Burak Coşar - 2022719036
SWE 573: Software Development Practice, Spring 2024
Project Final Deliverables

Student ID: 2022719036
Name: Burak Coşar
Date: 20-May-2024
Project Name: SocialHub
Deployment URL: https://swe573.burakcosar.com
Alternative Deployment Address (IPv4): http://51.20.37.61:8000
Git Repository URL: https://github.com/Burak-Cosar/SWE573-Cosar
Git Tag Version URL: https://github.com/Burak-Cosar/SWE573-Cosar/releases/tag/SocialHub

---

HONOR CODE

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

Burak Coşar

TABLE OF CONTENTS

REFERENCES

The main reference for the creation of this application is the Youtube channel, codemy.com. My Django knowledge is almost completely derived from the video series called “Django Wednesdays” on this channel. The way I built my general architecture, models, views and templates are mostly rearrangements or inspirations from this project. Apart from the work of Mr. John Elder (codemy.com), I browsed through several one-off solutions for specific problems, such as development forums such as stackoverflow.com, generative AI, and official or third party communities of certain services made use of (AWS, Docker, etc.).

OVERVIEW

This report is the documentation of the final deliverables of the term project “SocialHub” created for SWE573 course.

Project references, requirements, design, status of tasks, dockerization, deployment, installation instructions, user manual, test information and a video demo of the system is included in this report.

On a personal note, since this project has been my first full stack endeavor, I learned countless things from almost every little step I took. However; due to time constraints, implementation status (with regards to requirements) could only be finalized as partially completed. That being said, given enough time, I developed the level of confidence to envision all the requirements to be solved and implemented. This vision and confidence has been the number one takeaway for me from my efforts throughout the semester.

I would still like to name the other, concrete, takeaways as well:
- Using GitHub and the understanding of version control systems in general
- Using ORM for database management
- Django’s MVT system
- Dockerization of backend, PostgreSQL database and Nginx
- Deployment of this complete system on cloud (AWS)

I am looking forward to building on what I acquired from this course both in order to prepare for the upcoming semester and realize my personal ambitions around programming.

SOFTWARE REQUIREMENTS SPECIFICATION

Introduction

This document outlines the software requirements for a community platform, focusing on ownership and moderation, user interaction, community features, and general requirements to ensure compliance and user engagement.

Glossary

User: An individual who has registered and is participating in the platform. They can create, engage with content, join communities, and interact with other users.
Community: A channel within the platform where users can post content, discuss topics, and share information related to a specific interest or theme.
Community Owner: A user who has created a community. They have administrative rights over the community, including setting its rules, appointing moderators, managing membership and creating post templates.
Community Moderator: A user appointed by the community owner to help enforce community guidelines, manage content, and address user issues within the community.
Template: Templates are customized post forms, created and managed by the community owners and moderators.
Post: Content created by a user within a community. It can include various data types depending on the community templates and is meant for other community members to view, engage with, and discuss.
Comment: A response to a post made by a user. Comments allow for interaction and discussion directly related to the original post content.
Upvote: An action that a user can take to express positive feedback or agreement with a post or comment.
Downvote: An action that a user can take to express negative feedback or disagreement with a post or comment.
The System: The software platform that supports the community platform, including the frontend user interface, backend databases, and the algorithms.

Requirements

1. The System
1.1. The system shall provide user with a registration feature with username, email and password.
1.2. The system shall allow user to log in with username and password.
1.3. The system shall prohibit access to content for visitors who are not logged in.
1.4. The system shall provide users with a code of conduct that outlines the desired and undesired behaviors.
1.5. The system shall support multiple data types for content creation, including but not limited to text, photos, videos, geolocation, and various media formats.
1.6. The system shall not offer direct messaging or private messaging functionalities.
1.7. The system shall offer basic and advanced search functionalities. Basic Search shall allow users to perform quick searches using keywords or phrases. Advanced Search shall enable users to refine their searches using multiple criteria and filters including but not limited to users, communities, posts and post (template) types.
1.8. The system shall showcase the popular and recent activities in the home page of the application.

2. Users
2.1. Users shall be able to log in with their username and password to access their account and interact with the platform.
2.2. Users shall have the option to personalize their profiles. This personalization may include, but is not limited to, profile pictures, biographical details, and contact information.
2.3. Users shall be able to follow and unfollow other users.

