

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

магістр

(ступінь вищої освіти)

на тему

ДОСЛІДЖЕННЯ ТА РЕАЛІЗАЦІЯ
СЕРВІСУ БАГАТОКОРИСТУВАЦЬКОГО БЛОГІНГУ
RESEARCH AND IMPLEMENTATION OF
A MULTIUSER BLOGGING SERVICE

Виконав(ла): студент(ка) 2 курсу, групи КНТ-124м
Спеціальності 121 Інженерія програмного
(код і найменування спеціальності)

забезпечення

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

ЗІНОВ'ЄВ Д.Д.

(ПРІЗВИЩЕ та ініціали)

Керівник ІЛЬЯШЕНКО М.Б.

(ПРІЗВИЩЕ та ініціали)

Рецензент КОЦУР М.І.

(ПРІЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет КНТ
Кафедра програмних засобів
Ступінь вищої освіти магістр
Спеціальність 121 Інженерія програмного забезпечення
(код і найменування)
Освітня програма (спеціалізація) Інженерія програмного забезпечення
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
Сергій СУББОТІН
“ ” 2025 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

ЗІНОВ'ЄВА Данила Дмитровича

(ПРИЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Дослідження та реалізація сервісу багатокористувацького блогінгу. Research and implementation of a multiuser blogging service

керівник проєкту (роботи) к.т.н., доцент, ІЛЬЯШЕНКО Матвій Борисович,
(науковий ступінь, вчене звання, ПРИЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від “30” вересня 2025 року № 447

2. Строк подання студентом проєкту (роботи) 08 грудня 2025 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Матеріали і методи. 3. Опис програми. 4. Експлуатація, тестування та експериментальне дослідження програми.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, кількість слайдів, плакатів)

Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	ПРІЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	ІЛЪЯШЕНКО М.Б., доцент		
Нормоконтроль	КАЛІНІНА М.В., асистент		

7. Дата видачі завдання “30” вересня 2025 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області.	1 тиждень	Розділ 1
3	Вибір мови програмування та інших технологій розробки.	2 тиждень	Розділ 2
4	Розробка структури програми.	3 тиждень	Розділ 3
5	Розробка програми.	4-7 тижні	Розділи 3,4
6	Тестування та експериментальне дослідження програмного забезпечення.	8 тиждень	Розділ 4
7	Оформлення пояснювальної записки та документів до неї.	9-10 тижні	Додатки
8	Нормоконтроль та рецензування.	11 тиждень	
9	Захист роботи.	12 тиждень	

Студент(ка)

_____ Данило ЗІНОВ'ЄВ
(підпис) (Імя ПРІЗВИЩЕ)

Керівник проєкту (роботи)

_____ Матвій ІЛЪЯШЕНКО
(підпис) (Імя ПРІЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи магістра:
87 с., 3 табл., 39 рис., 3 дод., 19 джерел.

FLASK, PYTHON, БЛОГ, БЛОГІНГ, ЗАСТОСУНОК, МОВА
ПРОГРАМУВАННЯ, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, ФРЕЙМВОРК.

Об'єкт дослідження – процес розробки програмного забезпечення для багатокористувацького блогінгу.

Предмет дослідження – програмні засоби для багатокористувацького блогінгу.

Мета роботи – дослідження та розробка програмного забезпечення для багатокористувацького блогінгу.

Матеріали, методи та технічні засоби: мова програмування Python, фреймворк Flask.

Результати. Розроблено програмне забезпечення для багатокористувацького блогінгу за допомогою мови програмування Python та фреймворку Flask.

Практична цінність роботи полягає у розробці програмного забезпечення для багатокористувацького блогінгу.

Висновки. Виконано проєктування програмного забезпечення для багатокористувацького блогінгу. Розроблено програмне забезпечення для багатокористувацького блогінгу. Здійснено тестування розробленого програмного забезпечення для багатокористувацького блогінгу.

Галузь використання – засоби для ведення блогів.

ABSTRACT

Explanatory note to the diploma qualifying work of the master: 87 pages, 3 tables, 39 figures, 3 appendixes, 19 sources.

FLASK, PYTHON, BLOG, BLOGGING, APPLICATION, PROGRAMMING LANGUAGE, SOFTWARE, FRAMEWORK.

The object of research is the process of developing software for multi-user blogging.

The subject of the research is software for multi-user blogging.

The purpose of this work is to develop the software for multi-user blogging.

Materials, methods and technical tools: Python programming language, Flask framework.

Results. The software tools for multi-user blogging using the Python programming language and the Flask framework have been developed.

The practical value of the work lies in the development of software for multi-user blogging.

Conclusions. The software for multi-user blogging has been designed. The software for multi-user blogging has been developed. The developed software for multi-user blogging has been tested.

Scope of use – blogging tools.

ЗМІСТ

	С.
Перелік скорочень та умовних позначок	8
Вступ.....	9
1 Аналіз предметної області.....	11
1.1 Програмне забезпечення для ведення блогів.....	11
1.2 Аналіз програмного забезпечення для ведення блогів.....	23
1.3 Висновки за розділом 1	30
2 Матеріали і методи	32
2.1 Вибір мови програмування.....	32
2.2 Вибір фреймворку для створення програмного забезпечення для багатокористувацького блогінгу.....	34
2.3 Розробка моделі взаємодії користувача з програмним забезпеченням для багатокористувацького блогінгу.....	37
2.4 Висновки за розділом 2	40
3 Опис програми	42
3.1 Структура програмного забезпечення для багатокористувацького блогінгу.....	42
3.2 Функціонування програмного забезпечення для багатокористувацького блогінгу.....	45
3.3 Архітектура програмного забезпечення для багатокористувацького блогінгу.....	49
3.4 Проєктування інтерфейсу програмного забезпечення для багатокористувацького блогінгу.....	51
3.5 Висновки за розділом 3	54
4 Експлуатація, тестування та експериментальне дослідження програми.....	55
4.1 Призначення й умови застосування програми	55
4.2 Характеристики програми для багатокористувацького блогінгу.....	56
4.3 Інструкція по експлуатації програми	57
4.3.1 Звернення до програми.....	57

4.3.2 Вхідні й вихідні дані.....	57
4.3.3 Повідомлення	57
4.4 Виконання програмного забезпечення для багатокористувацького блогінгу.....	58
4.5 Тестування програмного забезпечення для багатокористувацького блогінгу.....	61
4.6 Висновки за розділом 4	61
Висновки	63
Перелік джерел посилань	66
Додаток А Технічне завдання	68
Додаток Б Фрагмент тексту програми	72
Додаток В Слайди презентації.....	80

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

API – Application Programming Interface;

MVC – Model-View-Controller;

ББ – багатокористувацький блогінг;

БД – база даних;

ПЗ – програмне забезпечення.

ВСТУП

Актуальність розроблення та використання програмного забезпечення для багатокористувацького блогінгу зумовлюється інтенсивним розвитком цифрових комунікацій, зростанням ролі контенту у формуванні інформаційного простору та необхідністю забезпечення ефективної взаємодії між авторами в межах єдиної платформи. Сучасні інформаційні потоки характеризуються високою динамікою, що вимагає оперативного створення, редагування та поширення матеріалів різного спрямування. У таких умовах програмні системи для ведення блогів виступають важливим інструментом для організації спільної роботи над контентом, забезпечення контролю якості матеріалів і підтримання їх актуальності [1], [2].

У поєднанні з розвитком хмарних технологій та сервісів, що підтримують спільну роботу, програмне забезпечення для ведення блогів набуває особливого значення для медіапроектів, освітніх платформ, корпоративних сайтів та професійних спільнот. Це зумовлює його перспективність як у комерційній, так і в некомерційній сфері, а також визначає необхідність удосконалення функціональних можливостей з урахуванням сучасних тенденцій цифрової взаємодії [3], [4].

Проте деякі програми для ведення блогів можуть мати підвищену складність адміністрування, оскільки забезпечення узгодженої взаємодії між численними користувачами потребує чітко налаштованих механізмів контролю доступу, модерації матеріалів та запобігання конфліктам при одночасному редагуванні. У разі відсутності належної оптимізації інтерфейсу та внутрішньої архітектури програми зростає ймовірність виникнення помилок під час публікації або оновлення вмісту блогу. Крім того, використання готових платформ із закритим вихідним кодом може обмежувати можливості налаштування функціоналу під специфічні потреби, що ускладнює інтеграцію з іншими інформаційними системами та програмними сервісами [3]-[6].

Тому актуальною є розробка програмного забезпечення для багатокористувацького блогінгу. У дипломній кваліфікаційній роботі магістра розв'язується актуальне завдання розробки та дослідження програмного забезпечення для багатокористувацького блогінгу.

Для досягнення поставленої мети у кваліфікаційній роботі магістра необхідно розв'язати такі задачі:

- виконати аналіз предметної області та програмних засобів для ведення блогів;
- розробити модель взаємодії користувача з програмним забезпеченням для багатокористувацького блогінгу;
- здійснити проєктування програмного забезпечення для багатокористувацького блогінгу;
- створити програмне забезпечення для багатокористувацького блогінгу;
- виконати тестування розробленого програмного забезпечення для багатокористувацького блогінгу.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Програмне забезпечення для ведення блогів

Блог являє собою Інтернет ресурс, призначений для поширення авторського контенту, що може охоплювати особисті міркування, аналітичні матеріали, навчальні тексти або інші тематичні публікації. Спочатку такий формат нагадував електронний щоденник, проте з часом перетворився на повноцінний інструмент комунікації та просування, який здатен приносити стабільний дохід. Сам термін походить від слова веблог, що вказує на його первинну природу як хронологічного онлайн записника [1].

На відміну від звичайних вебсайтів, блоги зазвичай оновлюються регулярно, а матеріали подаються у зворотному хронологічному порядку, де першим відображається найсвіжіший допис. Формат переважно підтримує неформальний або розмовний стиль викладу, а також інтерактивні можливості на кшталт коментарів та поширення у соціальних мережах. Тематика може бути будь-якою — від особистих розповідей до докладних інструкцій або рецензій [1].

Блог може бути як окремим ресурсом, так і складовою частиною більшого сайту. Наприклад, корпоративний портал здатен містити розділ блогу, щоб привертати увагу відвідувачів, розширювати інформування про діяльність і зміцнювати зв'язок із аудиторією. На відміну від статичних сайтів, що орієнтовані переважно на презентацію товарів чи послуг, блог зосереджений на формуванні інтересу через контент [1], [2].

Для початку роботи обирається програмна платформа, після чого створюється обліковий запис, налаштовується інтерфейс і публікуються перші матеріали. Окрім тексту, можливе додавання візуальних матеріалів і мультимедіа, а також інтеграція з соціальними мережами для розширення охоплення. Коли кількість відвідувачів досягає значного рівня, з'являється потенціал монетизації через рекламні мережі, партнерські програми або реалізацію власних продуктів [1], [2].

Блог слугує ефективним інструментом для досягнення різноманітних особистих і професійних цілей. Його використання дозволяє формувати позитивний імідж в онлайн середовищі, розширювати впізнаваність та зміцнювати власний бренд. Завдяки регулярному створенню якісного контенту можна закріпити репутацію експерта, що відкриває шлях до нових можливостей, включно з професійними запрошеннями, замовленнями чи співпрацею на вигідних умовах [1], [2].

Цей формат дає змогу отримувати стабільний дохід через різні джерела монетизації, зокрема партнерські програми, продаж власної продукції або навчальних матеріалів. Постійна робота над блогом сприяє розвитку навичок письма, редагування, пошукової оптимізації та ефективного використання соціальних мереж. Додатковою перевагою є встановлення професійних контактів, обмін досвідом і налагодження співпраці з іншими учасниками галузі [1], [2].

Ведення блогу може стати дієвим способом залучення відвідувачів до Інтернет ресурсу та перетворення їх на клієнтів за допомогою релевантних матеріалів, орієнтованих на цільову аудиторію. Цей інструмент ефективно працює як для розвитку особистих проєктів, так і для просування бізнесу, сприяючи одночасно зростанню впливу, доходів та кола зацікавлених читачів [1], [2].

Різноманітність форматів дає змогу обрати тематику відповідно до власних інтересів і цілей. Можливе створення матеріалів про подорожі, особисті події, модні тенденції, здоров'я, виховання, певні професійні ніші або корпоративні новини. Кожен напрям має свій стиль подачі та аудиторію, а правильно підібрана тематика дозволяє ефективно поєднати самовираження з фінансовими результатами [1], [2].

Ведення блогу розглядається як діяльність, що має як позитивні, так і негативні сторони. Для одних це можливість обмінюватися думками, формувати власну аудиторію та отримувати дохід через рекламу, продаж товарів чи послуг або використання партнерських програм. Для інших — це

заняття, яке потребує занадто багато часу та не завжди приносить очікувані результати. Через блог можна налагодити комунікацію з людьми зі схожими інтересами, підвищити впізнаваність і закріпити статус фахівця в певній темі. Крім того, цей інструмент здатен сприяти зростанню відвідуваності вебресурсу та залученню потенційних клієнтів за рахунок оптимізації публікацій під пошукові системи. Паралельно з цим розвиваються навички роботи з текстами, пошуковою оптимізацією, кодуванням і комунікацією в професійному середовищі [1], [2].

