# SWE 573: Software Development Practice - Spring 2024

# Project Report – Community Specific Information Management System (CSIMS)

**Doruk Büyükyıldırım**
ID: 2022719033

Date: 20.05.2024

## Table of Contents:

## 1. Honor Code

Related to the submission of all the project deliverables for the Swe573 2024 Spring semester project reported in this report, I Doruk Büyükyıldırım 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.

Doruk Büyükyıldırım

## 2. Project Information

### 2.1. Deployment URLs

**2.1.1.** Development Environment: https://dev.swe573.dorukb.com

**2.1.2.** Main Environment: https://swe573.dorukb.com

### 2.2. Repository URLs

**2.2.1.** Git repository: https://github.com/DorukBu/swe573

**2.2.2.** Git tag version URL: https://github.com/DorukBu/swe573/releases/tag/v0.9

### 2.3. Test Information for Deployments

**2.3.1.** Username: clouduser

**2.3.2.** Password: testpassword

## 3. Project Details

### 3.1. Overview

### 3.2. Software Requirements Specification

**FR-1** Users will authenticate with their username and passwords which they set during registration

**FR-2** There are 4 types of users -looking from a community perspective:

1. non-members/ wanderers: these type of users will not have access to private communities. They will be shown popular content when visiting main landing page and community pages ().
2. members: will have access to community -if the community is private, they should be invited and joined the community through invitation-, can create posts, can comment on posts
3. Community moderators/ moderator: Builders of community, community creators and other moderators can add a user as a moderator. Can create new post-templates, ban users.
4. Creator/ owner: The account which creates the community has the ownership. The ownership can be transferred. Owner is also a moderator. There can be multiple owners of a community. An owner cannot close community, only archive (read-only community) it.

**FR-3** There should be a basic search and an advanced search where users can filter: communities, post types (post template types), post creator, and filter for specific data fields belonging to a post type.

**FR-4** The main page of the application will display the following:

1. For non-users: Popular posts (recent, upward trending posts) in an order
2. For registered users: Popular posts from subscribed communities, non-member communities with similar interest (similarity can be inferred by analyzing other users subscribed communities)

**FR-5** Each post will be created by using a post template created by the moderators. A post template dictates the format of the post. It consists of any number of information fields set by the moderator (template creator). An information field consists of three elements:

1. Field name
2. Field data type
3. Requirement of the field (is the field required or not)

**FR-6** All XSD data types should be supported by the post templates

**FR-7** Moderators can kick people if they have to.

**FR-8** Moderators cannot kick other moderators, the owner(s) can.

**FR-9** The data cannot be deleted unless it is found to be harmful. Such content requires can be deleted by a moderator.