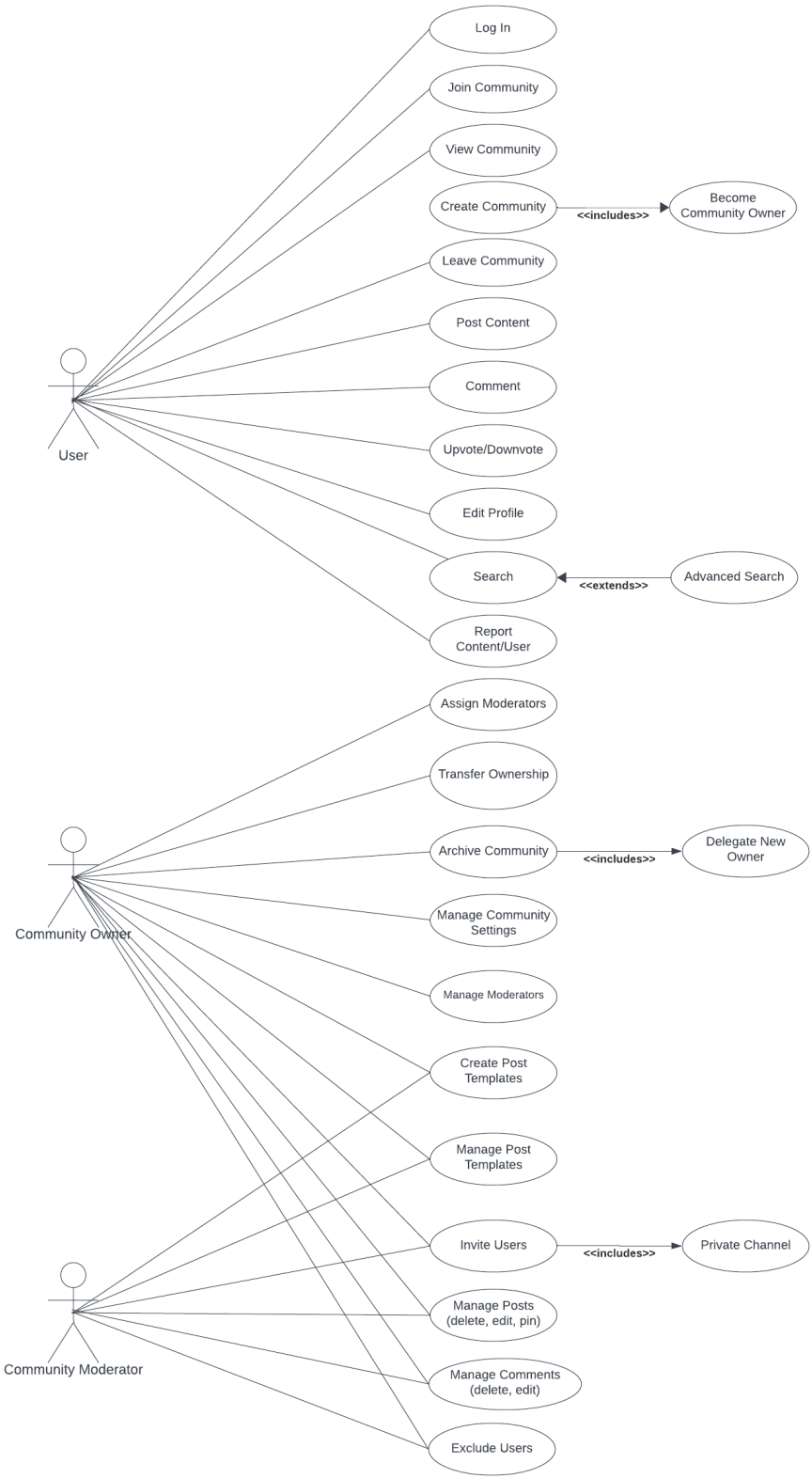
2.4. Users shall be able to report other users activities based on provided reporting categories. These categories may include but not limited to, 'Harassment', 'Spam', 'Inappropriate Content', 'Hate Speech', and 'Other'.
2.5. Users shall be able to join public communities so that they can create posts, comments and reactions in that community.
2.6. Users shall be able to create communities and become community owner of the created community.

