkurt/wiki/Final-Deliverables
Class Repository: https://github.com/ctrlaltkurt/SWE573-onuraltinkurt
Git Tag version : https://github.com/ctrlaltkurt/SWE573-onuraltinkurt/tags
Comflex Git Project Board : https://github.com/users/ctrlaltkurt/projects/1
Deployment URL : http://54.228.20.218:8001/home
User Manual : https://github.com/ctrlaltkurt/SWE573-onuraltinkurt/wiki/User-Manual

User info for the website. (new user can also be created)
Username: admin
Password: 12345

2. Overview and Scope

This project's aim is to develop a community specific information management web application that allows users to create and manage personalized communities and posts with roles such as community owners and moderators and regular users.

3. Software Requirements Specification

3.1. Functional Requirements

3.1.1. Account and Login Requirements

1. Users should be able to create accounts and register to the system.
2. Users should be able to log in to the system securely.
4. Users should be able to log out from the system.
5. Users should be able to change login credentials.

3.1.2. Community Interaction and Membership

6. The system should allow users to build communities.
7. Users should be able to join a community and become a member of multiple communities.
8. Users should be able to join private communities if any of the moderators allows join request.
9. Users should be able to create communities.
10. Communities should be able to be created as public or private.
11. Communities should have at least one community owner.
12. Communities should have unique names.
13. Communities should have a default post type when they are created.
14. Communities should be moderated by the moderators.
15. Communities should be able to have more than one moderator.

16. Communities should have an info that describes what is that community about that can be seen by anyone.
17. Communities should show the list of members to anyone.

3.1.3. Roles and Permissions

18. The system should include user roles as user (member), moderator, and community owner.
19. Users should be able to have user, moderator, and community owner roles at the same time.
20. Moderators should be able to allow or reject join requests to their private communities.
21. Moderators should be able to archive user's posts.
22. Moderators should be able to resign from the moderator role.
23. Moderators should be able to dismiss a member (user) from their communities.
24. Moderators should be able to create new post types for their communities.
25. Community owners should become the moderator in the community when he/she created it.
26. Community owners should be able to delegate a moderator role to community members (users).
27. Community owners should be able to resign from ownership.
28. Community owners should be able to delegate the community owner role to members (users) or moderators.
29. Community owners should be able to dismiss members (user) and moderators from the communities.

3.1.4. Content Creation and Management

30. Users should be able to report moderators to community owners.
31. Users should be able to report a post to moderators.
32. Users should be able to post as one of the community-specific post types.
33. Users who are members of a specific community should be able to propose a post type to a moderator of that community.
34. Users should be able to propose a post type to the moderator by creating a post type on his/her own and sending it to the moderator.
35. Users should be able to answer to any post in the community that they are a member of.
36. Users should be able to edit their posts and comments but an indication showing that the post is edited and the edit time should exist.
37. Users should be able to upvote or downvote a post.
38. All posts should have posting time and poster name information.

3.1.5. Search Functionality

39. Users should be able to search in the communities that they are a member of.
40. Users should be able to search information in the community in two ways: basic search and advanced search.
41. The basic search tool should be able to find information that is shared in the community as text.
42. The advanced search tool should be able to search information in different formats than text.

3.1.6. Homepage and Discovery

43. The homepage should be able to show posts to the users from the communities that they are a member of.

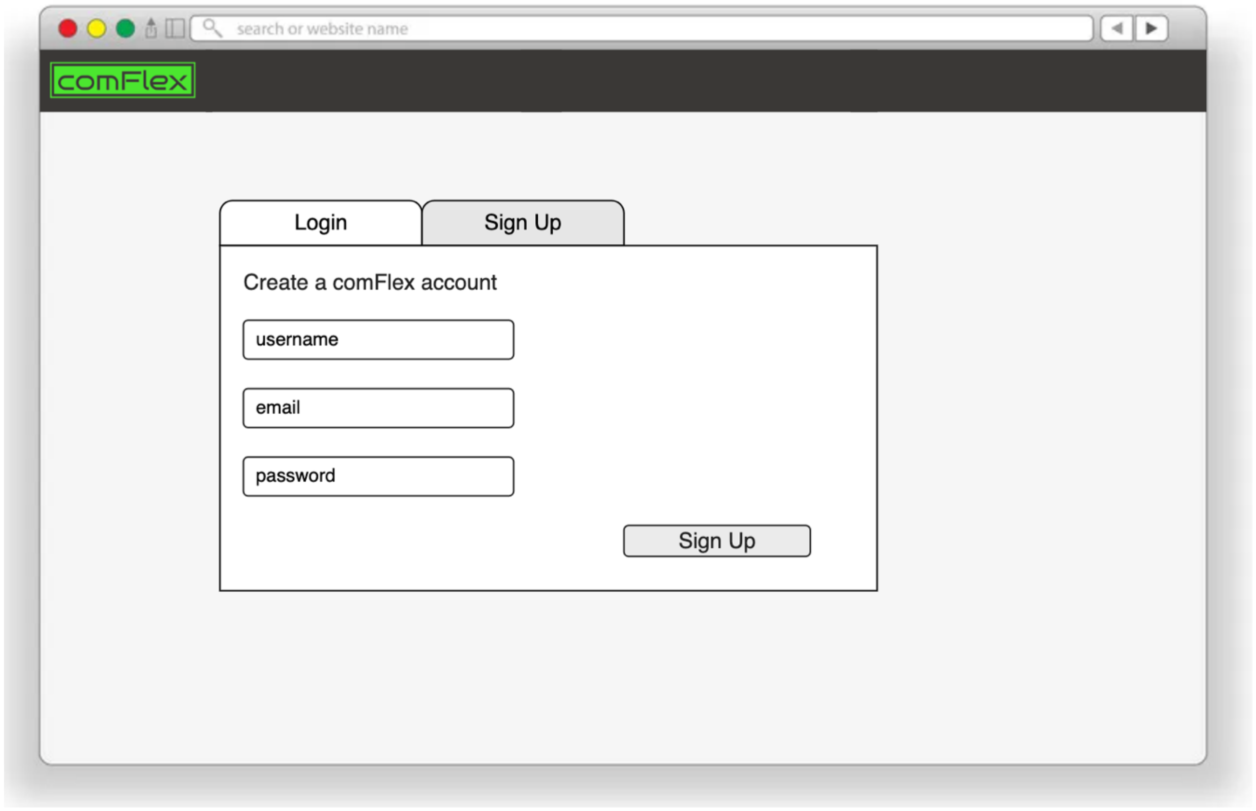
44. The homepage should be able to list popular communities that have the most members, the most posted recently, and the newest communities.
45. The homepage should be able to show the posts from the communities that the user interacted with the most in recent times.
46. The homepage should have an option (like "I am feeling lucky") to allow users to discover new communities by showing the popular posts from those communities.
47. The homepage should have an option to allow users to filter specific communities to be shown in the feed.

