Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)
Кафедра Інформатики
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

РОЗРОБКА ВЕБЗАСТОСУНКУ «СОЦІАЛЬНА МЕРЕЖА ДЛЯ ПУБЛІКАЦІЇ ТВОРЧИХ РОБІТ»
(тема)

Виконав:
студент 4 курсу, групи ІТІНФ-20-1
Супрун А.Є.
(прізвище, ініціали)

Спеціальності 122 Комп'ютерні науки
(код і повна назва спеціальності)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

Керівник доц. Руденко Д.О.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ Кобилін О.А.
(підпис) (прізвище, ініціали)

2024 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)

Кафедра Інформатики
(повна назва)

Рівень вищої освіти перший (бакалаврський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Супрун Анні Євгенівні
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка вебзастосунку «Соціальна мережа для публікації творчих робіт»

затверджена наказом університету від 20 травня 2024 року № 022 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 27 травня 2024 р.

3. Вихідні дані до роботи науково-методична та науково-технічна література, матеріали конференцій, дані інтернет-мережі, фреймворк з відкритим кодом Spring Framework, бібліотека з відкритим кодом React.

4. Перелік питань, що потрібно опрацювати в роботі
1. Теоретичний огляд функціоналу та методів створення соціальних мереж.
2. Моделювання компонентів соціальної мережі.
3. Реалізація вебзастосунку соціальної мережі.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) Актуальність проблеми взаємодії творчих користувачів у соціальній мережі за допомогою обміну повідомлень і публікації мультимедіа, постановка задачі.

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	08.04.2024	
2	Аналіз завдання, підбір літератури	08.04.24-12.04.24	
3	Аналіз літератури з досліджуваної проблеми	12.04.24-15.04.24	
4	Аналіз технічних засобів	16.04.24-17.04.24	
5	Розробка методу	18.04.24-02.05.24	
6	Програмна реалізація	02.05.24-21.05.24	
7	Оформлення пояснювальної записки	22.05.24-22.05.24	
8	Перевірка на плагіат	27.05.24	
9	Рецензування	28.05.24	
10	Підготовка презентації та доповіді	29.05.24-02.06.24	
11	Занесення роботи в електронний архів	03.06.24	
12	Попередній захист кваліфікаційної роботи	03.06.24	

Дата видачі завдання 8 квітня 2024 р.

Студент _____
(підпис)

Керівник роботи _____ _____доц. Руденко Д.О._____
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до кваліфікаційної роботи: 65 с., 2 табл., 40 рис., 37 джерел.

СОЦІАЛЬНО МЕРЕЖА, ВЕБЗАСТОСУНОК, MYSQL, JAVA, SPRING BOOT, JAVASCRIPT, REACT, MYSQL, БАЗА ДАНИХ, HYPERTEXT TRANSFER PROTOCOL, WEBSOCKET, REST API, SECURITY, JWT, МЕСЕНДЖЕР, ПРОФІЛЬ КОРИСТУВАЧА.

Об’єктом роботи є можливість взаємодії творчих користувачів у соціальній мережі за допомогою обміну повідомлень і публікації мультимедіа.

Метою роботи є розробка вебзастосунку соціальної мережі для публікації творчих робіт, зокрема бази даних, серверної та клієнтської частин.

Проведено аналіз існуючих успішних соціальних мереж, описано їх переваги та недоліки для творчих користувачів. Проаналізовано технології, що використовуються при створенні соціальних мереж. Розроблено алгоритми взаємодії користувачів із контентом і між собою, збереження інформації. Реалізовано користувацький інтерфейс для взаємодії із вебзастосунком.

У результаті роботи здійснена програмна реалізація соціальної мережі для публікації творчих робіт.

SOCIAL NETWORK, WEB APPLICATION, MYSQL, JAVA, SPRING BOOT, JAVASCRIPT, REACT, MYSQL, DATABASE, HTTP, WEBSOCKET, REST API, SECURITY, JWT, MESSENGER, USER PROFILE.

The object of the work is the interaction of creative users in the social network through the exchange of messages and the publication of multimedia.

The aim of the work is the development of a web application of a social network for publishing creative works, including the database, server and client parts.

The analysis of existing successful social networks was carried out, their advantages and disadvantages for creative users were described. The technologies used in the creation of social networks were analyzed. Algorithms for user interaction with content and with each other, information storage were developed. A user interface for interacting with the web application was implemented.

As a result of the work a software implementation of a social network for the publication of creative works was carried out.

ЗМІСТ

Перелік умовних позначень, символів, одиниць, скорочень і термінів 7
Вступ 8
1 Теоретичний огляд функціоналу та методів створення соціальних мереж .. 9
1.1 Огляд предметної області 9
1.2 Огляд функцій популярних соціальних мереж 9
1.3 Аналіз складових наведених соціальних мереж 14
1.3.1 Огляд мов програмування, що використовувалися при створенні наведених соціальних мереж 14
1.3.2 Огляд функціональності 14
1.4 Огляд технологій для розробки соціальної мережі 15
1.4.1 Огляд Java та фреймворка Spring Boot 15
1.4.2 Огляд JavaScript і бібліотеки React 17
1.4.3 Огляд СУБД MySQL 17
1.5 Постановка задачі 18
2 Моделювання компонентів соціальної мережі 19
2.1 Перелік функцій створюваної соціальної мережі 19
2.2 Архітектура соціальної мережі як мікросервісної системи 19
2.3 Структура бази даних та модель даних 20
2.3.1 Сутності і атрибути 20
2.3.2 Опис стовпців таблиць 22
2.3.3 Вибір інструментів для створення бази даних 25
2.4 Моделювання серверної частини 25
2.4.1 Вибір технологій для створення серверу 26
2.4.2 Компоненти серверу 28
2.4.3 Принцип Stateless 30
2.4.4 Автентифікація з використанням JWT токенів 31
2.4.5 Налаштування безпеки 32
2.4.6 WebSocket для миттєвого обміну повідомленнями 33

2.5 Моделювання клієнтської частини 34
2.5.1 Вибір технологій для створення клієнтської частини 34
2.5.2 Структура вебсайту 34
2.5.3 Створення UI з допомогою React 35
3 Реалізація вебзастосунку соціальної мережі 36
3.1 Середовища розробки 36
3.1.1 IntelliJ IDEA Community Edition 36
3.1.2 Visual Studio Code 36
3.1.3 Postman 37
3.2 Реалізація вебзастосунку 37
3.2.1 Типи користувачів 37
3.2.2 Реалізація серверу 38
3.3 Демонстрація роботи вебзастосунку соціальної мережі 42
3.3.1 Реєстрація та вхід 42
3.3.2 Профіль і рекомендації 43
3.3.3 Слідкування за користувачами і обмін повідомленнями 45
3.3.4 Публікації і взаємодія з ними 48
3.3.5 Групи 53
3.3.6 Сторінка адміністратора 58
3.4 Перспективи подальшого розвитку соціальної мережі 59
Висновки 61
Перелік джерел посилання 62

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

СУБД – система управління базами даних

БД – база даних

XML – Extensible Markup Language (розширювана мова розмітки)

HTTP – Hypertext Transfer Protocol (протокол передачі гіпертекстових документів)

JWT – JSON Web Token (стандарт токена доступу на основі JSON)

JSON – JavaScript Object Notation (текстовий формат обміну даними на основі JavaScript)

REST API – Representational State Transfer Application Programming Interface (інтерфейс програмування застосунків з представленням стану)

SQL – Structured Query Language (мова структурованих запитів)

JPA – Java Persistence API

DTO – Data Transfer Object (об’єкт передачі даних)

JPQL – Java Persistence Query Language (мова запитів JPA)

CORS – Cross-Origin Resource Sharing (спільне використання ресурсів між різними джерелами)

HTML – HyperText Markup Language (мова розмітки гіпертексту)

CSS – Cascading Style Sheets (каскадні таблиці стилів)

JS – JavaScript (мова програмування JavaScript)

UI – User Interface (інтерфейс користувача)

JSX – JavaScript XML

ВСТУП

У вік інтернету технології та штучний інтелект все більше стають невід'ємною частиною життя людини. Соціальні мережі надають широкий спектр інструментів і можливостей кожному, хто хоче поспілкуватися й поділитися частиною свого життя зі світом. Водночас із цим, доступність і легкість у використанні створюють величезний потік інформації та людей, який неможливо відфільтрувати самостійно і приділити кожному достатньо часу для глибокого знайомства та справжнього розуміння їх особистості.

Творчі люди зазвичай мають нішеві інтереси, які не завжди поділяє оточення. Через це знаходження однодумців може виявитися складнішим. До того ж, коли штучний інтелект за надзвичайно короткий час, до прикладу, здатний створювати те, на що художник витрачає години, вкладає душу та емоції, пересічний користувач може не оцінити старання останнього. Окрім того, найпопулярніші соціальні мережі зазвичай не надають того функціоналу, який хотілося би мати саме творцям.

Актуальність роботи полягає у створенні та наданні творчим особистостям такого простору, в якому можливо легко та комфортного знаходити однодумців, поширювати контент за інтересами, отримувати підтримку – соціальної мережі для публікації творчих робіт. Вона покликана бути вузькоспеціалізованим середовищем для розвитку спільноти творців, надихати, збагачувати креативний дух.

1 ТЕОРЕТИЧНИЙ ОГЛЯД ФУНКЦІОНАЛУ ТА МЕТОДІВ СТВОРЕННЯ СОЦІАЛЬНИХ МЕРЕЖ

1.1 Огляд предметної області

Соціальна мережа – це соціальна структура, утворена її учасниками на базі спільних інтересів, зв’язків, взаємодій. У контексті даної роботи, соціальна мережа – це застосунок, що дозволяє користувачам створювати профілі, публікувати фото, відео, текстовий контент, знаходити друзів, однодумців, створюючи спільноти за інтересами. Вони можуть мати різноманітні характеристики, але основні їхні атрибути є такими:

– фокусуються на контенті, що створюється користувачами для інших користувачів. Взаємодія відбувається за допомогою реакцій («Мені подобається», коментарів), що заохочують публікацію;

– наявний профіль, що відображає інформацію про користувача та створений ним контент. У ньому може бути використане реальне ім’я або нікнейм, фото профілю;

– наявні, зазвичай, довготривалі зв’язки між користувачами, що досягаються шляхом слідкування за профілем, відправкою повідомлень. Так і створюється, власне, поняття «мережа». На основі інформації про зв’язки у мережі алгоритми підбирають користувачам контент і профілі [1].

1.2 Огляд функцій популярних соціальних мереж

Одними з найбільш популярних соціальних мереж, які зараз використовують творці, є Instagram, X (що раніше мала назву Twitter), TikTok, Pinterest, DeviantArt та Facebook [2]. Ці платформи надають користувачам широкий вибір функцій для створення та публікації контенту, обміну інформацією, спілкування та взаємодії в реальному часі, можливості

для бізнесу. Розглядається функціонал наведених соціальних мереж, що є корисним саме для творців.

Instagram надає користувачам можливість публікувати фото, відео та історії за будь-якою тематикою. Також має функцію месенджеру. Достатньо зручна соціальна мережа, але має такі недоліки для творців, як незрозумілі алгоритми просування контенту, хештеги, що не показують найновіші публікації, нав'язливу рекламу, відсутність можливості створювати групи. На рисунку 1.1 зображений приклад сторінки користувача, вона достатньо зручна та зрозуміла [3].

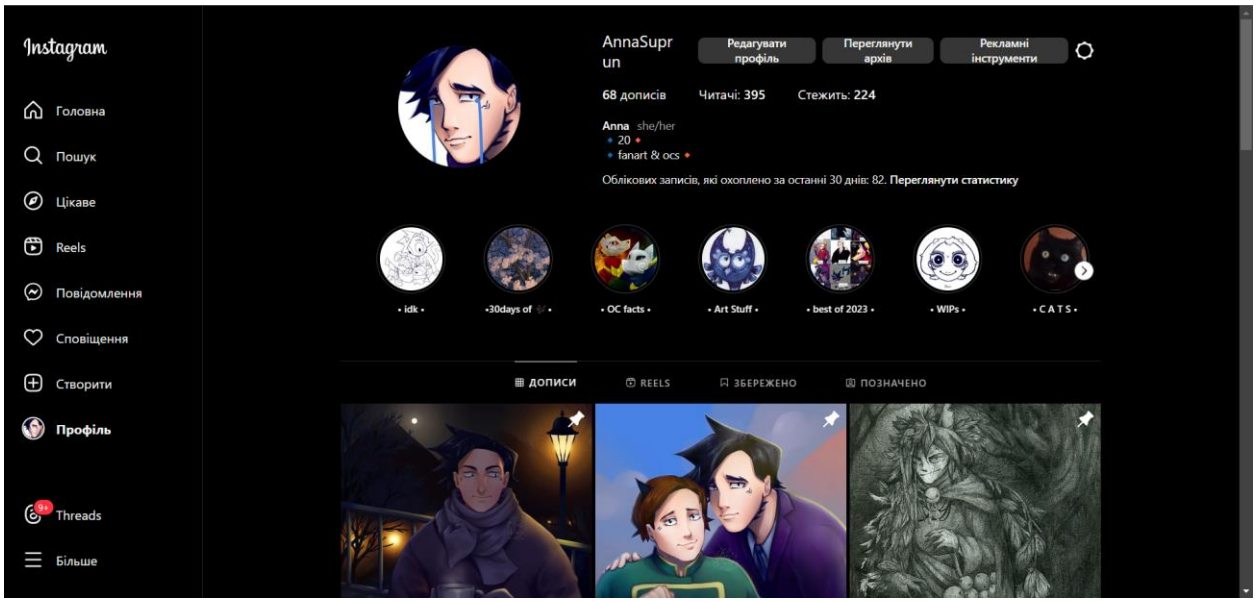

Рисунок 1.1 – Сторінка користувача соціальної мережі Instagram

X надає можливість публікувати фото, відео та текстові повідомлення, створювати групи, спілкуватися у месенджері, підписуватися на інших користувачів. Із недоліків можна виокремити те, що користувачі цієї платформи схильні до конфліктів, тому творці часто залишають її. Також її тематика не є лише творчою, тому алгоритми можуть підбирати користувачу контент, що йому не цікавий, що можна побачити у правій частині рисунку 1.2 (теми, що не стосуються творчості) [4].

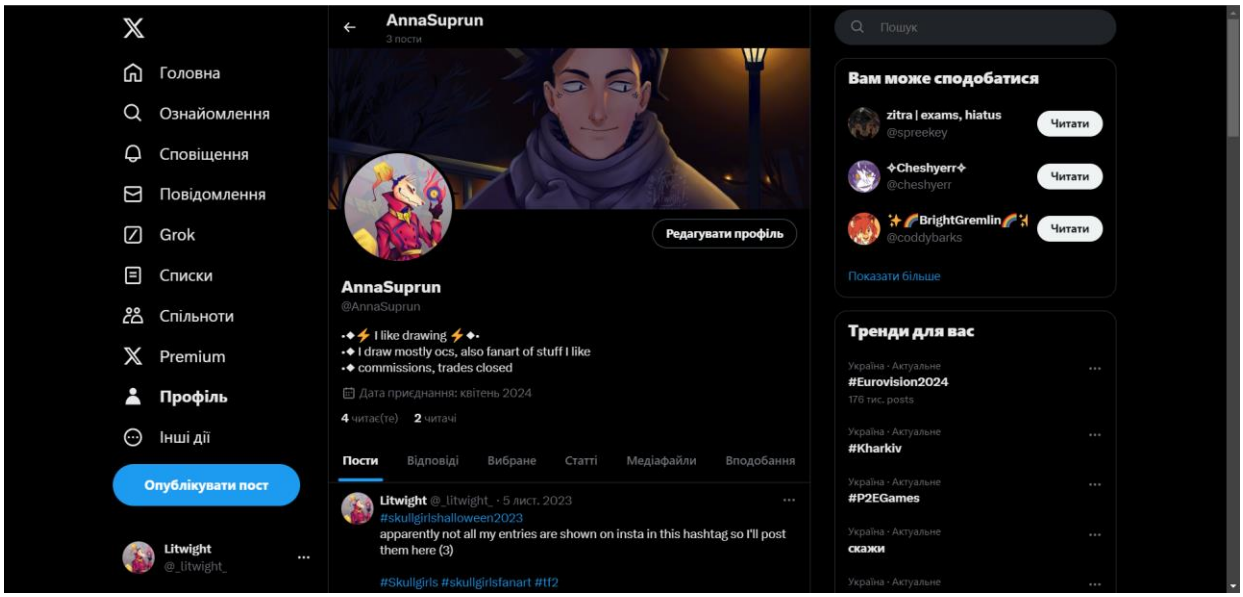

Рисунок 1.2 – Сторінка користувача соціальної мережі X

За допомогою TikTok можна публікувати відео (найголовніша функція даної платформи), фото та слайд шоу. Найбільша перевага – це потужний алгоритм підбору контенту, що також дає можливість новим користувачам швидко набирати тисячі переглядів і знаходити аудиторію (рис. 1.3). Є функція обміну повідомленнями, але лише між користувачами, що стежать один за одним [5].

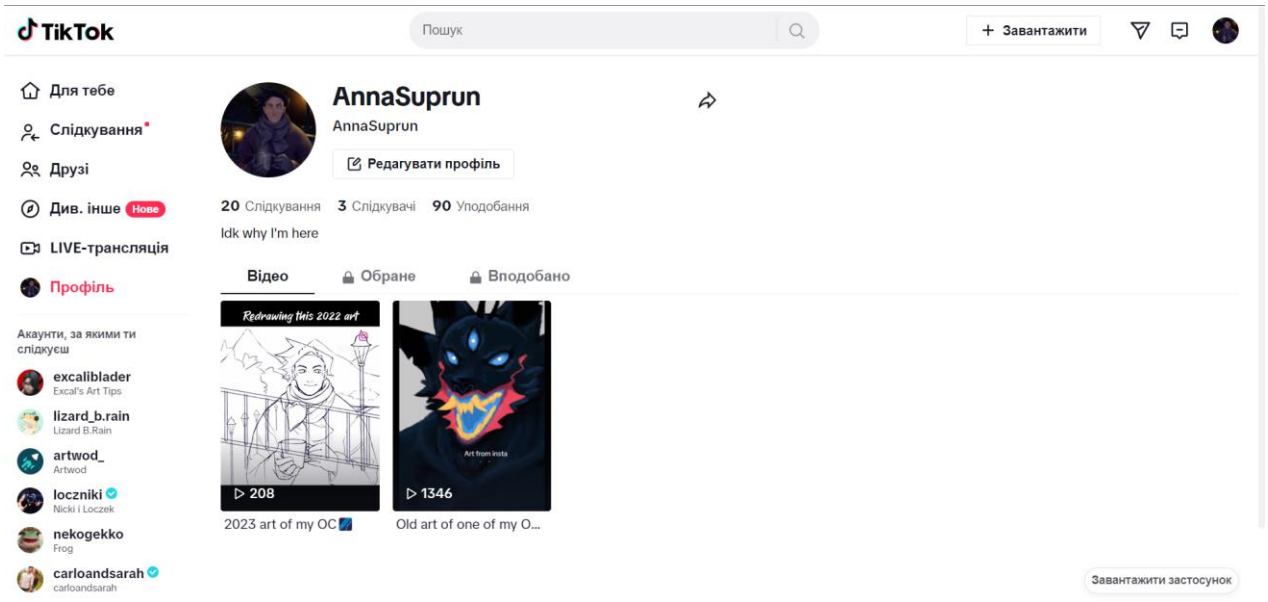

Рисунок 1.3 – Сторінка користувача соціальної мережі TikTok

Pinterest надає можливість публікувати відео та зображення, а однією з основних його функцій є створення «дошок», у які зберігаються контент за темами, які користувач сам виокремлює (рис. 1.4). Подібний до цього функціонал мають попередні приклади, але у Pinterest кожну «дошку» можна поділити на розділи та запрошувати інших користувачів бути редакторами для спільного створення, додання та видалення «пінів» (специфічна назва публікацій на цій платформі). Потужний алгоритм підбору контенту допомагає знаходити матеріали для натхнення, навчання у сфері творчості. Можливо надсилати повідомлення, залишати реакції і коментарі, але це не основний функціонал Pinterest [6].