3. Ownership and Moderation
3.1. A user shall become the community owner of a community they created.
3.2. Community owners shall control the privacy settings of their community on creation, choosing who can join (public or private).
3.3. Community owners shall have the ability to assign and manage moderators to help manage the community and distribute responsibilities.
3.4. A moderator shall not be able to kick other moderators.
3.5. Community owners shall have the ability to archive their community, permanently removing it from the platform while keeping existing content accessible.
3.6. Community owners shall be able to transfer ownership to another member.
3.7. Community owners shall be required to transfer ownership to another member to relinquish control of the community.
3.8. Community owner shall be able to change the name and description of the community.
3.9. Community moderators shall be able to delete content to ensure the community adheres to its guidelines.
3.10. Community moderators shall have the ability to exclude (ban) users and to revert this action so that a community environment can be maintained.

4. Communities
4.1. A community shall have a unique name.
4.2. A community shall have a description so that users can understand the content of the community.
4.3. A community shall have a visibility setting of either public or private.
4.4. A community shall have a default post template when created.
4.5. Users shall be able to view the content of a public community.
4.6. Users shall join a public community to create posts, comments and interactions.
4.7. Private communities shall not be visible to non-member users.
4.8. Private communities shall only be joined via invitations from channel owners and channel moderators.
4.9. Users shall be able to leave a community if they are members.

5. Posts
5.1. Users shall be able to create posts in the communities they are members of.
5.2. Posts shall be created based on available templates in the community.
5.3. Post templates shall be community specific and shall only be created and modifiable by community moderators.
5.4. Post templates creation shall consist of template name and desired fields.
5.5. Users shall be able to upvote, downvote and comment on the posts and other comments in a channel they are included.
5.6. Users shall have the ability to edit or delete their posts, comments and reactions to maintain control over their activities.
5.7. Users shall label posts to indicate their relevance to their categories.
5.8. The system shall indicate user post and comment edits or removals with timestamps, making such actions visible to other users.

DESIGN



Use Case Diagram

User Scenarios & Mockups for Sports Community are available on GitHub.

STATUS OF PROJECT

Below is the list of implemented features from requirements list, along with their status:

Feature	Completed Tasks	Incomplete Tasks	Status
Registration	Username, email, password	-	Complete
Login	Username, email, password	-	Complete
User Profile	Name, Surname, Username, email, Profile Removal (self)	Limited Personalization, Password Update	Complete
Authorization	Login required to view & interact	-	Complete
Code of Conduct	Page created	-	Complete
Supported Data Types	Text, TextArea, Number, Float, Date, Time, Image, Geolocation(number based)	Map based Geolocation, extended data types	Partial
Basic Search	Done	-	Complete
Advanced Search	-	Not Implemented	N/A
Feed	Recent Activities, Activity of Followed Users, Popular Communities	Extended capabilities	Partial
Following	Follow, Unfollow User	-	Complete
Reporting	-	Not Implemented	N/A
Up/Downvote	-	Not Implemented	N/A
Comment	Done	-	Complete
Community	Community creation, admin assignment and default template creation on initialization, private community, user invitation for private communities, template creation, post creation, list of members, list of posts, moderator assignment, join public communities, leave community	Community edit/ removal(archive), moderator capabilities, admin/owner assignment/transfer	Complete
Banning	-	Not Implemented	N/A
Post	Creation, edit (only title), remove	Edit (dynamic fields, timestamps)	Partial
Template	Creation, Use	Edit, Remove	Partial

STATUS OF DEPLOYMENT

The application is deployed on an EC2 instance on AWS. In terms of implementation, source code is pushed to EC2 and the dockerization is completed on EC2 instance with the docker compose build command.

The backend Django application, the Postgres database instance, and Nginx instance are all dockerized and running on this command, on the same EC2. Additionally, an Amazon S3 Bucket is in use for the storage of image files.

Deployment URL: https://swe573.burakcosar.com
Public IPv4 Address of EC2 Instance: http://51.20.37.61:8000
Amazon S3 Bucket: https://socialhubbucket.s3.eu-north-1.amazonaws.com/media

INSTALLATION INSTRUCTIONS

The system is available live on the specified addresses on top of this document.