**FR-10** Communities, data inside communities cannot be deleted but archived.

**FR-11** private communities will require an invitation process to get in.

**FR-12** An owner cannot leave a community without transferring the ownership.

**FR-13** Any type of user can put a profile picture/ avatar

**FR-14** Users can edit or delete content but, it should be made clear to other users.

**FR-15** There will be an upvote down-vote mechanism to promote dynamic participation

**FR-16** There will be a reporting system for the usage statistics, trending communities etc.

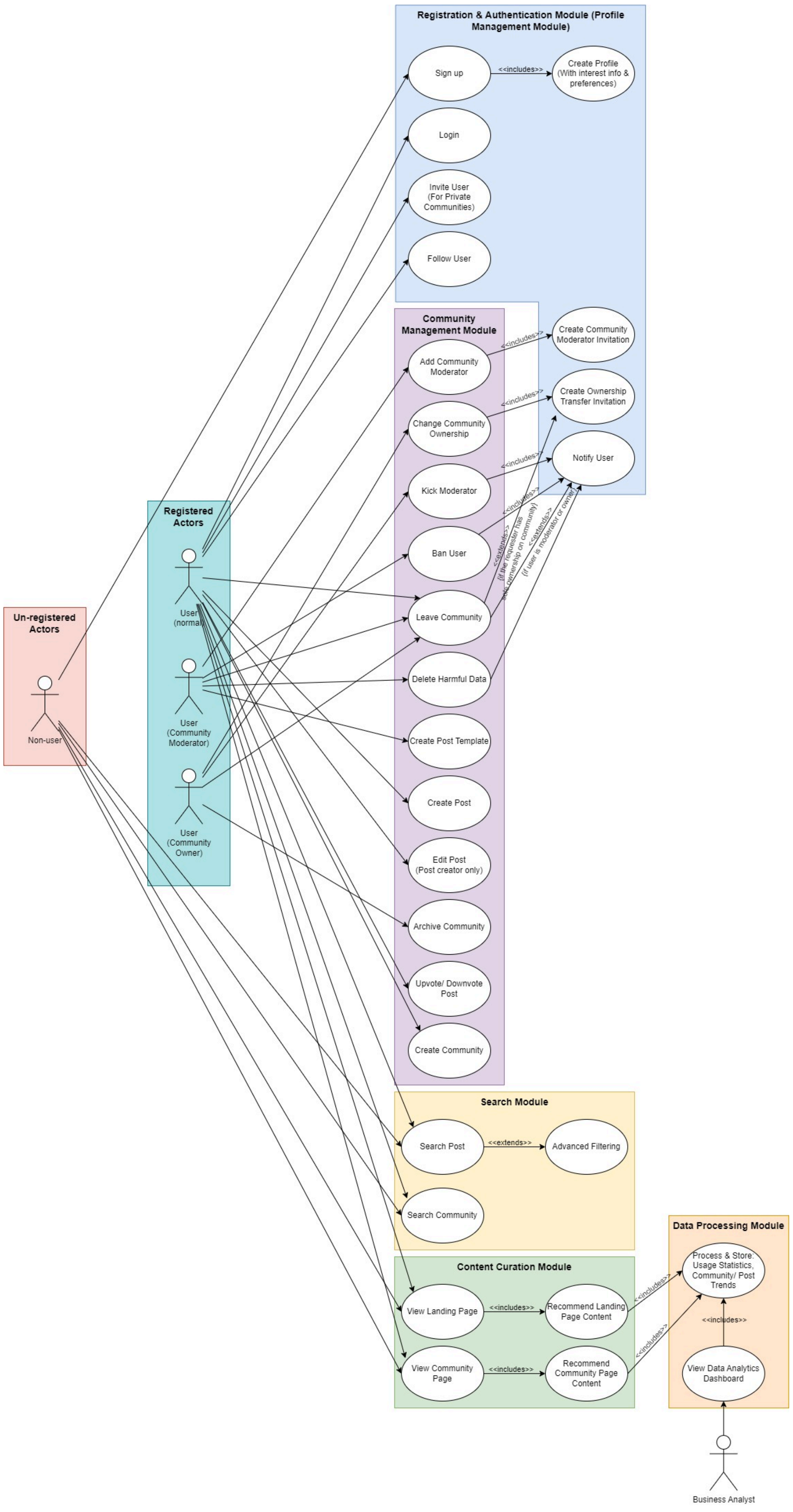
**FR-17** There will not be a direct messaging option for the system