Разом із тим процес підтримання блогу вимагає систематичних зусиль і значного часу, а конкуренція серед онлайн платформ є надзвичайно високою. Постійна потреба в оновленні контенту та просуванні матеріалів може стати виснажливою, а перші результати часто з'являються лише після тривалої роботи. Через велику кількість активних блогів складно привернути увагу та вирізнитися серед інших авторів [2].

Для початку роботи необхідно обрати підходящий інструмент. Платформа для блогінгу — це спеціальне програмне середовище, яке спрощує створення та адміністрування блогу. Серед відомих варіантів можна виділити WordPress, який вважається найпопулярнішою системою керування контентом завдяки простоті використання та широкому набору функцій. Medium орієнтований на тих, хто прагне публікувати тексти та ділитися ідеями з широкою читацькою аудиторією. Blogger від Google пропонує легке налаштування та зручність для новачків. Wix вирізняється візуальним редактором із функцією перетягування, що дозволяє створювати професійно оформлені сторінки без знання коду. Weebly забезпечує готові шаблони та інструменти для оптимізації, поєднуючи можливості блогу з функціями Інтернет магазину [1]-[3].

Створення власного блогу починається з вибору тематики, що здатна викликати стійкий інтерес у потенційної аудиторії та приносити прибуток через пов'язані товари чи послуги. Для цього зазвичай аналізуються власні

захоплення, досвід і знання, після чого перевіряється рівень попиту за допомогою інструментів дослідження ключових слів [2], [3].

Після визначення напряму розвитку відбувається формування спільноти читачів. Це досягається завдяки створенню змістовних матеріалів, що допомагають розв'язувати конкретні проблеми. Контент може мати різні форми, від текстових публікацій до відео чи електронних книг. Надання корисної інформації підвищує довіру й стимулює інтерес до запропонованих продуктів або послуг [2], [3].

Коли аудиторія вже сформована, постає питання монетизації. Для отримання прибутку можуть використовуватися прямі продажі, рекламні оголошення, партнерські програми чи спонсорські матеріали. Ефективність цих способів залежить від активності читачів і якості взаємодії з ними [2], [3].

Щоб блог залишався помітним серед інших ресурсів, важливим етапом є його просування. Значна частина зусиль спрямовується на збільшення видимості публікацій через соціальні мережі, гостьові статті, відповіді на тематичних програмних платформах та формування підписної бази електронної пошти [2], [3].

Систематичність публікацій, базові знання та вміння працювати з ключовими словами та створювати зворотні посилання допомагають утримувати стабільний трафік. Використання цифрових інструментів може прискорити підготовку матеріалів і покращити планування, проте контент повинен бути унікальним і мати реальну цінність для читача. Процес популяризації блогу вимагає часу й терпіння, оскільки результати зазвичай з'являються поступово, із зростанням довіри та лояльності аудиторії [2], [3].

У сучасному цифровому середовищі програми для блогінгу стали ключовим інструментом, який забезпечує створення, організацію та публікацію матеріалів без потреби у глибоких технічних знаннях. Такі платформи беруть на себе значну частину складних завдань, зокрема оптимізацію під пошукові системи, оформлення публікацій та налаштування

дизайну, надаючи при цьому можливість адаптувати ресурс під конкретні цілі та стиль автора [2], [3].

Вибір правильної платформи впливає на ефективність роботи блогу, його масштабованість та здатність залучати аудиторію. Інтуїтивно зрозумілий інтерфейс і широкий набір функцій допомагають як новачкам, так і досвідченим авторам публікувати якісний контент, оптимізувати його для SEO, підтримувати взаємодію з читачами та монетизувати діяльність [2], [3].

Програмні рішення для блогів охоплюють увесь процес ведення ресурсу — від первинного налаштування до подальшого аналізу відвідуваності та вдосконалення стратегії. Вони забезпечують зручність у керуванні матеріалами, підвищують креативний потенціал і допомагають розширити охоплення [2], [3].

Різноманіття доступних програмних інструментів дозволяє знайти оптимальний варіант для будь-яких завдань — від особистих проєктів до масштабних корпоративних сайтів. Обізнаність щодо особливостей роботи таких систем і розуміння ключових функцій дає змогу обрати рішення, яке найбільш відповідає вимогам та допоможе вивести блог на новий рівень розвитку [2], [3].

Програмне забезпечення (ПЗ) для ведення блогів являє собою інструмент, призначений для спрощення процесу створення, упорядкування та поширення публікацій у мережі. Таке ПЗ дозволяє зосередитися на змісті матеріалів, усуваючи потребу у складних технічних діях, пов'язаних з налаштуванням та підтримкою вебресурсу. Завдяки таким програмним рішенням стає можливим представлення ідей, досвіду та аналітичних матеріалів у впорядкованому та доступному форматі, орієнтованому на широку аудиторію [2], [3].

Різні варіанти таких систем відрізняються рівнем складності та набором функцій, що визначає їх придатність для певних завдань — від особистих проєктів до великих інформаційних платформ. Їхнє використання передбачає наявність засобів для структурування контенту, редагування

текстів із візуальним відображенням результату, застосування шаблонів для оформлення, оптимізації матеріалів для пошукових систем та підключення додаткових інструментів [2], [3].

У таких платформах часто передбачено функції для збору та аналізу статистики відвідувань, що допомагає глибше розуміти інтереси читачів і коригувати стратегію публікацій. За принципом організації роботи подібні рішення можуть надаватися як у форматі повністю обслуговуваних сервісів, так і у вигляді систем, що встановлюються на власні сервери, що розширює можливості контролю, але потребує більшої технічної підготовки [2], [3].

Окрему категорію становлять програмні проєкти з відкритим кодом, які дають змогу змінювати та розширювати функціонал відповідно до індивідуальних вимог. У підсумку такі інструменти виступають основою для ефективної роботи з блогами, поєднуючи технічну базу з можливостями для творчого вираження та просування контенту [2], [3].

Програмні рішення, орієнтовані на ведення блогів, створюються з урахуванням потреб публікації та підтримки текстового й мультимедійного контенту, тоді як інструменти для побудови вебресурсів мають ширший функціонал, охоплюючи різні формати онлайн проєктів. У першому випадку акцент робиться на зручності створення матеріалів, організації публікацій та оптимізації для пошуку. У другому – на можливості формування повноцінної структури сайту з багатьма розділами та додатковими сервісами [3], [4].

Серед основних переваг спеціалізованих рішень для блогінгу вирізняється інтуїтивність використання, що дозволяє зосередитися на змісті, а не на технічних налаштуваннях. Такі інструменти зазвичай пропонують готові шаблони, інтегровані можливості оптимізації та додаткові розширення, завдяки чому процес роботи з контентом стає швидшим і гнучкішим. За потреби можна масштабувати функціонал, налаштовуючи його під зростання аудиторії та зміни у форматі публікацій [3], [4].

Подібні платформи виконують роль системи, яка спрощує керування матеріалами та процесом їхнього розміщення, позбавляючи необхідності

працювати безпосередньо з кодом чи серверною частиною. Запуск нового проєкту зазвичай передбачає створення облікового запису, вибір візуального оформлення та налаштування елементів дизайну відповідно до стилю автора чи бренду. Завдяки цьому блог можна швидко підготувати до роботи та розпочати регулярну публікацію матеріалів, зберігаючи при цьому повний контроль над їхнім виглядом і структурою [3], [4].

Створення та впорядкування матеріалів у блозі зазвичай здійснюється через спеціальні онлайн інструменти, які дають змогу працювати з текстом у зручному візуальному форматі без потреби володіння мовами розмітки. Такі середовища надають можливість додавати текстові блоки, візуальні елементи та мультимедіа, а також налаштовувати їхній вигляд і розташування. Поряд із цим забезпечується підтримка форматування, що дає змогу змінювати структуру та стиль вмісту [3], [4].

До текстових матеріалів можна приєднувати зображення, аудіо, відео та інші ресурси, які зберігаються в інтегрованих бібліотеках і можуть автоматично оптимізуватися для швидкого завантаження. Завдяки цьому поліпшується робота сторінки та зберігається зручність для відвідувачів [4].

Для впорядкування записів використовуються тематичні групи та ключові позначення, що дають змогу об'єднувати публікації за змістом і полегшувати пошук. Передбачено також інструменти планування, які дозволяють визначати час і дату автоматичної появи матеріалу [3], [4].

Перед розміщенням у відкритому доступі зазвичай доступний попередній перегляд, що допомагає перевірити відображення тексту та медіа на різних типах пристроїв. Коли підготовка завершена, розміщення здійснюється одним натисканням, після чого матеріал стає доступним для читачів. Додатково надаються можливості оптимізації для пошукових систем, зокрема налаштування описів, адрес сторінок і характеристик зображень, що підвищує видимість контенту в Інтернеті [3], [4].

У процесі функціонування блогу велика частина роботи виконується непомітно для користувача. Усе зберігається в упорядкованому вигляді, щоб

за потреби можна було швидко отримати будь-який матеріал та відобразити його у відкритому доступі. Системи зберігають записи, медіафайли та допоміжні дані у структурованому форматі, що спрощує їх обробку та пошук. Доступність ресурсу в мережі забезпечується розміщенням на спеціальних серверах. Іноді цей процес відбувається у межах самої платформи, іноді вимагає залучення зовнішніх рішень. Паралельно впроваджуються механізми безпеки, які захищають дані від несанкціонованого доступу, а також автоматичне створення копій вмісту для можливості відновлення [3], [4].

Щоб робота сайту залишалася швидкою, застосовуються технології прискорення завантаження сторінок і розподілу вмісту через глобальні мережі, завдяки чому відвідувачі отримують інформацію з найзручнішого географічно джерела [3], [4].

Після запуску ресурс потребує постійного супроводу. Використовуються інструменти для відстеження відвідуваності, аналізу взаємодії аудиторії та оцінки ефективності матеріалів. Надаються можливості для зворотного зв'язку з читачами, обговорень і поширення публікацій у соціальних сервісах, що допомагає розширювати охоплення [3], [4].

Робота такої системи поєднує візуальну частину, яку бачить відвідувач, і внутрішні процеси, які забезпечують стабільність, швидкодію та захист. Усе це створює умови, за яких основна увага зосереджується на змісті, а не на технічних деталях [4], [5].

Під час обрання платформи для ведення блогу значення має відповідність інструменту завданням та рівню підготовки користувача. Від функціоналу, який пропонує сервіс, залежить легкість створення, редагування та вдосконалення публікацій. Важливо враховувати, наскільки зручно виконувати базові дії, які можливості налаштування доступні та чи передбачено масштабування під майбутні потреби [4], [5].

Значну роль відіграє інтерфейс, що впливає на швидкість освоєння інструменту. Для новачків особливо корисною є логічна структура меню та прості засоби редагування, тоді як досвідчені користувачі віддають перевагу

розширеним налаштуванням та доступу до технічних параметрів. Наявність візуальних конструкторів спрощує адаптацію зовнішнього вигляду сайту, дозволяючи змінювати структуру сторінок без знання коду [4], [5].

Можливості персоналізації визначають унікальність ресурсу. Платформи, що підтримують зміну шаблонів і тем, дають змогу налаштувати дизайн під індивідуальний стиль або тематику. Для більш глибоких змін цінується підтримка плагінів чи додаткових модулів, завдяки яким сайт можна розширити новими функціями. Деякі системи також допускають інтеграцію власного коду, що відкриває шлях до повного контролю над структурою та поведінкою блогу [4], [5].

Оптимізація для пошукових систем та маркетингові можливості відіграють важливу роль у підвищенні видимості блогу та залученні відвідувачів без додаткової реклами. Вибрана платформа повинна пропонувати інструменти для налаштування контенту під алгоритми пошукових систем або мати можливість інтегруватися з відповідними сервісами, що спрощують оптимізацію [4], [5].

Інструменти для пошукової оптимізації можуть автоматично підбирати параметри оптимізації або давати змогу користувачеві регулювати ключові елементи контенту, включно із заголовками та описами. Деякі системи також забезпечують обмежену автоматичну оптимізацію швидкості завантаження та адаптивності для мобільних пристроїв [4], [5].

Маркетинговий аспект включає легкість обміну публікаціями через соціальні канали, що дозволяє залучати аудиторію безпосередньо з популярних мереж. Інтеграція з соціальними платформами спрощує планування поширення контенту та економить час на публікації. Додатково цінним є зв'язок із інструментами розсилок, що дозволяє збирати контакти відвідувачів і формувати регулярну комунікацію з аудиторією [4], [5].

Фінансові міркування теж впливають на вибір платформи. Деякі системи доступні безкоштовно та дозволяють розпочати роботу з мінімальними витратами, хоча можуть мати обмеження щодо дизайну та

функцій. Інші пропонують платні пакети з розширеними можливостями, включно з персональними доменами, професійними темами та інструментами оптимізації. Витрати на хостинг та обслуговування залежать від моделі платформи: деякі включають ці послуги у вартість підписки, тоді як інші вимагають окремої оплати зовнішніх ресурсів [5], [6].

Для ефективного використання платформи важлива підтримка та наявність навчальних матеріалів. Платформи з доступною технічною підтримкою та детальною документацією дозволяють швидко вирішувати проблеми та освоювати нові функції. Активні спільноти користувачів також сприяють обміну досвідом та допомагають знаходити рішення для складних завдань, що виникають у процесі ведення блогу [5], [6].