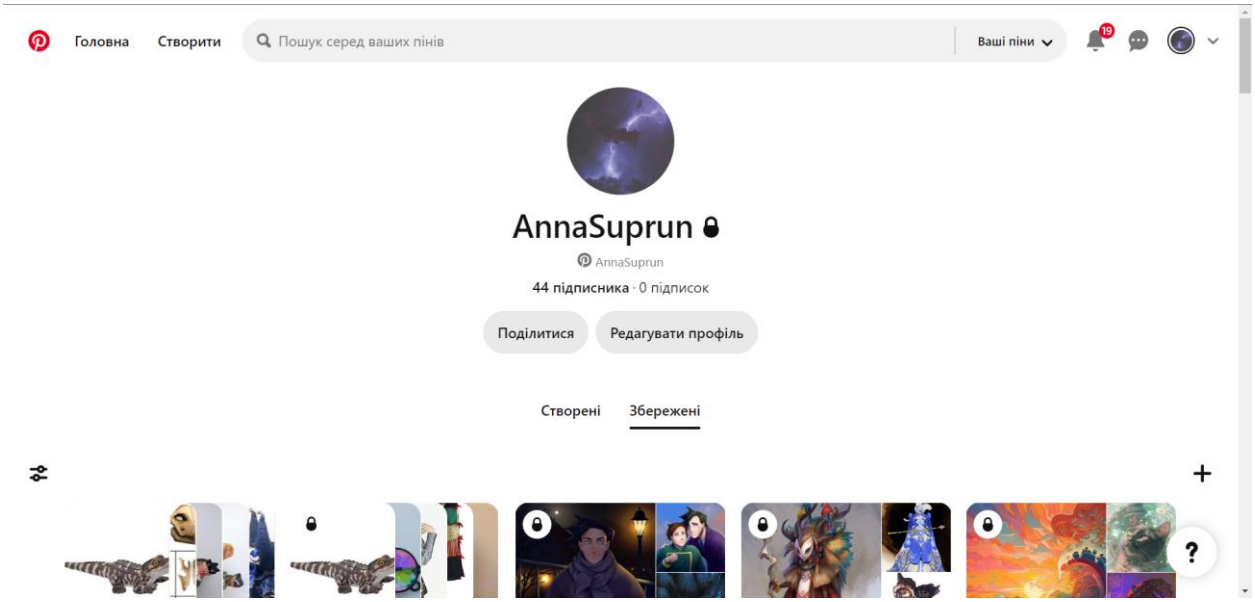

Рисунок 1.4 – Сторінка користувача соціальної мережі Pinterest

DeviantArt має функції публікування зображень, відео, слідкування за користувачами, спілкування у коментарях і чаті. Є можливість створення груп і ведення блогу. Однією із функцій, що вирізняють його з-поміж інших соціальних мереж – це можливість налаштування вигляду власного профілю (зміна кольору, додання різних меню, додання обкладинки тощо), що можна побачити на рисунку 1.5 [7].

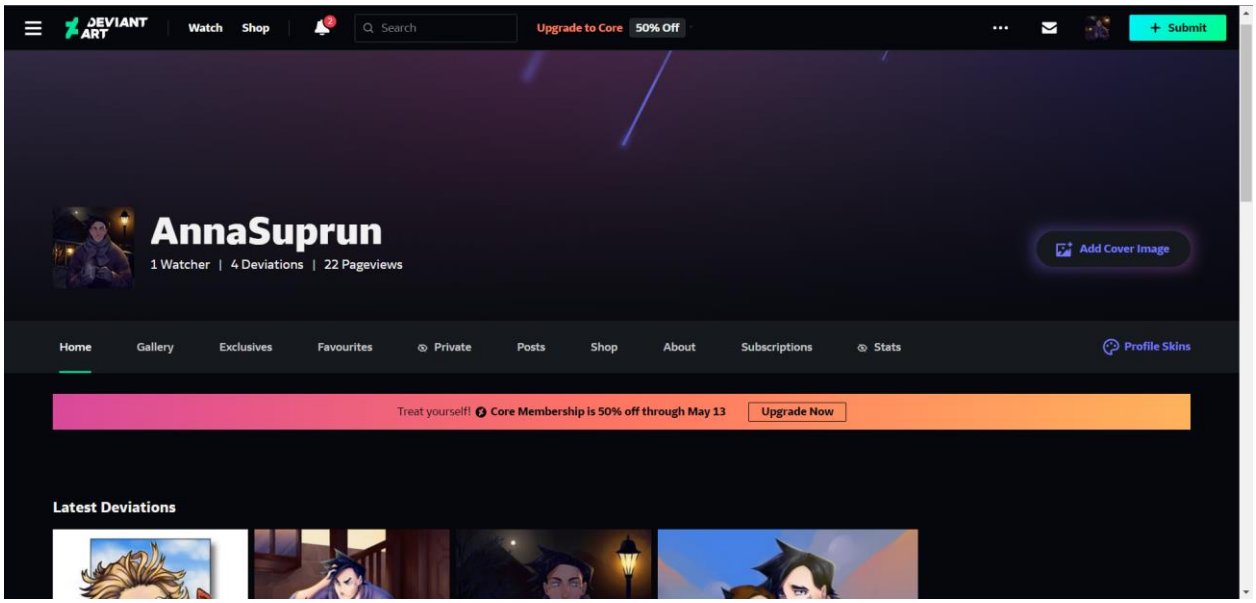

Рисунок 1.5 – Сторінка користувача соціальної мережі DeviantArt

Facebook має багато функцій на будь-який смак, що, звичайно, включає можливість публікувати фото, відео, текстовий контент, створювати групи, слідкувати за користувачами, надсилати повідомлення тощо. Найкраще публікувати творчі роботи або у профілі, або створити для цього спеціальну групу (рис. 1.6) [8].

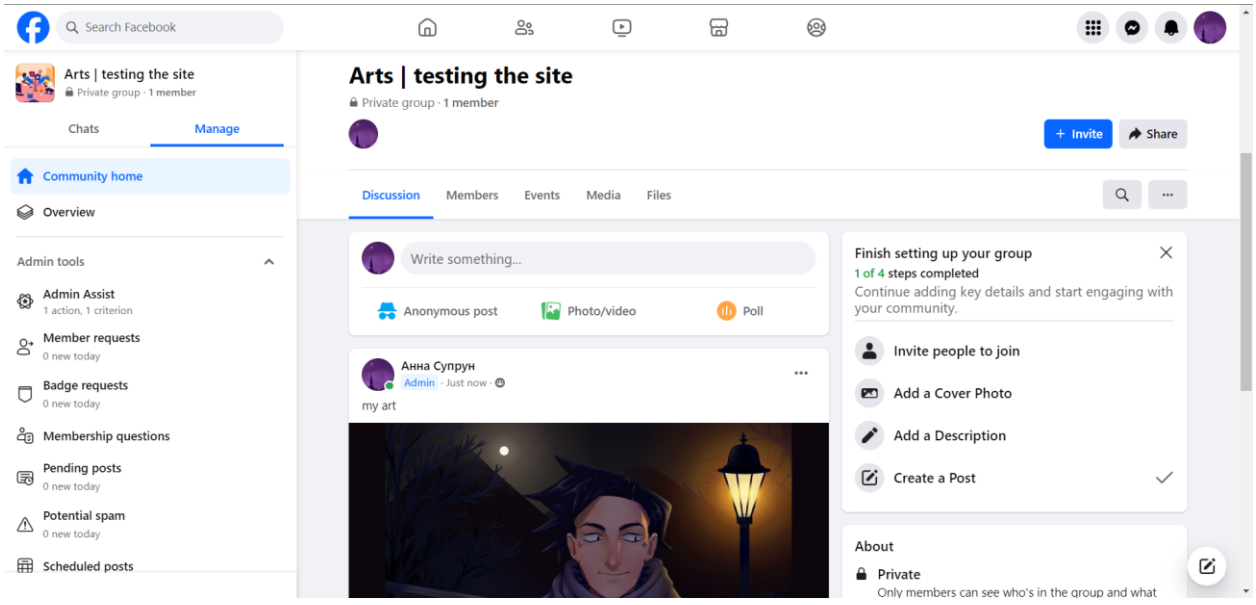

Рисунок 1.6 – Сторінка користувача соціальної мережі Facebook

1.3 Аналіз складових наведених соціальних мереж

1.3.1 Огляд мов програмування, що використовувалися при створенні наведених соціальних мереж

Наведені соціальні мережі використовують різноманітні мови програмування, перелік яких наведено у таблиці 1.1 [9, 10].

Таблиця 1.1 – Перелік мов програмування і БД соціальних мереж

Соціальна мережа	**Серверна частина**	**Клієнтська частина**	**СУБД**
Instagram	Python	JavaScript	PostgreSQL, Cassandra, Redis
X	C++, Java, Scala, Ruby (Ruby on Rails)	JavaScript	MySQL
TikTok	Python, C, Java, Swift	Java, Kotlin (для Android)	MongoDB, Cassandra, MySQL
Pinterest	Python (Django), Erlang, Elixir	JavaScript	MySQL, Redis
DeviantArt	PHP, C++	JavaScript	недостатньо даних
Facebook	Hack/HHVM, Python, C++, Java, Erlang, D, Haskell	JavaScript, Typescript, Flow	MariaDB, MySQL, HBase, Cassandra

1.3.2 Огляд функціональності

Огляд функціональності перелічених соціальних мереж виявив ряд спільних функціональних можливостей:

– публікація контенту. Користувачі мають можливість публікувати фото, відео у більшості випадків, текстові повідомлення та інші медіа-файли;

– спілкування та мережа контактів. Усі соціальні мережі надають засоби для спілкування з іншими користувачами, залишення коментарів, реакцій та обміну приватними повідомленнями;

– групи та спільноти. Багато з цих платформ дозволяють створювати групи за певною тематикою або інтересами, де користувачі можуть обмінюватися ідеями та контентом;

– алгоритми просування контенту.

1.4 Огляд технологій для розробки соціальної мережі

Розробка соціальних мереж вимагає використання різноманітних технологій та інструментів, щоб забезпечити їхню ефективність, масштабованість та зручність для користувачів.

Для розробки серверної частини вебзастосунку гарним рішенням буде використати мову програмування Java та фреймворк Spring Boot.

Клієнтська частина може бути реалізована за допомогою мови програмування JavaScript і бібліотеки React.

У поєднанні ці технології допомагають створювати потужні та зручні вебзастосунки, забезпечуючи надійну та швидку роботу, інтуїтивно зрозумілий інтерфейс та багатий функціонал для користувачів, що дуже важливо для соціальної мережі [11, 12].

1.4.1 Огляд Java та фреймворка Spring Boot

Мова програмування Java має багато переваг, що робить її хорошим вибором для розробки різноманітних програм, від малих вебсайтів до складних платформ та сервісів. Серед них:

– надійність та стабільність. Java має вбудовані механізми безпеки та обробки помилок, що робить її відмінним вибором для створення застосунків;

– незалежність від платформи. Програми, написані на Java, можуть працювати на будь-якій платформі, яка підтримує віртуальну машину Java (JVM), що робить їх переносними та універсальними;

– об’єктно-орієнтований підхід, на якому побудована Java, сприяє створенню модульного та зрозумілого коду;

– велика спільнота розробників, яка підтримує багато корисних бібліотек, фреймворків та інструментів розробки;

– широке застосування. Java використовується в різних сферах від веброзробки та мобільних застосунків до вбудованих систем та великих корпоративних проєктів;

– велика продуктивність завдяки оптимізації віртуальної машини Java та ефективній роботі з пам’яттю.

Spring Boot – це фреймворк для розробки вебзастосунків на мові Java, який має багато переваг:

– надає можливість швидко почати роботу з новим проєктом завдяки своєму автоматичному конфігуруванню;

– має вбудований сервер, що дозволяє запускати застосунки без необхідності окремої установки сервера;

– автоматично керує залежностями проєкту, що спрощує процес додавання, оновлення та управління бібліотеками;

– надає автоматичну конфігурацію за замовчуванням на основі класу шляху та залежностей проєкту, що зменшує необхідність вручну налаштовувати застосунок;

– інтегрується з великою кількістю додаткових модулів та бібліотек Spring, що дозволяє розширювати його функціонал та використовувати різноманітні інструменти для розробки.

1.4.2 Огляд JavaScript і бібліотеки React

JavaScript – це мова програмування, яка використовується для створення динамічних та інтерактивних вебсайтів. Вона володіє широким функціоналом, включаючи можливість маніпулювати вмістом сторінки, взаємодіяти з користувачем, валідувати дані та виконувати асинхронні запити до сервера без перезавантаження сторінки.

JavaScript є незамінною складовою для створення сучасних вебзастосунків, і однією із її найкращих бібліотек є React. Вона відома своєю простотою та ефективністю у роботі з компонентами, що дозволяє розробникам легко створювати складні інтерфейси. React використовує концепцію віртуальної об’єктної моделі документа (DOM) для оптимізації швидкодії та реактивного оновлення сторінок без перезавантаження. Вона також дозволяє використовувати JSX – розширення JavaScript, яке дозволяє описувати структуру компонентів у вигляді HTML-подібного коду. React має багато переваг, таких як ефективність, гнучкість і продуктивність, має активну спільноту розробників [13, 14].

1.4.3 Огляд СУБД MySQL

MySQL – потужна та надійна система управління базами даних, що відома своєю широкою популярністю та використанням у різних сферах. MySQL відома своєю надійністю та стабільністю, забезпечує швидкий доступ до даних та ефективну обробку запитів, підтримує багато мов програмування та платформ, що дозволяє інтегрувати її з різноманітними середовищами розробки, має велику кількість функцій та можливостей, таких як транзакції, індексація, підтримка зовнішніх ключів та багато інших, що робить її потужним інструментом для роботи з даними [15].

1.5 Постановка задачі

Хоча кожна із перелічених соціальних мереж надає багатий функціонал, їхня користувацька база дуже різноманітна, більша її частина не зацікавлена у тематиці творчості [16]. Також не всі з варіантів надають повний список потрібного функціоналу, наприклад, створення спільнот.

Таким чином, створення нового простору для публікації творчих робіт є актуальним. Тому ставиться завдання програмної реалізації соціальної мережі із використанням Java, Spring Boot Framework, JavaScript і React.

Об’єктом роботи є можливість взаємодії творчих користувачів у соціальній мережі за допомогою обміну повідомлень і публікації мультимедіа.

Метою роботи є розробка вебзастосунку соціальної мережі для публікації творчих робіт, зокрема бази даних, серверної та клієнтської частин.

Для досягнення мети необхідно вирішити такі завдання:

– провести аналіз існуючих соціальних мереж, їх функцій, елементів, принципів роботи;

– освоїти технології, необхідні для реалізації;

– створити базу даних для збереження даних;

– реалізувати серверну частину, яка буде мати зв’язок із БД і клієнтською частиною;

– реалізувати клієнтську частину, яка буде використовуватися для зв’язку із сервером;

– провести тестування готового застосунку.

2 МОДЕЛЮВАННЯ КОМПОНЕНТІВ СОЦІАЛЬНОЇ МЕРЕЖІ

2.1 Перелік функцій створюваної соціальної мережі

Вебзастосунок соціальної мережі, як й інші вебзастосунки, буде складатися із бази даних, серверної та клієнтської частин. Ці компоненти будуть реалізовані задля виконання таких функцій застосунку:

- автентифікація користувача (реєстрація та логін);
- створення та редагування профіля користувача;
- публікація фото у профілі та групах;
- редагування своїх постів і перегляд постів інших користувачів;
- взаємодія із фото за допомогою реакцій та коментарів;
- слідкування за іншими користувачами;
- створення та приєднання до груп користувачів;
- спілкування користувачів у реальному часі (отримання та надсилання текстових повідомлень).

2.2 Архітектура соціальної мережі як мікросервісної системи

Соціальна мережа буде складатися із трьох основних компонентів: бази даних, сервера та клієнта, що будуть мікросервісами. Це підхід до розробки застосунків, при якому програма складається з невеликих, незалежних компонентів, кожен з яких є окремим сервісом. Переваги мікросервісів такі:

- мікросервіси дозволяють розбити програму на невеликі, незалежні компоненти, які можуть бути розгорнуті та масштабовані незалежно один від одного;
- кожен мікросервіс працює у своєму власному контексті, що забезпечує ізоляцію від інших сервісів. Це дозволяє уникнути проблем,

пов’язаних з монолітними програмами, такими як взаємні залежності та конфлікти версій;

– мікросервісна архітектура дозволяє командам розробників працювати над невеликими, самостійними сервісами, що сприяє прискореній розробці, тестуванню та постачанню нового функціоналу;

– завдяки тому, що мікросервіси працюють незалежно, відмова одного сервісу не призводить до відмови всієї системи. Це дозволяє забезпечити більш високий рівень доступності та надійності програми;

– кожен мікросервіс може використовувати різні технології, мови програмування та фреймворки залежно від потреб. Це дозволяє вибирати найкращі інструменти для конкретного завдання та уникати прив’язки до певних технологій.

У даному випадку сервер та клієнт можуть бути розглянуті як окремі мікросервіси, оскільки вони можуть бути розгорнуті та масштабовані незалежно один від одного. Сервер обробляє запити від клієнтів та взаємодіє з базою даних для отримання, зміни та збереження даних. Він буде надавати API для взаємодії з клієнтською частиною програми, обробляти бізнес-логіку та керувати безпекою та автентифікацією. Клієнтом буде програма, за допомогою якого користувачі взаємодіють із соціальною мережею. Клієнт надсилає запити на сервер для отримання даних або виконання певних дій, а потім відображає ці дані користувачеві у зручному вигляді [17, 18].

2.3 Структура бази даних та модель даних

2.3.1 Сутності і атрибути

У базі даних будуть зберігатися дані про користувачів, створений контент, взаємодії тощо. Для розробки бази даних програми необхідно визначити, які сутності будуть використовуватися в роботі. Сутність – це

об’єкт, який є одиницею даних, що має унікальну ідентифікацію та зберігається в базі даних. Сутності в базі даних перелічені у таблиці 2.1.

Таблиця 2.1 – Перелік сутностей бази даних соціальної мережі

Назва сутності	**Пояснення**	**Опис сутності**
User	Користувачі	Користувач соціальної мережі
Role	Ролі	Роль користувача, призначена для визначення його можливостей (звичайний користувач або адміністратор)
Post	Публікації	Публікація, створювана користувачем
Like	Реакції	Реакція на публікацію
Comment	Коментарі	Коментар до публікації
Club	Групи	Група, до якої можуть приєднуватися користувачі
Dialog	Діалоги (чати)	Діалог (чат) між користувачами
Message	Повідомлення	Повідомлення у чаті
Banned User	Заблоковані користувачі	Користувач, котрого заблокував адміністратор (наприклад, порушник правил)
User Role	Ролі користувачів	Сутність для зв’язку між сутностями User та Role, що мають зв’язок M:N
User Connection	Зв’язки користувачів	Сутність для позначення зв’язку (слідкування) між користувачами
Club Member	Учасники групи	Сутність для зв’язку між сутностями User та Club, що мають зв’язок M:N

Користувач обов’язково має роль, може створювати багато груп і публікацій (у своєму профілі або у групі), залишати багато реакцій, коментарів, слідкувати за іншими користувачами та групами, може бути заблокований або розблокований, відправляти повідомлення будь-яким користувачам. Коментарі й реакції обов’язково мають публікацію, під якою вони були залишені. Також вони та повідомлення мають відправника, повідомлення на додачу мають отримувача та діалог, у котрому були залишені. Групи можуть мати пости та користувачів, що за ними слідкують, а

можуть бути пустими. Для забезпечення коректного збереження даних і загалом роботи програми потрібно розробити правильну структуру таблиць.

На рисунку 2.1 зображений опис структури таблиць.

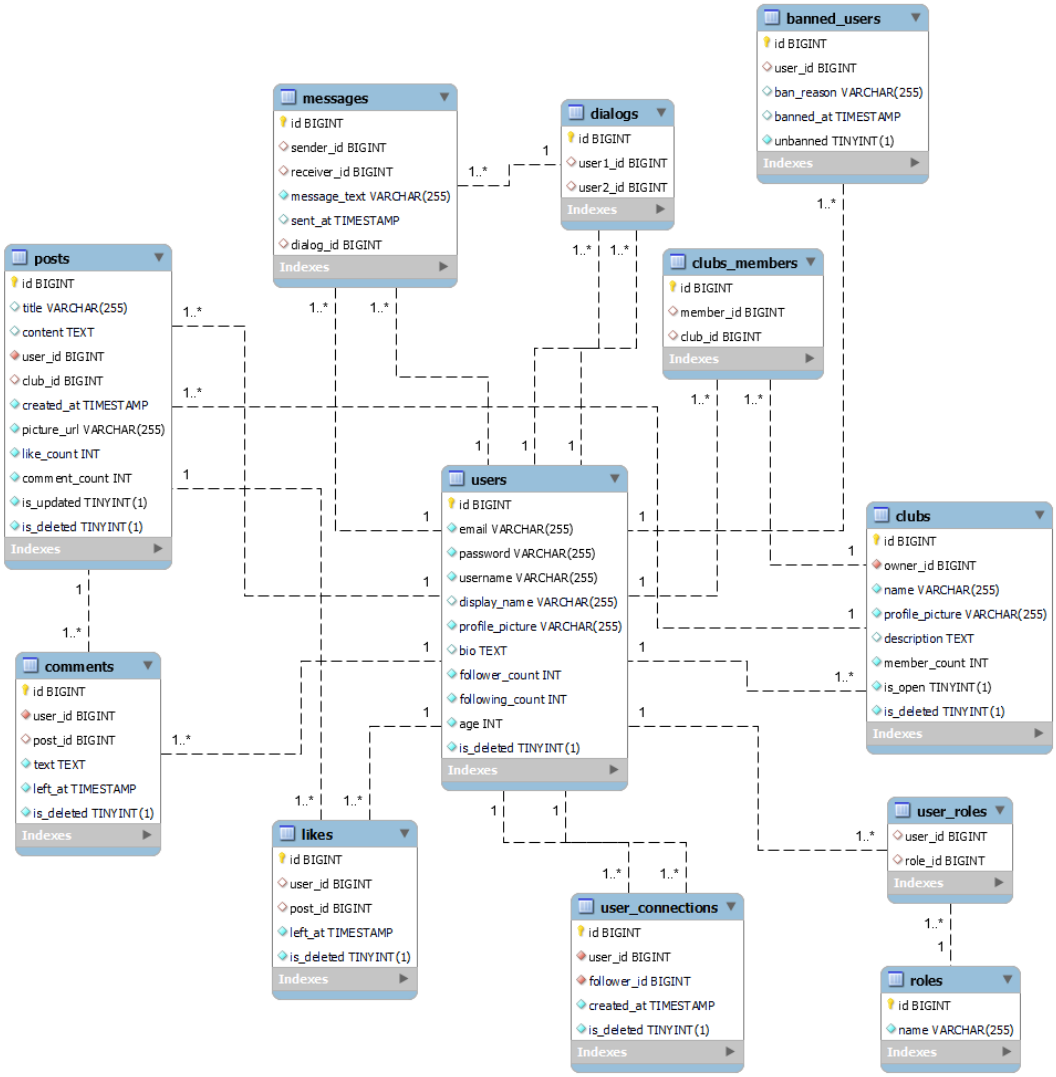

Рисунок 2.1 – Опис структури таблиць

2.3.2 Опис стовпців таблиць

Отже, коли спосіб створення таблиць обрано, визначаються стовпці таблиць. Перш за все, соціальна мережа без користувачів не має сенсу існувати, тож вони повинні мати можливість реєструватися та входити в обліковий запис. Для автентифікації вони будуть використовувати пошту та

пароль, тож до таблиці User буде додано email і password. Також всі записи матимуть id.