To run the system locally:
- Make sure you have Docker installed in your OS
- Pull the code from GitHub repository (specified on top of this document)
- Open terminal on the main directory (where docker file and docker-compose.yml exist)
- Run “docker-compose up —build” on terminal
- The system will be up and running

Note: In certain cases, during the very first launch, if DB is initialized after application, there might be a DB error. In such cases, stop the running containers and re-build them.

USER MANUAL

Since the live deployment has a solid Bootstrap front-end, it is pretty intuitive to navigate through the Website. Below are some instructions to navigate a new user through the journey:

User Registration: Users can not access the Website content without logging in. Thus, if not logged in, a user will be welcomed by the options to “register” or “login”. Simply click the green “Register” button to proceed to register with Username, Email Address and Password.

User Login: If not logged in, a user will be welcomed by the options to “**register**” or “**login**”. Simply click the gray “Login” button to login with Username and Password.

Homepage: Homepage consists of left, center, right columns. On the left you can access the list of all communities, view popular communities based on member count and your subscribed communities. In the middle, there are recent posts and posts from users you follow. On the right, you can access the code of conduct.

Code of Conduct: On the right side of the homepage, there is a blue button directing to Code of Conduct. Click this button to view the **Code of Conduct**.

Communities: On the left side of the **homepage**, click the green button “**View All Communities**” to view **public** and (if any) **private communities** you are invited or a part of.

When you are in the communities page, you can use the green “**Create New Community**” button to create your own community. Creating a new community will automatically make you the administrator of that specific community. Every user can view a join a **public community**; however, if you choose to create a **private community,** you can use the yellow button “Invite Users” in your community to invite the users of your choice to the community.

On the community page, you can see the list of members of the community on the right side. If you are the administrator, you can **make moderator** or **remove moderator** using this list.

If you are invited to a **private community**, the **invitation to community** will pop both under “**Pending Invites**” section on the left side or in the main body of the list of communities in the communities page.

For the communities that you do not administer, you will have the option to **join**/**leave.** This feature can be used within the community or in the communities list, using the self titled buttons (green button for **join**, red for **leave**).

Templates: As a non-administrator user of a community, in order to create a **post,** you need to select a template type from the dropdown menu. **Templates** are different post forms, each one with their specific set of fields for their purposes.

If you are the **administrator** of a **community**, you can **create a template.** In order to create a template, use the green button “**Create New Template**” in the community page, under **Admin panel** on the left side. Every template begins with a “**title**” section that can not be removed and default for every template. After that, you can click “**Add Field**” button to give the field a name and choose the data type of the field from dropdown menu.

Posts: To create a **post**, in the community page, simply select a template type from the dropdown menu and click the green button “**Create Post**”. Fill in the specified post fields to post.

Once you created a post, you can view the post by clicking gray “**View**” button on the community page. You can edit the title of the post you created by clicking the yellow button “**Edit Post**” or delete it by clicking the red “**Delete**” button.

Users: When you are in a community, you can view the list of users in the community on the right side with a blue “**View**” button on the side. By either clicking this button or clicking on the user name (where applicable) you can visit the user’s profile to see their followers, the users they follow and the communities they are part of. You can **Follow/Unfollow** users on the profile page using green/red buttons.