Використання спеціалізованих програмних систем для ведення блогів забезпечує широкі можливості для створення та керування контентом, роблячи цей процес більш ефективним та професійним. Такі програмні платформи дозволяють організовувати роботу з матеріалами без складних технічних знань, що робить їх доступними як для початківців, так і для досвідчених авторів. Завдяки інтегрованим інструментам управління та дизайну, створення публікацій та оформлення сторінок стає інтуїтивно зрозумілим і швидким [5], [6].

Простота роботи проявляється в наявності зручних панелей керування, що дозволяють легко переходити між різними розділами та функціями, від написання тексту до налаштування зовнішнього вигляду блогу. Більшість систем не вимагає володіння мовами програмування, оскільки забезпечує можливість перетягування елементів та використання готових макетів, що значно прискорює процес запуску блогу [5], [6].

Можливості налаштування дозволяють підбирати оформлення та функціональні компоненти відповідно до індивідуальних потреб і стилю. Доступні різноманітні шаблони та теми, які можна редагувати за кольорами, шрифтами та структурою сторінок, а також додавати віджети та розширення для інтерактивності, інтеграції з соціальними мережами або електронною

комерцією. Оптимізація для пошукових систем забезпечує підвищення видимості контенту, полегшує залучення органічного трафіку та покращує рейтинг у пошукових результатах. Такі інструменти дозволяють налаштовувати ключові елементи сторінок, автоматично генерувати описи та аналізувати ефективність контенту. Додатково платформи підтримують інтеграції для більш просунутих налаштувань, включно з управлінням внутрішніми посиланнями або створенням карт сайту [5], [6].

Швидкість завантаження та адаптивність під мобільні пристрої також є важливими перевагами. Оптимізовані платформи гарантують комфортне відображення на будь-яких екранах, що підвищує користувацький досвід і сприяє кращій індексації в пошукових системах. У сукупності всі ці можливості роблять процес ведення блогу більш професійним, зручним і продуктивним, дозволяючи зосередитися на створенні цікавого та якісного контенту [5], [6].

Програмні системи для ведення блогів полегшують організацію та управління матеріалами, забезпечуючи можливість планувати публікації, структурувати контент та зручно керувати медіафайлами. Завдяки таким інструментам ведення блогу стає більш послідовним і структурованим, що дозволяє підтримувати регулярність і порядок у розміщенні нових матеріалів. Автоматичне планування публікацій дозволяє створювати контент заздалегідь і задавати час його появи на сайті без потреби вручну запускати процес. Категоризація та маркування матеріалів допомагають легко знаходити пов'язані публікації та покращують навігацію для читачів. Керування медіафайлами здійснюється через інтегровані сховища, де можна зберігати та додавати зображення, відео та інші елементи, що робить блог візуально привабливим без додаткових зусиль [5], [6].

Захист контенту та даних користувачів забезпечується за допомогою різноманітних функцій безпеки. Регулярне автоматичне резервне копіювання гарантує збереження матеріалів і можливість їх відновлення у разі проблем. Шифрування даних через сертифікати безпеки забезпечує безпечну взаємодію

з платформою, а вбудовані інструменти запобігають поширеним загрозам, включно з атаками шкідливого програмного забезпечення та спробами несанкціонованого доступу [5], [6].

Доступність таких систем робить їх економічно привабливими для користувачів з різними бюджетними можливостями. Є варіанти без початкових вкладень, що дозволяють почати роботу без витрат, а також платні плани, які пропонують розширені функції, масштабованість та можливість інтеграції з додатковими сервісами [5], [6].

Програмні платформи для блогів також підтримують активні спільноти користувачів і надають широкий спектр ресурсів для навчання. Документація та навчальні матеріали дозволяють швидко освоїти основні та розширені функції, а служби підтримки забезпечують оперативну допомогу при виникненні технічних питань. Спільноти користувачів сприяють обміну досвідом, порадами та ідеями, що робить процес ведення блогу більш комфортним і надихає на розвиток контенту [5], [6].

Незважаючи на численні переваги платформ для ведення блогів, включно з легкістю використання, гнучкими налаштуваннями та можливостями пошукової оптимізації, слід враховувати існування певних обмежень. До них належать обмежена свобода дизайну, складнощі з масштабуванням, потенційні загрози безпеки [5], [6].

Визначено, що блог являє собою Інтернет ресурс, призначений для поширення авторського контенту, що може охоплювати особисті міркування, аналітичні матеріали, навчальні тексти або інші тематичні публікації. На відміну від звичайних вебсайтів, блоги зазвичай оновлюються регулярно, а матеріали подаються у зворотному хронологічному порядку, де першим відображається найсвіжіший допис. Програмне забезпечення для ведення блогів являє собою інструмент, призначений для спрощення процесу створення, упорядкування та поширення публікацій у мережі. Таке ПЗ дозволяє зосередитися на змісті матеріалів, усуваючи потребу у складних технічних діях, пов'язаних з налаштуванням та підтримкою вебресурсу.

1.2 Аналіз програмного забезпечення для ведення блогів

Проаналізуємо програмне забезпечення для ведення блогів [7]-[15].

Програмне забезпечення Tumblr (рис. 1.1) є сервісом, що поєднує функціонал блогінгу та соціальної мережі, орієнтованим на публікацію коротких і візуально привабливих матеріалів. Платформа Tumblr дозволяє розміщувати як текстові записи, так і мультимедійний контент, включно з фотографіями, гіперпосиланнями, GIF-зображеннями, аудіо та відео файлами. Основний акцент у програмному забезпеченні Tumblr зроблено на простоті взаємодії та швидкості публікації, що робить сервіс привабливим для ведення особистих блогів у формі хобі [7], [12], [13].

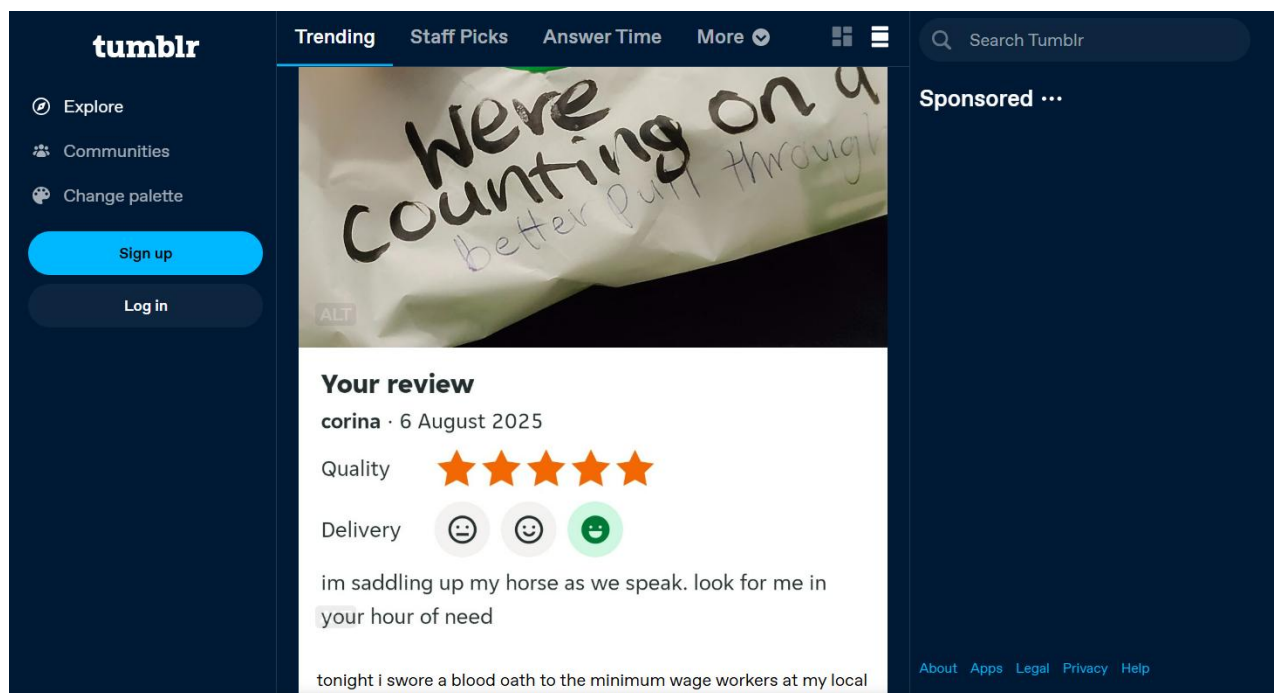


Рисунок 1.1 – Програмне забезпечення Tumblr [13]

Використання ресурсу Tumblr не передбачає витрат на хостинг або доступ до основного функціоналу. Підключення власного доменного імені потребує окремого оформлення через реєстратора, але інтеграція його з платформою виконується без додаткових платежів. Дизайн блогу можна

налаштовувати шляхом вибору серед безкоштовних шаблонів або придбання спеціальних тем різної вартості [7], [12], [13].

Платформа Tumblr вирізняється соціальною складовою, зокрема завдяки використанню хештегів, що сприяє знаходженню контенту іншими користувачами, а також наявності активних спільнот, присвячених популярним темам. Окрім базового функціоналу, запроваджено можливість отримання доходу через платні підписки на контент, хоча загалом ресурс не позиціонується як комерційно орієнтований інструмент і не має розвинених засобів монетизації [7], [12], [13].

Основними характеристиками програмного забезпечення Tumblr є такі [7], [12], [13]:

- призначення: програмне забезпечення Tumblr може використовуватися для публікації коротких текстів і мультимедійних матеріалів, що поєднує функції блогінгу та соціальної мережі [7], [12], [13];

- доступність: використання програмного забезпечення Tumblr безкоштовне, без витрат на хостинг або основний функціонал [7], [12], [13];

- налаштування дизайну: програмне забезпечення Tumblr надає можливість вибору безкоштовних тем або придбання преміальних шаблонів для персоналізації блогу [7], [12], [13];

- соціальна інтеграція: програмне забезпечення Tumblr підтримує застосування хештегів та активних спільнот для полегшення пошуку та обміну контентом [7], [12], [13];

- монетизація: програмне забезпечення Tumblr забезпечує наявність платних підписок і партнерських посилань, хоча платформа не орієнтована на комерційне використання [7], [12], [13];

- обмеження: відсутність плагінів, додаткових розширень та інструментів пошукової оптимізації [7], [12], [13];

- зручність використання: програмне забезпечення Tumblr надає простий інтерфейс, забезпечує швидке створення блогу та інтуїтивну навігацію користувачів [7], [12], [13].

Обмеження сервісу Tumblr полягають у відсутності плагінів, розширень та вбудованих інструментів пошукової оптимізації, що звужує можливості для професійного розвитку блогу. Незважаючи на це, Tumblr приваблює своєю простотою, інтуїтивно зрозумілим інтерфейсом та дружньою атмосферою спільнот, що робить його оптимальним вибором для творчого самовираження і пошуку однодумців, а не для побудови комерційних проєктів [7], [12], [13].

Програмне забезпечення Medium (рис. 1.2) є онлайн платформою, що дозволяє створювати особистий профіль та публікувати матеріали, які тут називають історіями. Опублікований контент демонструється читачам, зацікавленим у темі, та може охоплювати широку аудиторію завдяки налаштованим алгоритмам відбору [7], [12], [14].