Користувач має профіль, у котрому є вся інформація про нього: username, за яким інші користувачі можуть його знаходити, display_name для позначення того, як звертатися до користувача (він може не заповнювати це поле, тоді воно буде однаковим із username), bio – коротка інформація про користувача, profile_picture – фото профілю, age – вік. Follower_count та following_count – кількість користувачів, що слідкують за користувачем, і кількість тих, за ким він сам слідкує відповідно.

Користувач має роль. Вона визначає перелік дій, які йому доступні. Ролі зберігатимуться в окремій таблиці roles із стовпцями id і name. Так як безліч користувачів можуть мати одну роль, і один користувач може мати декілька ролей, створюється таблиця user_roles із стовпцями user_id і role_id.

Для реалізації шаблону проєктування Soft Delete також додається стовпець is_deleted. Цей шаблон дозволяє не видаляти дані з таблиці, а помічати їх як видалені. Це запобігає випадковому видаленню даних, збереженню даних для проведення статистики, в даному випадку, дає можливість користувачу видалити свій обліковий запис і відновити його, якщо він цього захоче. Надалі цей шаблон, а також стовпець id для зручної ідентифікації запису будуть використовуватися й у наступних таблицях.

Користувач може слідкувати за іншими користувачами задля перегляду їхніх публікацій. Для цього створюється таблиця user_connections зі стовпцями user_id і follower_id, які є зовнішніми ключами та посилаються на id таблиці users. user_id – користувач, follower_id – той, що за ним слідкує.

Користувач може бути заблокованим, тоді він збережеться в таблиці banned_users. Зовнішній ключ user_id посилається на users, стовпець ban_reason – причина, з якої користувача заблоковано, banned_at – час, коли користувача заблоковано, unbanned – може приймати значення true, коли користувач стане розблокований, false – навпаки. Цей стовпець потрібен для статистики, чи є користувач схильним до порушення правил та блокування.

Користувач може створювати групи, до яких будуть заходити та приєднуватися інші. Вони зберігатимуться у таблиці clubs. Owner_id – зовнішній ключ на таблицю users, вказує на користувача, що створив групу, name, profile_picture, description, member_count – ім’я, фото групи, опис, кількість учасників групи відповідно. Is_open – статус групи, вказує, чи зараз можна слідкувати за групою.

Аналогічно до користувачів і ролей, дані про учасників груп зберігаються у таблиці club_members. У ній зовнішні ключі member_id і club_id посилаються на таблиці user і club відповідною.

Між собою користувачі можуть спілкуватися за допомогою повідомлень. Між ними створюється таблиця dialogs зі стовпцями user1_id та user2_id, що є зовнішніми ключами на таблицю users.

Користувачі обмінюються повідомленнями, що зберігаються у таблиці messages зі стовпцями sender_id, receiver_id (відправник та отримувач, зовнішні ключі на таблицю users), dialog_id (зовнішній ключ на таблицю dialog), message_text (текст повідомлення) і sent_at (час, коли повідомлення було надіслано).

Користувач може створювати публікації. Вони зберігаються в таблиці posts. Стовпець user_id – зовнішній ключ на таблицю user, користувач, що створив публікацію, club_id – зовнішній ключ на таблицю club, група, у якій була створена публікація (залежно від того, чи опубліковано її у профілі або у групі, може бути пустим), title, content – заголовок і текст публікації, created_at – час створення, picture_url – посилання на зображення у публікації на сервері, like_count, comment_count – кількість реакцій і коментарів під публікацією, is_updated – вказує на те, чи була публікація відредагована.

Таблиця likes зберігає реакції, містить user_id, post_id – зовнішні ключі на users і posts, і left_at – час залишення реакції.

Таблиця comments зберігає коментарі, містить user_id, post_id – зовнішні ключі на users і posts і left_at – час залишення реакції і text – текст коментаря.

2.3.3 Вибір інструментів для створення бази даних

Таблиці бази даних будуть створюватися за допомогою такого інструмента, як Liquibase.

Liquibase – це інструмент управління версіями бази даних, який дозволяє розробникам визначати та застосовувати зміни у структурі бази даних у вигляді міграційних файлів. Він забезпечує контроль версій схеми бази даних та автоматичне застосування міграцій під час запуску програми. Liquibase підтримує різні системи управління базами даних, включаючи обрану для реалізації БД MySQL, PostgreSQL, Oracle, SQL Server та інші. Має хорошу інтеграцію з Java та Spring Framework, що дозволяє використовувати його в Java-застосунках, включаючи Spring Boot.

Для створення таблиць буде обраний формат даних YAML (YAML Ain't Markup Language). Це зручний формат серіалізації даних, що легко може бути прочитаний людиною.

Для зручності код для створення кожної таблиці буде в окремому changelog файлі. Врешті-решт, після його написання, назви файлів додаються до файлу db.changelog-master.yaml. Цей файл використовується в Liquibase для визначення послідовності змін бази даних, які потрібно застосувати. Коли Liquibase запускається, він читає цей файл і послідовно виконує кожну зміну, описану в ньому, щоб оновити схему бази даних до потрібного стану. db.changelog-master.yaml містить посилання інші файли changelog, які містять самі зміни схеми бази даних.

2.4 Моделювання серверної частини

Серверна частина (backend) – це частина вебзастосунку, яка обробляє запити від клієнтської частини (frontend) і керує взаємодією з базою даних, а також іншими зовнішніми сервісами або системами.

Основні функції серверної частини включають:

– реалізацію бізнес-логіки. Вона визначає, як застосунок повинен взаємодіяти з даними та які дії слід виконувати у відповідь на різні запити;

– обробку запитів. Сервер отримує HTTP-запити від клієнтської частини та обробляє їх відповідно до бізнес-логіки програми;

– керування даними. Сервер взаємодіє з базою даних для отримання, зміни та збереження даних;

– автентифікацію та авторизацію, забезпечення безпеки програми, перевіряючи облікові дані користувачів, а також їх права доступу до різних ресурсів та функціональності;

– взаємодію із зовнішніми сервісами, такими як API сторонніх сервісів або внутрішні мікросервіси для виконання різних операцій [19, 20].

2.4.1 Вибір технологій для створення серверу

Для реалізацій сервера були обрані такі технології та інструменти: Maven, Spring Framework та Spring Boot, Spring WebSocket, Hibernate, MySQL Connector, Project Lombok, MapStruct, JWT, Liquibase.

Apache Maven – інструмент для управління проєктами та автоматизації збірки програмного забезпечення, заснований на концепції управління залежностями. Він потрібен для спрощення збірки Java-проєктів, управління залежностями, виконання завдань, пов’язаних з розробкою, тестуванням та розгортанням програм. Maven використовує конфігураційні файли у форматі XML з назвою pom.xml для опису проєкту та його залежностей.

Spring Framework – це фреймворк для розробки застосунків на Java. Він надає комплексний набір інструментів та функціональність для спрощення створення різних типів програм, включаючи вебзастосунки, мікросервіси, програми для обробки даних та багато іншого.

Spring Boot – це фреймворк на основі Spring Framework. Він надає набір інструментів для швидкого розгортання та налаштування програм з мінімальними зусиллями з боку розробника. Має вбудовані сервери застосунків (у даному випадку, Tomcat). Додані до файлу pom.xml залежності spring-boot-starter, spring-boot-starter-data-jpa, spring-boot-starter-web, spring-boot-starter-security є наборфми для швидкого створення Spring Boot програм із доступом до БД за допомогою JPA, а також забезпечення безпеки.

Spring WebSocket – модуль Spring Framework, що надає підтримку для розробки програм, які використовують протокол WebSocket для двостороннього зв'язку між клієнтом та сервером. Протокол WebSocket дозволяє встановити постійне з'єднання між клієнтом та сервером, що забезпечує обмін повідомленнями в реальному часі. До pom.xml додається залежність spring-boot-starter-websocket.

MySQL Connector надає JDBC-драйвер для підключення до бази даних MySQL (залежність mysql-connector-java).

Project Lombok – це бібліотека для Java, яка допомагає зменшити кількість шаблонного коду завдяки використанню анотацій. Вона надає інструкції для автоматичної генерації методів, таких як гетери, сетери, конструктори, методи toString(), equals() та hashCode(), що спрощує та прискорює процес розробки. Однією з ключових особливостей Lombok є те, що вона додається в проєкт як залежність збірки, а не як залежність під час виконання, що робить її ідеальним інструментом для використання в середовищах розробки без додавання додаткового коду в підсумковий JAR (Java ARchive) файл. До pom.xml додається залежність lombok.

MapStruct – фреймворк Java для генерації коду, що спрощує перетворення одних об'єктів в інші. Він надає інструкції для опису мапінгів між класами та генерує відповідний код під час компіляції. За допомогою MapStruct визначається інтерфейс з методами перетворення одного класу в інший, а потім генерується реалізація цих методів на основі анотацій, які використовуються для вказівки правил перетворення. Це робить процес

перетворення більш зрозумілим і безпечним, покращує продуктивність. В даному випадку, MapStruct буде використовуватися для перетворень між DTO та сутностями [21]. До pom.xml додається залежність mapstruct.

Залежності jjwt-api, jjwt-impl, jjwt-jackson використовуватимуться для роботи з JWT для автентифікації й авторизації користувачів у вебзастосунку.

Для підключення Liquibase використовується залежність liquibase-core.

2.4.2 Компоненти серверу

Компонентами архітектури серверу будуть класи сутностей, репозиторії, сервіси, контролери, інші допоміжні класи тощо.

Сутності представляють об’єкти системи, які зберігаються в базі даних. Кожна сутність зазвичай відповідає таблиці у базі даних і має атрибути, що описують її стан. Сутності зазвичай використовуються разом із репозиторіями для виконання операцій CRUD (Create, Read, Update, Delete) з даними. Вони надають зручний інтерфейс для взаємодії з базою даних та забезпечують цілісність даних у застосунку. У даному застосунку сутностями будуть User (таблиця users), Role (таблиця roles), BannedUser (таблиця banned_users), UserConnection (таблиця user_connections), Club (таблиця clubs), Post (таблиця posts), Like (таблиця likes), Comment (таблиця comments), Dialog (таблиця dialogs), Message (таблиця messages) [22].

Репозиторії грають ключову роль у контексті архітектури, заснованої на шаблоні проєктування DAO (Data Access Object). Репозиторії забезпечують єдиний інтерфейс доступу до даних із бази даних. Вони інкапсулюють специфічні деталі взаємодії з базою даних, такі як створення SQL-запитів, використання JPA. Репозиторії дозволяють уникнути дублювання коду, пов’язаного з доступом до даних, оскільки вони надають стандартні методи виконання операцій CRUD (Create, Read, Update, Delete) сутностей. Вони надають абстракцію доступу до даних, дозволяючи сервісам

зосереджуватись на бізнес-логіці без необхідності дбати про деталі реалізації доступу до даних [23].

У застосунку будуть інтерфейси репозиторіїв для кожної сутності. Вони будуть розширяти інтерфейс JpaRepository. JpaRepository є частиною Spring Data JPA та надає зручний спосіб взаємодії з базою даних у програмах, що використовують JPA для роботи з об'єктно-реляційною моделлю даних.

JpaRepository надає можливість без необхідності написання власних SQL-запитів використовувати стандартні методи для створення, читання, оновлення та видалення сутностей (save(), findById(), findAll(), deleteById() тощо) і створювати методи пошуку сутностей за заданими критеріями без явного написання SQL-запитів (наприклад, якщо User має поле username, findByUsername(String username) – метод для пошуку користувачів на ім'я користувача). JpaRepository дозволяє легко організовувати результати запитів із підтримкою пагінації та сортування. Це дуже корисно для соціальної мережі, бо у ній зазвичай є великі обсяги даних, що заради покращення продуктивності потрібно розбивати на сторінки. Є можливість визначати іменовані запити в репозиторії за допомогою анотації @Query та JPQL. Це дозволяє писати власні SQL запити для виконання складних операцій, які не можуть бути виконані за допомогою стандартних методів. JpaRepository автоматично керує транзакціями під час операцій з базою даних, що забезпечує узгодженість даних та цілісність транзакцій.

Сервіси в архітектурі застосунків відіграють важливу роль у забезпеченні поділу відповідальності та управлінні бізнес-логікою. Вони допомагають розділити код на логічні блоки, кожен із яких відповідає за певний функціонал. Це покращує підтримуваність та масштабованість коду. Сервіси містять реалізацію бізнес-логіки програми, обробляють запити від контролерів, виконують необхідні операції з даними та повертають результати, приховують деталі доступу до даних та інкапсулювати їх у собі. Можуть виконувати валідацію вхідних запитів, перевіряючи їх на відповідність бізнес-правилам та вимогам безпеки. Вони також можуть

обробляти та перетворювати дані, отримані від клієнтів, у формат, зрозумілий для інших компонентів програми [24].

Контролери виконують ключову роль у взаємодії з клієнтською частиною програми та управлінні потоком даних. Вони приймають HTTP-запити від клієнта та визначають, як програма реагуватиме на ці запити. Після отримання запиту контролер викликає методи сервісів для обробки бізнес-логіки, взаємодії з базою даних. Після обробки запиту контролер генерує відповідь, яка може бути відображена на стороні клієнта. Це може бути HTML-сторінка, JSON-об’єкт або будь-який інший формат даних, який передається клієнту через протокол HTTP [25].

У даному випадку будуть використовуватися REST-контролери. Вони надають відповіді у вигляді даних у форматі JSON відповідають на запити згідно архітектурного стилю REST. Основні принципи REST включають використання уніфікованих ідентифікаторів ресурсів (URI), використання стандартних методів HTTP (GET, POST, PUT, DELETE) для взаємодії з ресурсами, а також stateless обмін повідомленнями.

2.4.3 Принцип Stateless

Stateless – це принцип архітектури, при якому кожен запит до сервера містить всю необхідну інформацію для сервера для розуміння та обробки цього запиту. Сервер не зберігає жодного стану про клієнта між послідовними запитами. Кожен запит вважається незалежним від попередніх запитів, містить усю необхідну інформацію, щоб сервер міг обробити його, що дозволяє серверу бути нечутливим до стану клієнта або попередніх запитів. Так як сервер не зберігає стан клієнта, можна легко розподіляти навантаження між різними серверами без необхідності синхронізації стану. Це особливо корисно для соціальної мережі, якою можуть користуватися тисячі або навіть мільйони користувачів. Такий підхід дозволяє уникнути

проблем, пов’язаних із зберіганням стану на сервері, таких як змішаний стан або конфлікти синхронізації, що може підвищити надійність системи [26].

2.4.4 Автентифікація з використанням JWT токенів

Stateless автентифікація буде використовувати маркери, у даному випадку, JSON Web Token, котрі містять інформацію про користувача. JWT – це рядок, який складається з трьох серіалізованих з Base64 частин, розділених крапками. Розшифрований токен містить два JSON-рядки – заголовок (header) з навантаженням (payload) та підпис (signature). Заголовок містить метадані про тип токену та алгоритм шифрування, який використовується для підпису токену. Тіло містить корисну інформацію, яка пов’язана з токеном, таку як ідентифікатор користувача, термін дії токену, ролі користувача тощо. Підпис генерується на основі заголовка, тіла та секретного ключа за допомогою вказаного алгоритму шифрування. Він дозволяє перевірити цілісність та автентичність токену. JWT розшифровується на стороні сервера, який має доступ до секретного ключа, використовуючи алгоритм шифрування, вказаний у заголовку. Після декодування токену сервер може перевірити його цілісність та здійснити необхідні дії відповідно до інформації в тілі токену, наприклад, автентифікувати користувача чи надати доступ до певних ресурсів [27].

У даному випадку буде використовуватися алгоритм шифрування HMAC-SHA (Keyed-Hash Message Authentication Code with Secure Hash Algorithm). Це криптографічний алгоритм, що використовується для обчислення хеш-коду (підпису) повідомлення за допомогою секретного ключа. В алгоритмі використовується безпечна хеш-функція SHA (Secure Hash Algorithm), а ключ використовується для забезпечення конфіденційності та цілісності даних.

Буде написаний клас JwtUtil для формування JWT токену. Необхідний секретний ключ (secret), що має залишатися конфіденційним і бути достатньо надійним, щоб протистояти атакам brute force методом. Під час створення екземпляру JwtUtil, цей ключ ініціалізується рядком також із файлу application.properties. Цей рядок перетворюється у ключ за допомогою методу hmacShaKeyFor класу Keys.

JwtUtil використовуватиметься у класі JwtAuthenticationFilter, що буде перевіряти наявність і валідацію JWT в заголовку запиту Authorization. Якщо токен є валідним, фільтр встановить автентифікацію для користувача, здійснюючи пошук користувача за іменем, яке міститься в токені. Після цього фільтр передає обробку запиту наступному етапу у ланцюжку фільтрів.

2.4.5 Налаштування безпеки

Безпека буде налаштована за допомогою потужного фреймворк для захисту вебзастосунків Spring Security. Він надає широкий спектр функцій для автентифікації, авторизації та управління безпекою застосунків.

Буде створено конфігураційний клас SecurityConfig, що міститиме налаштування для фільтрів безпеки та доступу до ресурсів. У його методі securityFilterChain встановлюються CORS правила для дозволу віддаленого доступу до ресурсів з інших доменів. Це дозволить клієнтській частині надсилати запити на сервер. Далі налаштовується авторизація запитів за допомогою HttpSecurity, де вказуються правила доступу до різних кінцевих точок (URL-адрес, за якими можна отримати доступ до певних функцій застосунку). Неавтентифіковані користувачі матимуть доступ до реєстрації та логіну, а усі інші кінцеві точкин доступні лише автентифікованим користувачам. Далі встановлюється stateless сесії, оскільки використовується JWT для аутентифікації та додається фільтр JwtAuthenticationFilter для перевірки та обробки JWT токенів для автентифікації [28].

Також у SecurityConfig конфігурується AuthenticationManager для обробки запитів на автентифікацію.

2.4.6 WebSocket для миттєвого обміну повідомленнями

WebSocket – це протокол зв’язку між клієнтом і сервером, який дозволяє встановити постійне двостороннє з’єднання. Основна його відмінність від HTTP в тому, що HTTP працює на засаді «запит-відповідь», тобто клієнт посилає запит, сервер обробляє його і надсилає відповідь, після чого з’єднання закривається. У випадку WebSocket воно відкривається один раз, обидва боки надсилають дані один одному в реальному часі без необхідності постійно посилати запити, отримувати відповіді.

У чатах WebSocket дуже корисний, оскільки він забезпечує миттєву доставку повідомлень між учасниками. Традиційний підхід на основі HTTP може вимагати постійних запитів клієнта до сервера для перевірки нових повідомлень або оновлень, що може призвести до затримок та перевантаження сервера. Використання WebSocket дозволяє уникнути цих проблем, оскільки з’єднання лишається активним, і сервер може негайно надсилати нові повідомлення клієнтам, що забезпечує швидку та ефективну комунікацію без перезавантаження сторінки.

Створюється конфігураційний клас WebSocketConfig, де реєструються кінцеві точки WebSocket, визначається, які з них будуть доступні для клієнтів. У даному випадку, «/ws» буде шляхом для WebSocket-з’єднання, яке буде доступне для клієнтів за адресою клієнтської частини. Визначається, які маршрути повідомлень будуть використовуватися для внутрішнього обміну повідомленнями між сервером та клієнтом. Повідомлення, які надсилаються клієнтам, будуть маршрутизовані через тему «/topic» [29, 30].