4. User Scenarios

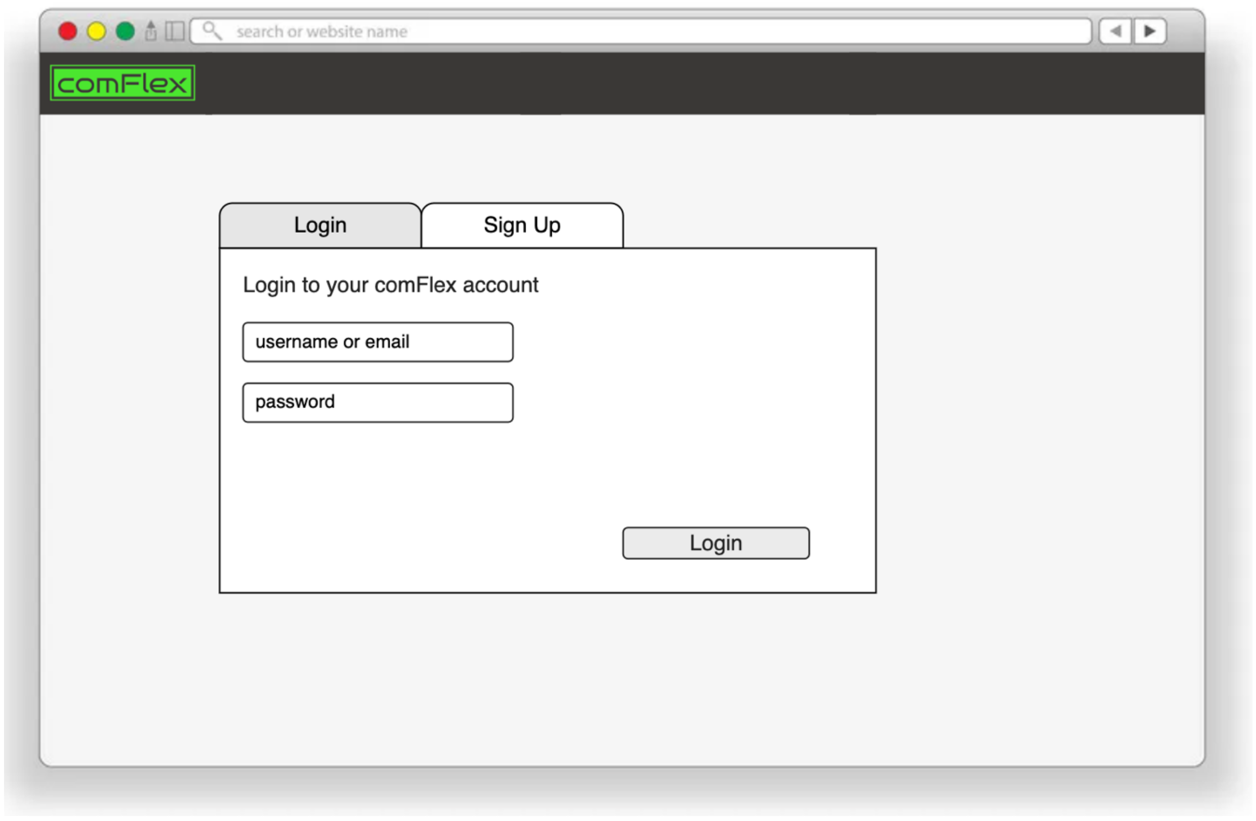
4.1. User Scenario 1 – New User

Ali Patioğlu
Ali is in need to find some opinions from pet parents about cat foods for his picky cat "Necat". Necat does not like the regular cat food, so Ali needs to find someone who has a similar type of cat and learn what they did to solve this issue. He enters to the platform for the first time and searches the "cat" keyword just to see what is in the website related with cats, then he finds the community "Cat Food" and enters it. He checks the posts and does not see anything related with it. He decides to create a post.