**FR-18** Users should be able to follow other users and see their activities

## 3.3. Design Documents

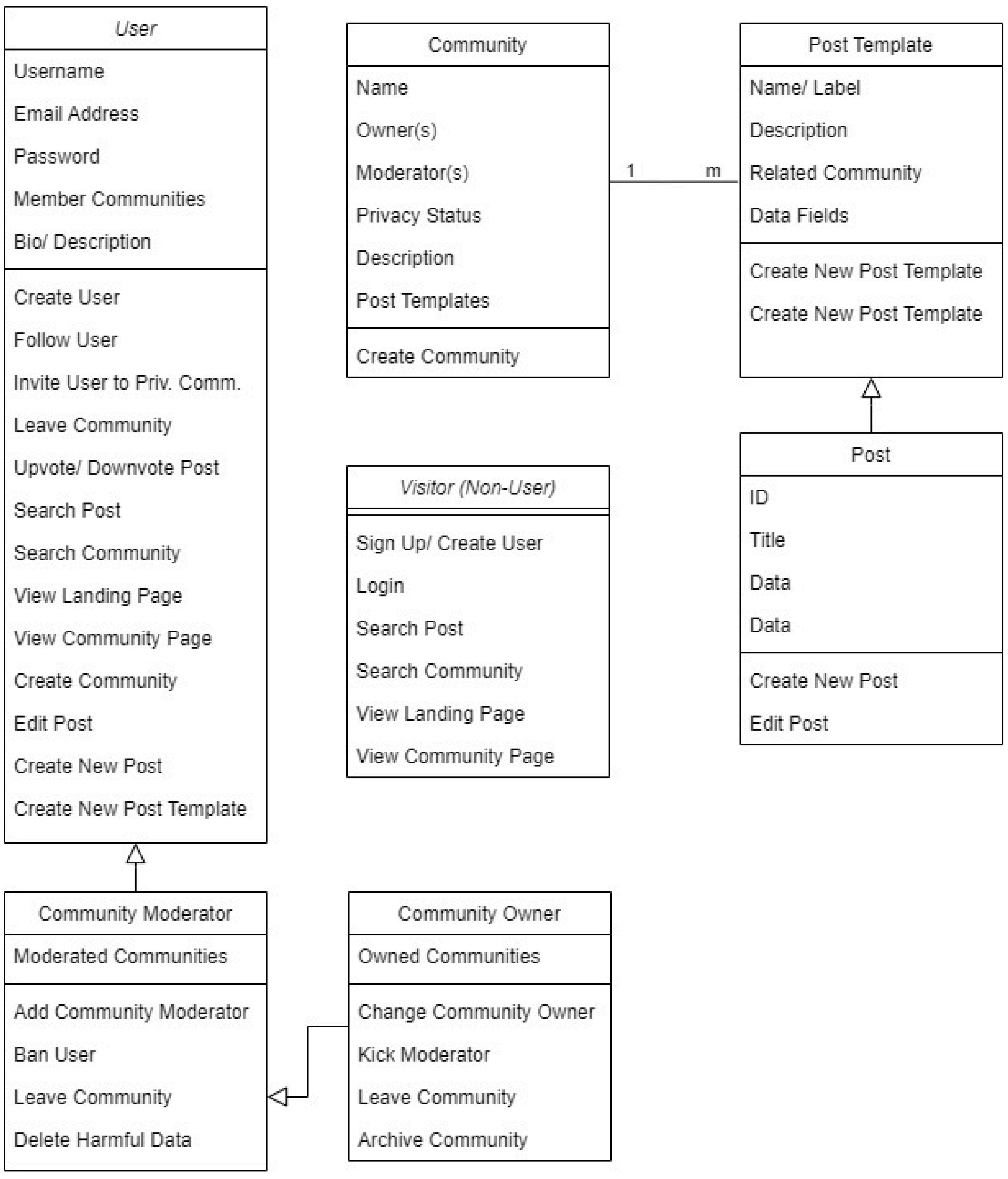
During the design phase of the project, the following design documents were produced

### 3.3.1. Use Case Diagram



Registration & Authentication Module (Profile Management Module)
Sign up
<<includes>>
Create Profile (With interest info & preferences)
Login
Invite User (For Private Communities)
Follow User
Community Management Module
Add Community Moderator
<<includes>>
Create Community Moderator Invitation
Change Community Ownership
<<includes>>
Create Ownership Transfer Invitation
Kick Moderator
<<includes>>
Notify User
Ban User
<<includes>>
<<extends>>
{if the requester has sole ownership on community}
<<extends>>
{if user is moderator or owner}
Leave Community
Delete Harmful Data
Create Post Template
Create Post
Edit Post (Post creator only)
Archive Community
Upvote/ Downvote Post
Create Community
Registered Actors
User (normal)
User (Community Moderator)
User (Community Owner)
Un-registered Actors
Non-user
Search Module
Search Post
<<extends>>
Advanced Filtering
Search Community
Content Curation Module
View Landing Page
<<includes>>
Recommend Landing Page Content
View Community Page
<<includes>>
Recommend Community Page Content
Data Processing Module
<<includes>>
<<includes>>
Process & Store: Usage Statistics, Community/ Post Trends
<<includes>>
View Data Analytics Dashboard
Business Analyst

### 3.3.2. Class Diagram



## 3.4.Project Status

### 3.4.1. Completed Requirements

FR-1, FR-2 (member and non-member user types), FR-5 (only basic post template is implemented), FR-18