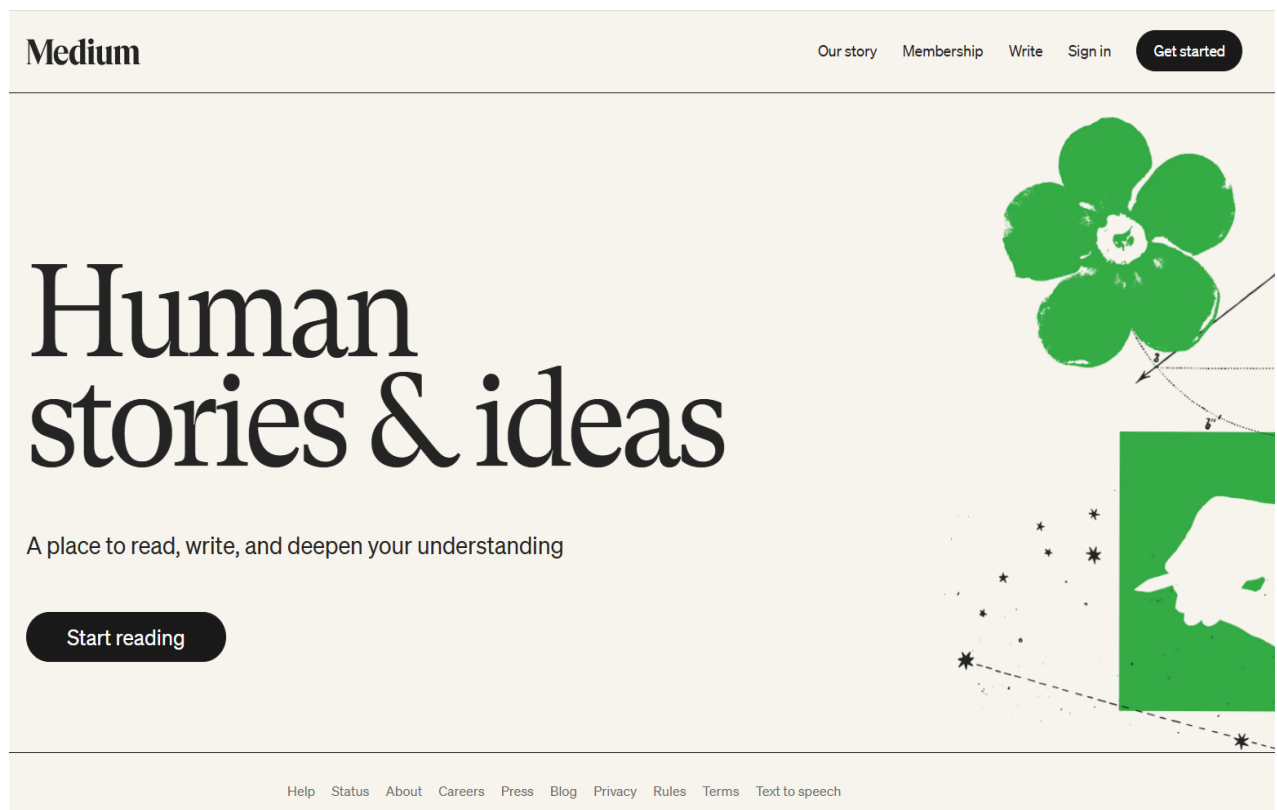


Рисунок 1.2 – Програмне забезпечення Medium [14]

Програмне забезпечення Medium особливо цінується авторами, які прагнуть поділитися досвідом або розповідями у форматі сторітелінгу, що

активно підтримується середовищем платформи. Існує можливість створювати публікації, тобто колекції текстів одного чи кількох авторів, які можуть виглядати як онлайн журнали, і ця функція стає доступною за наявності членства. Для монетизації передбачена партнерська програма», що надає змогу отримувати винагороду за публікації за умови наявності достатньої кількості підписників. Писати матеріали можна безкоштовно, але розширені можливості, зокрема налаштування публікацій і необмежений доступ до читання інших текстів, відкриваються після оформлення платного членства. Також можна підключити власний домен, придбавши його у реєстратора, без додаткової плати платформі [7], [12], [14].

Програмне забезпечення Medium вирізняється високим рівнем авторитету у пошукових системах, що підвищує шанси публікацій отримати хороші позиції в результатах пошуку. Водночас можливості оформлення обмежені, відсутні сторонні інструменти та теми, а також діють правила, що забороняють певні типи контенту. Платформа підходить авторам, які прагнуть зручно й швидко розмістити власні роботи онлайн, зокрема тим, хто хоче продемонструвати тексти потенційним клієнтам без необхідності створення окремого вебсайту [7], [12], [14].

Програмне забезпечення Medium має такі особливості [7], [12], [14]:

- формат публікацій: контент подається у вигляді історій, що дозволяє створювати матеріали різної тематики та обсягу [7], [12], [14];
- підтримка написання історій: програмне забезпечення Medium сприяє розвитку спеціального стилю написання публікацій, що приваблює аудиторію та підвищує залученість користувачів [7], [12], [14];
- створення публікацій: доступна функція програмного забезпечення Medium для об'єднання статей одного чи кількох авторів у єдиний інформаційний ресурс [7], [12], [14];
- партнерська програма: у програмному забезпеченні Medium передбачена можливість монетизації матеріалів за умови досягнення мінімальної кількості підписників [7], [12], [14];

- доменне підключення: можна використовувати власне доменне ім'я для профілю або публікацій без додаткової оплати платформі [7], [12], [14];
- авторитет у пошуку: високий рейтинг домену сприяє кращій видимості публікацій у результатах пошукових систем [7], [12], [14];
- обмеження дизайну: можливості налаштування зовнішнього вигляду контенту є мінімальними [7], [12], [14];
- відсутність доповнень: сторонні інструменти та розширення не підтримуються, функціонал програмного забезпечення Medium визначається обраним типом облікового запису [7], [12], [14];
- політика контенту: у програмному забезпеченні Medium встановлено заборону на публікацію матеріалів певного характеру, включно з темами азартних ігор [7], [12], [14];
- цільова аудиторія: програмне забезпечення Medium підходить авторам, що прагнуть швидко розмістити роботи та продемонструвати їх потенційним читачам чи замовникам [7], [12], [14].

Серед недоліків програмного забезпечення Medium можна відзначити обмежені можливості персоналізації оформлення, відсутність доступу до тем і шаблонів, а також суворі правила щодо вмісту, які забороняють розміщення матеріалів певної категорії. Крім того, читачі під час перегляду можуть легко переключитися на інші публікації, запропоновані платформою, що знижує утримання уваги на конкретному авторі, а відсутність інтеграції зі сторонніми інструментами ускладнює розширення функціоналу блогу [7], [12], [14].

Програмне забезпечення Write.as (рис. 1.3) є платформою для блогінгу, створеною для тих, хто цінує простоту та відсутність візуального перевантаження. Її мінімалістичний підхід дозволяє зосередитися виключно на змісті, не відволікаючись на складні налаштування чи дизайн. Інтерфейс працює за принципом простого текстового редактора з автоматичним збереженням, що робить його зручним для ведення особистих нотаток або публікації матеріалів у форматі віртуального журналу [12], [15].

write.as

Type words, put them on the internet.

Start writing

Write an essay without signing up,
or [sign up](#) now.

Рисунок 1.3 – Програмне забезпечення Write.as [15]

Програмне забезпечення Write.as не підтримує рекламу та доступна безкоштовно, проте передбачає додаткові платні функції. Для роботи з текстом використовується система розмітки, яка передбачає застосування спеціальних символів для створення заголовків, списків та посилань. Освоєння цього синтаксису не є складним, але потребує певної практики, що може стати бар'єром для тих, хто звик працювати з кнопками форматування. Завдяки своїй легкості та стриманому оформленню програмне забезпечення Write.as часто обирають автори, яким потрібна тиха, відокремлена від зайвих елементів простір для творчості, а також команди, що бажають обмінюватися матеріалами в режимі онлайн [12], [15].

Програмне забезпечення Write.as має такі особливості [12], [15]:

- мінімалістичний дизайн: інтерфейс програмного забезпечення Write.as побудований таким чином, щоб не відволікати користувача від створення контенту, зосереджуючи увагу лише на тексті [12], [15];
- відсутність реклами: у безкоштовній версії програмного забезпечення Write.as немає жодних рекламних блоків, що забезпечує чистоту та зручність перегляду матеріалів [12], [15];
- підтримка Markdown: для форматування тексту застосовується система розмітки, яка дозволяє створювати заголовки, списки та посилання за допомогою спеціальних символів [12], [15];

- автоматичне збереження: редактор зберігає всі зміни у реальному часі, що мінімізує ризик втрати даних під час роботи [12], [15];
- командна робота: у програмному забезпеченні Write.as передбачена можливість обміну матеріалами та спільної роботи над текстами за умови оформлення платної підписки [12], [15].

Серед недоліків програмного забезпечення Write.as можна відзначити необхідність опанування синтаксису системи розмітки для форматування тексту, що може ускладнити роботу користувачам, які звикли до візуальних редакторів. Також обмежені можливості налаштування зовнішнього вигляду блогів і відсутність розширених інструментів оптимізації, що робить платформу менш придатною для складних проєктів або комерційного використання [12], [15].

Порівняльну характеристику програмного забезпечення для багатокористувацького блогінгу наведено у таблиці 1.1.

Таблиця 1.1 – Порівняння програмного забезпечення для багатокористувацького блогінгу

Критерій порівняння	Tumblr [13]	Medium [14]	Write.as [15]
Простота використання	+	+	+—
Можливості монетизації	+—	+	—
Налаштування дизайну	+	+—	—
Підтримка медіаконтенту	+	+	+—
Відсутність реклами	+—	+	+
Придатність для командної роботи	—	+—	+

За результатами проведеного аналізу можна зробити висновок, що у наш час існує досить багато програмних засобів для ведення блогів. Проте деякі програмні засоби для ведення блогів можуть мати підвищену складність

адміністрування, оскільки забезпечення узгодженої взаємодії між численними користувачами потребує чітко налаштованих механізмів контролю доступу, модерації матеріалів та запобігання конфліктам при одночасному редагуванні. У разі відсутності належної оптимізації інтерфейсу та внутрішньої архітектури програми зростає ймовірність виникнення помилок під час публікації або оновлення вмісту блогу. Крім того, використання готових платформ із закритим вихідним кодом може обмежувати можливості налаштування функціоналу під специфічні потреби, що ускладнює інтеграцію з іншими інформаційними системами та сервісами. Тому актуальною є розробка програмного забезпечення для багатокористувацького блогінгу.

При розробці програмного забезпечення для багатокористувацького блогінгу необхідно забезпечити такі функціональні вимоги:

- підтримка можливості відображення публікацій у блозі;
- підтримка можливості роботи з програмою різних користувачів;
- можливість редагування інформації користувача в особистому профілі;
- можливість створення нових постів у блозі;
- підтримка можливості редагування та видалення власних постів користувача;
- підтримка можливості адміністратора створення, редагування та видалення інформації про користувачів програми.

1.3 Висновки за розділом 1

Визначено, що блог являє собою Інтернет ресурс, призначений для поширення авторського контенту, що може охоплювати особисті міркування, аналітичні матеріали, навчальні тексти або інші тематичні публікації. На відміну від звичайних вебсайтів, блоги зазвичай оновлюються регулярно, а матеріали подаються у зворотному хронологічному порядку, де першим відображається найсвіжіший допис. Програмне забезпечення для ведення

блогів являє собою інструмент, призначений для спрощення процесу створення, упорядкування та поширення публікацій у мережі. Таке ПЗ дозволяє зосередитися на змісті матеріалів, усуваючи потребу у складних технічних діях, пов'язаних з налаштуванням та підтримкою вебресурсу.

За результатами проведеного аналізу зроблено висновок, що у наш час існує досить багато програмних засобів для ведення блогів. Проте деякі програмні засоби для ведення блогів можуть мати підвищену складність адміністрування, оскільки забезпечення узгодженої взаємодії між численними користувачами потребує чітко налаштованих механізмів контролю доступу, модерації матеріалів та запобігання конфліктам при одночасному редагуванні. У разі відсутності належної оптимізації інтерфейсу та внутрішньої архітектури програми зростає ймовірність виникнення помилок під час публікації або оновлення вмісту блогу. Крім того, використання готових платформ із закритим вихідним кодом може обмежувати можливості налаштування функціоналу під специфічні потреби, що ускладнює інтеграцію з іншими інформаційними системами та сервісами. Тому актуальною є розробка програмного забезпечення для багатокористувацького блогінгу.

Сформульовано функціональні вимоги до програмного забезпечення для багатокористувацького блогінгу.

2 МАТЕРІАЛИ І МЕТОДИ

2.1 Вибір мови програмування

При розробці програмного забезпечення для багатокористувацького блогінгу було використано мову програмування Python [16], [17].

Мова програмування Python вирізняється універсальністю та здатністю ефективно застосовуватися в різних галузях розробки, починаючи від вебпрограмування і закінчуючи аналізом даних, машинним навчанням та автоматизацією бізнес-процесів. Її популярність зумовлена не лише простотою синтаксису, але й високою читабельністю коду, що полегшує командну розробку та супровід великих проєктів. Завдяки підтримці об'єктно-орієнтованого, функціонального та процедурного підходів Python забезпечує гнучкість у виборі архітектурних рішень і дає змогу поєднувати різні парадигми програмування в межах одного проєкту [16], [17].

Широка стандартна бібліотека та багатий вибір сторонніх модулів дозволяють швидко реалізовувати складні завдання без потреби у створенні функціоналу з нуля. Сумісність із сучасними технологіями, такими як хмарні сервіси, контейнери та мікросервісна архітектура, робить Python придатним для створення масштабованих і розподілених систем. Активна міжнародна спільнота розробників сприяє швидкому вдосконаленню мови, випуску оновлень та розвитку екосистеми інструментів [16], [17].

Мова Python підтримує кілька парадигм програмування, зокрема об'єктно-орієнтовану, функціональну та процедурну, що забезпечує гнучкість під час вибору архітектурного підходу та дозволяє поєднувати різні методи розробки в межах одного продукту. Гнучкість у поєднанні з відносно низьким порогом входу забезпечує доступність Python як для початківців, так і для досвідчених інженерів, а його стабільне поширення у сфері освіти, науки та комерційних проєктів свідчить про довготривалу перспективність використання цієї мови [16], [17].

Використання мови програмування Python для розроблення програмного забезпечення багатокористувацького блогінгу є доцільним завдяки поєднанню простоти синтаксису, широких можливостей інтеграції та розвиненої екосистеми бібліотек. Застосування цієї мови дає змогу скоротити час розробки за рахунок зрозумілої структури коду та наявності готових інструментів для реалізації ключових функцій, таких як управління користувачами, обробка запитів, робота з базами даних та забезпечення безпеки [16], [17].

Python має потужні фреймворки, зокрема Django та Flask, які дозволяють створювати вебзастосунки з підтримкою багатокористувацького режиму, впроваджувати систему ролей і прав доступу, інтегрувати механізми модерації та засоби захисту від несанкціонованих дій. Наявність численних модулів для роботи з API, хмарними сервісами та інструментами автоматизації забезпечує гнучкість під час розширення функціональності блогової платформи [16], [17].