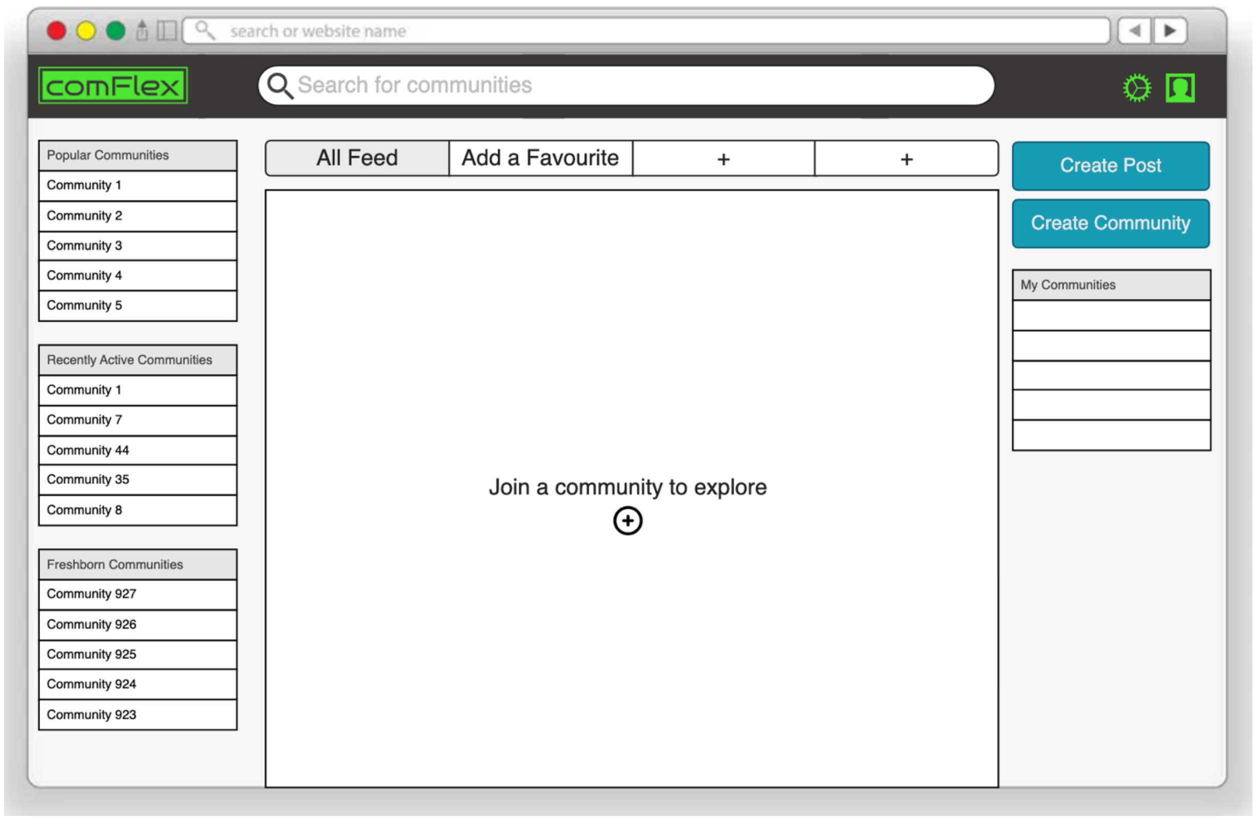
Mockup-1.1 - Sign up screen



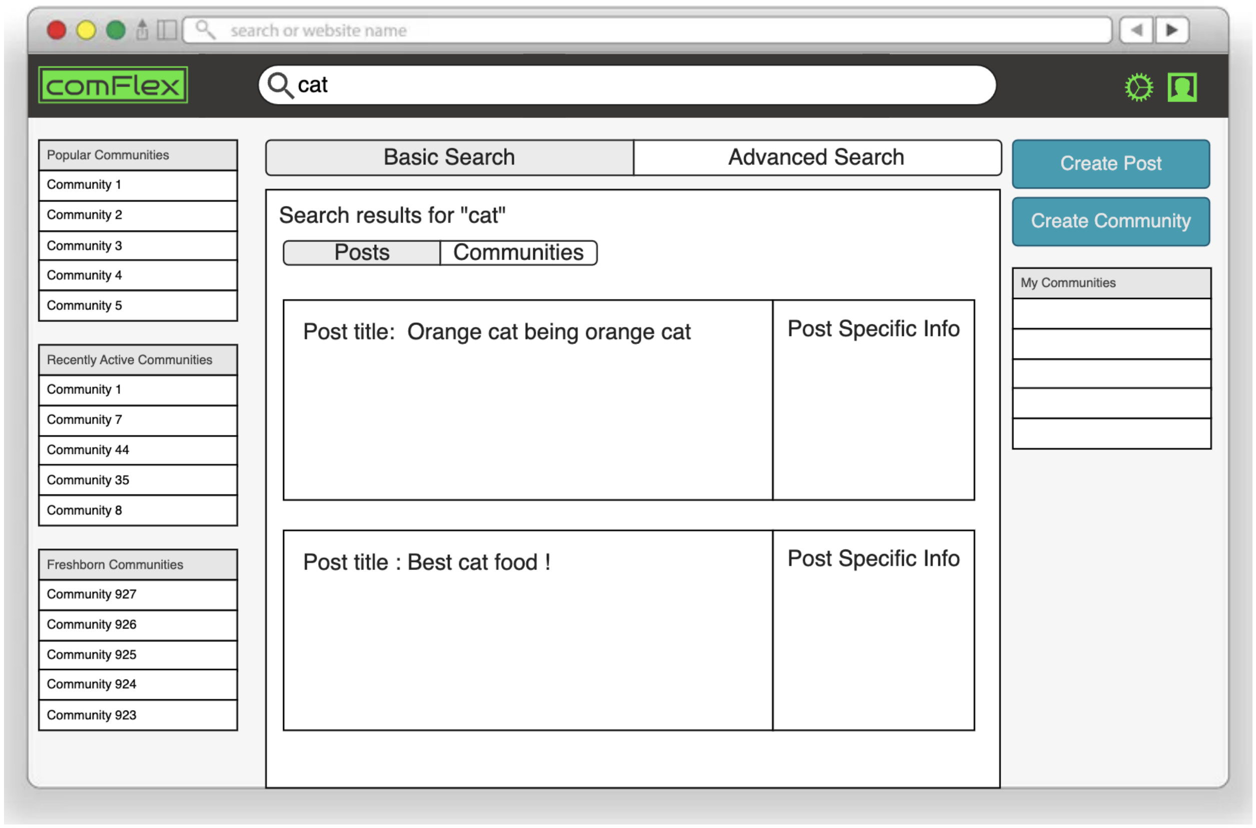
Mockup-1.2-Sign in screen



Mockup-1.3 – Frontpage



Mockup-1.4.1- Searching posts

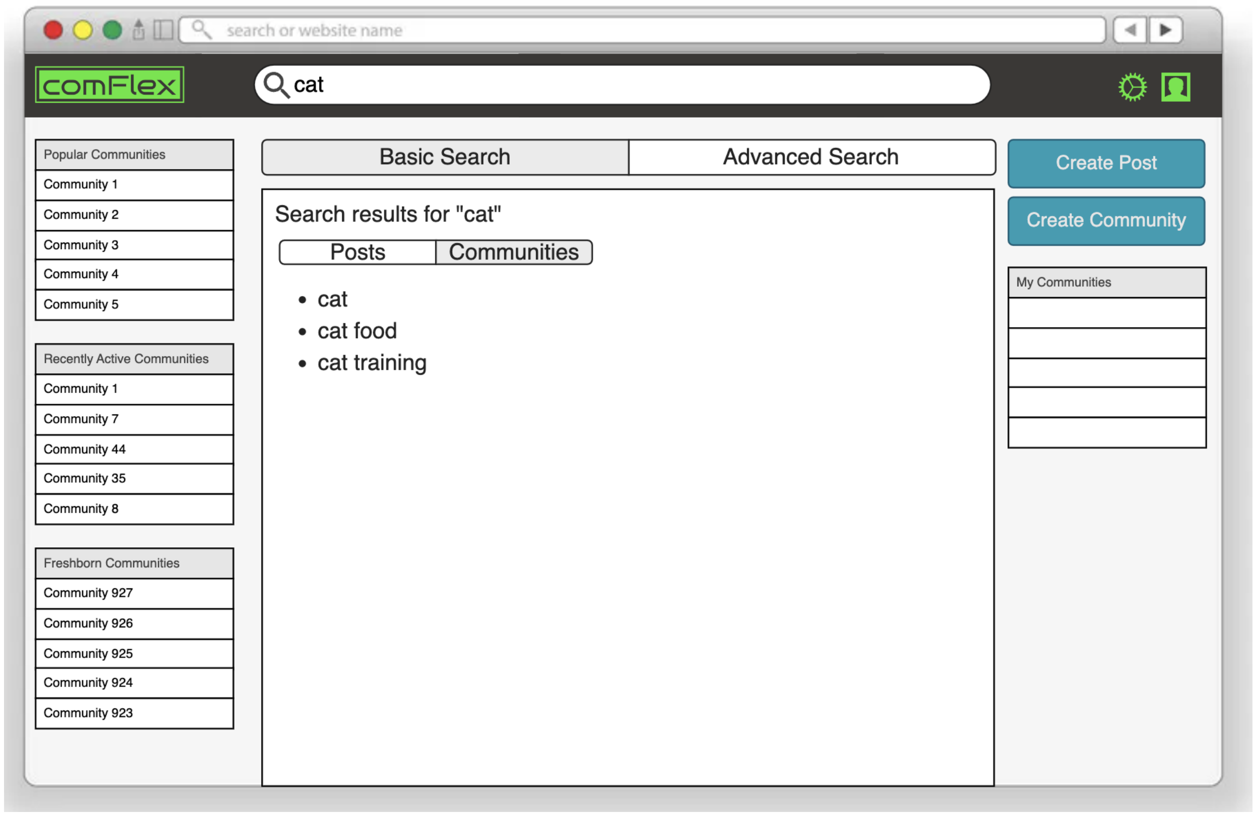


Search results for "cat"

Post title: Orange cat being orange cat

Post title : Best cat food !