***Note:** Full list of requirements can be found under 3.2*

### 3.4.2. Not completed Requirements

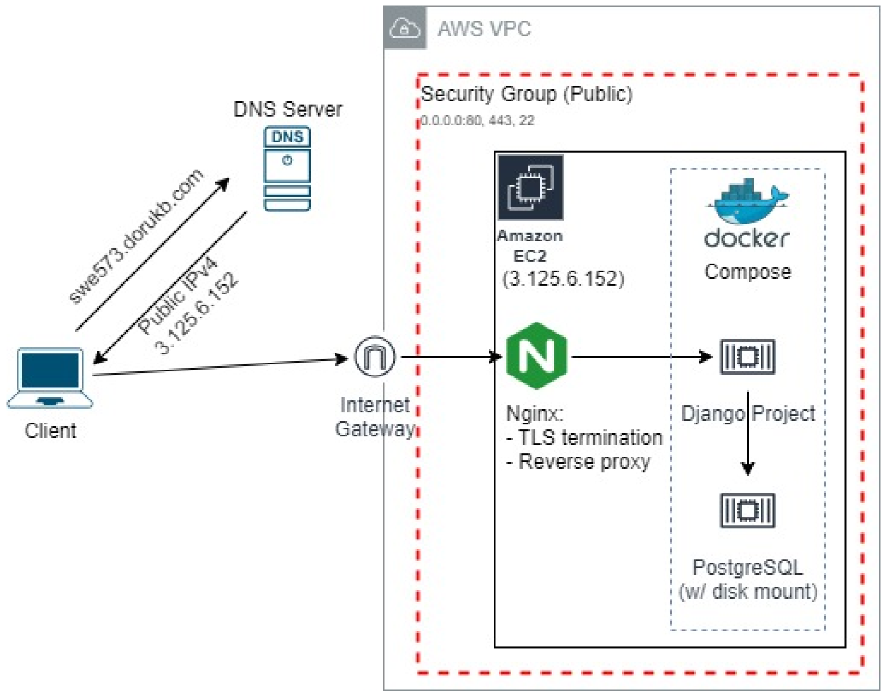
FR-2 (moderator and owner type not implemented), FR-3, FR-4, FR-5 (custom post template creation), FR-6, FR-7, FR-8, FR-9, FR-10, FR-11, FR-12, FR-13, FR-14, FR-15, FR-16, FR-17

## 3.5. Deployment Status

The project has been dockerized and can be deployed/ run utilizing the `docker compose` command, which also initiates the required SQL DBMS and needed database tables.

The project has been deployed as two environments (dev, main) on Amazon Web Services.

The following diagram illustrates the network architecture and infrastructure behind the cloud deployment:



**Steps taken for cloud deployment in detail:**

- A free tier Ec2 instance is started in Amazon Web Services (AWS).
- The instance is attached to a Virtual Private Cloud (VPC) with Internet Gateway and a Security Group (SG).
- The SG is configured to allow traffic from the internet on ports 22 (for ssh connection), 80 (for http requests), and 443 (for https requests).
- A DNS record (A record) is created to point to the public IP of the Ec2 instance inside AWS
- An Nginx Web Server service is created and run inside the Ec2 instance to provide TLS termination and reverse proxying the traffic to the Django project listening on port 8000.
- The required TLS certificate is generated automatically by Certbot software for the DNS record (swe573.dorukb.com).
- A Linux systemd service is generated and started to run docker compose up command as a daemon service.
- GitHub Actions pipelines (configured as pipeline yaml files) deploy the updated code when a push/ merge event on designated branch takes place (automatically triggered).

### 3.5.1. Project URLs:

Main URL: https://swe573.dorukb.com (auto-deployed from main branch)
Development URL: https://dev.swe573.dorukb.com (auto-deployed from dev branch)

| | |
|---|---|
| Dockerfile: | *csims/Dockerfile* |
| docker-compose.yaml: | *csims/docker-compose.yaml* |
| dev deployment pipeline: | *.github/workflows/dev_deploy.yaml* |
| main deployment pipeline: | *.github/workflows/main_deploy.yaml* |

## 3.6. Installation Instructions

**There are only three requirements to run the project in any environment:**

1. Internet connection
2. docker version 20.10.x or greater
3. docker compose version v2.2.x or greater

Installing Docker and Docker Compose can be achieved, and commonly done by installing Docker Desktop on Mac or Windows devices with GUI. You can find the installation instructions (for different platforms) for Docker Desktop on this page: https://docs.docker.com/desktop/

If you would like to install the docker engine without UI (like you would with a Linux server distribution like Ubuntu server) you should follow the steps described under the specific platform on this page: https://docs.docker.com/engine/install/ This guide goes through docker engine, docker-compose-plugin (required for `docker compose` command usage), and other commonly used tools for a docker development environment.

**To build and run the application locally:**

**3.6.1.** The docker compose up command should be executed inside the csims directory (/swe573/tree/main/csims).

**3.6.2.** The required software packages will be downloaded during the initial docker compose up run, which may take some time during the first run. Following runs will take much less time due to Docker's caching feature.