Ефективність мови підсилюється наявністю спеціалізованих фреймворків, таких як Flask та Django, що надають засоби для організації багатокористувацької взаємодії, контролю прав доступу, модерації контенту та захисту від шкідливих дій. Значна кількість бібліотек для роботи з API, сервісами хмарного зберігання й автоматизаційними платформами відкриває широкі можливості для масштабування функціоналу та впровадження додаткових сервісів [16], [17].

Висока кросплатформність Python спрощує розгортання застосунків на різних операційних системах і хостинг середовищах, а активна спільнота розробників гарантує постійну підтримку та оновлення технологічної бази. Завдяки цим властивостям Python дозволяє створювати масштабовані, безпечні та зручні у супроводі рішення, що відповідають вимогам сучасних програмних систем для ведення блогів [16], [17].

Обґрунтування вибору мови програмування для реалізації програмного забезпечення для багатокористувацького блогінгу наведено у таблиці 2.1.

Таблиця 2.1 – Обґрунтування вибору мови програмування для реалізації програмного забезпечення для багатокористувацького блогінгу

Критерій порівняння мов програмування	Мова програмування		
	Python	PHP	JavaScript
Читабельність коду	+	+—	+—
Розвинена спільнота	+	+	+—
Інтеграція з сучасними технологіями	+	+—	+
Зручність для створення ПЗ для багатокористувацького блогінгу	+	—	—
Безпека	+	+—	+—

Отже, для реалізації програмного забезпечення для багатокористувацького блогінгу обрано мову програмування Python, що обумовлено поєднанням доступності синтаксису, розвинутої інфраструктури інструментів і бібліотек, а також високого рівня інтеграційних можливостей. Використання готових модулів дає змогу суттєво скоротити час реалізації ключових функцій, зокрема управління користувачами, обробки HTTP-запитів, взаємодії з базами даних та впровадження механізмів безпеки.

2.2 Вибір фреймворку для створення програмного забезпечення для багатокористувацького блогінгу

В якості фреймворку для створення програмного забезпечення для багатокористувацького блогінгу було обрано Flask [18], [19].

Фреймворк Flask вирізняється своєю універсальністю та здатністю адаптуватися до різноманітних завдань, починаючи від створення невеликих прототипів і закінчуючи повнофункціональними вебзастосунками з розподіленою архітектурою. Його модульна структура дозволяє розширювати

базовий функціонал за допомогою численних сторонніх бібліотек, що дає змогу впроваджувати складні механізми без перевантаження основного коду. Гнучкість налаштувань забезпечує можливість вибору оптимальних технологічних рішень для конкретного проєкту, зберігаючи при цьому простоту інтеграції з базами даних, сервісами аутентифікації та системами обробки медіаконтенту [18], [19].

Flask відзначається здатністю налаштовуватися під різні сценарії використання, від розробки простих концептуальних зразків до створення розгалужених вебсистем із комплексною архітектурою. Його побудова на модульних засадах дає можливість поступово розширювати базові можливості за рахунок додавання зовнішніх бібліотек, що сприяє впровадженню складних функцій без надмірного ускладнення коду. Гнучкість параметрів конфігурації відкриває простір для вибору найбільш ефективних технологічних рішень, зберігаючи водночас простоту взаємодії з базами даних, сервісами ідентифікації користувачів та інструментами обробки мультимедійного контенту [18], [19].

Завдяки легкій архітектурі Flask спрощує процес тестування та розгортання вебзастосунків, що особливо важливо при розробці продуктів, які потребують регулярного оновлення та швидкої адаптації до змін вимог. Його сумісність із сучасними підходами, такими як мікросервісна архітектура та контейнеризація, робить його ефективним інструментом для створення масштабованих рішень. Активна спільнота користувачів і розробників постійно доповнює екосистему новими розширеннями та прикладами, що сприяє швидкому впровадженню нових можливостей та підтриманню високого рівня надійності створених систем [18], [19].

Невелики та логічно побудована структура Flask полегшує як тестування, так і розгортання вебрішень, що має особливе значення для продуктів, котрі регулярно оновлюються або швидко адаптуються до змінних вимог. Підтримка сучасних підходів, зокрема контейнеризації та мікросервісної архітектури, робить його придатним для створення

масштабованих застосунків, які легко інтегруються в розподілені системи. Активна спільнота постійно доповнює екосистему новими інструментами, розширеннями та прикладами реалізації, що пришвидшує впровадження оновлень і підвищує стабільність готових продуктів [18], [19].

Використання фреймворку Flask для розроблення програмного забезпечення багатокористувацького блогінгу є доцільним завдяки його легкості, гнучкості та здатності забезпечувати високу швидкість створення вебзастосунків із налаштованою під потреби архітектурою. Його мінімалістичний підхід надає розробникові повний контроль над структурою проєкту та дозволяє інтегрувати лише ті компоненти, які необхідні для реалізації конкретних функцій, що особливо важливо у випадках, коли система повинна залишатися оптимізованою та масштабованою [18], [19].

Завдяки добре організованому прикладному інтерфейсу програмування (Application Programming Interface, API) та підтримці широкого спектра розширень Flask дозволяє безперешкодно реалізовувати авторизацію, керування ролями користувачів, обробку форм, взаємодію з базами даних та інтеграцію з зовнішніми сервісами. Це створює умови для ефективного впровадження механізмів багатокористувацької взаємодії, модерації та публікації контенту [18], [19].

Фреймворк забезпечує сумісність із сучасними технологіями та стандартами веброзробки, підтримує роботу з REST API, асинхронними запитами та хмарними платформами, що дозволяє створювати гнучкі та швидкодіючі рішення. У поєднанні з високою читабельністю коду на мові програмування Python та активною спільнотою фреймворку Flask стає надійною основою для розробки багатокористувацьких блогових програмних систем, здатних відповідати вимогам продуктивності, безпеки та зручності експлуатації користувачами [18], [19].

Обґрунтування вибору фреймворку для створення програмного забезпечення для багатокористувацького блогінгу наведено у таблиці 2.2.

Таблиця 2.2 – Обґрунтування вибору фреймворку для створення програмного забезпечення для багатокористувацького блогінгу

Критерій порівняння фреймворків	Фреймворки		
	Flask	Django	Pyramid
Гнучкість архітектури	+	+—	+
Швидкість розробки	+	+—	+—
Легкість налаштування	+	—	+—
Спільнота та підтримка	+	+	+—
Продуктивність для малих і середніх проєктів	+	+—	+—

Таким чином, для створення програмного забезпечення для багатокористувацького блогінгу обрано фреймворк Flask, що забезпечує високу продуктивність, надійність та гнучкість ПЗ, а також є надійною платформою для розробки сучасних програмних систем для підтримки блогів з багатокористувацьким доступом, високим рівнем безпеки та зручністю експлуатації.

2.3 Розробка моделі взаємодії користувача з програмним забезпеченням для багатокористувацького блогінгу

З метою можливості розробки програмного забезпечення для багатокористувацького блогінгу створимо відповідну модель у вигляді діаграми прецедентів.

При розробленні моделі взаємодії користувача з програмним забезпеченням для багатокористувацького блогінгу враховуємо те, що гості програми мають можливості дивитися контент. Користувач може додатково завантажувати контент, коментувати контент, скаржитися на контент, а також

виконувати авторизацію. Адміністратор може виконувати адміністрування блогу, зокрема, модерацію контенту, модерацію скарг, видалення контенту.

Основний успішний сценарій користувача при роботі з програмним забезпеченням для багатокористувацького блогінгу можна описати таким чином:

- користувач програмного забезпечення для багатокористувацького блогінгу авторизується;
- після авторизації користувача сервер програмного забезпечення для багатокористувацького блогінгу у відповідь повертає основну сторінку вебсайту для багатокористувацького блогінгу;
- користувач бачить відповідну сторінку програмного забезпечення для багатокористувацького блогінгу;
- користувач читає статті у блогах;
- користувач за бажанням коментує обрані статті у блогах;
- користувач завершує роботу з програмним забезпеченням для багатокористувацького блогінгу.

Опишемо основні дії адміністратора програмного забезпечення для багатокористувацького блогінгу:

- адміністратор програмного забезпечення для багатокористувацького блогінгу авторизується;
- після авторизації адміністратора сервер програмного забезпечення для багатокористувацького блогінгу у відповідь на запит повертає основну сторінку для багатокористувацького блогінгу;
- адміністратор програмного забезпечення для багатокористувацького блогінгу бачить відповідну сторінку та обирає блоги для модерації;
- адміністратор програмного забезпечення для багатокористувацького блогінгу модерує інформації та за необхідності видаляє контент, що не відповідає правилам;
- адміністратор програмного забезпечення для багатокористувацького блогінгу завершує роботу з програмою.

Враховуючи сценарії роботи користувачів програмного забезпечення для багатокористувацького блогінгу (гість, користувач, адміністратор) при роботі з програмою, модель взаємодії користувача з програмним забезпеченням для багатокористувацького блогінгу подамо у вигляді діаграми прецедентів, зображеної на рис. 2.1.

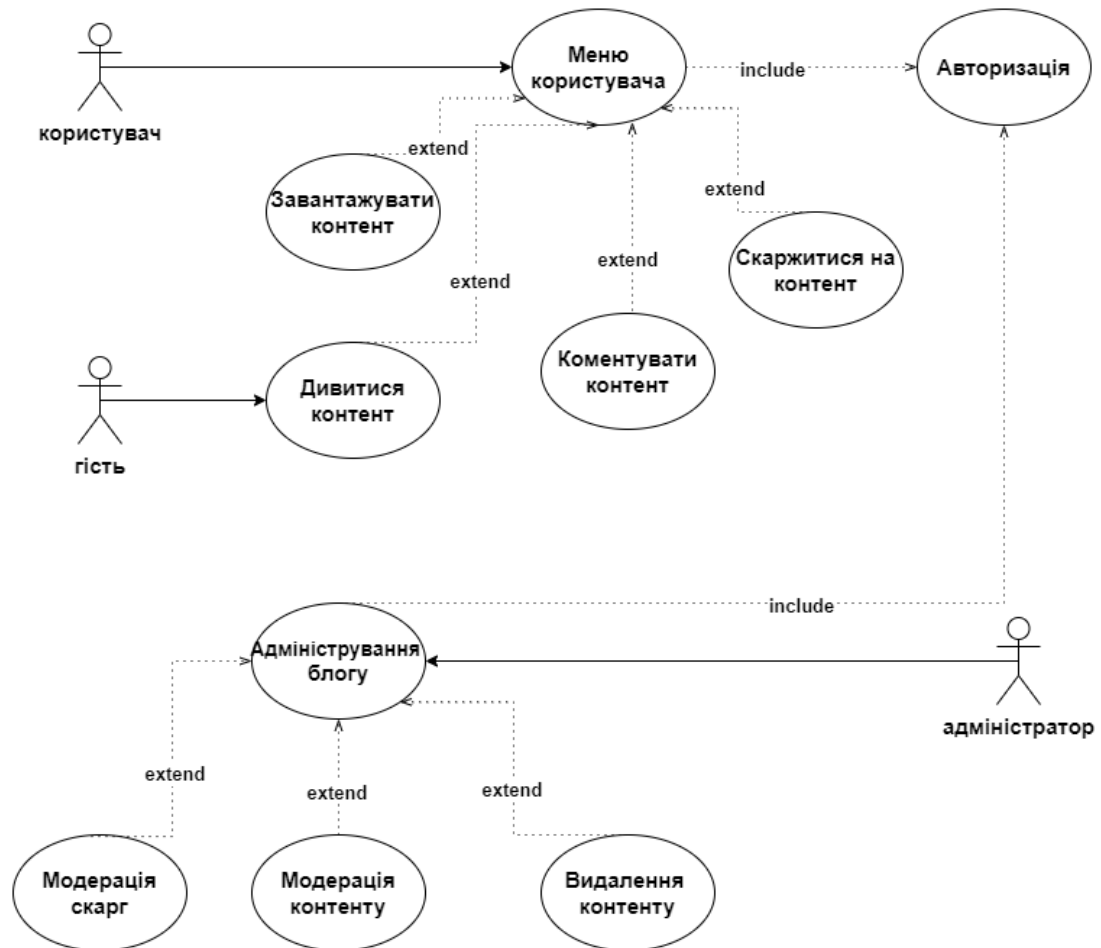


Рисунок 2.1 – Модель взаємодії користувача з програмним забезпеченням для багатокористувацького блогінгу

Діаграма прецедентів програмного забезпечення для багатокористувацького блогінгу відображає взаємодію трьох основних учасників системи (Гостя, Користувача та Адміністратора) із функціональними можливостями платформи. Гість є найменш привілейованим учасником і має доступ лише до перегляду опублікованого контенту. Цей

рівень взаємодії не передбачає авторизації та не надає можливостей для створення або редагування матеріалів.

Користувач, на відміну від Гостя, володіє розширеним набором дій. Окрім перегляду матеріалів, він може додавати власний контент, залишати коментарі, надсилати скарги на публікації, які порушують правила, а також проходити процедуру авторизації для отримання доступу до персоналізованих функцій. Така роль спрямована на активну участь у формуванні наповнення блогу змістом та підтримку інтерактивності спільноти.