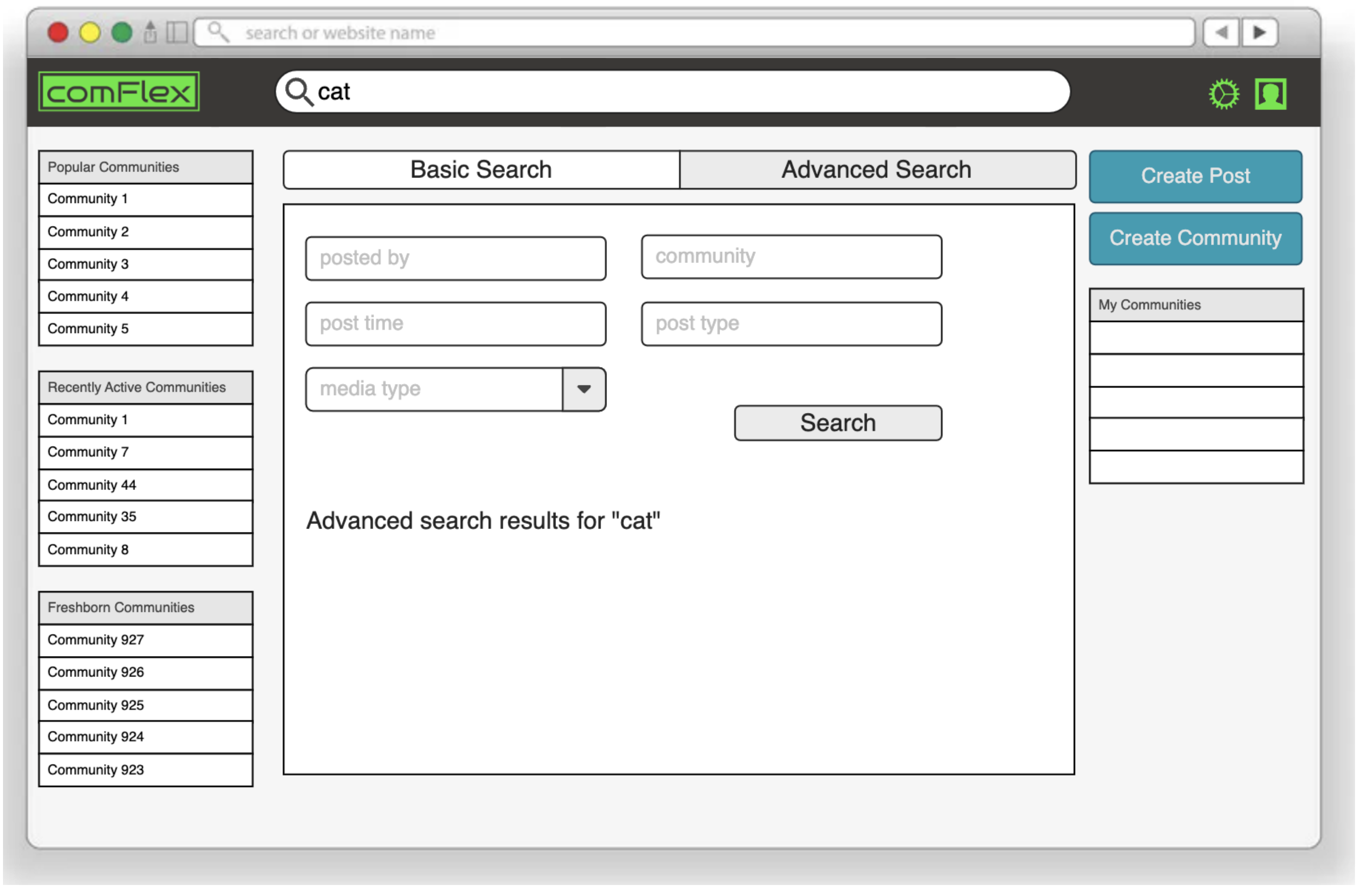
Mockup-1.4.2- Searching communities



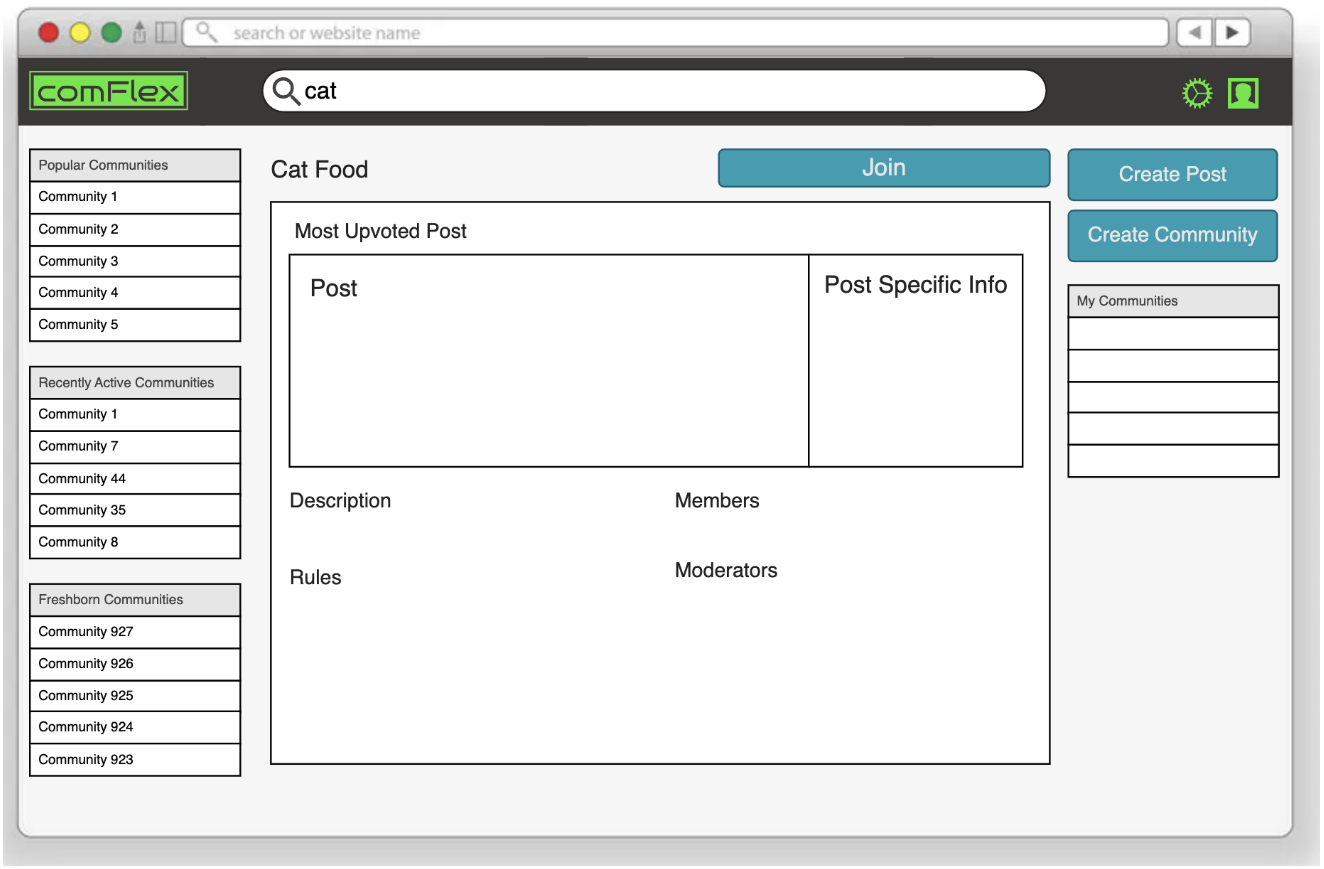
Search results for "cat"