2.5 Моделювання клієнтської частини

2.5.1 Вибір технологій для створення клієнтської частини

Клієнтська частина відповідає за відображення інтерфейсу користувача та взаємодію з сервером. Для її реалізації обрано бібліотеки React – інструмент для створення швидких, ефективних й інтерактивних вебзастосунків, Axios для здійснення HTTP-запитів з JavaScript, котра є простою у використанні та зручно обробляє помилки, і StompJs, що дозволяє легко підключатися до WebSocket-серверів за допомогою протоколу STOMP (Simple Text Oriented Messaging Protocol) і надсилати повідомлення між клієнтом і сервером [31].

2.5.2 Структура вебсайту

При аналізі успішних соціальних мереж виявлено, яка структура та дизайн вебсторінок є найбільш ефективними для забезпечення комфортного користувацького досвіду. Відповідно до цього розроблені макети вебсторінок (рис. 2.2, 2.3).

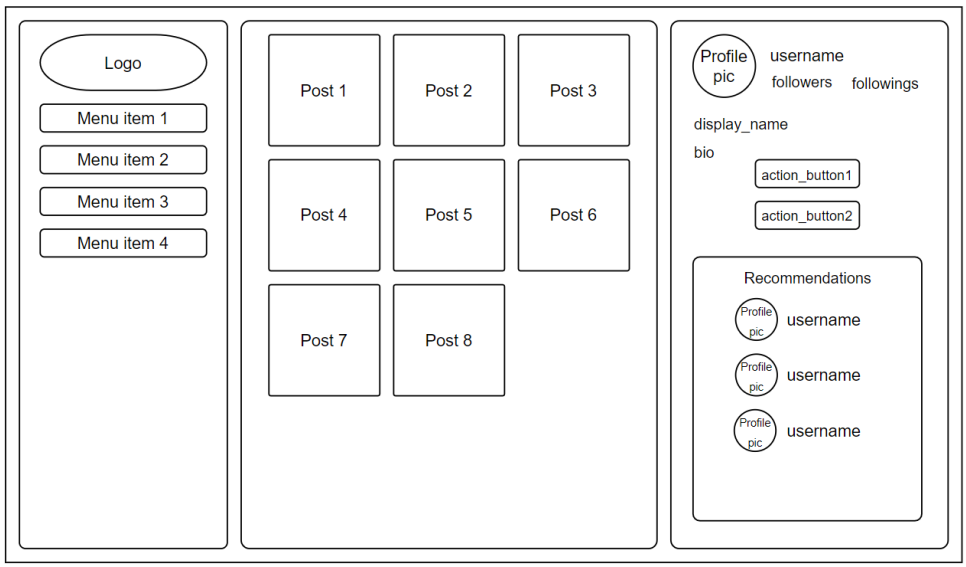

Рисунок 2.2 – Макет сторінки профілю користувача

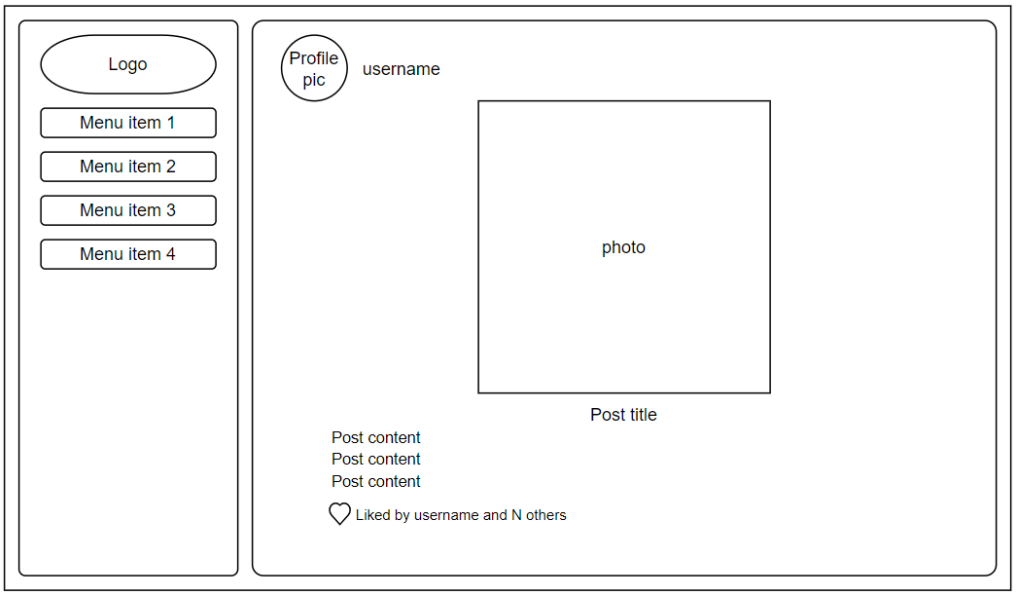

Рисунок 2.3 – Макет сторінки публікації

2.5.3 Створення UI з допомогою React

Для створення сторінок з React визначається їхні структура та компоненти. Це включає створення компонентів для різних частин сторінки. Розмітка сторінок створюється з використанням JSX, де компоненти розміщуються так, щоб відображатися в потрібному порядку та місці. Для керування даними та станом на сторінці використовується хук useState. Це дозволяє компонентам React оновлювати свій стан і перерисовуватись при зміні даних. Після цього додаються обробники подій та логіка взаємодії (обробники натискання кнопок, надсилання запитів на сервер). Застосовуються стилі до сторінки за допомогою CSS, щоб зробити сторінку красивою та зручною для користувачів [32].

Для соціальної мережі створюються сторінки профілю та груп, де відображаються інформація про них і публікації, сторінки пошуку користувачів і груп, створення, редагування публікацій та іншої інформації, сторінки публікацій із фото, текстом, реакціями й коментарями, сторінки відображення повідомлень між користувачами тощо.

3 РЕАЛІЗАЦІЯ ВЕБЗАСТОСУНКУ СОЦІАЛЬНОЇ МЕРЕЖІ

3.1 Середовища розробки

В якості програмного забезпечення для створення соціальної мережі було обрано IntelliJ IDEA Community Edition для розробки серверу та Visual Studio Code для розробки клієнтської частини. Для тестування роботи серверу використовувався застосунок Postman.

3.1.1 IntelliJ IDEA Community Edition

IntelliJ IDEA Community Edition – це безкоштовне IDE (інтегроване середовище розробки), що призначене для роботи із деякими мовами програмування, включаючи Java. Воно має широкий набір функцій, серед яких виявлення помилок та надання варіантів виправлення, автодоповнення коду з огляду на контекст, інструменти рефакторингу, інтегрована система збірки, підтримка фреймворків і бібліотек, що полегшує розробку на різних технологічних стеках, тощо. Має зручний зрозумілий дизайн, широкі можливості налаштування під потреби конкретного розробника, інтегрується із системи збірки (а саме Maven, що використовується у роботі) [33].

3.1.2 Visual Studio Code

Visual Studio Code – це безкоштовний та потужний редактор коду, розроблений компанією Microsoft. Має невеликий розмір та низьке споживання ресурсів, що робить його швидким та ефективним, підтримує роботу з величезною кількістю мов програмування, однією з яких є JS, мова розмітки HTML та стилізації вебсторінок CSS, має велику екосистему

розширень, що дозволяє додавати йому функціональність, автодоповнення коду, рефакторинг [34].

3.1.3 Postman

Postman – це інструмент для тестування API. Він дозволяє розробникам створювати, тестувати, документувати та спілкуватися з різними типами API. За допомогою нього можна налаштовувати різні типи запитів (GET, POST, PUT, DELETE тощо), додавати параметри, заголовки, тіло запиту та перевіряти відповіді сервера. Має інтуїтивно зрозумілий та легкий у використанні інтерфейс, що дозволяє швидко створювати та виконувати HTTP-запити без необхідності написання власного коду, дозволяє створювати тестові набори, які автоматично виконують послідовність запитів та перевіряють, чи повернув сервер очікувану відповідь, підтримує роботу зі змінними, середовищами, можливість імпорту та експорту колекцій, автоматичну генерацію документації API та багато іншого [35].

3.2 Реалізація вебзастосунку

3.2.1 Типи користувачів

Є три типи користувачів: анонімний користувач, авторизований користувач, адміністратор. Анонімному користувачу дозволений доступ лише до сторінок реєстрації та входу в обліковий запис. Авторизованому користувачу є доступ до основного функціоналу – редагування профілю, публікації фото, спілкування, створення груп тощо. Адміністратор, окрім перелічених дій, може блокувати та розблоковувати користувачів, до прикладу, тих, що порушують правила платформи.

3.2.2 Реалізація серверу

За допомогою IntelliJ IDEA створено проєкт із усіма необхідними залежностями.

Створено 10 сутностей: User, Role, BannedUser, UserConnection, Club, Post, Like, Comment, Dialog і Message.

Створено інтерфейси маперів UserMapper, ClubMapper, PostMapper, MessageMapper, LikeMapper, EntityConversionService, CommentMapper, ClubMapper для перетворення об’єктів на DTO та навпаки для того, щоб не надсилати клієнту зайву інформацію. Також створено необхідні записи (records) для DTO. Приклад одного з інтерфейсів маперів наведено в лістингу 3.1.

Лістинг 3.1 Інтерфейс маперу повідомлень:

```
@Mapper (config = MapperConfig.class, uses = {EntityConversionService.class})
public interface CommentMapper {
@Mapping(source = "user.id", target = "userId")
@Mapping(source = "post.id", target = "postId")
@Mapping(source = "text", target = "text")
@Named("commentsToDto")
CommentDto toDto(Comment comment);

@Mapping(source = "userId", target = "user", qualifiedByName = "userFromId")
@Mapping(source = "postId", target = "post", qualifiedByName = "postFromId")
Comment toModel(CommentRequestDto commentDto);
}
```

Створено інтерфейси репозиторіїв для доступу до даних із БД: UserRepository, RoleRepository, BannedUserRepository, UserConnectionRepository, ClubRepository, PostRepository, LikeRepository, CommentRepository, DialogRepository, MessageRepository. У них перелічено сигнатури методів із анотацією @Query і запитами на мові JPQL для отримання необхідних даних. Приклади таких методів наведені у лістингах 3.2 і 3.3.

Лістинг 3.2 Метод пошуку користувача за частиною username з пагінацією:

```
@Query("SELECT u FROM User u JOIN u.roles r " +
    "WHERE r.name != 'ROLE_ADMIN' " +
    "AND LOWER(u.username) LIKE " +
    "CONCAT('%', LOWER(:username), '%')")
List<User> findByPartialUsername(String username, Pageable pageable);
```

Лістинг 3.3 Метод пошуку всіх повідомлень із діалогу з пагінацією:

```
@Query("SELECT m FROM Message m WHERE " +
    "m.dialog.id = :dialogId " +
    "ORDER BY m.sentAt DESC")
List<Message> findMessagesByDialogId(Long dialogId, Pageable
pageable);
```

Реалізовано інтерфейси сервісів та їх імплементації для роботи із репозиторіями та реалізації бізнес-логіки програми: UserService, AuthenticationService, UserConnectionService, PostService, LikeService, CommentService, MessageService, DialogService, ClubService, а також CustomUserDetailsService, що імплементує UserDetailsService із springframework.security, і FileUploadService для збереження зображень на

сервері. Він потрібен для надання файлу, отриманому з клієнта, унікального імені, що буде збережено до БД.

Створено контролери для взаємодії з клієнтською частиною:

– AuthenticationController: відповідає за реєстрацію та вхід користувачів до системи;

– ClubController: працює із запитами на створення, редагування, отримання інформації про групи;

– FeedController: відповідає за надання даних про стрічку нових публікацій від облікових записів у списку «Following» користувача);

– FollowersController: відповідає за отримання інформації про зв’язки користувачів, їх створення чи розірвання;

– ImageController: відповідає за надання зображень;

– MessageController: реалізує можливість спілкування користувачів у реальному часі;

– MessageRestController: відповідає за надання повідомлень і діалогів із БД;

– PostController: отримання інформації про публікації, їх редагування, створення, видалення;

– PostInteractionController: працює із запитами на залишення і отримання реакцій і коментарів на публікаціях;

– ProfileController: отримання інформації про профіль користувача, його редагування;

– SearchController: відповідає за обробку запитів на пошук користувачів та груп.

Створено конфігураційні класи MapperConfig, SecurityConfig, WebSocketConfig, класи для роботи із JWT – JwtUtil, JwtAuthenticationFilter.

Роботу застосунку протестовано за допомогою Postman.

Реалізовано простий алгоритм підбору контенту для користувачів – відображення всіх публікацій з облікових записів, за котрими стежить користувач за датою створення (найновіші згори).

Також реалізовано алгоритм рекомендацій користувачів, за якими можна стежити. Схематично приклад його роботи зображений на рисунку 3.1.

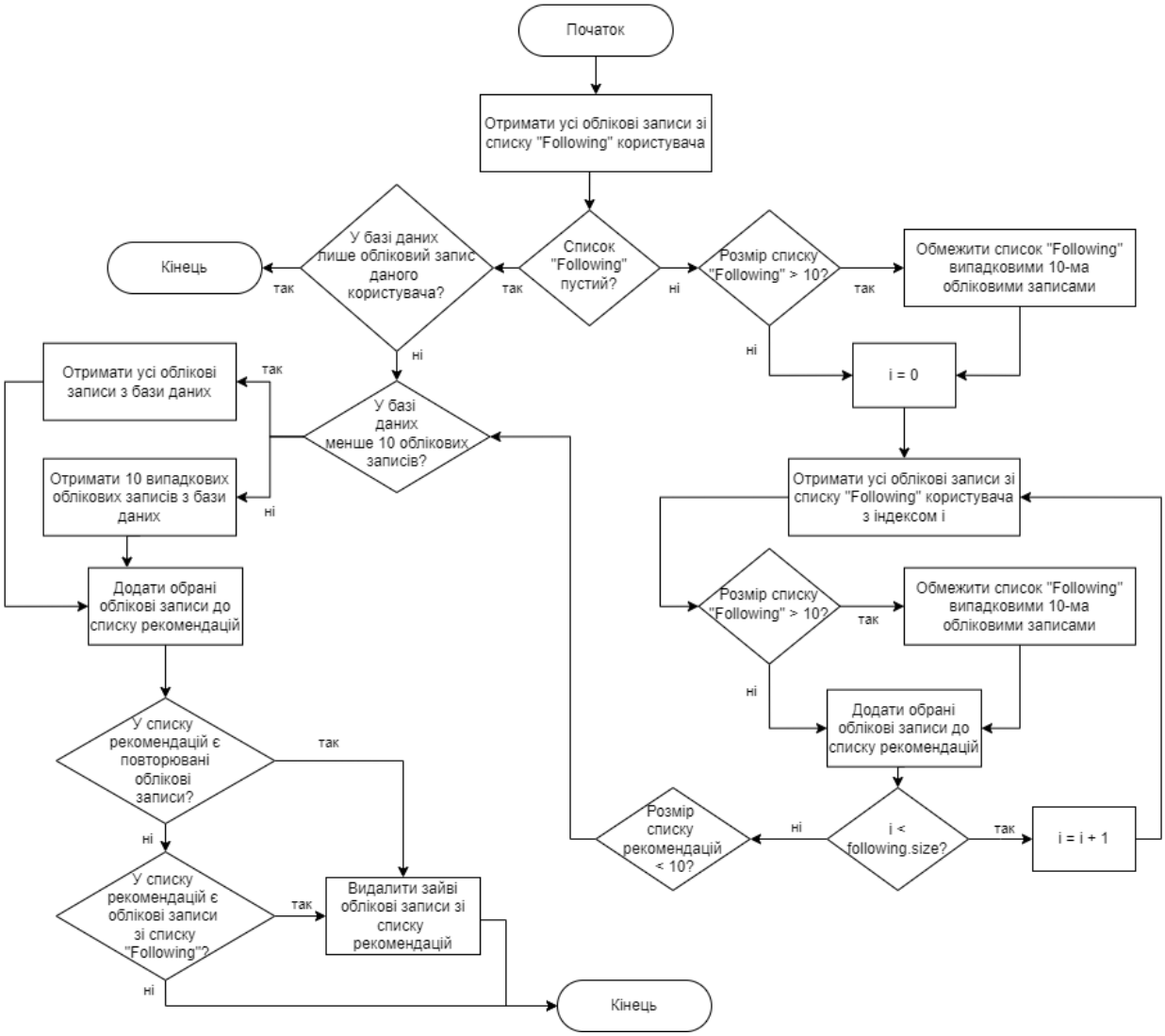

Рисунок 3.1 – Приклад роботи алгоритму рекомендацій користувачів

Алгоритм відбирає облікові записи, за якими стежать користувачі із списку «Following». Якщо у користувача в ньому недостатньо облікових записів для підбору, обираються випадкові облікові записи із всієї соціальної мережі. Алгоритм не пропонує ті, за якими користувач вже стежить [36].

3.2.1 Реалізація клієнта

За допомогою Visual Studio Code створено React-застосунок, у якому створюються всі необхідні компоненти для вебсторінок. Створено компоненти кнопок, меню, публікацій, модальні вікна (для відображення списків зв’язків між користувачами та облікових записів, що залишили реакції під публікаціями), повідомлень, форм тощо. Реалізовано сторінки входу в систему, реєстрації, профілю, перегляду публікацій і груп, їх створення і редагування, сторінки діалогів, пошуку, блокування користувачів – для адміністратора. Створено і додано стилі для елементів на сторінках.

3.3 Демонстрація роботи вебзастосунку соціальної мережі

3.3.1 Реєстрація та вхід

Неавторизований користувач може або зареєструватися (рис. 3.2), або увійти у свій профіль (рис. 3.3).

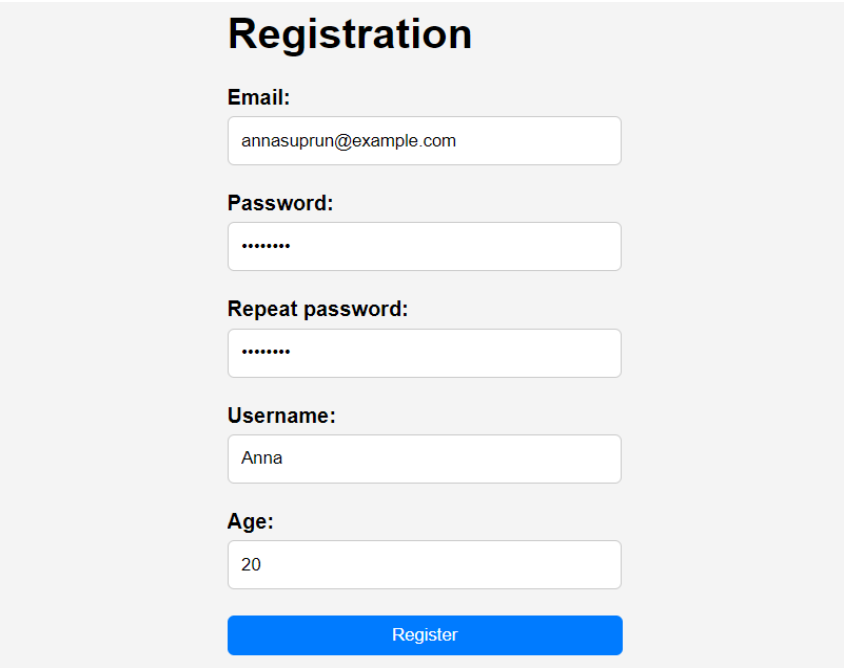

Рисунок 3.2 – Форма реєстрації

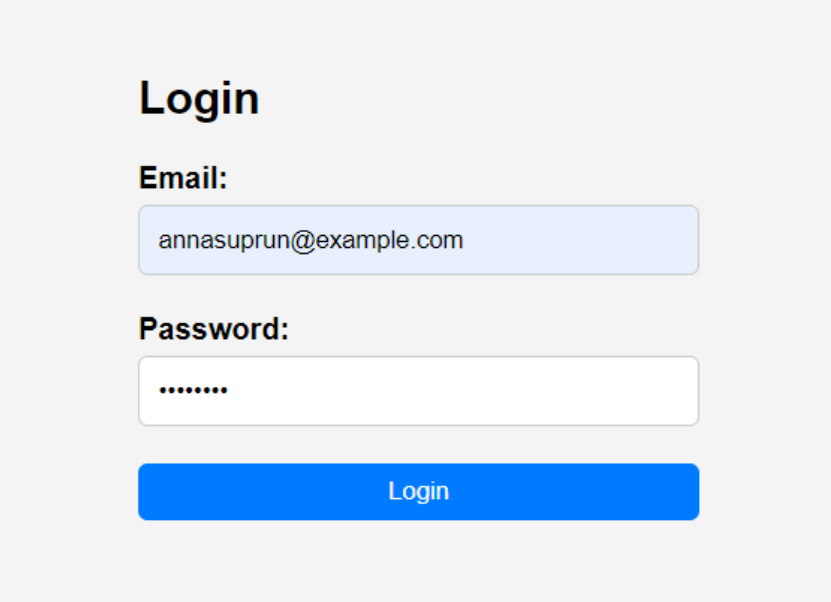

Рисунок 3.3 – Форма входу в систему

3.3.2 Профіль і рекомендації

Після реєстрації або входу користувач переходить на сторінку свого профілю (рис. 3.4). Поки він ще нічого не опублікував і не заповнив інформацію про себе, тому профіль пустий. Також користувач бачить рекомендації – інших творців із мережі, за якими він може слідкувати.