Адміністратор виконує управлінські функції, пов'язані з підтриманням належної роботи системи та дотриманням правил спільноти. Його обов'язки охоплюють модерацію публікацій, розгляд і обробку скарг, а також видалення матеріалів, що не відповідають встановленим нормам. Таким чином, адміністратор відповідає за стабільність, безпеку та якість інформаційного середовища платформи.

Діаграма прецедентів відображає ці ролі та їхні взаємозв'язки з функціональними можливостями системи, демонструючи розподіл прав доступу та специфіку взаємодії кожного типу користувачів із програмним забезпеченням.

Таким чином, створено модель взаємодії користувача з програмним забезпеченням для багатокористувацького блогінгу, яка подана у вигляді діаграми прецедентів, що відображає ролі та їхні взаємозв'язки з функціональними можливостями програми, демонструючи розподіл прав доступу та специфіку взаємодії кожного типу користувачів із програмним забезпеченням.

2.4 Висновки за розділом 2

Для реалізації програмного забезпечення для багатокористувацького блогінгу обрано мову програмування Python, що обумовлено поєднанням доступності синтаксису, розвиненої інфраструктури інструментів і бібліотек, а

також високого рівня інтеграційних можливостей. Використання готових модулів дає змогу суттєво скоротити час реалізації ключових функцій, зокрема управління користувачами, обробки HTTP-запитів, взаємодії з базами даних та впровадження механізмів безпеки.

Для створення програмного забезпечення для багатокористувацького блогінгу обрано фреймворк Flask, що забезпечує високу продуктивність, надійність та гнучкість ПЗ, а також є надійною платформою для розробки сучасних програмних систем для підтримки блогів з багатокористувацьким доступом, високим рівнем безпеки та зручністю експлуатації.

Створено модель взаємодії користувача з програмним забезпеченням для багатокористувацького блогінгу, яка подана у вигляді діаграми прецедентів, що відображає ролі та їхні взаємозв'язки з функціональними можливостями програми, демонструючи розподіл прав доступу та специфіку взаємодії кожного типу користувачів із програмним забезпеченням.

3 ОПИС ПРОГРАМИ

3.1 Структура програмного забезпечення для багатокористувацького блогінгу

Структуру програмного забезпечення для багатокористувацького блогінгу наведено у вигляді діаграми класів на рис. 3.1.

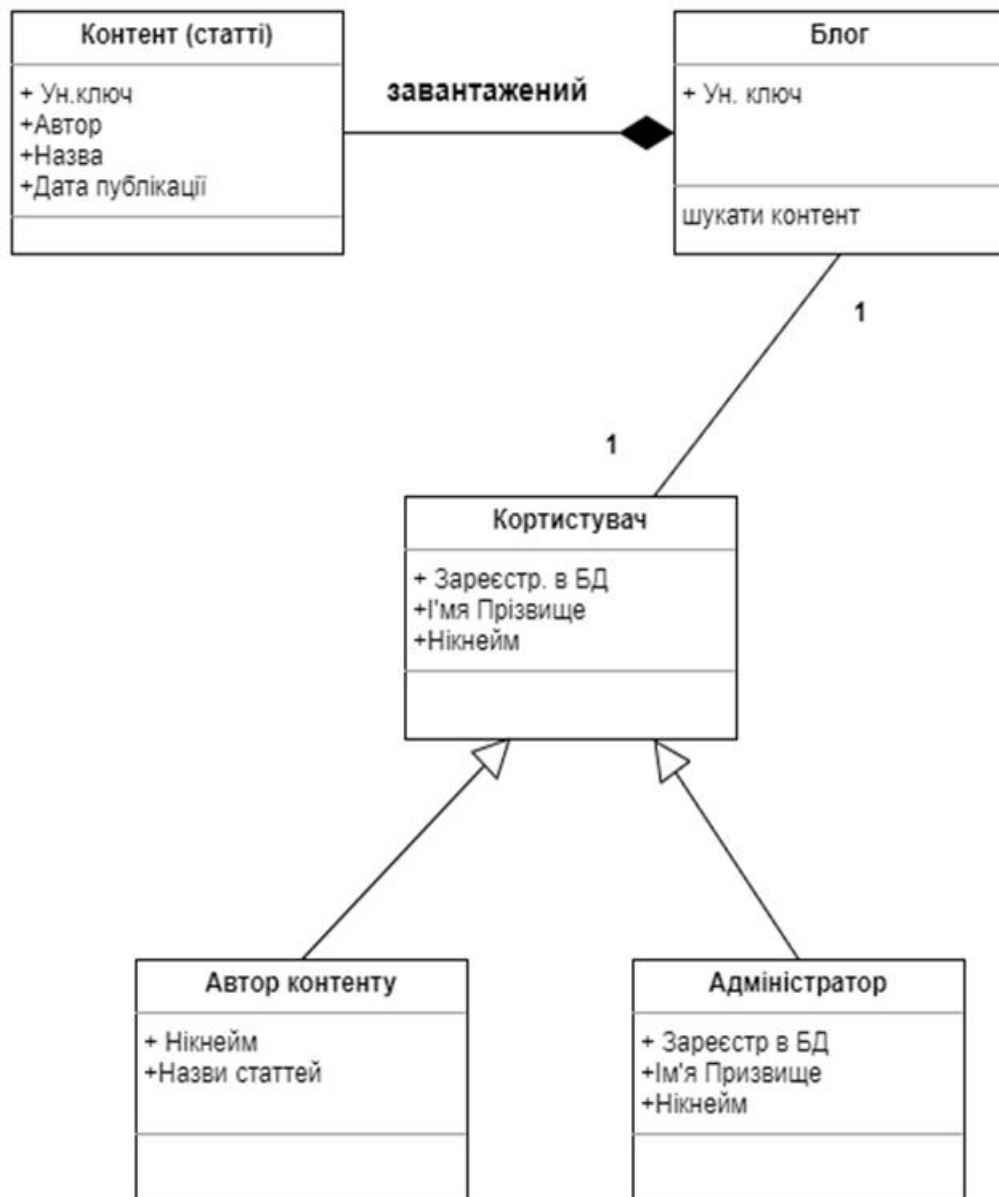


Рисунок 3.1 – Структура програмного забезпечення для багатокористувацького блогінгу у вигляді діаграми класів

Як видно з рис. 3.1, як основними класами програмного забезпечення для багатокористувацького блогінгу є такі: Блог, Контент (статті), Користувач, Автор контенту, Адміністратор.

Діаграма класів програмного забезпечення для багатокористувацького блогінгу демонструє структуру основних сутностей системи та взаємозв'язки між ними.

Клас Блог виступає центральним елементом, який об'єднує весь функціонал платформи. Він містить посилання на об'єкти класу «Контент», відповідає за зберігання колекції статей і забезпечує методи для їх додавання, пошуку та відображення.

Клас Контент представляє окрему публікацію в блозі, що може містити текст, зображення чи інші медіа. До його атрибутів належать заголовок, основний текст, дата створення та інформація про автора. Також цей клас має методи редагування, публікації та видалення матеріалу.

Клас Користувач описує загальні характеристики будь-якого учасника системи — ім'я, логін, контактні дані, а також базові методи для авторизації та перегляду контенту.

Клас Автор контенту є спеціалізацією «Користувача» та розширює його функціонал можливістю створення та редагування публікацій, додавання медіа, а також управління власними матеріалами.

Клас Адміністратор також наслідує властивості «Користувача», але додатково наділений інструментами для модерації контенту, розгляду скарг та управління всією структурою блогу.

Зв'язки між класами відображають ієрархію та взаємодію: Автор контенту і Адміністратор є підкласами Користувача, а Блог агрегує об'єкти Контент, що створюються авторами та контролюються адміністраторами. Така модель чітко визначає розподіл ролей і відповідальностей у межах системи програмного забезпечення для багатокористувацького блогінгу.

Діаграму компонентів програмного забезпечення для багатокористувацького блогінгу наведено на рис. 3.2.

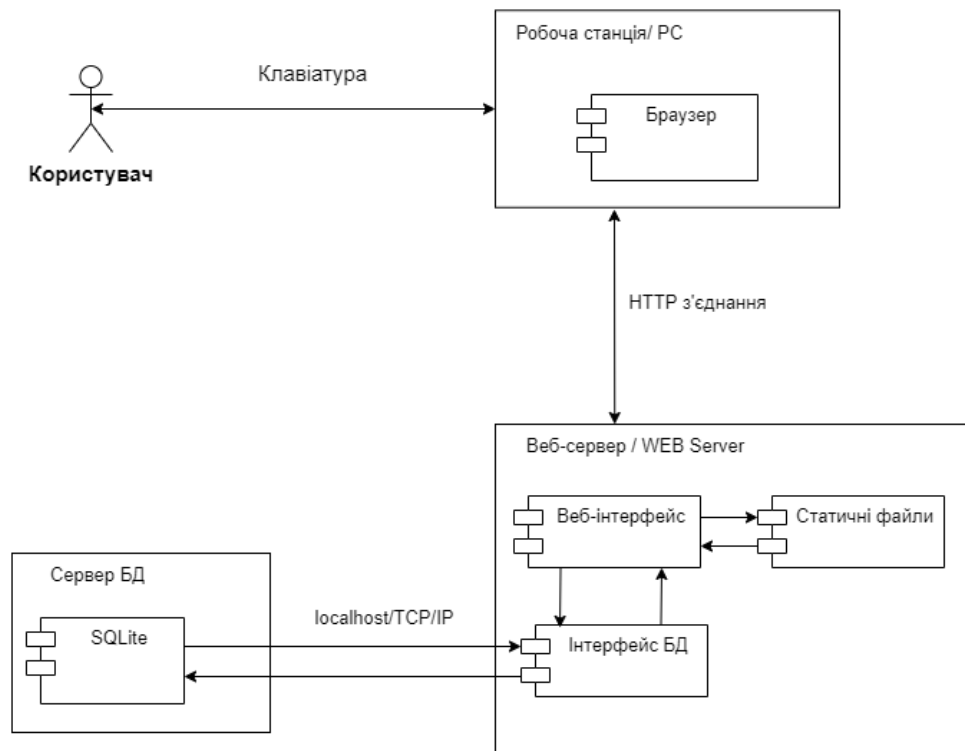


Рисунок 3.2 – Діаграма компонентів програмного забезпечення для багатокористувацького блогінгу

Діаграма компонентів програмного забезпечення для багатокористувацького блогінгу відображає основні структурні елементи системи та канали взаємодії між ними.

Користувач здійснює доступ до платформи через персональний комп'ютер. Усередині цього компонента виділено підкомпонент браузера, що виконує функцію клієнтського середовища для обміну даними з вебзастосунком. Саме браузер ініціює запити до сервера, відправляючи їх у форматі HTTP.

HTTP-з'єднання спрямоване до вебсервера, який є центральною ланкою між користувачем та внутрішньою логікою системи. Структура вебсервера включає кілька взаємопов'язаних підкомпонентів. Вебінтерфейс відповідає за обробку клієнтських запитів, формування відповідей та передачу їх назад браузеру. Інтерфейс бази даних забезпечує комунікацію між вебзастосунком і сервером зберігання даних, передаючи запити на зчитування чи зміну інформації. Статичні файли містять ресурси, що не потребують

динамічної генерації, зокрема зображення, CSS-стилі та скрипти, і передаються безпосередньо користувачеві.

Взаємодія з сервером бази даних здійснюється двосторонньо: вебсервер надсилає запити на отримання або зміну даних та приймає від нього відповідні результати. Така схема забезпечує відокремлення інтерфейсної, логічної та інформаційної частин системи, що сприяє підвищенню масштабованості, зручності супроводу та ефективності роботи сервісу багатокористувацького блогінгу.

3.2 Функціонування програмного забезпечення для багатокористувацького блогінгу

Функціонування програмного забезпечення для багатокористувацького блогінгу подамо за допомогою діаграми станів, зображеної на рис. 3.3.

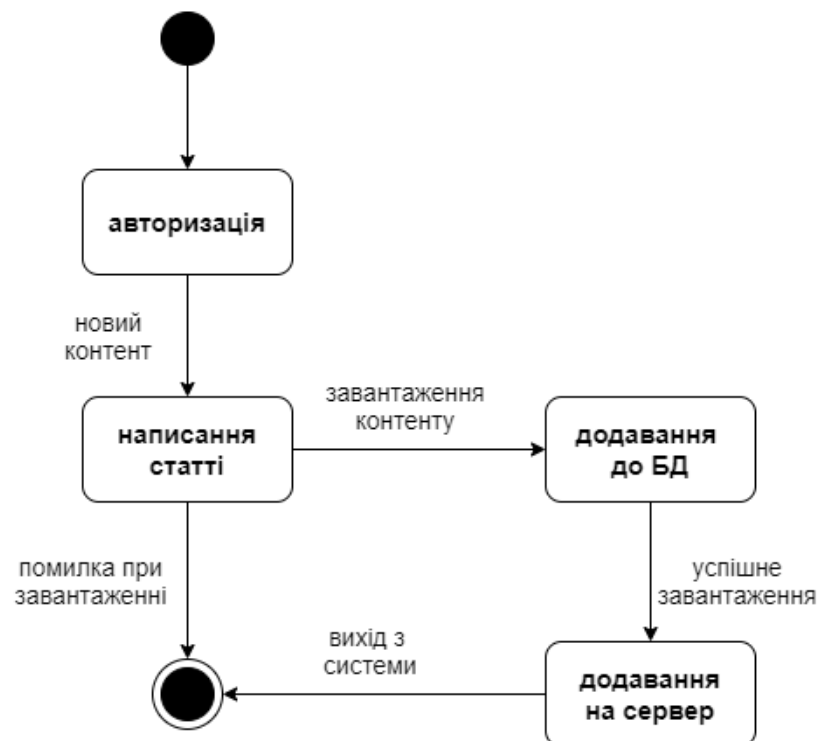


Рисунок 3.3 – Схема функціонування програмного забезпечення для багатокористувацького блогінгу у вигляді діаграми станів

Діаграма станів програмного забезпечення для багатокористувацького блогінгу відображає послідовність переходів між ключовими етапами роботи користувача під час публікації матеріалу.