- cat
- cat food
- cat training

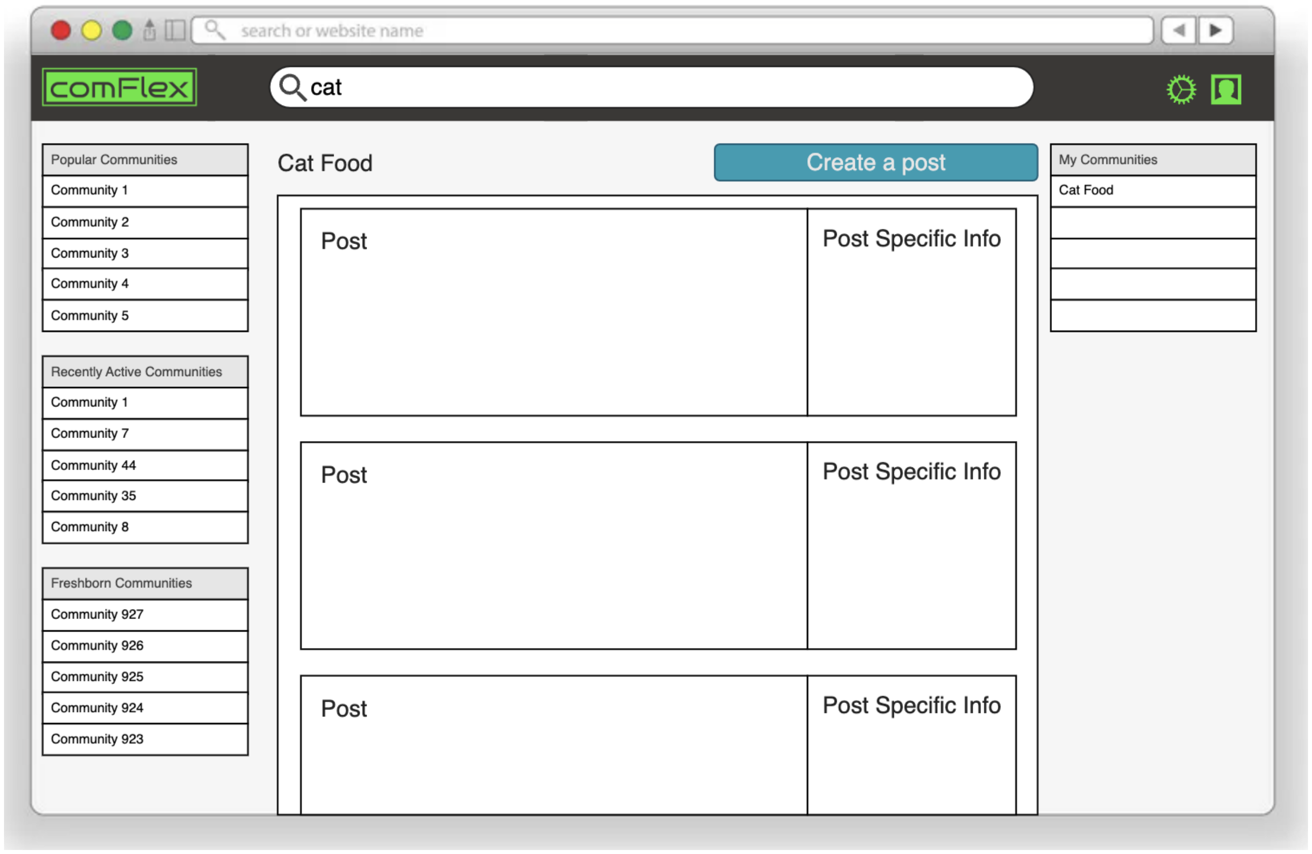
Mockup-1.4.3- Advanced searching



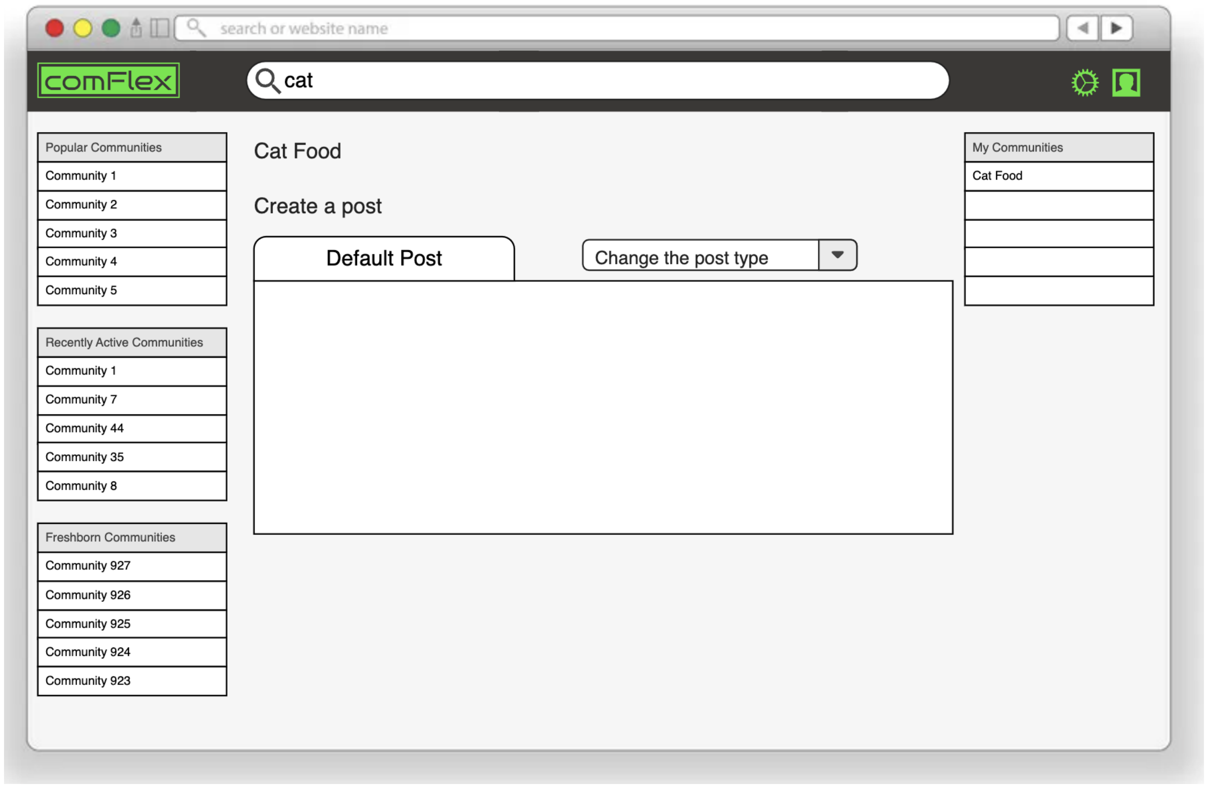
Mockup-1.5- Enters the community page, sees the rules, members.