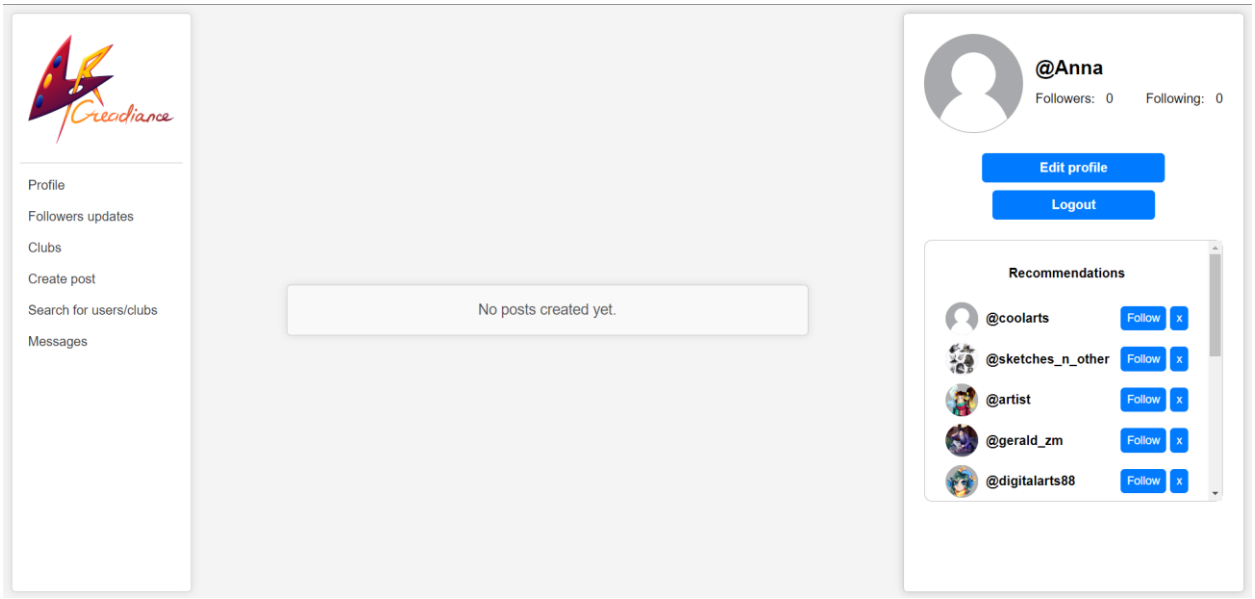

Рисунок 3.4 – Сторінка входу в систему

Перш за все, користувач може заповнити інформацію про себе, натиснувши кнопку «Edit profile». Він перейде на сторінку редагування профілю, де можна змінити ім’я, опис і фото (рис. 3.5).

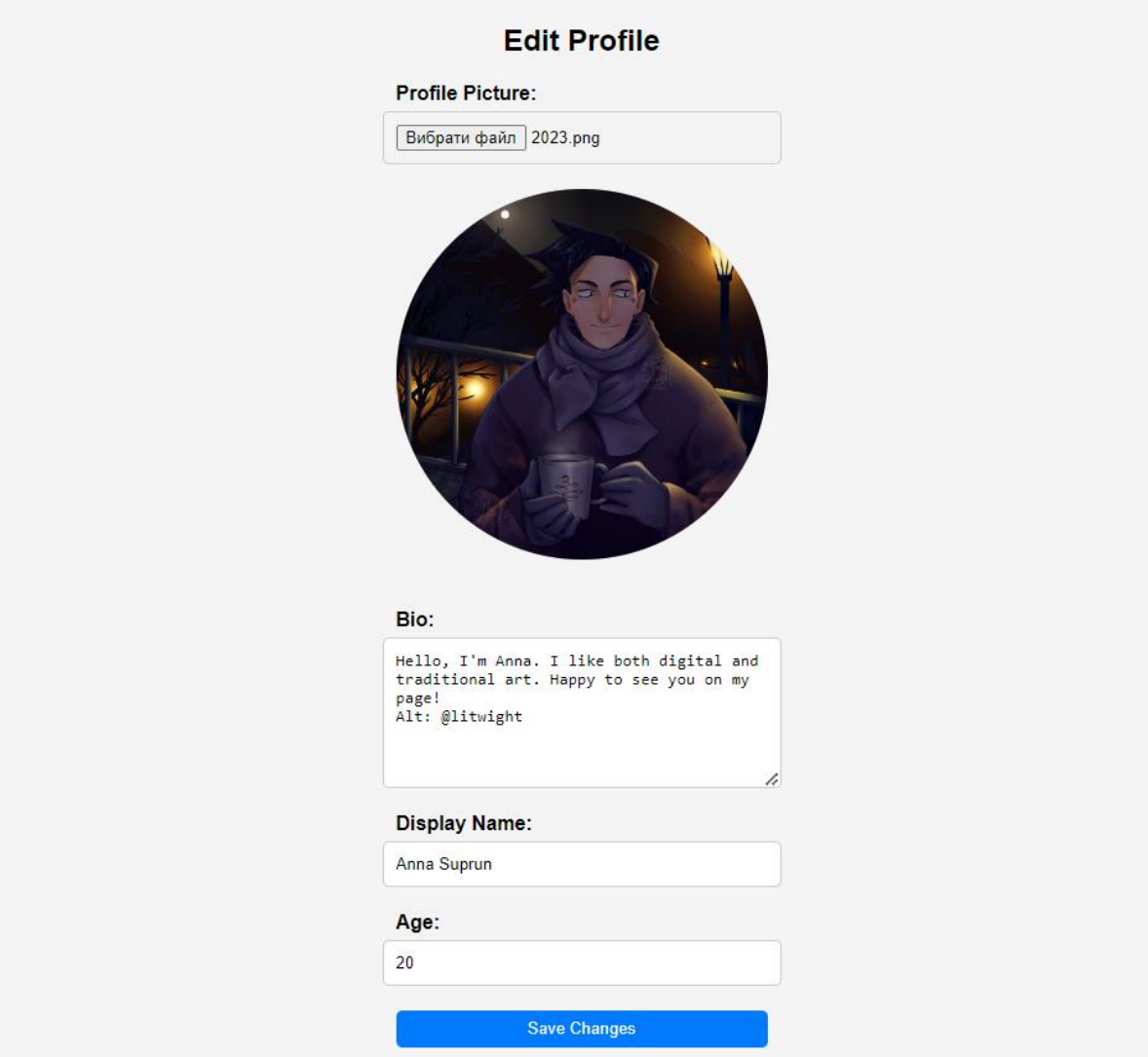

Рисунок 3.5 – Форма редагування профілю

Після редагування нова інформація відобразиться у профілі (рис. 3.6). Якщо користувач хоче створити посилання на іншого користувача у своєму профілі або у тексті чи заголовку публікації, він може написати його ім’я, перед ним поставивши символ «@». Тоді на це слово можна натиснути і перейти на сторінку іншого користувача.

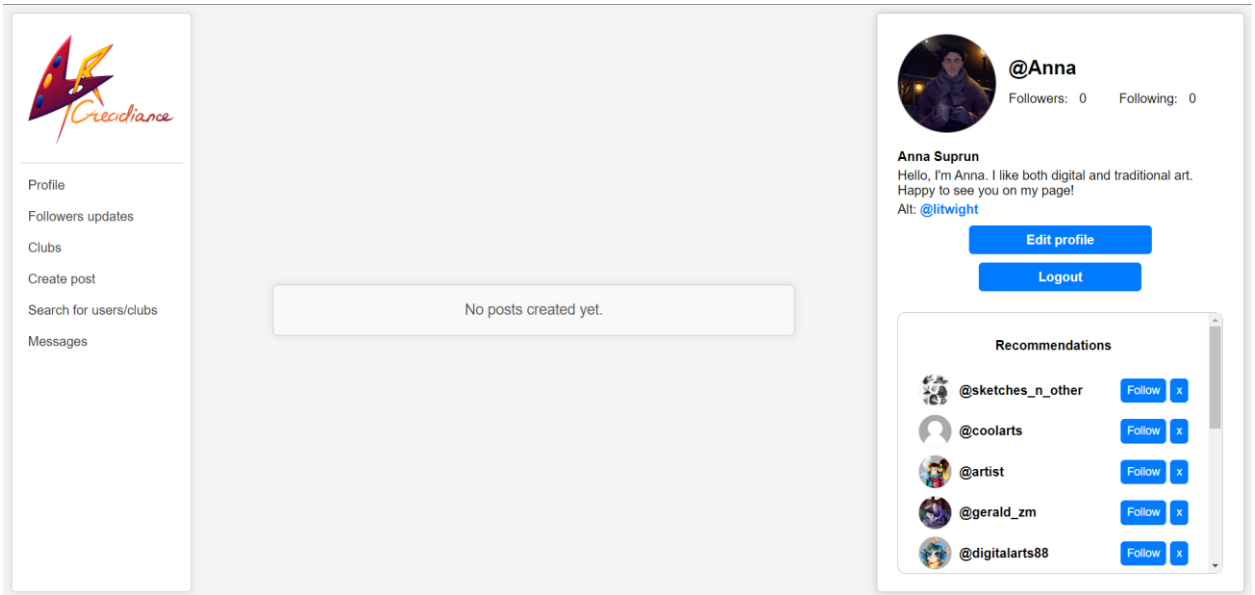

Рисунок 3.6 – Профіль після редагування

3.3.3 Слідкування за користувачами і обмін повідомленнями

Користувач може слідкувати за іншими зі списку рекомендацій, натиснувши на кнопку «Follow». Натиснувши на кнопку «Following», він може побачити список тих, за ким слідкує. Звідти ж може перестати слідкувати за ними (рис. 3.7).

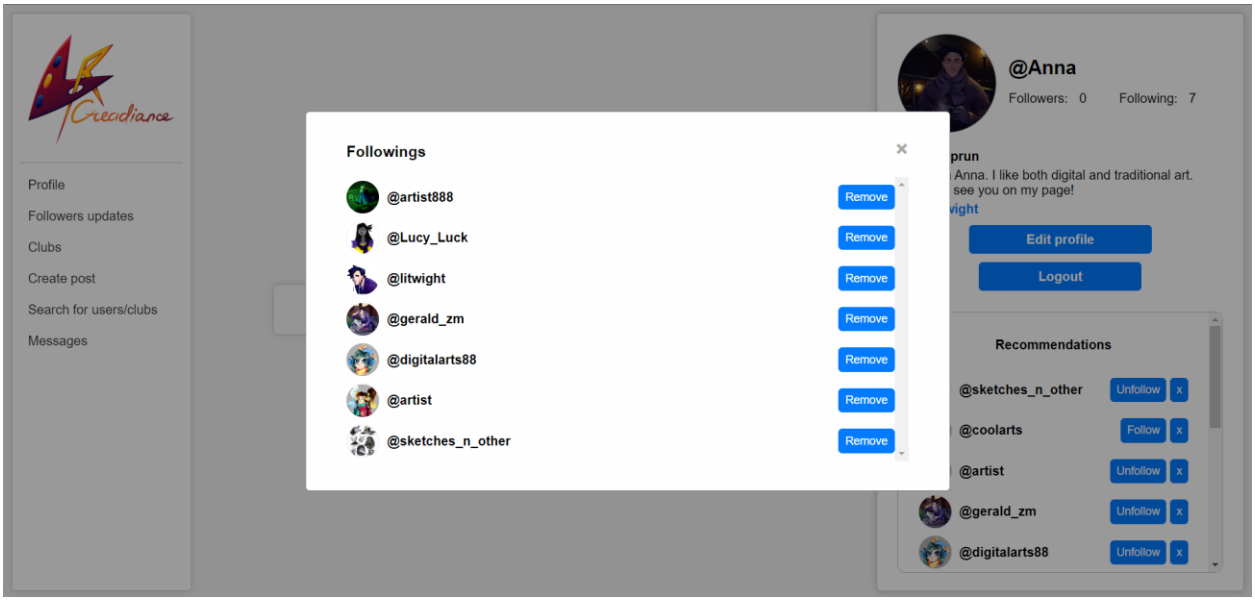

Рисунок 3.7 – Модальне вікно зі списком тих, за ким слідкує користувач

Звідси він може перейти на сторінки цих користувачів (рис. 3.8). Тут він може написати власнику сторінки, відписатися або перейти до публікацій.

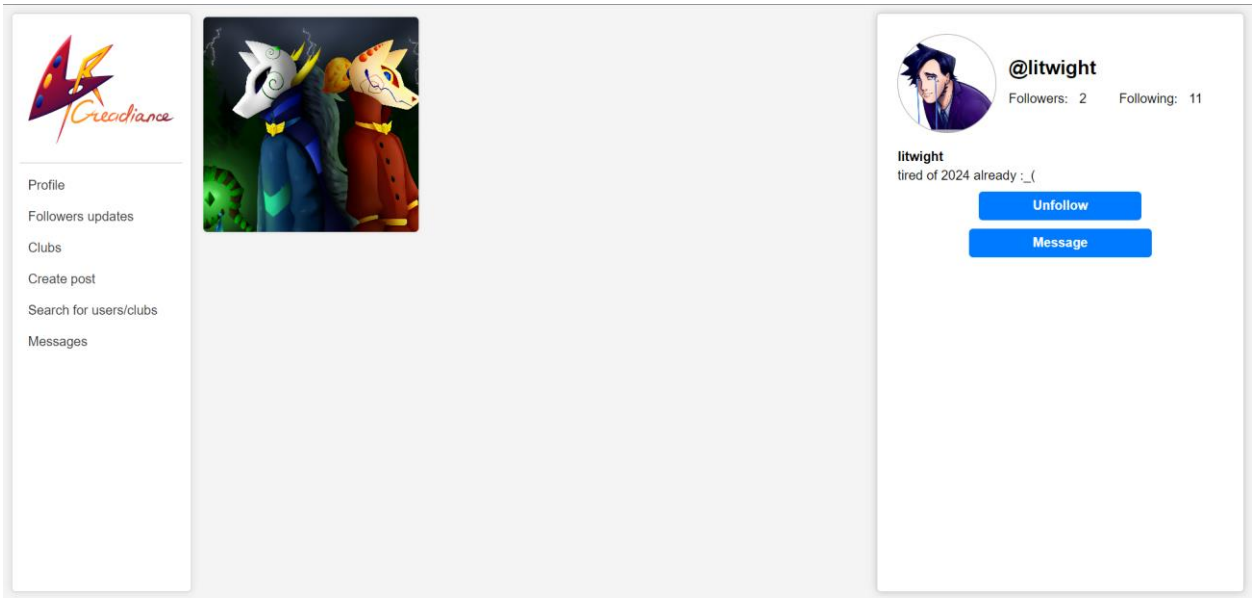

Рисунок 3.8 – Сторінка іншого користувача

Користувач обирає написати повідомлення. Він натискає кнопку «Message» і переходить до сторінки діалогу (рис. 3.9).

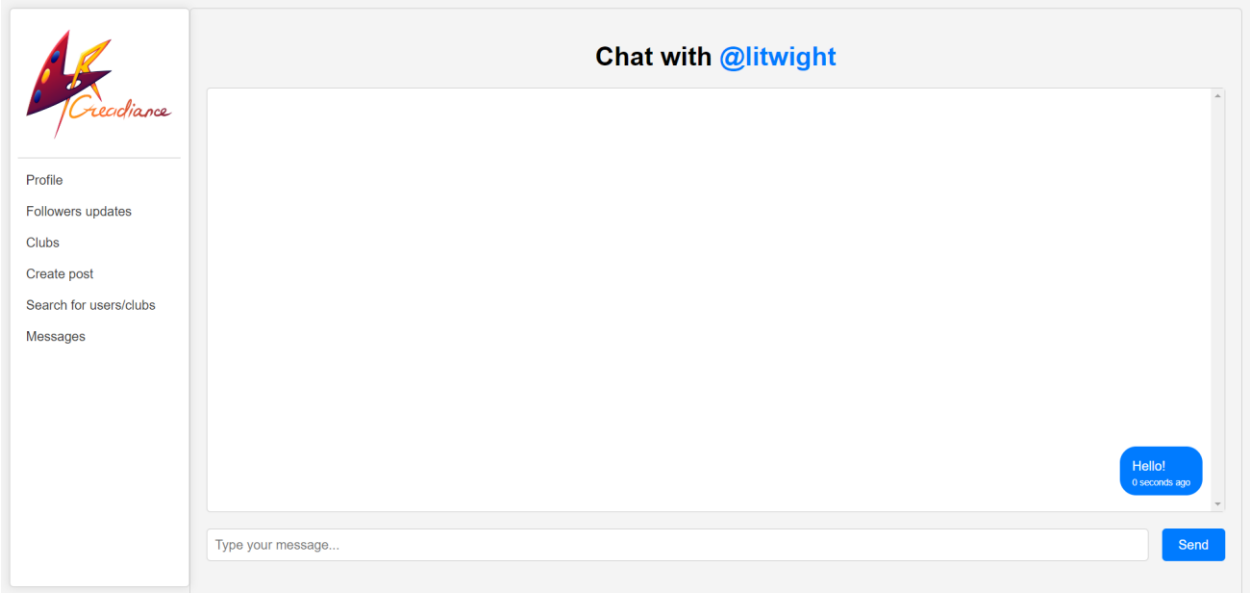

Рисунок 3.9 – Сторінка діалогу

Інший користувач отримує повідомлення та може написати щось у відповідь (рис. 3.10). Для демонстрації відкрито 2 браузера, в яких було здійснено вхід у 2 різних облікових записи.

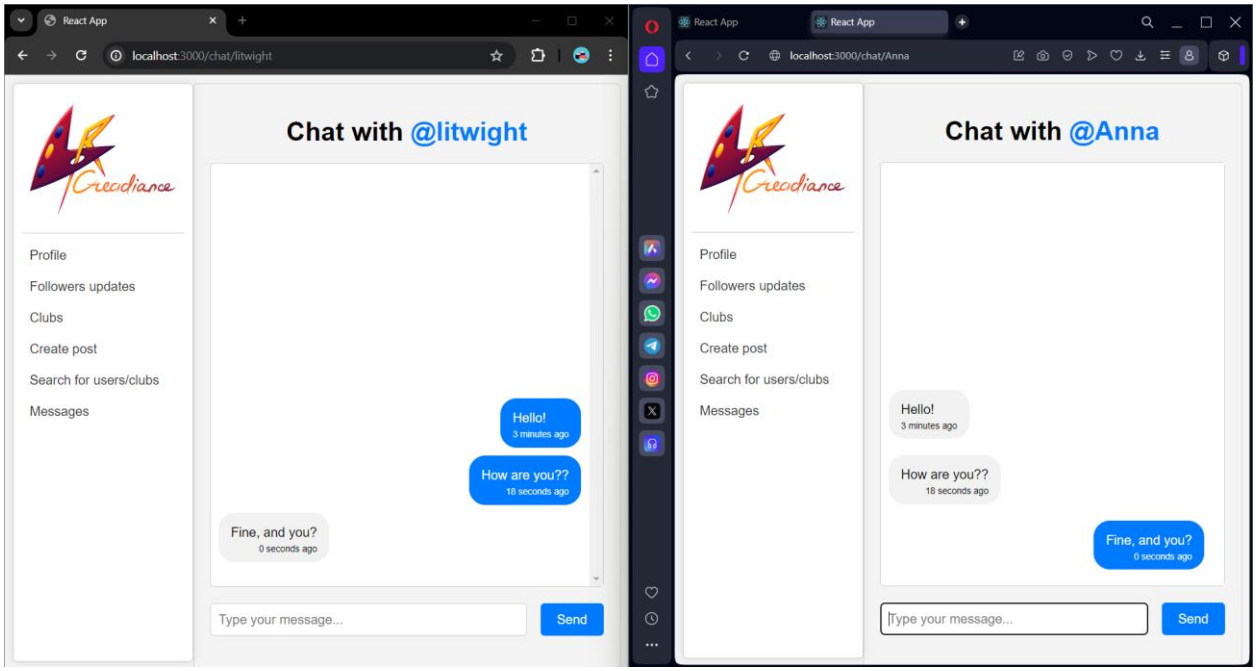

Рисунок 3.10 – Демонстрація діалогу в реальному часі

Користувач може подивитися усі діалоги, натиснувши в меню на кнопку «Messages» (рис. 3.11).