**3.6.3.** The PostgreSQL database will run automatically and create /db_data folder for database files.

**3.6.4.** During the docker compose up the database migrations required by Django will run automatically.

**3.6.5.** Execute docker compose down to stop the project. After shutting down data in the database will be persistent under /db_data folder. Any user, user, community, etc. created in a previous run will be available if data under /db_data is not deleted.

**Note:** */db_data* folder is added to *.gitignore* file so that database data is not pushed to remote repository.

## 3.7.User Manual

The functionality is close to what a web community platform would provide.

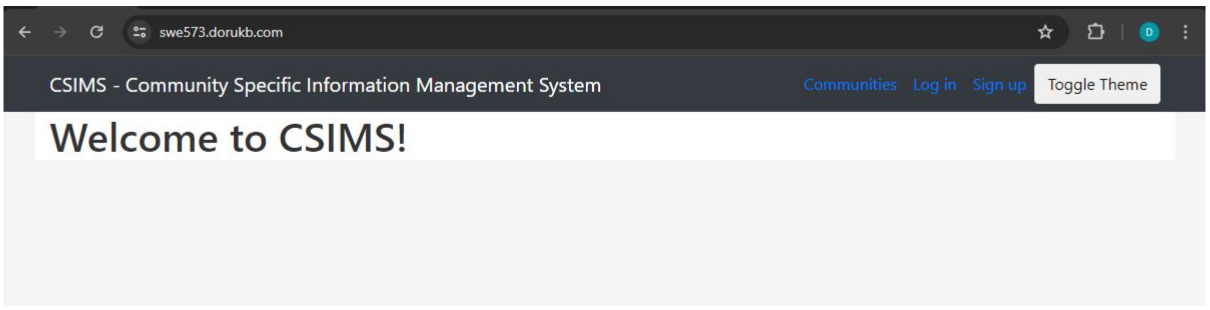
- The visitors can sign in, log in, create communities, create posts in communities, join communities
- Non signed in users cannot create communities or posts, they cannot join communities.
- Only the creator of post can delete the post

### 3.7.1. User screens:

#### 3.7.1.1. First landing page for non-logged in user:

The user (not-logged in) is greeted with the main page containing 4 options on the upper right hand side (from right to left):

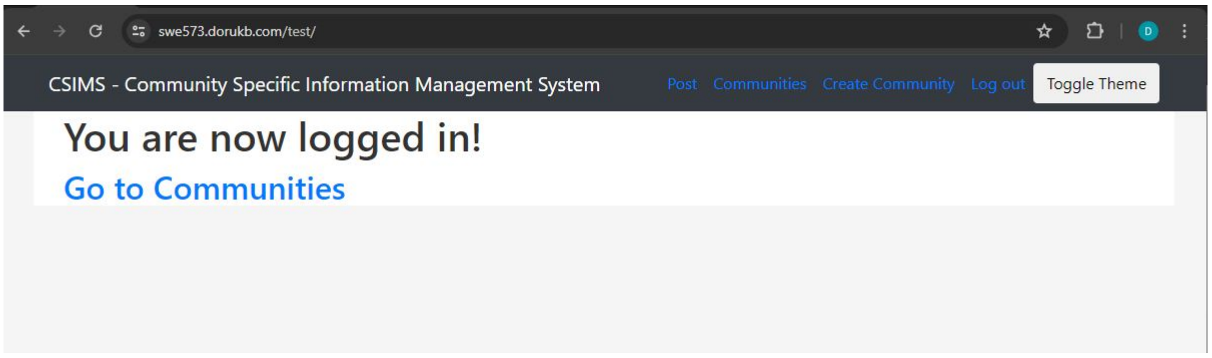
1. Toggle Theme: toggles theme between a light and a dark one
2. Sign up page: Redirects to the sign up page for first registration
3. Log in page: Redirects to log in page for already registered users
4. Communities page: non-registered users can also see the communities and associated posts but cannot create new communities or posts



Welcome to CSIMS!