Mockup-1.6- Joins the community

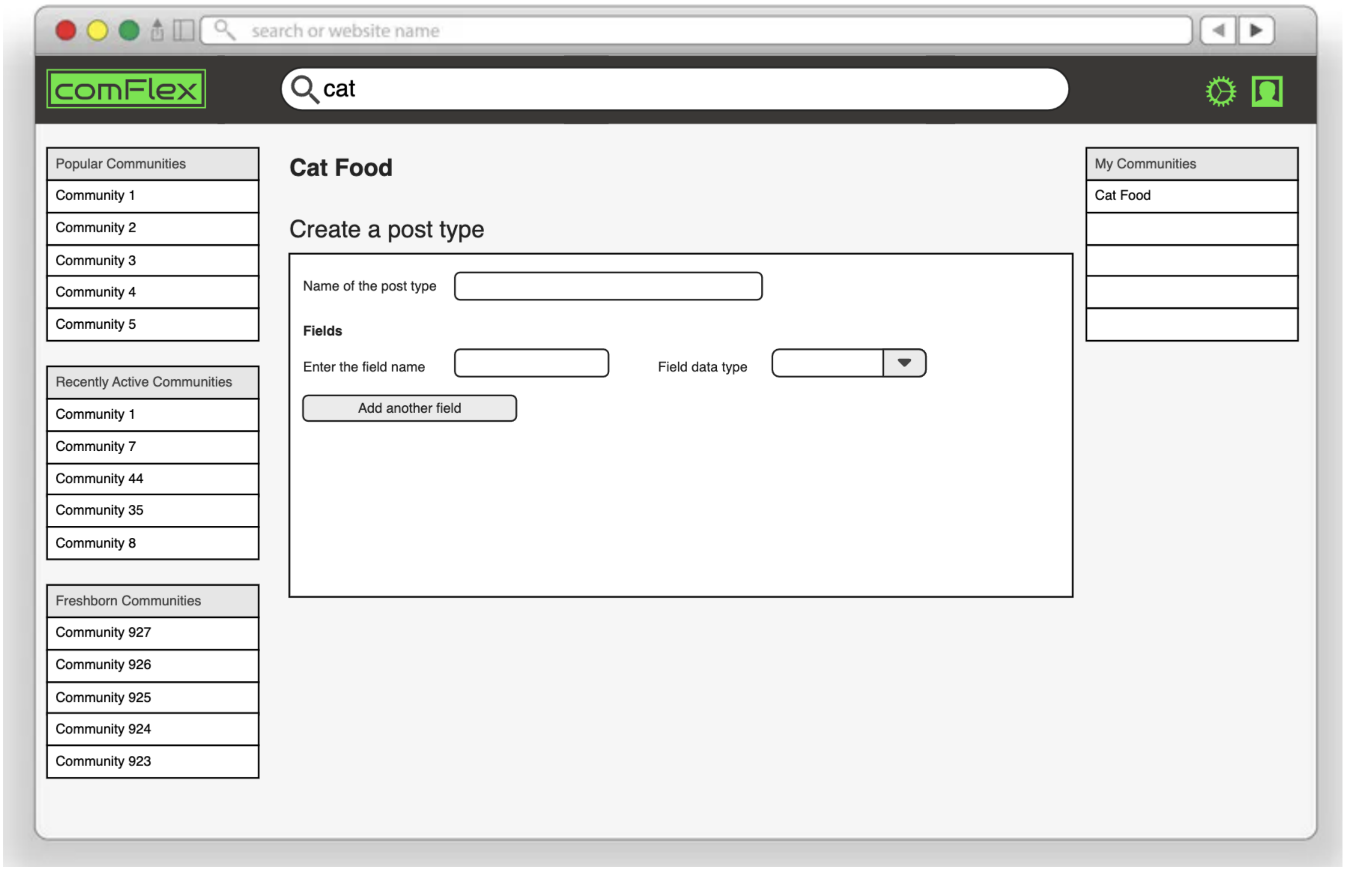


Mockup-1.7- Tries to post, sees default post types



4.2. User Scenario 2 – User becomes moderator

Mockup-2.1- creating new post type



5. Design documents

5.1. Design Decisions

This web application is developed with Django. Django follows an MVC architecture model with slight variations. The Comflex application consists of two main components: user account management and the core community functionality. There are several key models in the project: User, Community, PostType, PostTypeField and Posting forming the backbone of the application. These models are supported by multiple views, including create_community, add_post_type, create_post, modify_post, and various user and community management views. In the context of MVC architecture, views in Django act as controllers, while templates function as the views.

UML Diagrams were used to visualize and develop the overall application. The main design decision of Comflex was to facilitate community-specific posts and member management, with users, communities and posts forming the core connections. Community owners can create custom post types and assign moderators to help manage the community. Moderators can create post types and manage members, ensuring community guidelines are upheld.

The application also supports advanced search functionality, allowing users to search for communities and posts based on various criteria. Users can interact with posts through likes and dislikes, enhancing engagement within the community. To ensure a seamless user experience, the application has been designed to be easily extendable and maintainable.

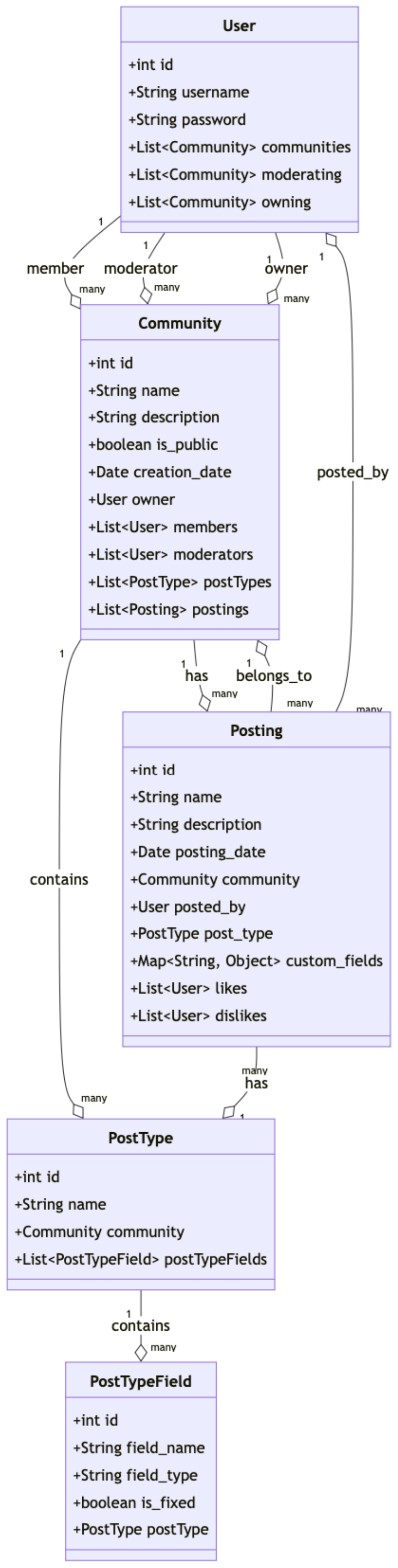
Comflex aims to provide a robust platform for community building and interaction, leveraging the power of Django's framework to create a dynamic and user-friendly environment. For development and testing purposes, the application can be easily run locally using Docker, ensuring consistency across different environments and simplifying deployment processes.