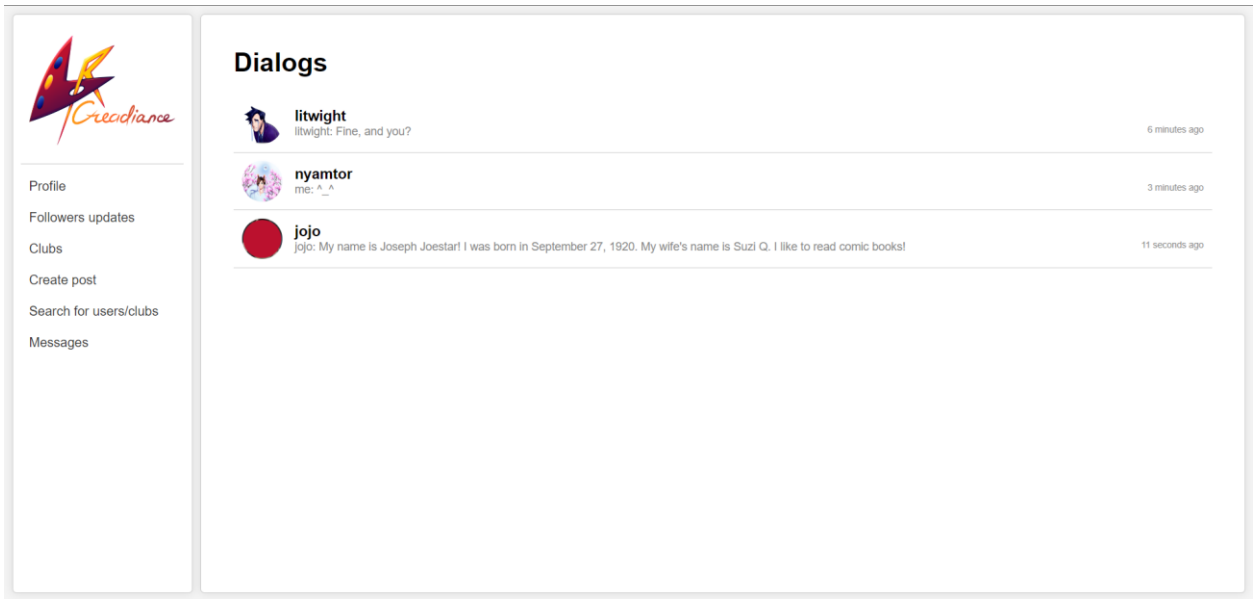

Рисунок 3.11 – Сторінка всіх діалогів

3.3.4 Публікації і взаємодія з ними

Користувач може перейти до публікації за сторінки її власника, групи або натискаючи кнопку «Followers updates» в меню (рис. 3.12).

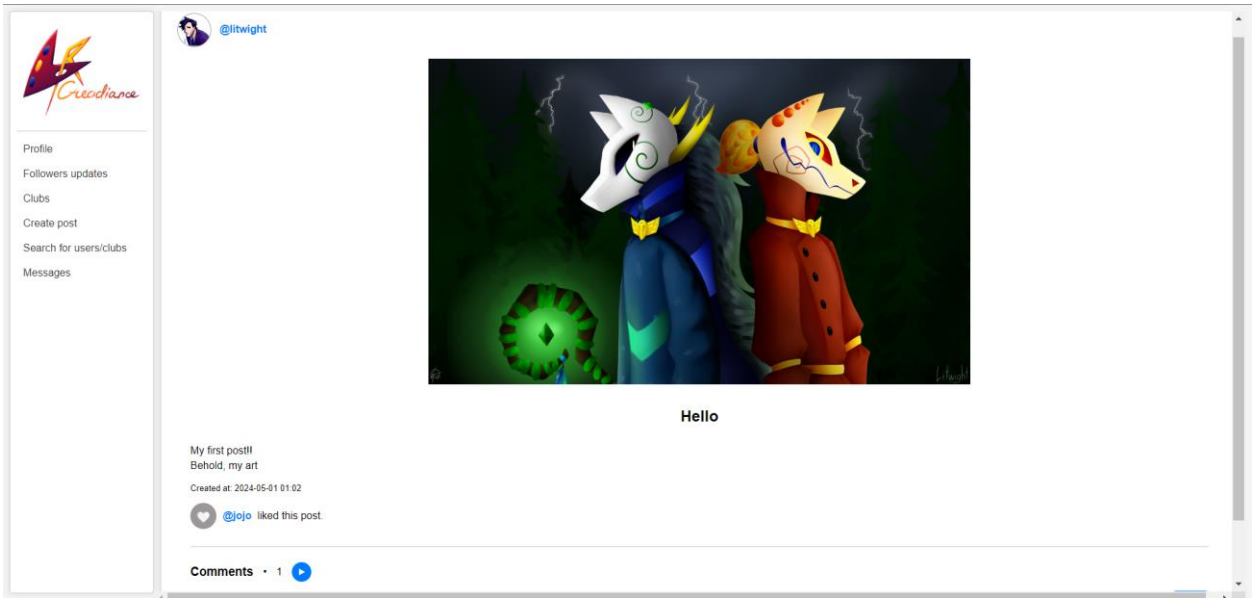

Рисунок 3.12 – Сторінка публікації

Користувач може залишити реакцію і/або коментар під публікацією (рис. 3.13).

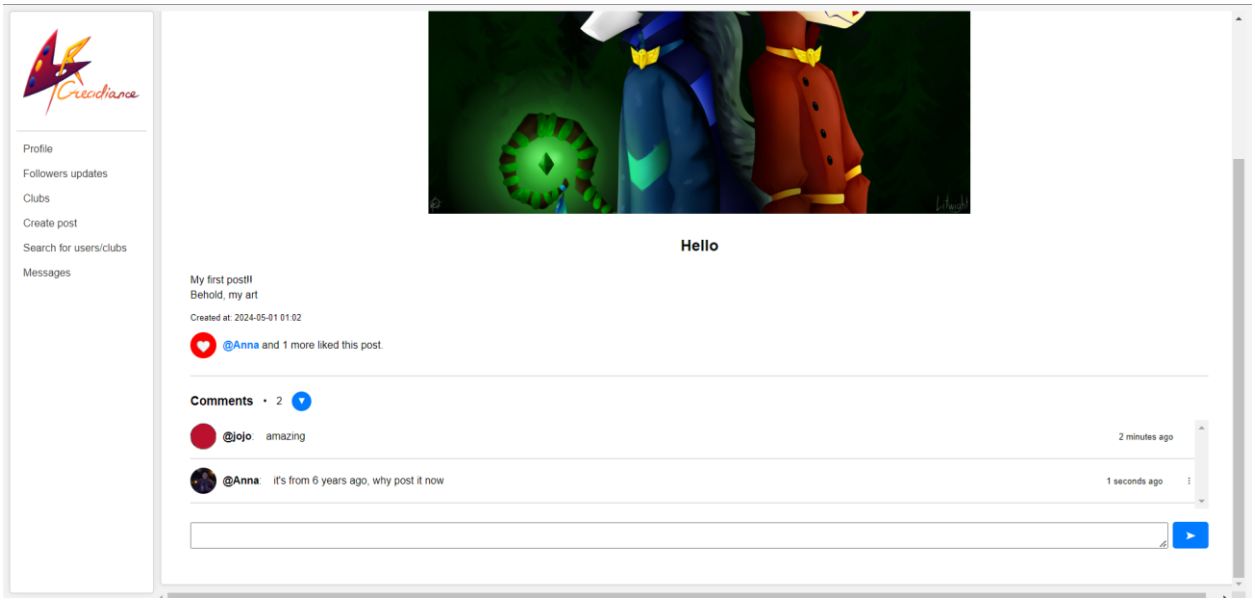

Рисунок 3.13 – Реакція і коментар

Можна переглянути, хто залишив реакції під публікацією, натиснувши на текст коло кнопки серця (рис. 3.14).

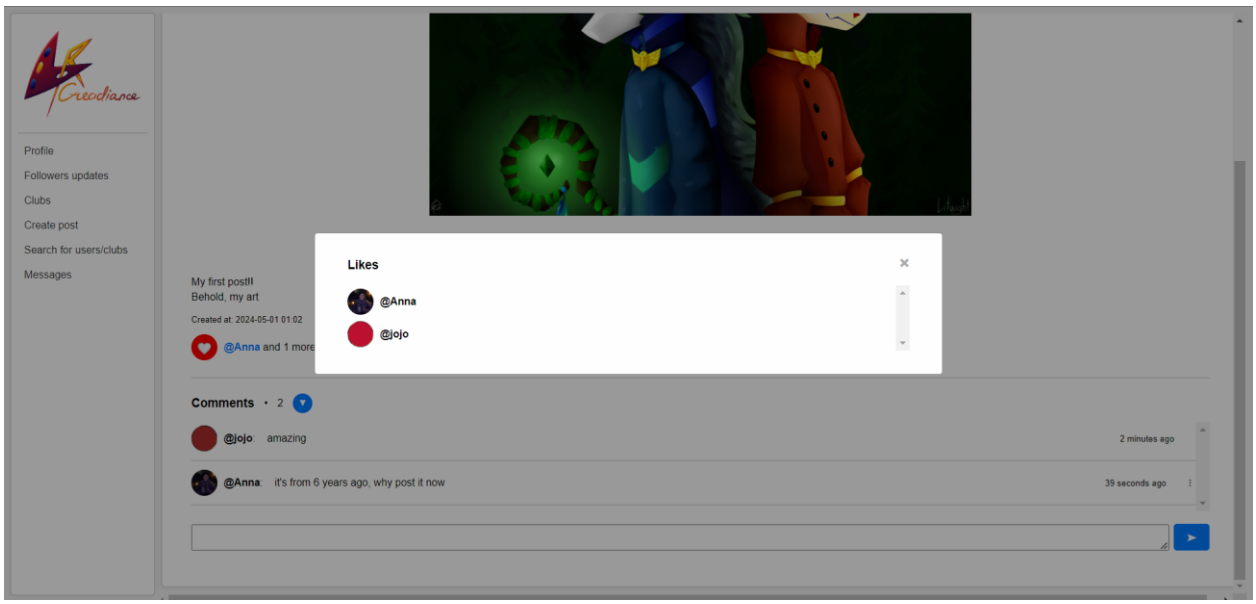

Рисунок 3.14 – Модальне вікно із всіма, хто залишив реакцію

Можна відредагувати або видалити свій коментар, натиснувши на три крапки з правої сторони від нього (рис. 3.15). Також автор публікації може видаляти коментарі інших.

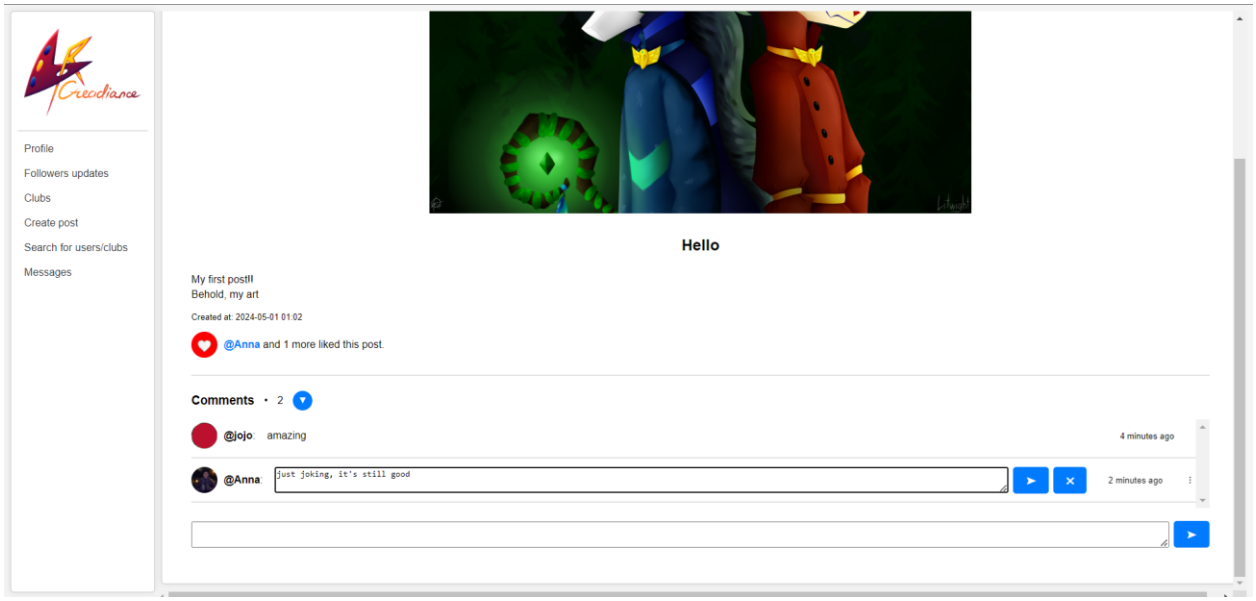

Рисунок 3.15 – Редагування коментаря

Натиснувши в меню «Create post», користувач може створити публікацію (рис. 3.16).

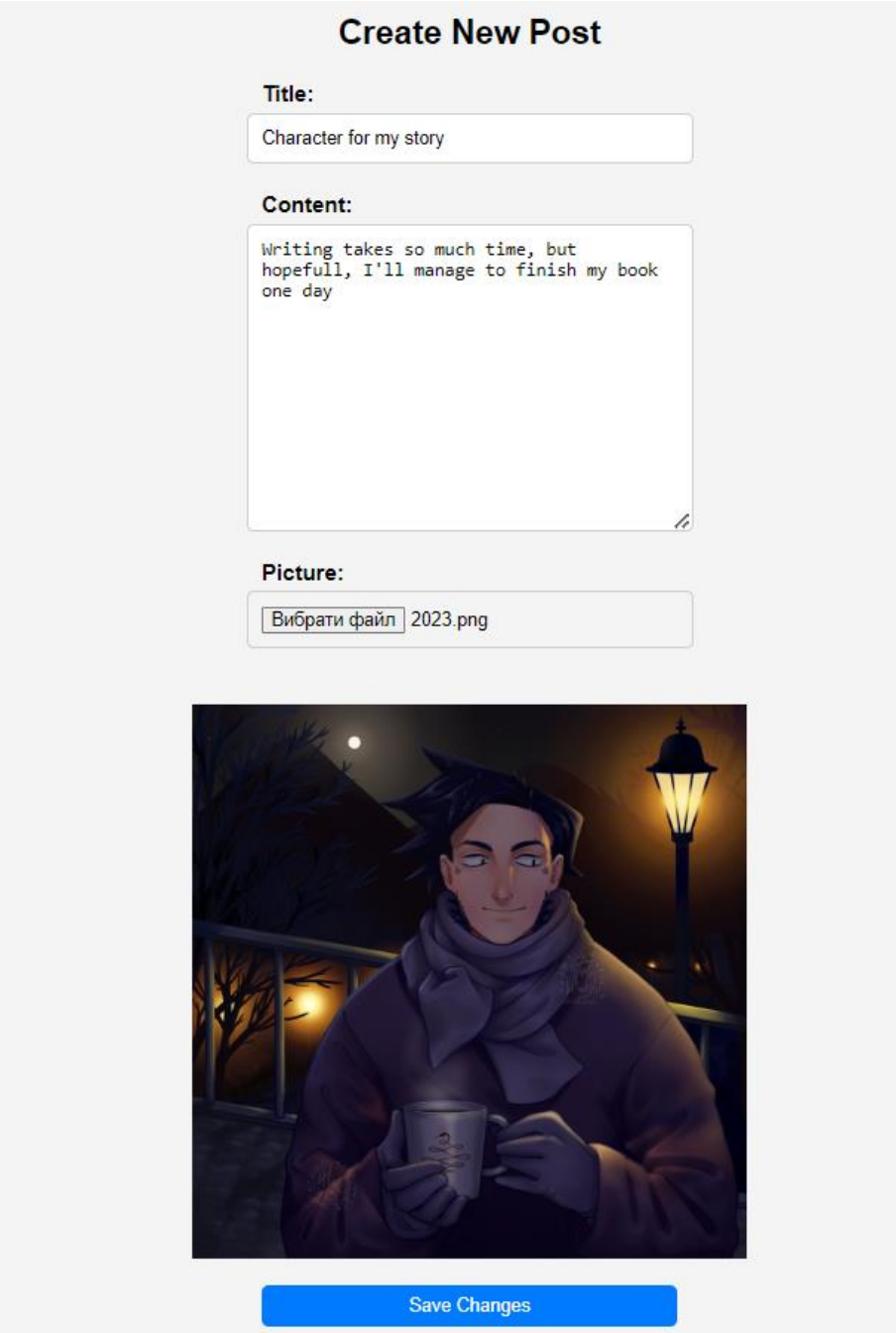

Рисунок 3.16 – Форма створення публікації

Зі створенням нових публікацій старі відображаються нижче та правіше за нові (рис. 3.17).

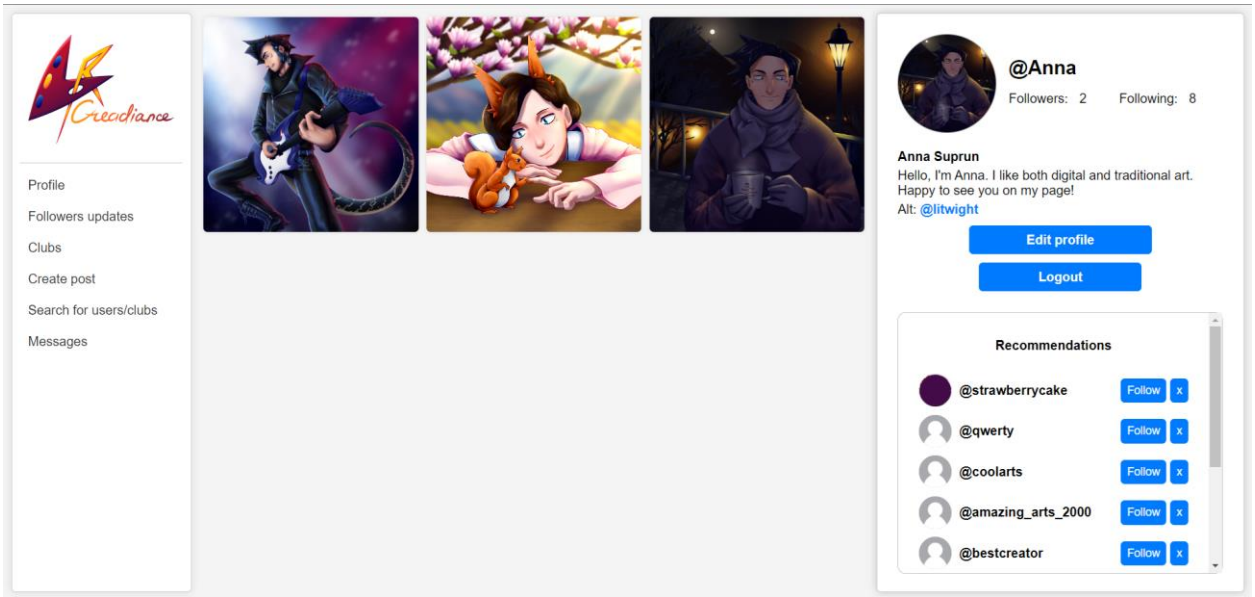

Рисунок 3.17 – Профіль із новими публікаціями

Користувач може редагувати свою публікацію, натиснувши кнопку «Edit» (рис. 3.18). Він не може змінювати фото у ній. При натисканні він перейде до сторінки редагування (рис. 3.19). Редагований пост буде відмічений як «redacted» (рис. 3.20).