#### 3.7.1.2. Landing page for logged in users

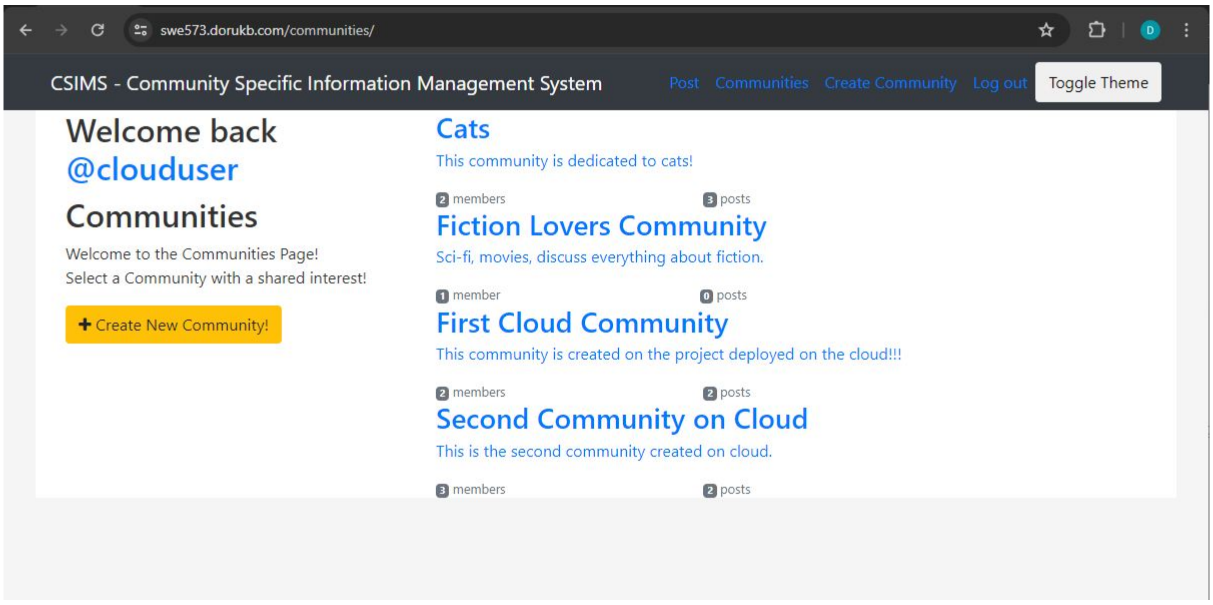
The logged in users are redirected to a landing page with two more options: create community page and posts page (for creating new posts) links.



You are now logged in!
Go to Communities

#### 3.7.1.3. Communities page for logged in users

The communities page for a logged in user is the same for a gues user except the create community option


swe573.dorukb.com/communities/

CSIMS - Community Specific Information Management System

Welcome back @clouduser

Communities

Welcome to the Communities Page!
Select a Community with a shared interest!

Cats
This community is dedicated to cats!
2 members 3 posts

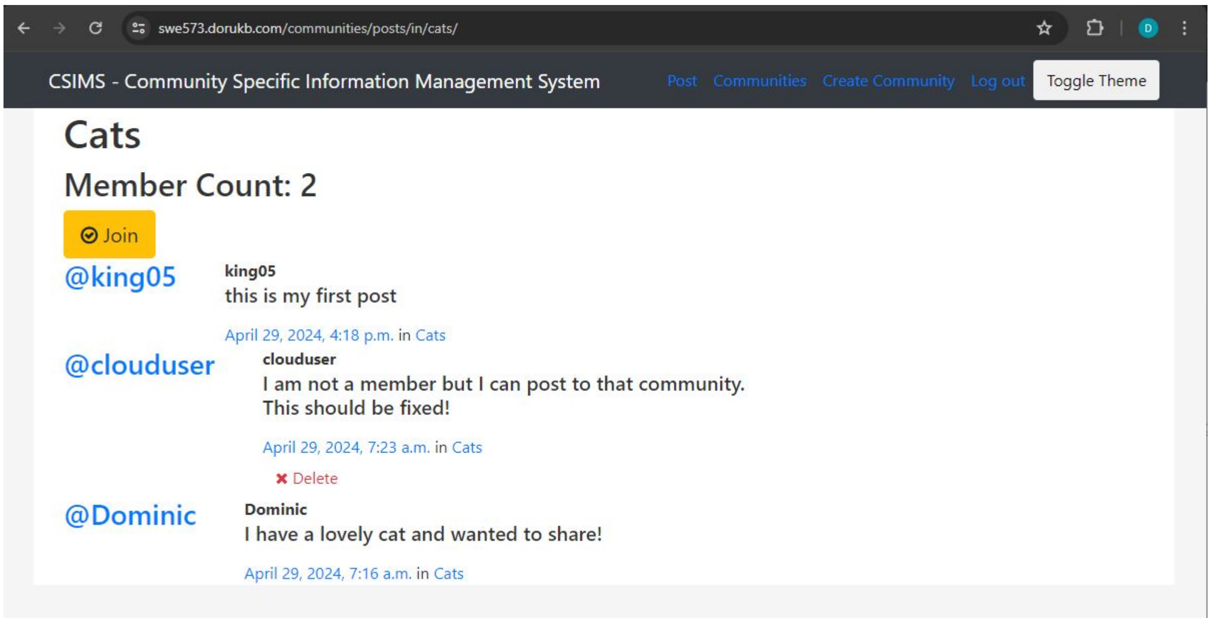
Fiction Lovers Community
Sci-fi, movies, discuss everything about fiction.
1 member 0 posts