5.2. Some Specific Functionalities

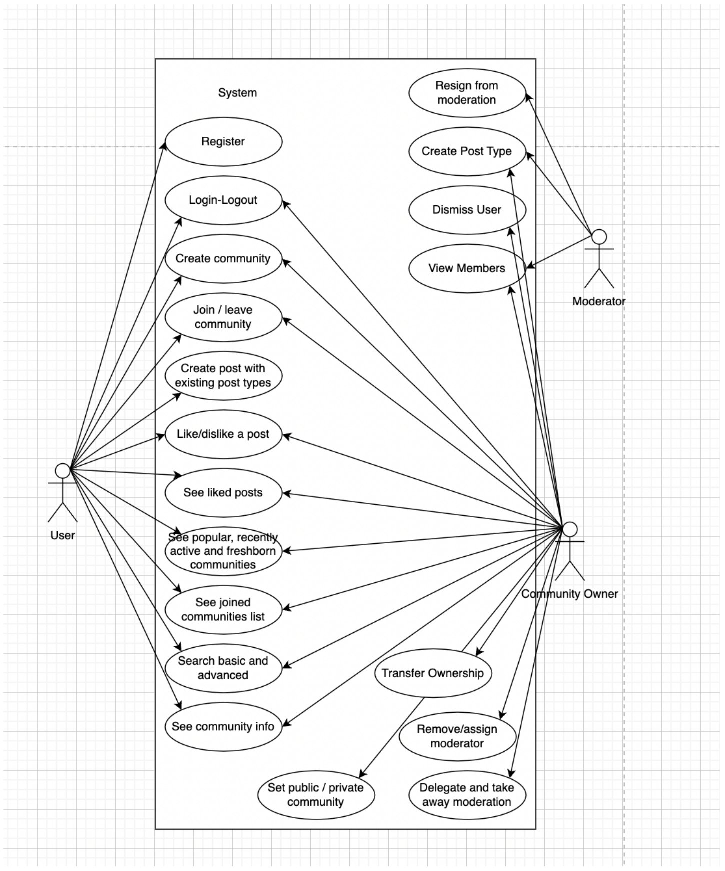
- In terms of user roles, users can create communities and become the community owners, assign and remove moderators, create post types, dismiss users, view community members, transfer ownership, and leave communities.
- Non-logged-in users can see posts with more than one like on the homepage, but they cannot interact with posts until they log in.
- The Discover page lists up to ten popular communities based on the number of posts, displays ten of the newest communities by their creation date, shows up to ten recently active communities sorted by the date of the last post and highlights the top five most crowded communities based on member count.
- Within each community page, posts can be sorted by date or by the number of likes, providing flexibility in how users view and engage with the content. These specifications ensure that Comflex provides a structured and engaging platform for community building and interaction, while maintaining clear distinctions between different user roles and their capabilities.
- Both community owners and moderators have access to a page listing all community members, where they can manage memberships.
- Advanced search functionality allows users to perform searches for communities and posts using various filters such as name, description, creation date, member count, post count, and last post date.

5.3. UML Diagrams

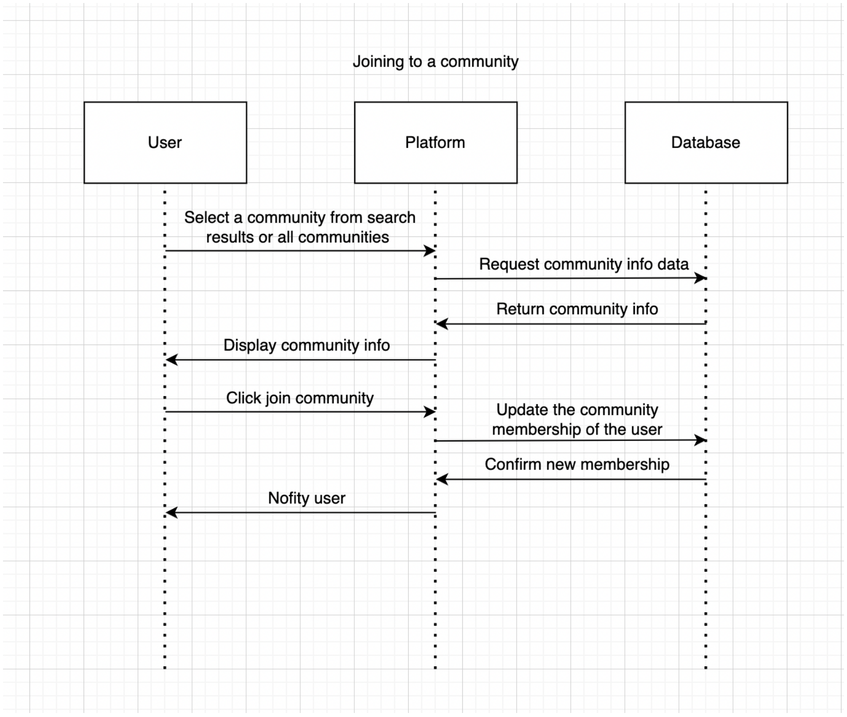
5.3.1. Class Diagram



5.3.2. Use Case Diagram



5.3.3. Sequence Diagram



6. Status of your project: the status of each requirement

6.1. Completed Requirements

Account and Login Requirements

1.Users should be able to create accounts and register to the system
2.Users should be able to log in to the system securely
3.Users should be able to log out from the system

Community Interaction and Membership

5.The system should allow users to build communities
6.Users should be able to join a community and become a member of multiple communities
8.Users should be able to create communities
9.Communities should be able to be created as public or private
10.Communities should have at least one moderator
11.Communities should have unique names
12.Communities should have a default post type when they are created
13.Communities should be moderated by the moderators
14.Communities should be able to have more than one moderator
15.Communities should have an info that describes what is that community about that can be seen by anyone
16.Communities should show the list of members to anyone: (CHANGED : Community member list is shown only to the community owner and moderators)

Roles and Permissions

17.The system should include user roles as user (member), moderator, and community owner
18.Users should be able to have user, moderator, and community owner roles at the same time
21.Moderators should be able to resign from the moderator role
22.Moderators should be able to dismiss a member (user) from their communities
23.Moderators should be able to create new post types for their communities
24.Community owners should become the moderator in the community when he/she created it(CHANGED)
25.Community owners should be able to delegate a moderator role to community members (users)
26.Community owners should be able to resign from ownership
27.Community owners should be able to delegate the community owner role to members (users) or moderators
28.Community owners should be able to dismiss members (user) and moderators from the communities

Content Creation and Management

31.Users should be able to post as one of the community-specific post types
35.Users should be able to edit their posts and comments but an indication showing that the post is edited and the edit time should exist(PARTIALLY COMPLETED: Users can edit their posts)
36.Users should be able to upvote or downvote a post
37.All posts should have posting time and poster name information

Search Functionality

38.Users should be able to search in the communities that they are a member of(CHANGED)
39.Users should be able to search information in the community in two ways: basic search and advanced search
40.The basic search tool should be able to find information that is shared in the community as text (CHANGED)
41.The advanced search tool should be able to search information in different formats than text (CHANGED)

Homepage and Discovery

42.The homepage should be able to show posts to the users from the communities that they are a member of

43.The homepage should be able to list popular communities that have the most members, the most posted recently, and the newest communities: COMPLETED (CHANGED)
44.The homepage should be able to show the posts from the communities that the user interacted with the most in recent times

6.2. Incomplete Requirements

Account and Login Requirements

4.Users should be able to change login credentials

Community Interaction and Membership

7.Users should be able to join private communities if any of the moderators allows join request

Roles and Permissions

19.Moderators should be able to allow or reject join requests to their private communities
20.Moderators should be able to archive user's posts

Content Creation and Management

29.Users should be able to report moderators to community owners
30.Users should be able to report a post to moderators
32.Users who are members of a specific community should be able to propose a post type to a moderator of that community
33.Users should be able to propose a post type to the moderator by creating a post type on his/her own and sending it to the moderator
34.Users should be able to answer to any post in the community that they are a member of

Homepage and Discovery Status

45.The homepage should have an option (like "I am feeling lucky") to allow users to discover new communities by showing the popular posts from those communities
46.The homepage should have an option to allow users to filter specific communities to be shown in the feed

7. Status of Deployment

Website is deployed and properly working on the EC2 instance of AWS.