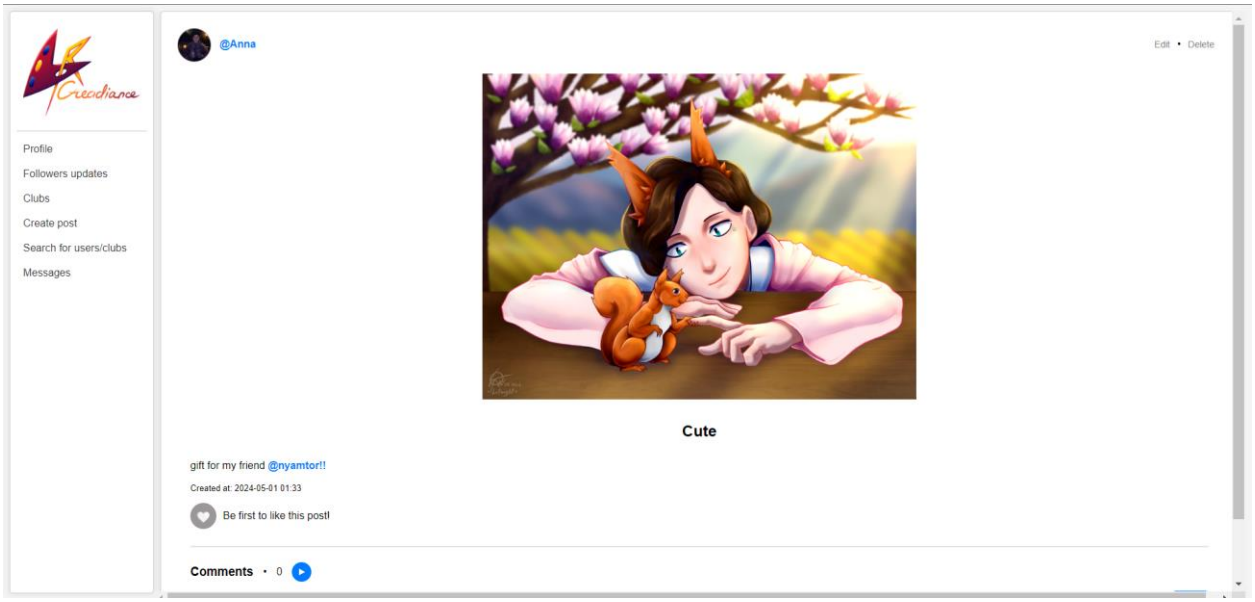

Рисунок 3.18 – Перегляд своєї публікації

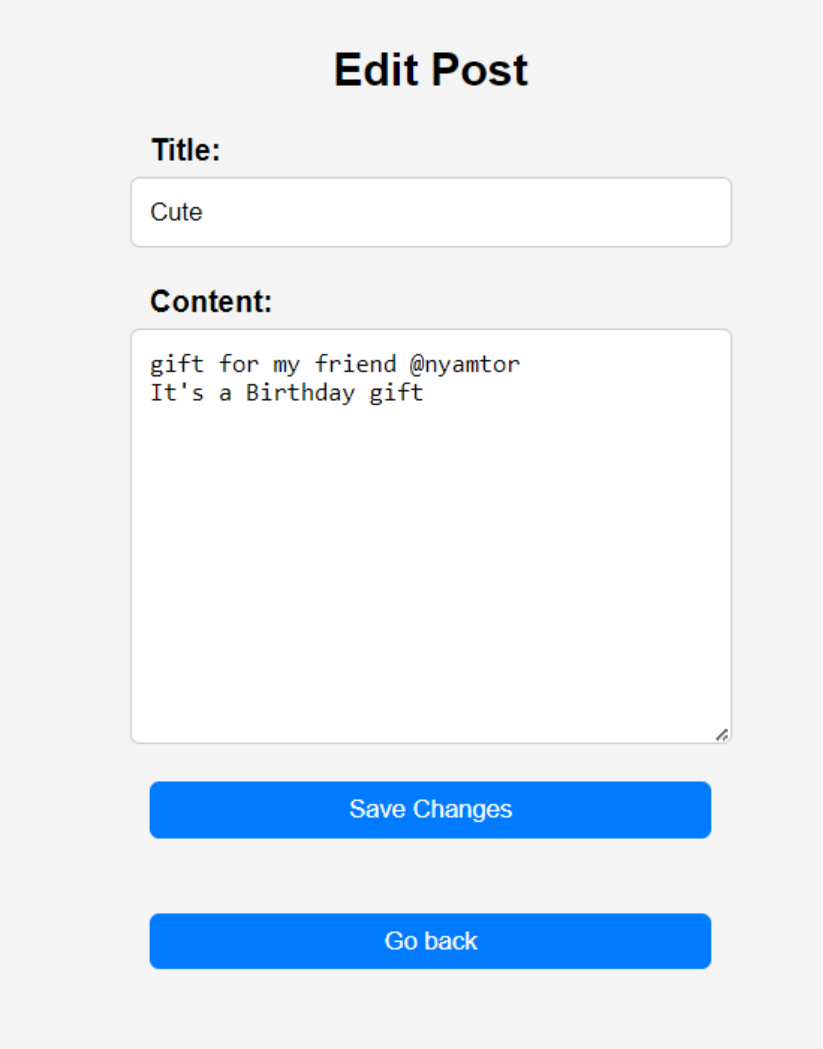

Рисунок 3.19 – Форма редагування публікації

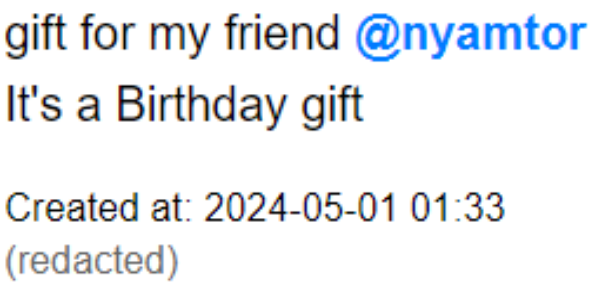

Рисунок 3.20 – Відредагований текст публікації

Натиснувши в меню на кнопку «Followers updates», користувач може побачити найновіші публікації тих, за ким він слідкує (рис. 3.21).

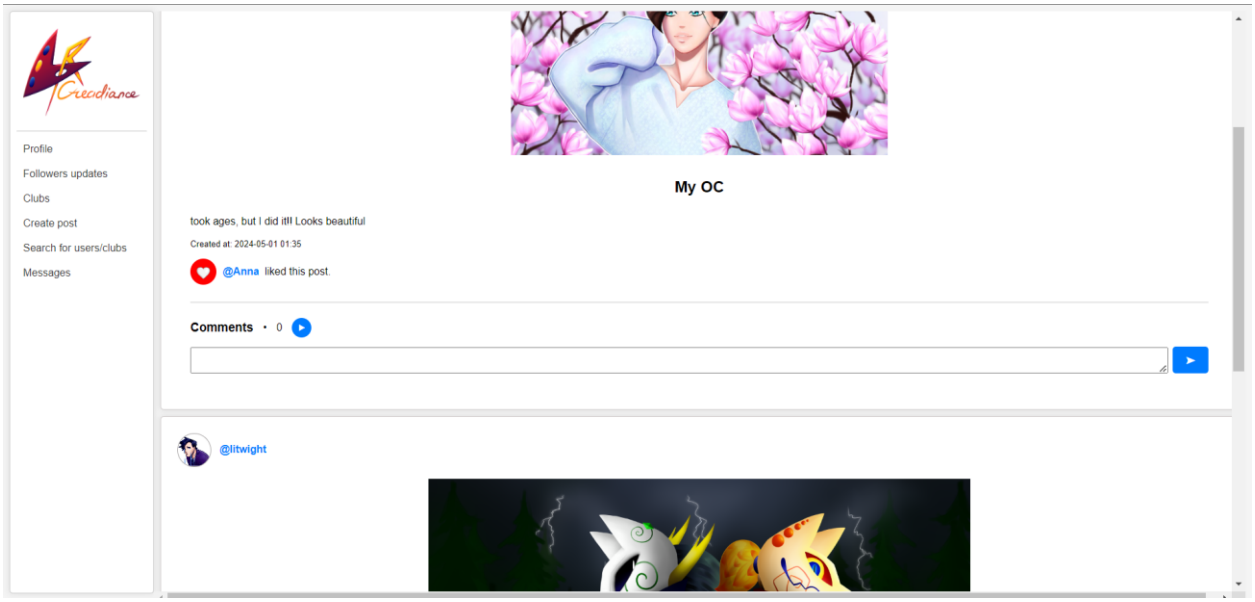

Рисунок 3.21 – Публікації інших користувачів

3.3.5 Групи

Натиснувши кнопку «Clubs», користувач може перейти до сторінки груп, у яких він є учасником. Якщо він ще не є учасником жодної, на цій сторінці йому пропонується перейти на пошукову сторінку, де за іменем можна шукати як групи, так й інших користувачів (рис. 3.22).

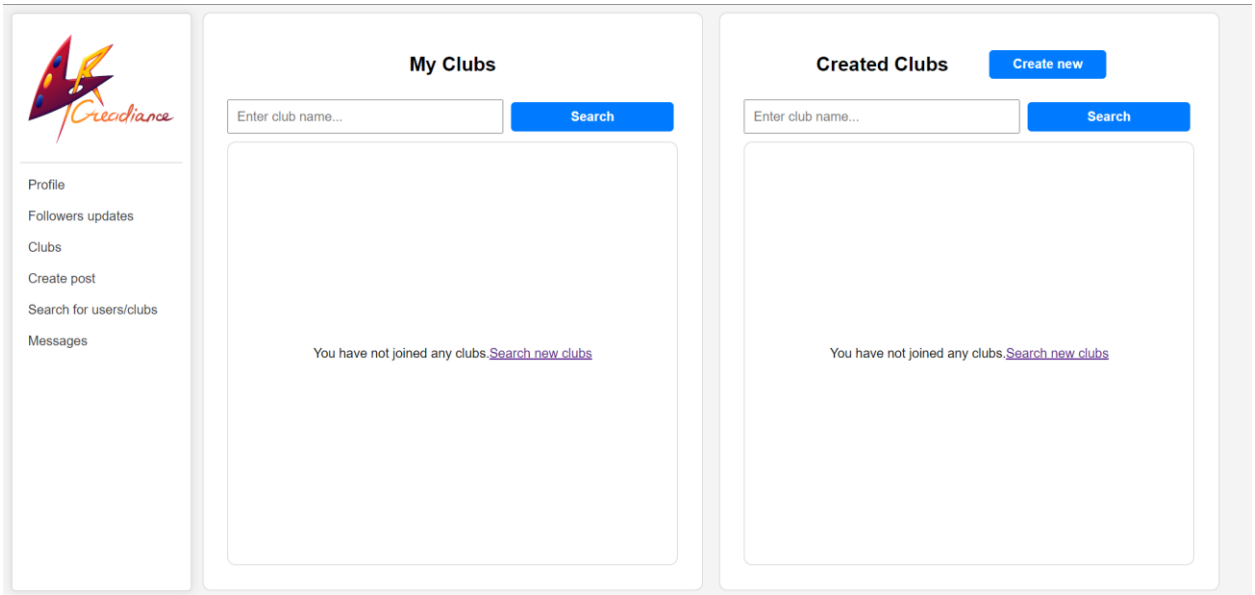

Рисунок 3.22 – Сторінка груп користувача, що не є учасником жодної

На сторінку пошуку також можна перейти із меню, натискаючи кнопку «Search for users/clubs». На ній користувач може вводити повне або часткове ім’я (рис. 3.23). Натискаючи на певний результат пошуку, можна перейти на його сторінку.

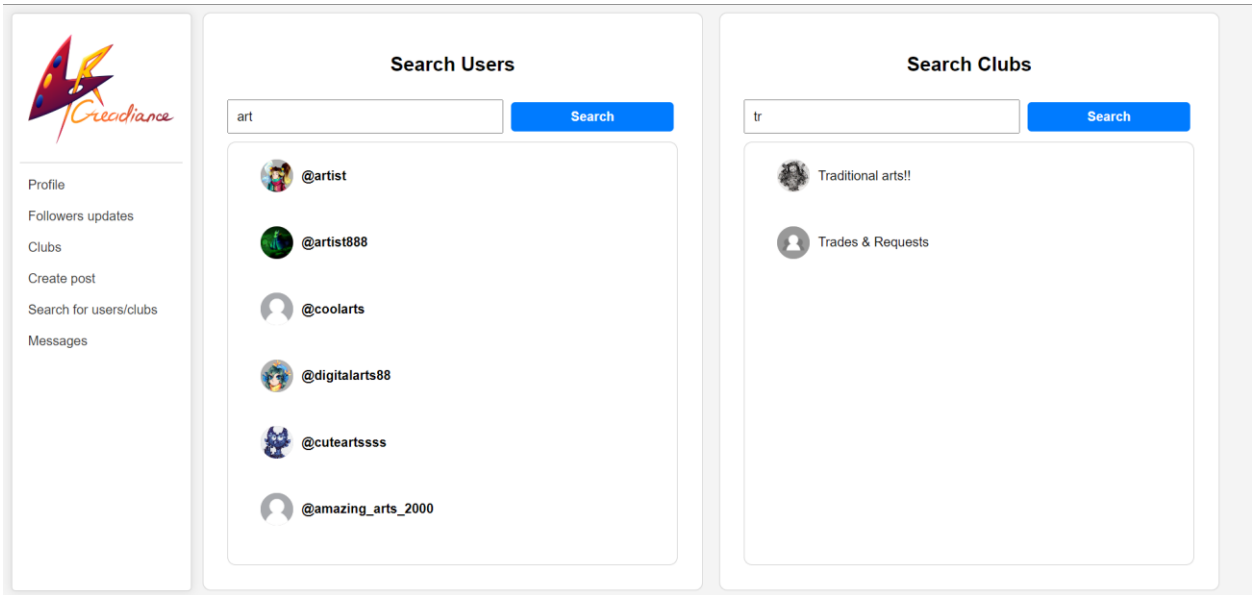

Рисунок 3.23 – Пошукова сторінка із результатами пошуку

Користувач може приєднатися до групи, натиснувши кнопку «Join club» (рис 3.24).

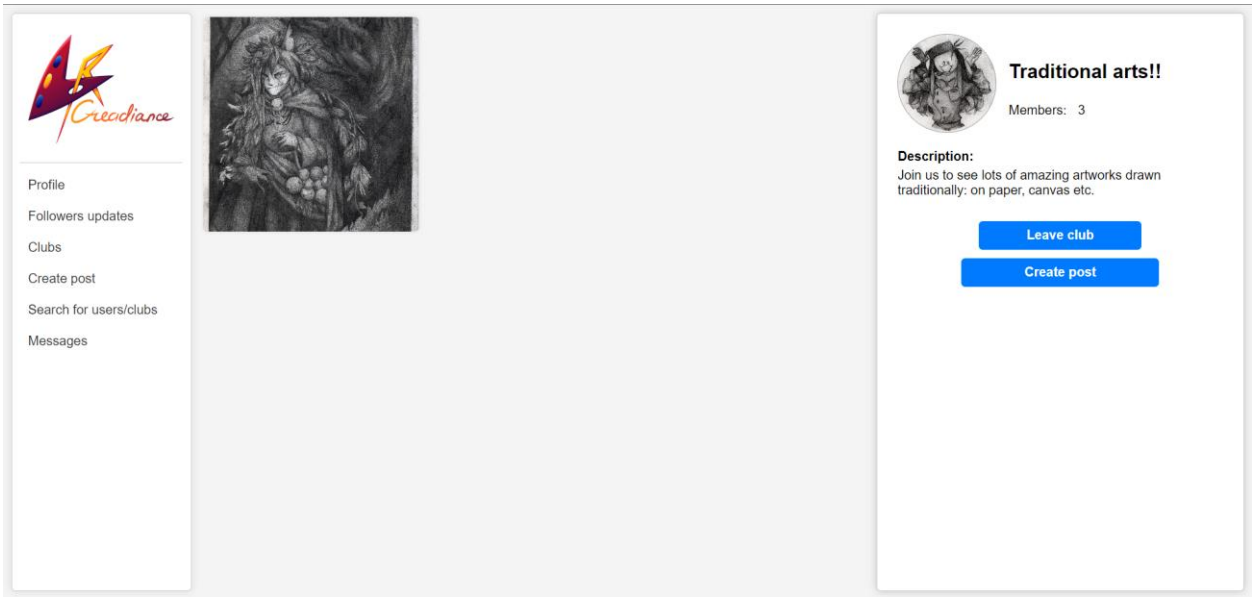

Рисунок 3.24 – Сторінка групи після приєднання користувача до неї

Є можливість створити публікацію у групі, натиснувши кнопку «Create post» (рис. 3.25).

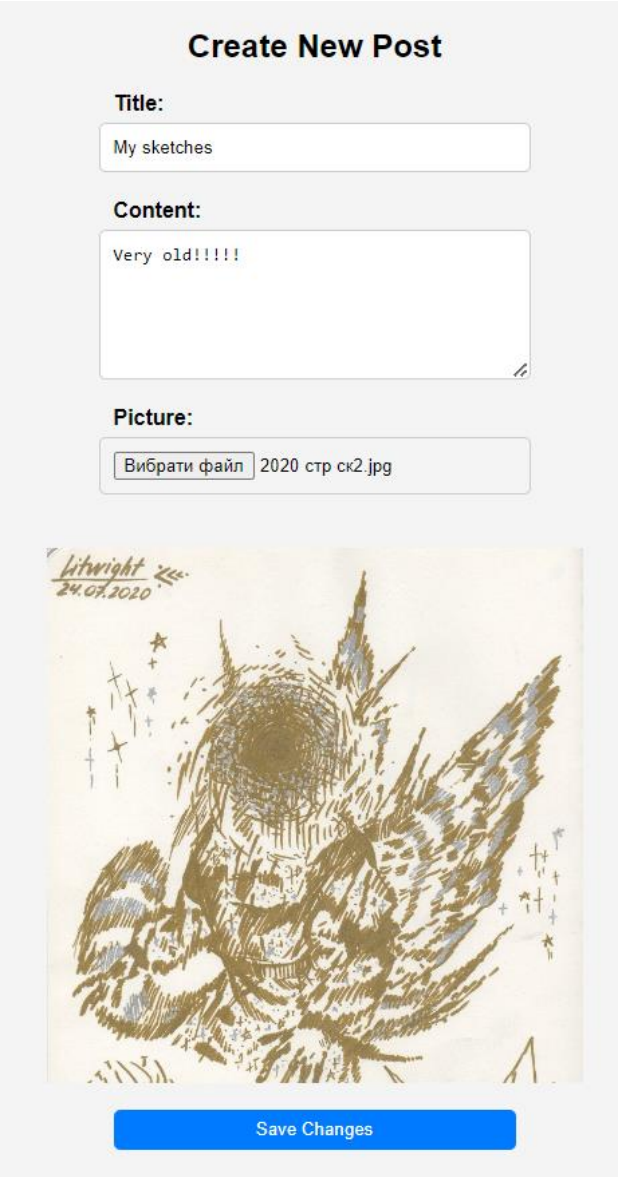

Рисунок 3.25 – Форма створення публікації

В результаті публікація відображається на сторінці групи. Натискаючи на кнопку «Members», є можливість переглянути всіх учасників групи (рис. 3.26).

Так само, як і з користувачами зі списків «Followers» і «Following», користувач може переходити на їхні сторінки.

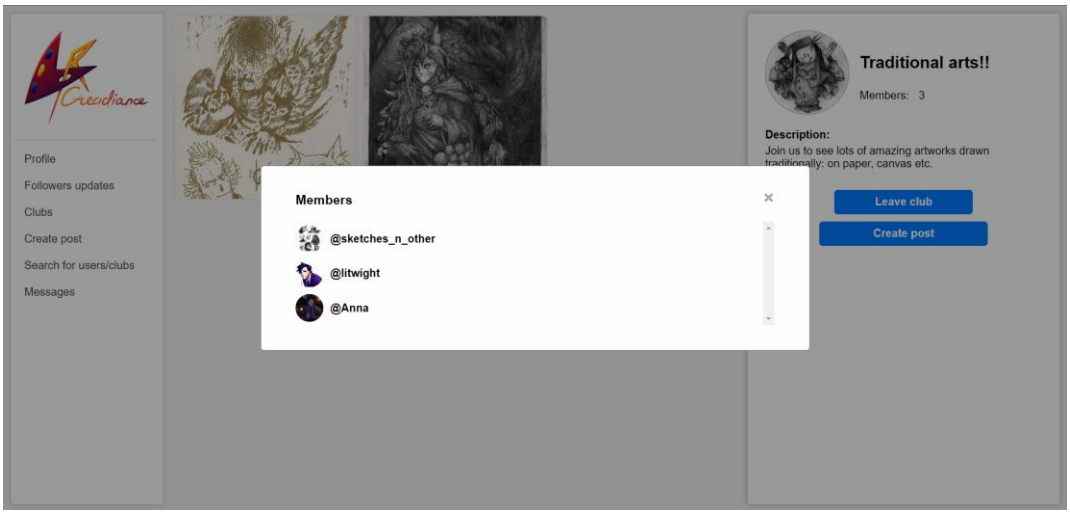

Рисунок 3.26 – Вікно з учасниками групи

Користувач може створити свою власну групу. Для цього він переходить на сторінку груп і натискає кнопку «Create new». Тоді він перейде до сторінки створення групи (рис. 3.27).

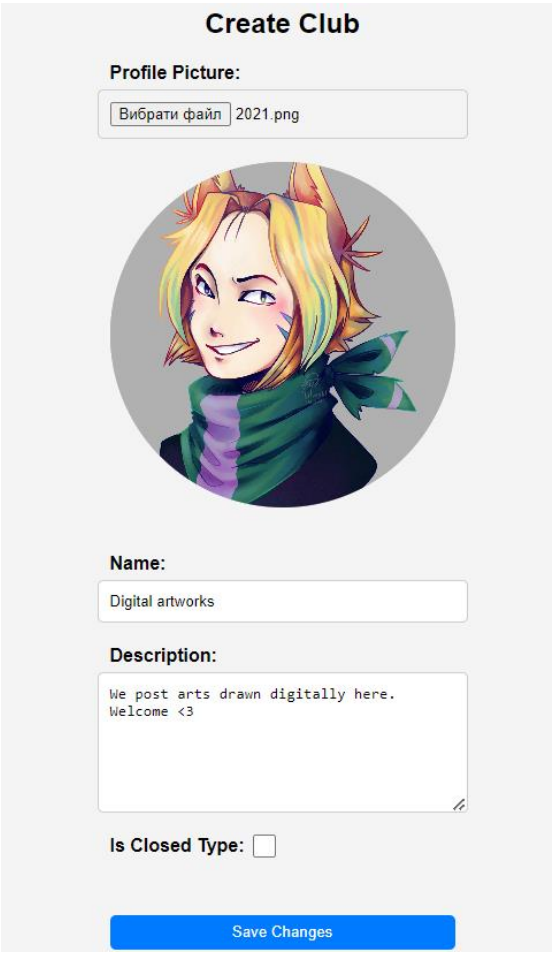

Рисунок 3.27 – Форма створення групи

Користувач може обрати опцію закритої групи (Is Closed Type). Якщо вона обрана, група не буде відображатися в пошуку, але все ще буде видна для тих користувачів, що є її учасниками. Власник може змінити тип групи в будь-який момент на сторінці редагування, і тоді вона знову буде відображатися іншим користувачам у пошуку.