Початковою точкою є авторизація, яка відкриває доступ до функціональних можливостей системи. Після успішного входу користувач може ініціювати дію новий контент, що переводить систему в стан написання статті. На цьому етапі відбувається створення та підготовка текстового матеріалу.

Зі стану Написання статті можливі два варіанти розвитку подій. Якщо під час спроби відправлення виникає помилка при завантаженні, процес переходить у стан завершення, що означає припинення поточної сесії або відмову від публікації. Якщо ж автор успішно ініціює завантаження контенту, система переходить у стан Додавання до бази даних, де створений матеріал зберігається у структурованому вигляді для подальшого використання.

Після цього, у разі успішного завантаження, відбувається перехід у стан Додавання на сервер, де контент стає доступним іншим користувачам платформи. Завершальним етапом є вихід із системи, після якого процес знову опиняється у стані завершення, готуючись до можливої наступної взаємодії з користувачем.

Така схема відображає логіку руху контенту від моменту його створення до публікації, з урахуванням можливих помилок та завершення роботи системи.

Діаграму діяльності у процесі авторизації програмного забезпечення багатокористувацького блогінгу наведено на рис. 3.4. Діаграма діяльності програмного забезпечення для багатокористувацького блогінгу відображає процес авторизації як послідовність дій із можливими варіантами розвитку подій. Початковим етапом є ініціалізація авторизації, після якої відбувається розгалуження у вигляді ромба, що означає необхідність паралельного виконання двох операцій. З одного боку здійснюється введення логіну, з

іншого введення паролю. Обидві дії виконуються незалежно, але є взаємопов'язаними і мають завершитися для переходу до наступного кроку.

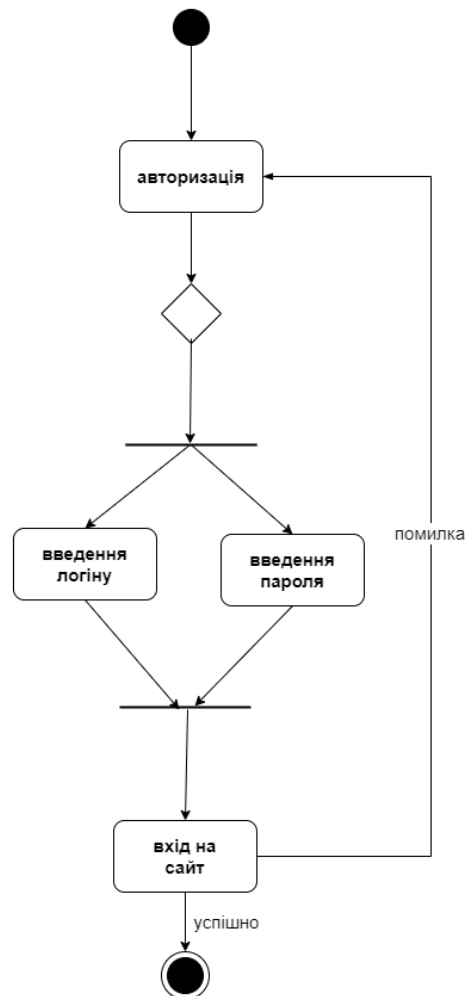


Рисунок 3.4 – Діаграма діяльності у процесі авторизації програмного забезпечення багатокористувацького блогінгу

Після цього відбувається об'єднання потоків у єдину дію Вхід на сайт. На цьому етапі система програмного забезпечення багатокористувацького блогінгу перевіряє коректність наданих облікових даних. Якщо виявлено невідповідність або помилку, користувач програмного забезпечення багатокористувацького блогінгу повертається до початкового стану авторизації, щоб повторити спробу. У разі успішної перевірки відбувається завершення авторизації, що відкриває доступ до функцій програмного забезпечення багатокористувацького блогінгу.

Така діаграма наочно демонструє логіку перевірки доступу та підкреслює необхідність правильного введення як логіну, так і паролю для подальшої роботи в системі програмного забезпечення багатокористувацького блогінгу.

Діаграму послідовностей програмного забезпечення багатокористувацького блогінгу наведено на рис. 3.5.

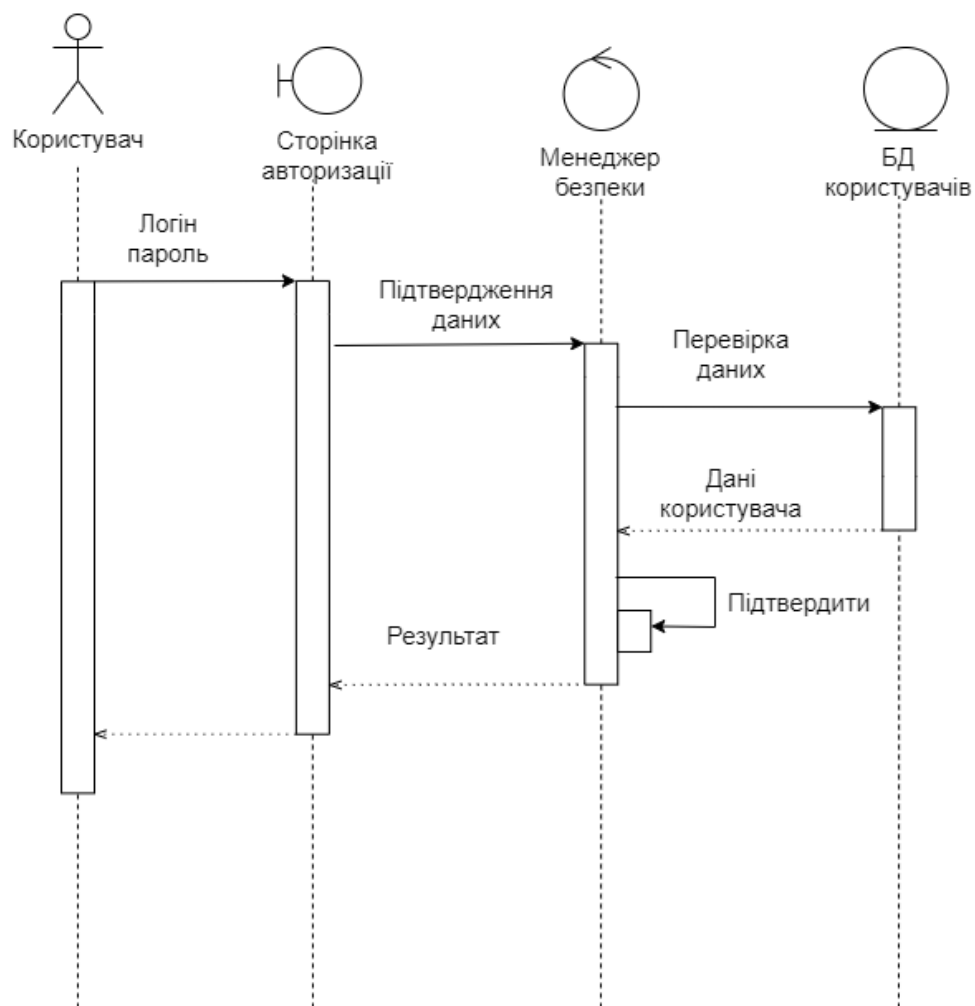


Рисунок 3.5 – Діаграма послідовностей програмного забезпечення багатокористувацького блогінгу

Діаграма послідовностей для програмного забезпечення багатокористувацького блогінгу відображає взаємодію між основними учасниками процесу авторизації в хронологічному порядку. Початковою дією є передавання користувачем своїх облікових даних у вигляді логіну та паролю

на сторінку авторизації. Ця сторінка, виконавши базову обробку введеної інформації, надсилає запит із підтвердженням даних до менеджера безпеки. Далі менеджер безпеки звертається до бази даних, надсилаючи запит на перевірку коректності наданих облікових відомостей.

У відповідь база даних передає менеджеру безпеки відомості про користувача, що дозволяє розпочати етап внутрішньої валідації, відображений самопосиланням у вигляді стрілки, яка повертається до того ж менеджера безпеки. Завершивши перевірку, менеджер безпеки формує остаточний результат авторизації та надсилає його одночасно на сторінку авторизації й безпосередньо користувачеві.

Така послідовність дій демонструє логічний ланцюг обміну інформацією та чітко ілюструє, як відбувається підтвердження доступу в програмному забезпеченні для багатокористувацького блогінгу, забезпечуючи захист і контроль над процесом входу.

3.3 Архітектура програмного забезпечення для багатокористувацького блогінгу

Архітектура програмного забезпечення багатокористувацького блогінгу побудована з використанням шаблону проектування Модель-Вид-Представлення (Model-View-Controller, MVC), що забезпечує чітке розділення функціональних складових системи. У межах цього підходу модель відповідає за зберігання, обробку та управління даними, реалізуючи бізнес-логіку незалежно від зовнішнього вигляду інтерфейсу чи механізмів обробки дій користувача. Представлення програмного забезпечення багатокористувацького блогінгу виступає засобом відображення актуального стану моделі, забезпечуючи візуалізацію інформації та зручність взаємодії з нею. Контролер програмного забезпечення багатокористувацького блогінгу виконує роль посередника між користувачем і моделлю, реагуючи на події, що надходять від інтерфейсу, та ініціюючи зміни в даних.

Схему архітектури програмного забезпечення багатокористувацького блогінгу, подану за допомогою шаблону MVC, наведено на рис. 3.6.

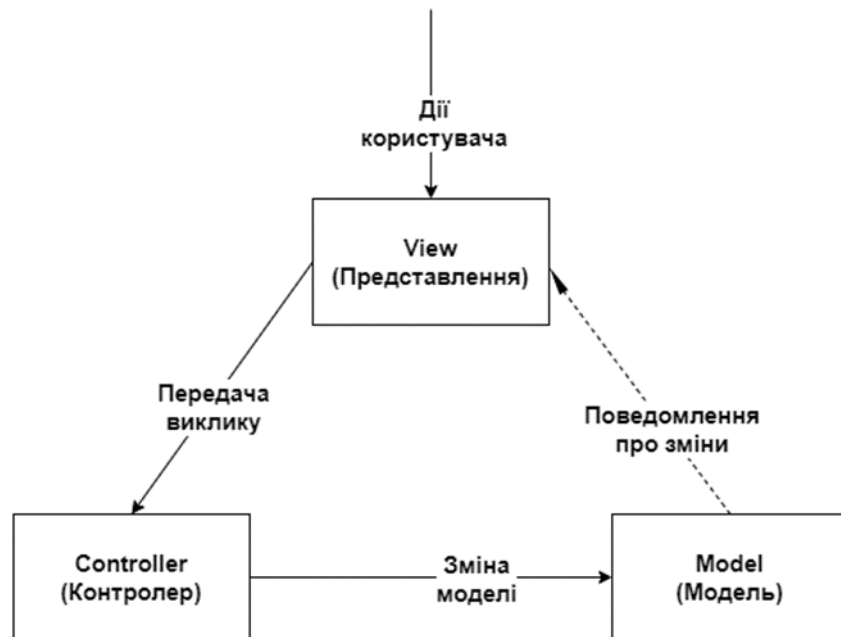


Рисунок 3.6 – Архітектура програмного забезпечення багатокористувацького блогінгу, подана за допомогою шаблону MVC

Перевага такого підходу полягає в тому, що модуль, відповідальний за логіку, залишається ізольованим від компонентів інтерфейсу та контролю, що спрощує модифікацію, тестування й масштабування системи. Під час роботи програмного забезпечення багатокористувацького блогінгу користувач взаємодіє з інтерфейсом, передаючи запити, які контролер обробляє та спрямовує до моделі. Після внесення змін у дані модель програмного забезпечення багатокористувацького блогінгу повідомляє про це представлення, яке відображає оновлену інформацію, підтримуючи узгодженість і оперативність відображення вмісту. Така організація програмного забезпечення багатокористувацького блогінгу дозволяє ефективно підтримувати й розвивати блог без порушення структури його основних компонентів.