8. Installation instructions

- Istall the source code from https://github.com/ctrlaltkurt/SWE573-onuraltinkurt/releases/tag/v0.9
- Open the zip and go to the comflex folder "cd downloadeddirectory/comflex"
- Run the following command :"source env/bin/activate"
- Run the following command: "pip install django"
- Run the following command: "cd comflex_website"
- run the following command "python3 manage.py runserver 8001"

9. User manual

For the user manual
https://github.com/ctrlaltkurt/SWE573-onuraltinkurt/wiki/User-Manual

9.1. Introduction

Welcome to Comflex, a dynamic platform designed to help you create and manage communities with ease.

9.2. Getting Started

Registration

- Visit the Comflex homepage.
- Click on the "Register" button in the navigation bar.
- Fill in your username, email, and password.
- Click "Submit" to create your account.

Login

- Click on the "Login" button in the navigation bar.
- Enter your email and password.
- Click "Submit" to log in.

9.3. User Roles

Community Owner

- Create Community
- Assign Moderator
- Remove Moderator
- Create Post Type
- Dismiss Users
- View Community Members
- Transfer Ownership
- Leave Community

Moderator

- Create Post Type
- Dismiss Users (except other moderators and the community owner)
- Resign as Moderator
- View Community Members

Member

- Join and Leave Communities
- Create Posts
- View and Engage with Posts
- Participate in Community Activities

9.4. Creating and Managing Communities

Creating a Community

- Click on the "Create" dropdown in the navigation bar.
- Select "New Community".

- Fill in the community name, description, and other details.
- Click "Submit" to create the community.
- Upon successful creation, you will be redirected to a page confirming the creation.

Managing Communities

- Update Community Info: Click "Modify Community" on the community page.
- Transfer Ownership: Click "Transfer Ownership and Leave" and follow the prompts.
- Dismiss Users: Go to the community members list and click "Dismiss" next to the user’s name.

9.5. Posting and Post Types

Creating a Post Type

- Navigate to your community page.
- Click "Create a Post Type".
- Fill in the post type name and add custom fields.
- Click "Submit" to save the post type.

Creating a Post

- Click "Create Post" in your community.
- Select a post type.
- Fill in the post details.
- Click "Submit" to publish the post.

9.6. Advanced Search

Community Search

- Click "Advanced Search" in the navigation bar.
- Fill in the community search criteria (name, description, creation date, etc.).
- Click "Search" to view results.

Post Search

- Click "Advanced Search" in the navigation bar.
- Switch to the "Posts" tab.
- Fill in the post search criteria (name, description, posting date, etc.).
- Click "Search" to view results.

9.7. Community Moderation and Member Management

Assigning Moderators

- Navigate to your community page.
- Click "Add Moderator".
- Select a member from the dropdown list.
- Click "Submit" to assign the role.

Viewing and Managing Members

- Click "View Members" on the community page.
- View the list of members.
- Dismiss members by clicking the "Dismiss" button next to their name (if you are an owner or moderator).

10. Test results (user)

10.1. User Test Results

10.1.1.Account and Login

1. Create an Account
    - Expected Outcome: User is able to create an account successfully and is redirected to the login page.
    - Actual Outcome: Account creation is successful, and the user is redirected to the login page.
    - Issues/Bugs: None.

2. Login to the System
    - Expected Outcome: User is able to log in successfully and is redirected to the homepage.
    - Actual Outcome: Login is successful, and the user is redirected to the homepage.
    - Issues/Bugs: None.

3. Logout from the System

    - Expected Outcome: User is logged out successfully and redirected to the home page.
    - Actual Outcome: Logout is successful, and the user is redirected to the home page.
    - Issues/Bugs: None.

10.1.2. Community Interaction and Membership

4. Create a Community

    - Expected Outcome: User is able to create a community and is redirected to a confirmation page.
    - Actual Outcome: Community creation is successful and the user is redirected to the confirmation page.
    - Issues/Bugs: None.

5. Join a Community

    - Expected Outcome: User is able to join a public community and the buttons changes to indicate membership.
    - Actual Outcome: Joining a community is successful, and the buttons updates accordingly.
    - Issues/Bugs: None.

6. View Community Info

    - Expected Outcome: Community info should display name, description, number of members, posts count, and creation date.
    - Actual Outcome: Community info displays correctly.
    - Issues/Bugs: None.

7. View Community Members

- Expected Outcome: List of community members should be visible to community owners and moderators.
- Actual Outcome: List of community members is visible to community owners and moderators.
- Issues/Bugs: Change in requirement implementation. No bugs.

10.1.3. Roles and Permissions

8. Assign Moderator Role

- Expected Outcome: Community owner is able to assign a moderator role to a member.
- Actual Outcome: Assignment of moderator role is successful.
- Issues/Bugs: None.

9. Remove Moderator Role

- Expected Outcome: Community owner is able to remove a moderator role from a member.
- Actual Outcome: Removal of moderator role is successful.
- Issues/Bugs: None

10. Resign from Moderator Role

- Expected Outcome: Moderator is able to resign from the role.
- Actual Outcome: Resignation from moderator role is successful.
- Issues/Bugs: None.

11. Dismiss a Member from the Community

- Expected Outcome: Community owner or moderator is able to dismiss a member.
- Actual Outcome: Dismissal of a member is successful.
- Issues/Bugs: None.

10.1.4. Content Creation and Management

12. Create a Post Type

- Expected Outcome: Community owner or moderator is able to create a new post type.
- Actual Outcome: Creation of post type is successful.
- Issues/Bugs: The name of the image field must be “picture” to be able to display images.

13. Create a Post

- Expected Outcome: User is able to create a post using a specific post type.
- Actual Outcome: Post creation is successful.
- Issues/Bugs: None.

14. Upvote/Downvote a Post

- Expected Outcome: User is able to upvote or downvote a post, and the vote count updates.
- Actual Outcome: Upvoting and downvoting functionality works correctly.
- Issues/Bugs: None

15. Basic Search

- Expected Outcome: Basic search should return relevant results.
- Actual Outcome: Basic search returns correct results.
- Issues/Bugs: None.

16. Advanced Search

- Expected Outcome: Advanced search should return relevant results based on multiple criteria.
- Actual Outcome: Advanced search returns correct results.
- Issues/Bugs: None.

17. View Homepage

- Expected Outcome: Homepage displays posts from communities the user is a member of.
- Actual Outcome: Homepage displays the correct posts.
- Issues/Bugs: None.

18. Discover New Communities

- Expected Outcome: Discover page lists new, popular, and recently active communities correctly.
- Actual Outcome: Discover page displays new, popular, and recently active communities as expected.
- Issues/Bugs: None.

19. Sort Posts in Community Page

- Expected Outcome: User should be able to sort posts by date and number of likes.
- Actual Outcome: Sorting functionality works correctly.
- Issues/Bugs: None.