Після успішного створення групи користувач переходить на її сторінку (рис. 3.28). Звідси він може перейти до її редагування або створення нової публікації. Така публікація буде відображатися лише на сторінці цієї групи.

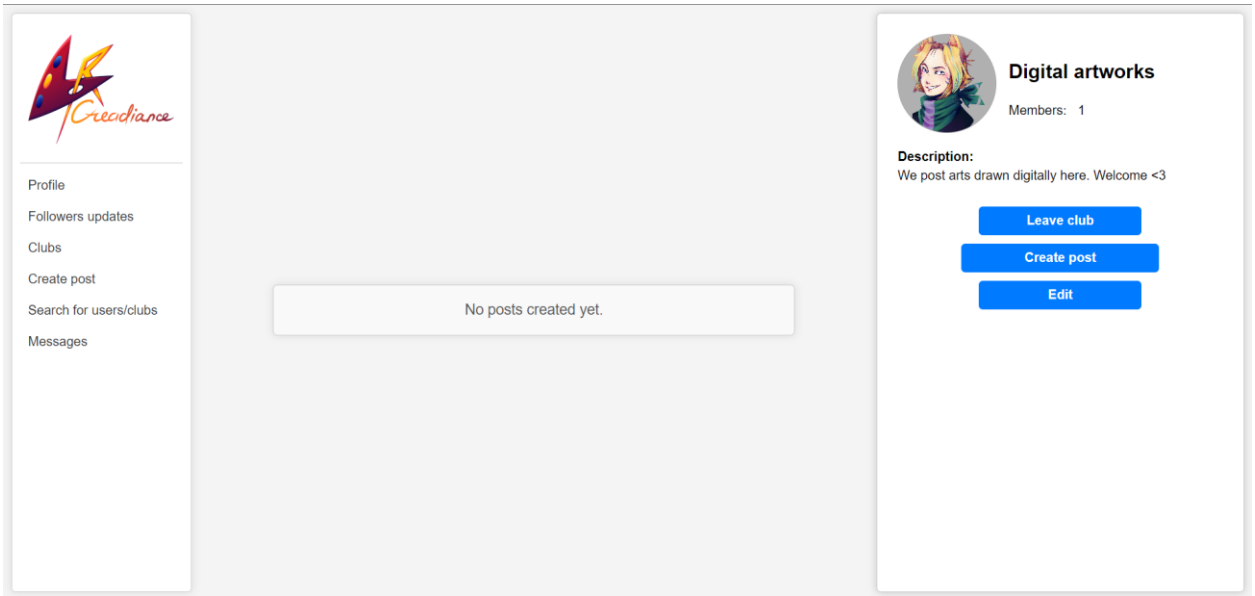

Рисунок 3.28 – Сторінка створеної групи

Користувач може перестати бути учасником створеної ним групи, але все ще може її редагувати.

Свої групи та ті, у яких користувач є власником, він може переглянути на сторінці груп, натиснувши в меню кнопку «Clubs» (рис. 3.29).

Більш старі публікації, коментарі, а також користувачі та групи що є нижче у списках, завантажуються, коли користувач прокручує сторінку донизу.

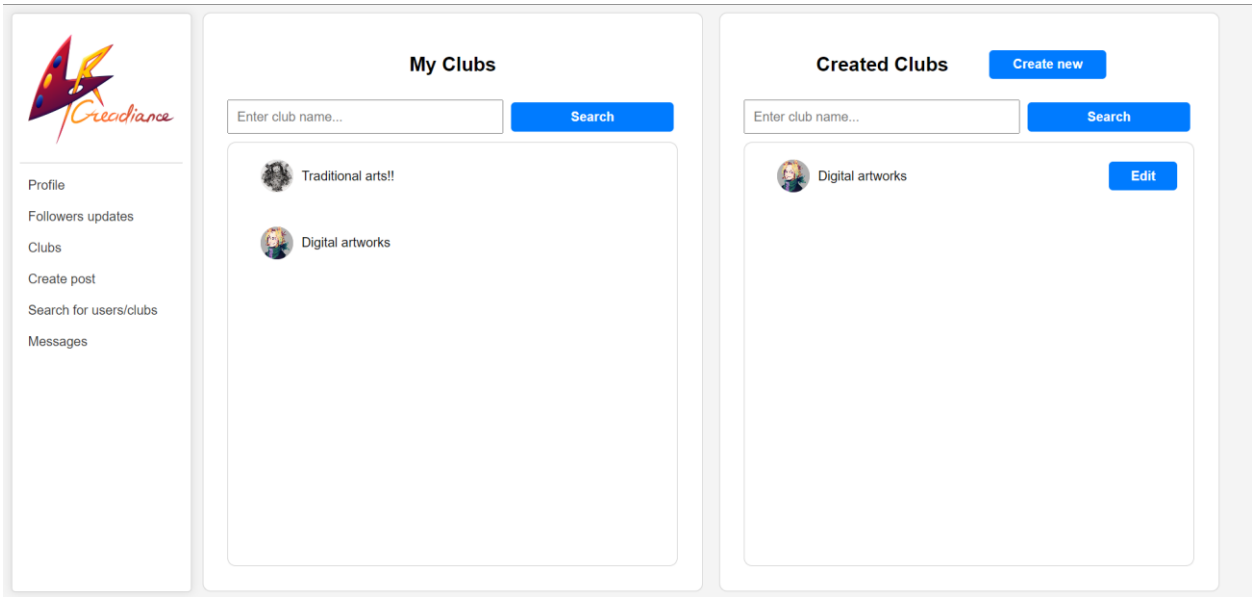

Рисунок 3.29 – Сторінка груп користувача

3.3.6 Сторінка адміністратора

Адміністратор може блокувати користувачів на його сторінці. Він знаходить користувача за ім’ям і може ввести причину, з якої користувач був заблокований (рис. 3.30).

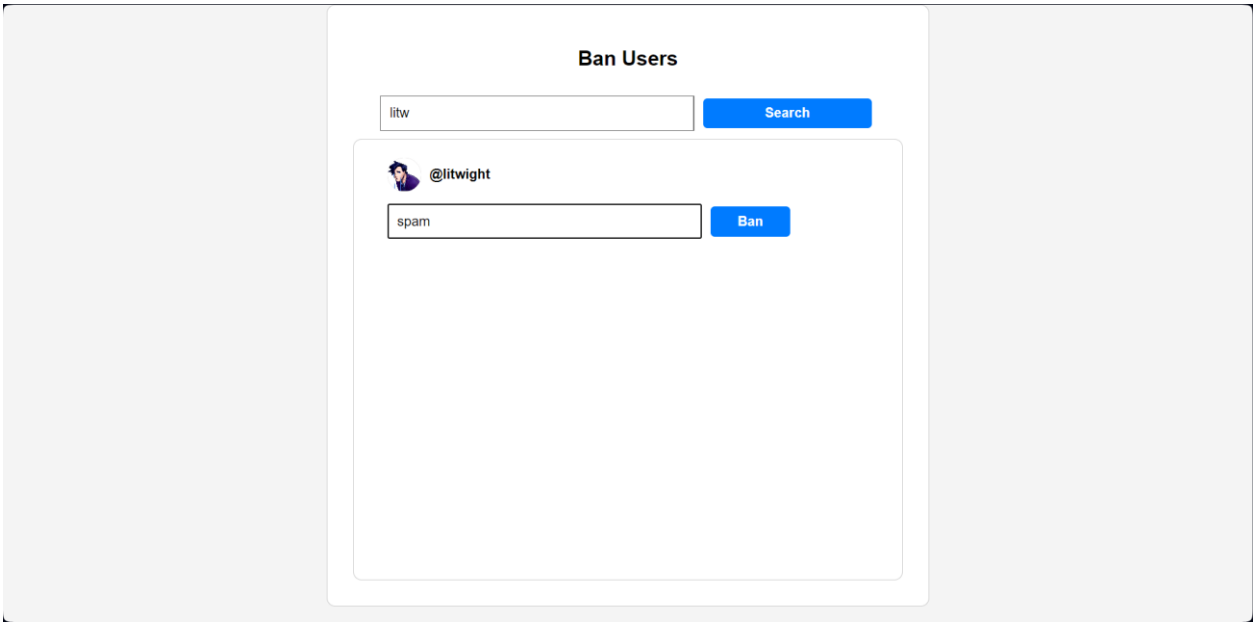

Рисунок 3.30 – Сторінка адміністратора, можливість блокування

Після вводу причини адміністратор натискає кнопку «Ban», користувач стає заблокованим і не може увійти в систему за допомогою сторінки «Login». Також може розблокувати користувача за допомогою кнопки «Unban», тоді користувач зможе користуватися соціальною мережею як раніше (рис. 3.31).

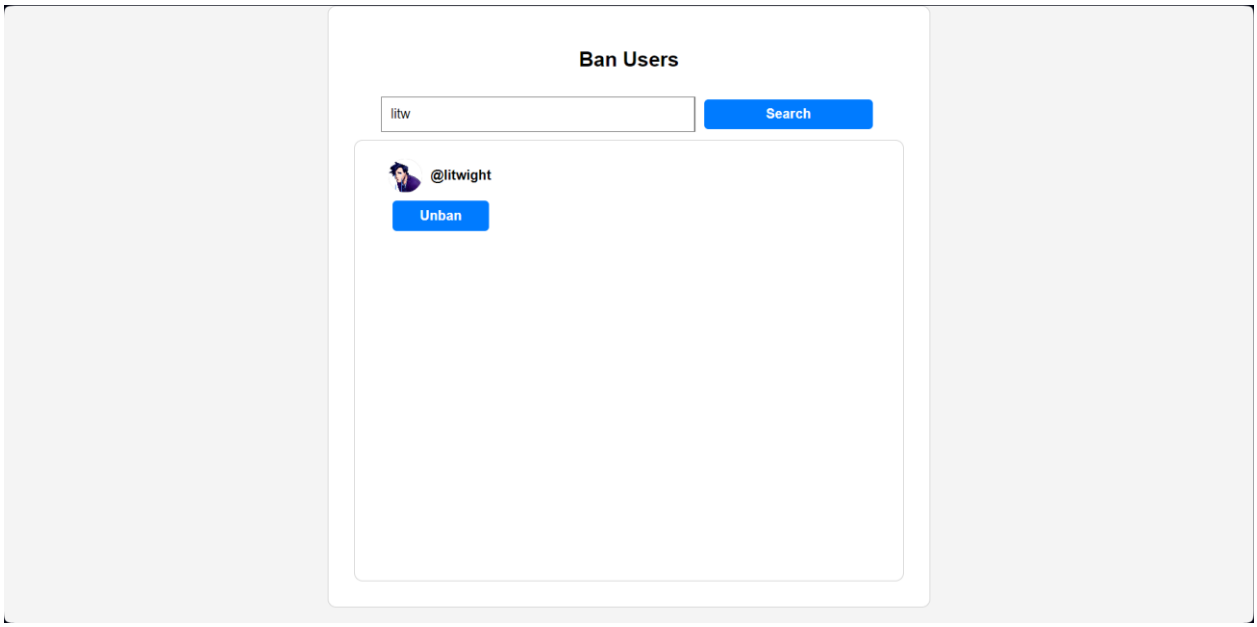

Рисунок 3.31 – Сторінка адміністратора, можливість розблокування

3.4 Перспективи подальшого розвитку соціальної мережі

Можливі такі варіанти розвитку функціоналу застосунку:

– розвиток алгоритмів підбору контенту користувачам не тільки на основі найновіших публікацій користувачів зі списку «Following», а ще і на основі інтересів;

– покращення алгоритмів рекомендацій груп і користувачів;

– додання можливості публікації аудіо і відеофайлів;

– додання можливості пошуку контенту за хештегами;

– додання можливості перегляду статистики по профілю, групах і публікаціях;

– додання можливості редагування фото і відео на платформі, використання фільтрів;

– додання системи оповіщень про нові публікації, повідомлення, реакції, коментарі тощо;

– покращення кастомізації профілю (можливості налаштування інтерфейсу користувача з метою відповідності особистим вимогам і уподобанням користувача). Кастомізація може включати зміну кольорів, шрифтів, розміщення різних елементів, налаштування функціональності профілю тощо;

– створення власних алгоритмів або інтеграція із сервісами, що надають аналіз контенту задля виявлення фото і відео, створених штучним інтелектом;

– інтеграція із платіжними системами, можливість співпраці із рекламодавцями задля публікації реклами;

– розширення функціоналу спілкування в чатах (надсилання фото, відео, аудіо, файлі, редагування, видалення повідомлень, стікерів, емодзі тощо), створення чатів на декілька людей;

– розширення можливостей груп, наприклад, чати груп, додання розділів тощо;

– додання можливості зміни мови інтерфейсу, зміни тем вебсайту (темна, світла, інші);

– додання інструментів створення контенту прямо на платформі (інструменти для малювання тощо);

– додання навчального контенту (статті, відео, трансляції лекцій і майстер-класів тощо).

Отже, перспективи подальшого розвитку даного застосунку є достатньо широкими.

ВИСНОВКИ

У рамках кваліфікаційної роботи було розроблено вебзастосунок соціальної мережі для публікації творчих робіт.

Під час розробки цього проєкту було проведено аналіз існуючих соціальних мереж, їхнього функціоналу, елементів дизайну тощо. Виявлено елементи, що є спільними для кожної мережі, їхні переваги, що варто було б перенести до даного застосунку, та недоліки, яких можна уникнути при розробці.

Обрано та опановано необхідні для розробки технології. За допомогою Java і Spring Framework розроблено мікросервіс серверної частини, із використанням JavaScript і бібліотеки React реалізовано мікросервіс клієнтської частини, створено базу даних застосунку. Розроблено дизайн інтерфейсу користувача.

Розроблений вебзастосунок соціальної мережі дозволяє користувачам публікувати свої творчі роботи, взаємодіяти між собою за допомогою повідомлень, коментарів і реакцій, приєднуватися до груп тощо.

Після тестування застосунку виявлено варіанти подальшого розширення та покращення його функціоналу.

Результати роботи апробовано у вигляді тез доповідей під час XXVIII Міжнародного молодіжного форуму «РАДІОЕЛЕКТРОНІКА І МОЛОДЬ У XXI СТОЛІТТІ», онлайн конференції «КОМП'ЮТЕРНИЙ ЗІР, СИСТЕМНИЙ АНАЛІЗ ТА МАТЕМАТИЧНЕ МОДЕЛЮВАННЯ» [37].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. What is social networking? URL: https://www.techtarget.com/whatis/definition/social-networking (дата звернення 16.04.2024).

2. Social Media for Artists: Top Platforms and Strategies to Choose. URL: https://www.socialpilot.co/blog/social-media-for-artists (дата звернення 16.04.2024).

3. Instagram. URL: https://www.instagram.com/ (дата звернення 17.04.2024).

4. X. URL: https://twitter.com/home (дата звернення 17.04.2024).

5. TikTok. URL: https://www.tiktok.com/ (дата звернення 17.04.2024).

6. Pinterest. URL: https://www.pinterest.ca/ (дата звернення 17.04.2024).

7. DeviantArt. URL: https://www.deviantart.com/ (дата звернення 17.04.2024).

8. Facebook. URL: https://www.facebook.com/ (дата звернення 17.04.2024).

9. Programming languages used in most popular websites. URL: https://en.wikipedia.org/wiki/Programming_languages_used_in_most_popular_websites (дата звернення 17.04.2024).

10. Threads, Instagram, TikTok: Top Social Media Apps and What Technologies Stand Behind Them. URL: https://codegym.cc/groups/posts/18412-threads-instagram-tiktok-top-social-media-apps-and-what-technologies-stand-behind-them (дата звернення 17.04.2024).

11. Гороховатський В.О., Творошенко І.С. (2022) Аналіз багатовимірних даних за описом у формі множини компонент: монографія. Харків: ХНУРЕ, 124 с.

12. Tvoroshenko, I., & Andrieieva, A. (2021). Development of web applications for remote learning of English.

13. Танянський, О., & Руденко, Д. (2018). Порівняльний аналіз популярних JavaScript-фреймворків та бібліотек для front-end розробки.

14. Творошенко, І. С. (2021). Технології прийняття рішень в інформаційних системах: навч. посібник. Харків: ХНУРЕ.

15. Tvoroshenko I., and Gorokhovatskyi V. (2022) The Application of Hybrid Intelligence Systems for Dynamic Data Analysis, *International Journal of Engineering and Information Systems*, 6(2), pp. 40-48.

16. Top 10 Most Relevant Topics on Instagram. URL: https://blog.fanpagekarma.com/2019/04/18/top-10-most-relevant-topics-on-instagram/ (дата звернення 18.04.2024).

17. Gorokhovatskyi V., Tvoroshenko I., Yakovleva O., Hudáková M., and Gorokhovatskyi O. (2024) Application a committee of Kohonen neural networks to training of image classifier based on description of descriptors set, *IEEE Access*, vol. 12, pp. 73376-73385.

18. Bodyanskiy, Y. V., Shafronenko, A., & Rudenko, D. (2019). Online Neuro Fuzzy Clustering of Data with Omissions and Outliers based on Completion Strategy. In CMIS (pp. 18-27).

19. Abu-Jassar, A., Rudenko, D., & Abdalla, H. (2024). Digital medical image as an object of processing and analysis. Multidisciplinary Journal of Science and Technology, 4(1), 28-35.

20. Pomazan, V., Tvoroshenko, I., & Gorokhovatskyi, V. (2023). Development of an application for recognizing emotions using convolutional neural networks, International Journal of Academic Information Systems Research, 7(7), pp. 25-36.

21. Руденко Д.О., Бондар В.О. (2020). Огляд можливостей використання стратегій об’єктно-орієнтованого маппінгу для зіставлення сутностей при розробці web додатків. Матеріали III Міжнародної науково-практичної конференції «Priority Directions of Science and Technology Development» Київ, Україна.с.377-380.

22. Valeria, B., Hryhorii, K., Diana, R., & Vyacheslav, L. (2023). Phase Portrait Models as a Tool for Analyzing Banking Activities.

23. Shafronenko, A. Y., Bodyanskiy, Y. V., & Rudenko, D. A. (2020). Adaptive Neuro-Fuzzy Methods for Distorted Data Clustering.

24. Шафроненко, А. Ю., Бодянський, Є. В., & Руденко, Д. О. (2023). Модифікований рекурентний метод достовірної нечіткої кластеризації з використанням оптимізаційної процедури на основі косяків риб. Системи обробки інформації, (1 (172)), 92-96.

25. Руденко, Д. О., & Кухарчук, В. А. (2022, October). СУЧАСНІ НАПРЯМКИ ЗАСТОСУВАННЯ МЕТОДІВ СУПЕР РОЗДІЛЬНОСТІ ЗОБРАЖЕННЯ. In The 8 th International scientific and practical conference "Modern research in world science"(October 29-31, 2022) SPC "Sci-conf. com. ua", Lviv, Ukraine. 2022. 1828 p. (p. 393).

26. Lyashenko, V., & Rudenko, D. (2021). Modeling Deformation of Spur Gear.

27. Андрєєва, А. Ю., & Руденко, Д. О. (2022, October). ЯК БЛОКЧЕЙН ТОКЕНІЗАЦІЯ ЗМІНЮЄ СВІТ. In The 2 nd International scientific and practical conference "Science and innovation of modern world"(October 26-28, 2022) Cognum Publishing House, London, United Kingdom. 2022. 948 p. (p. 233).

28. Introduction to Java Config for Spring Security. URL: https://www.baeldung.com/java-config-spring-security (дата звернення 23.04.2024).

29. Кобилін, О.А., & Творошенко, І.С. (2021). Методи цифрової обробки зображень: навч. посібник. Харків: ХНУРЕ.

30. Using WebSocket to build an interactive web application. URL: https://spring.io/guides/gs/messaging-stomp-websocket (дата звернення 22.04.2024).

31. Rudenko, D., Serhiienko, O., Zeleniy, O., & Lyashenko, V. (2022). Model for Predictive Analysis of International Trade Based on the Dynamics of Stock Indices (Example of Data from the USA, Canada and UK).

32. React Tutorial. URL: https://www.w3schools.com/react/default.asp (дата звернення 23.04.2024).

33. Features overview URL: https://www.jetbrains.com/idea/features/ (дата звернення 23.04.2024).

34. Visual Studio Code Docs. URL: https://code.visualstudio.com/docs (дата звернення 23.04.2024).

35. What is Postman? URL: https://www.postman.com/product/what-is-postman/ (дата звернення 23.04.2024).

36. Гороховатський, В.О., & Творошенко, І.С. (2021). Методи інтелектуального аналізу та оброблення даних: навч. посібник.

37. Супрун А.Є. Аналіз існуючих інформаційних систем громадських організацій. *Радіоелектроніка та молодь у XXI столітті: тези доповідей 28-го Міжнародного молодіжного форуму (Харків, 16–18 квітня 2024 р.). Харків: ХНУРЕ*, 2024. Т. 7. С. 114-115.