First Cloud Community
This community is created on the project deployed on the cloud!!!
2 members 2 posts

Second Community on Cloud
This is the second community created on cloud.
3 members 2 posts

### 3.7.1.4. Page of a community as a logged in user

The logged in user can see the posts belonging to a community and delete only their post if desired


swe573.dorukb.com/communities/posts/in/cats/

CSIMS - Community Specific Information Management System

Cats

Member Count: 2

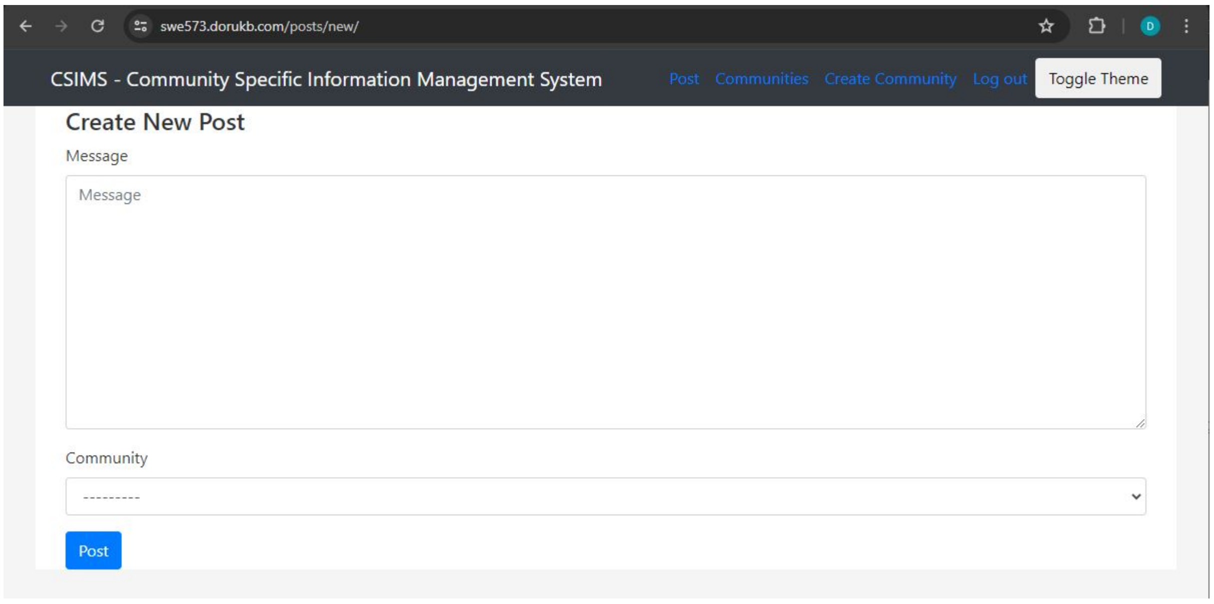
@king05
king05
this is my first post
April 29, 2024, 4:18 p.m. in Cats

@clouduser
clouduser
I am not a member but I can post to that community. This should be fixed!
April 29, 2024, 7:23 a.m. in Cats

@Dominic
Dominic
I have a lovely cat and wanted to share!
April 29, 2024, 7:16 a.m. in Cats

#### 3.7.1.5. Post creation page for logged in users

Post creation page can only be used as a logged in user. Post content and target community is required as inputs



### 3.8. Test Results

### 3.9. Demonstration (Video)

The following Youtube link contains the demonstration video (3:15): https://youtu.be/J0q1_KV5RyI

## 4. Use Rights