TESTING

User Testing

1. **Registration**
   - Navigate to the website address
   - Click “Register” button
   - Enter Username, Email Address, Password
   - Click “Submit” button
   - Verify that you are logged in as the registered user
2. **Login**
   - Navigate to the website address
   - Click “Login” button
   - Enter Username and Password
   - Click “Submit” button
   - Successfully view homepage
   - Verify that navigation bar has “Profile” button
   - Click “Profile” and view your profile settings
3. **Create Public Community**
   - Login to the system
   - Click “View All Communities” button on homepage
   - Click “Create New Community” button on communities page
   - Enter community name and description. Do not check “Private” setting.
   - View the page of the community created
   - Verify that you are the admin of the community:
     - Verify that you are marked as “admin” on the members list.
     - Verify that ADMIN panel is visible to you on the left hand side of the community.
     - Verify there is a “Create New Template” button under the panel.
4. **Create Private Community**
   - Login to the system
   - Click “View All Communities” button on homepage
   - Click “Create New Community” button on communities page
   - Enter community name and description. Check “Private” setting.
   - View the page of the community created
   - Verify that you are the admin of the community:
     - Verify that you are marked as “admin” on the members list.
     - Verify that ADMIN panel is visible to you on the left hand side of the community.
     - Verify there is a “Create New Template” and “Invite Users” buttons under the panel.
5. **Invite User to Private Community**
   - Login to the system
   - Click “View All Communities” button on homepage
   - Click “View” button of the private community you administer, if none exists Create Private Community
   - Click “Invite Users” button on the ADMIN panel
   - On the Invite Users page, select the user(s) you would like to invite
   - Verify that invited users are visible in the “Invited Users” list on the right panel
6. **Create Template**
   - Login to the system
   - Click “View All Communities” button on homepage
   - Click “View” button of the private community you administer, if none exists Create Public or Private Community
   - Click “Create New Template” button on the ADMIN panel
   - Enter template name and description
   - Add fields by their names and data types
   - Click “Submit”
   - Verify you are redirected to the community page
   - Verify that the new template exists in the post creation dropdown
7. **Make/Delete Moderator**
   - Login to the system

- Click “View All Communities” button on homepage
- Click “View” button of the private community you administer
- View the list of members on the right side
- Click “Make Moderator” button on the user of your choice
- Verify that the user has “Moderator” mark
- Click “Delete Mod” button on the user with the ribbon
- Verify that the user’s “Moderator” mark is gone

8. Join Community

- Login to the system
- Click “View All Communities” button on homepage
- Two paths:
  Path 1:
  - Click “Join” button of a community of your choice
  - Verify redirection to the community’s page

  Path 2:
  - Click “View” button of a community that you are not member of
  - Click “Join” button in the community’s page
- Verify your username exists on members list on the right side of the community page

9. Leave Community

- Login to the system
- Click “View All Communities” button on homepage
- Two paths:
  Path 1:
  - Click “Leave” button of a community that you are member of
  - Verify “Leave” button replaced by “Join” button

  Path 2:
  - Click “View” button of a community of your choice that you are member of
  - Click “Leave” button in the community’s page
  - Click “View” button of the community you just left
  - Verify your username removed from members list on the right side of the community page

10. Create Post

- Login to the system
- Click “View All Communities” button on homepage
- Click “View” button of a community of your choice that you are member of
- Select template type and click “Create Post” button
- Fill in the fields in the post form
- Click “Post”
- Verify the new post exists on the community page

11. Edit Post

- Login to the system
- Click “View All Communities” button on homepage
- Click “View” button of a community of your choice that you are member of
- Click “View” button of a post you are the creator of
- Click “Edit Post” button
- Change the title of the post and click “Save Changes”
- Verify the title of the post changed

12. Delete Post

- Login to the system
- Click “View All Communities” button on homepage
- Click “View” button of a community of your choice that you are member of
- Click “View” button of a post you are the creator of
- Click “Delete” button
- Verify the post is not available in the community

13. Follow User

- Login to the system
- Two paths:
  Path 1:
  - Click “View All Communities” button on homepage
  - Click “View” button of a community of your choice
  - Click “View” button of a user on the right side

Path 2:
    - Click username under a post on the homepage
- On the user profile page, click “Follow” button
- Verify your username is under “Followed by” section of the user
- On the homepage, verify the post from followed user are visible on “Followed” feed (if any posts from the user are available)

14. Unfollow User
- Login to the system
- Visit the profile of the user followed in the “Follow” test
- On the user profile page, click “Unfollow” button
- Verify your username is removed from under “Followed by” section of the user

Unit Testing

Due to time constraints and lack of experience/knowledge of unit testing, I was unable to complete unit testing efforts. However, since my application has a solid front-end, user tests above sufficed me to check the functions, both on database and UI.

Demo Video

Demo video can be accessed from the following URL: https://drive.google.com/file/d/1dwl1YAWt8r4m1o-6fEDigJkdCs1twJA0/view?usp=sharing