3.4 Проектування інтерфейсу програмного забезпечення для багатокористувацького блогінгу

Проектування інтерфейсу взаємодії користувача з програмним забезпеченням для багатокористувацького блогінгу виконано з урахуванням вимог технічного завдання. При розробленні програмного забезпечення для багатокористувацького блогінгу враховані функціональні вимоги до програми:

- підтримка можливості відображення публікацій у блозі;
- підтримка можливості роботи з програмою різних користувачів;
- можливість редагування інформації користувача в особистому профілі;
- можливість створення нових постів у блозі;
- підтримка можливості редагування та видалення власних постів користувача;
- підтримка можливості адміністратора створення, редагування та видалення інформації про користувачів програми.

Інтерфейс прототипу сторінки програмного забезпечення для багатокористувацького блогінгу наведено на рис. 3.7.

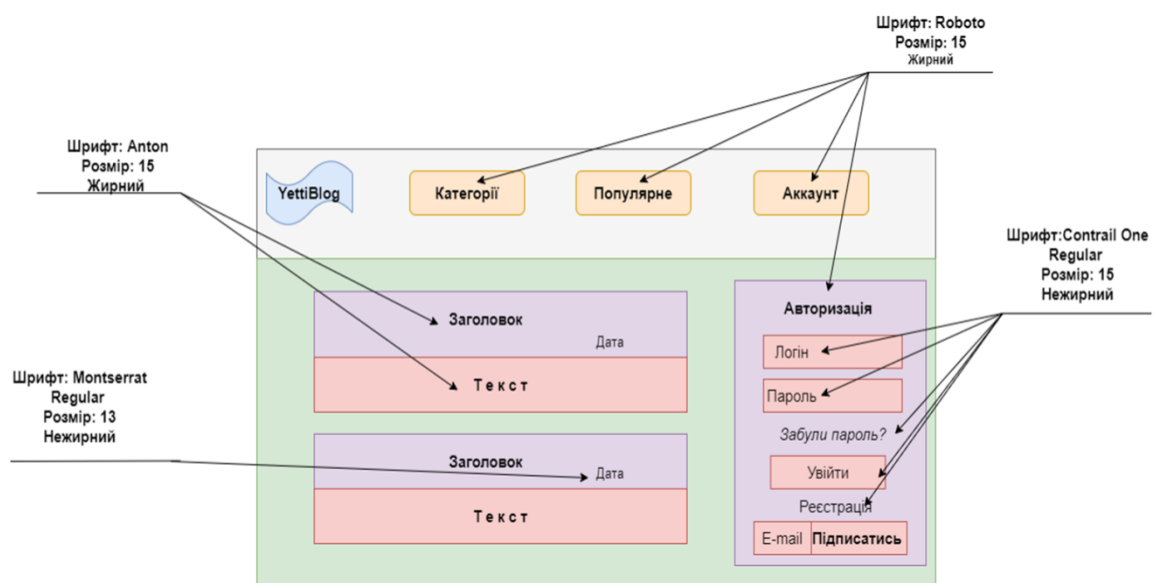


Рисунок 3.7 – Інтерфейс прототипу сторінки програмного забезпечення для багатокористувацького блогінгу

Інтерфейс прототипу сторінки реєстрації наведено на рис. 3.8.

The wireframe shows a registration page layout. At the top, there is a header bar containing a 'LOGO' placeholder, a 'Категорії' (Categories) button, a 'Популярне' (Popular) button, and an 'Аккаунт' (Account) button. Below the header, the main content area is titled 'Реєстрація' (Registration). It features three stacked input fields labeled 'Логін' (Login), 'Пароль' (Password), and 'Повторити пароль' (Repeat password). A 'УВІЙТИ' (Log In) button is positioned at the bottom right of the registration form.

Рисунок 3.8 – Інтерфейс прототипу сторінки реєстрації

Інтерфейс прототипу сторінки користувача наведено на рис. 3.9.

The wireframe shows a user profile page layout. The header bar is identical to the registration page, with 'LOGO', 'Категорії', 'Популярне', and 'Аккаунт' buttons. The main content area is titled 'ПРОФІЛЬ <НІКНЕЙМ>' (Profile <Username>). It contains five buttons: 'Мої статті' (My articles) and 'Редагувати дані' (Edit data) in the top row; 'Написати статтю' (Write article) and 'Пошта' (Email) in the middle row; and a wide 'Рейтинг користувача' (User rating) button at the bottom.

Рисунок 3.9 – Інтерфейс прототипу сторінки користувача

Інтерфейс прототипу меню редагування наведено на рис. 3.10.

Інтерфейс прототипу меню редагування. У верхній частині розташовані елементи: **LOGO** (у вигляді хвилястої лінії), **Категорії**, **Популярне** та **Аккаунт**. Основна частина меню має заголовок **Меню редагування** та три великі текстові поля: **НАЗВА СТАТТІ**, **ОПИС** та **ТЕГИ**. У правому нижньому куті розташований кнопка **ПІДТВЕРДИТИ**.

Рисунок 3.10 – Інтерфейс прототипу меню редагування

Інтерфейс прототипу сторінки зі статтями користувача наведено на рис. 3.11.

Інтерфейс прототипу сторінки зі статтями користувача. У верхній частині розташовані елементи: **LOGO** (у вигляді хвилястої лінії), **Категорії**, **Популярне** та **Аккаунт**. Основна частина сторінки має заголовок **МОЇ СТАТТІ** та три колонки. Кожна колонка містить три елементи: **Назва**, **Опис статті** та **Редагувати**.

Рисунок 3.11 – Інтерфейс прототипу сторінки зі статтями користувача

3.5 Висновки за розділом 3

Розроблено програмне забезпечення для багатокористувацького блогінгу.

Запропоновано структуру програмного забезпечення для багатокористувацького блогінгу. Визначено, що основними класами програмного забезпечення для багатокористувацького блогінгу є такі: Блог, Контент (статті), Користувач, Автор контенту, Адміністратор. Розроблено діаграму компонентів програмного забезпечення для багатокористувацького блогінгу відображає основні структурні елементи системи та канали взаємодії між ними.

Функціонування програмного забезпечення для багатокористувацького блогінгу подано за допомогою діаграми станів, яка відображає послідовність переходів між ключовими етапами роботи користувача під час публікації матеріалу.

Визначено, що архітектура програмного забезпечення багатокористувацького блогінгу побудована з використанням шаблону Model-View-Controller, що забезпечує чітке розділення функціональних складових системи.

Виконано проєктування інтерфейсу взаємодії користувача з програмним забезпеченням для багатокористувацького блогінгу.

4 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОГРАМИ

4.1 Призначення й умови застосування програми

Програма призначена для багатокористувацького блогінгу. Програма написана на мові Python, що обумовлено поєднанням доступності синтаксису, розвиненої інфраструктури інструментів і бібліотек, а також високого рівня інтеграційних можливостей. В якості фреймворку для створення програмного забезпечення для багатокористувацького блогінгу обрано Flask, що забезпечує високу продуктивність, надійність та гнучкість ПЗ, а також є надійною платформою для розробки сучасних програмних систем для підтримки блогів з багатокористувацьким доступом, високим рівнем безпеки та зручністю експлуатації.

Програмне забезпечення для багатокористувацького блогінгу забезпечує виконання таких функцій:

- підтримка можливості відображення публікацій у блозі;
- підтримка можливості роботи з програмою різних користувачів;
- можливість редагування інформації користувача в особистому профілі;
- можливість створення нових постів у блозі;
- підтримка можливості редагування та видалення власних постів користувача;
- підтримка можливості адміністратора створення, редагування та видалення інформації про користувачів програми.

Для експлуатації програмного забезпечення для багатокористувацького блогінгу необхідно таке апаратне та програмне забезпечення:

- процесор з тактовою частотою не нижче 2,0 ГГц, що забезпечує достатню обчислювальну потужність для обробки запитів від декількох користувачів одночасно;

- обсяг оперативної пам'яті має становити щонайменше 4 ГБ, що дозволяє підтримувати стабільну роботу серверних процесів та клієнтських з'єднань;

- накопичувач із вільним простором не менше 20 ГБ;

- стабільне з'єднання з мережею Інтернет із пропускною здатністю від 10 Мбіт/с для швидкого обміну інформацією між клієнтами та сервером.

Програмна складова повинна включати операційну систему (Linux, Windows або macOS у підтримуваних версіях), вебсервер, інтерпретатор Python версії 3.8 чи новішої, а також систему управління базами даних. Для коректної роботи інтерфейсу користувача потрібен браузер.

Функціональні характеристики розробленого програмного забезпечення для багатокористувацького блогінгу наведено у технічному завданні (додаток А). Фрагмент тексту програмного забезпечення для багатокористувацького блогінгу наведено у додатку Б.

4.2 Характеристики програми для багатокористувацького блогінгу

Програмне забезпечення багатокористувацького блогінгу характеризується здатністю забезпечувати взаємодію різних користувачів у режимі реального часу.

Програмне забезпечення багатокористувацького блогінгу підтримує створення, редагування та публікацію контенту з можливістю додавання мультимедійних матеріалів. Передбачено механізми авторизації та автентифікації, що дозволяють контролювати доступ до функціоналу та розмежовувати права користувачів.

Інтерфейс програмного забезпечення багатокористувацького блогінгу орієнтовано на зручність використання та швидку навігацію, а серверна частина оптимізована для стабільної роботи під навантаженням.

Реалізовано інтеграцію з базами даних для зберігання контенту, користувацьких профілів і статистичних даних, а також підтримку обміну інформацією через сучасні вебтехнології. Програмне забезпечення багатокористувацького блогінгу здатне адаптуватися до змін у вимогах, зберігаючи продуктивність і безпеку на високому рівні.

4.3 Інструкція по експлуатації програми

4.3.1 Звернення до програми

Для запуску програмного забезпечення для багатокористувацького блогінгу на серверній частини необхідно запустити відповідний ру-файл. Для звертання до клієнтської сторони необхідно перейти на відповідну вебсторінку.

4.3.2 Вхідні й вихідні дані

Вхідними даними до програмного забезпечення для багатокористувацького блогінгу є інформація про користувача, про блог, пости у блозі.

Вихідними даними програмного забезпечення для багатокористувацького блогінгу є вебсторінки з постами користувачів, вебсторінка з профілем користувача та ін.

4.3.3 Повідомлення

Користувач програмного забезпечення для багатокористувацького блогінгу може отримати такі повідомлення: повідомлення про успішну реєстрацію, повідомлення про видалення посту у блозі та ін.

4.4 Виконання програмного забезпечення для багатокористувацького блогінгу

Після запуску програмного забезпечення для багатокористувацького блогінгу користувачу відображається головна сторінка (рис. 4.1), де наведено останні публікації у блозі. Для входу до системи зареєстрований користувач повинен перейти за посиланням Login та ввести свої дані (рис. 4.2).

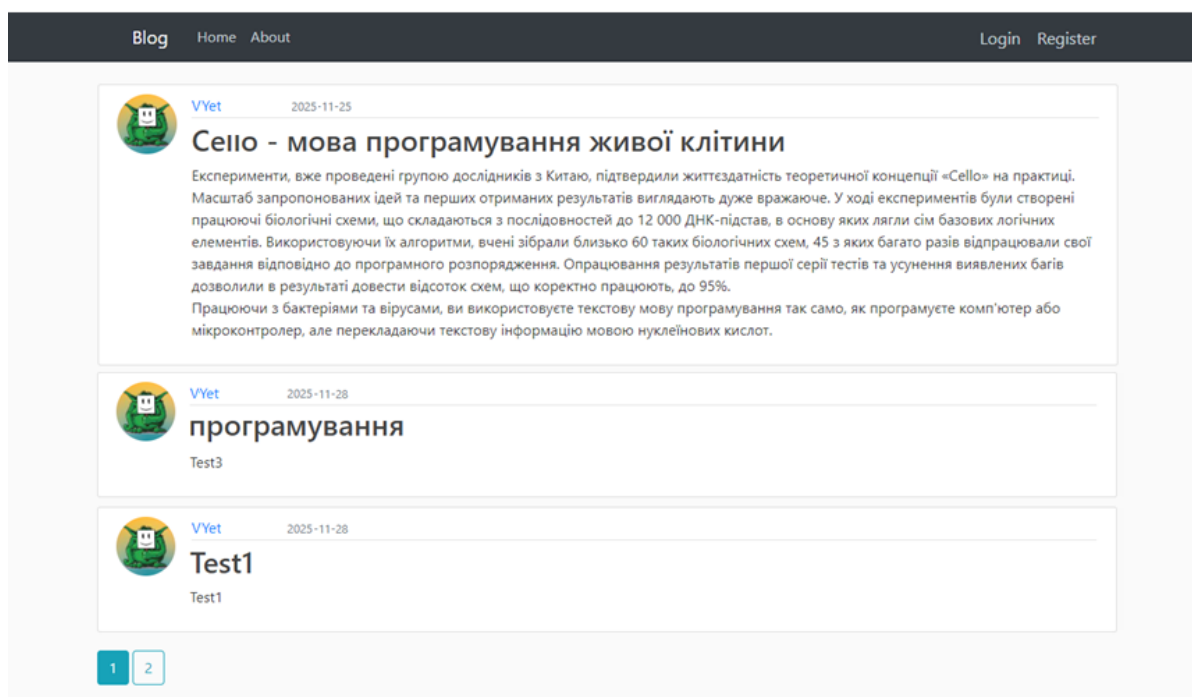


Рисунок 4.1 – Інтерфейс головної сторінки програмного забезпечення багатокористувацького блогінгу

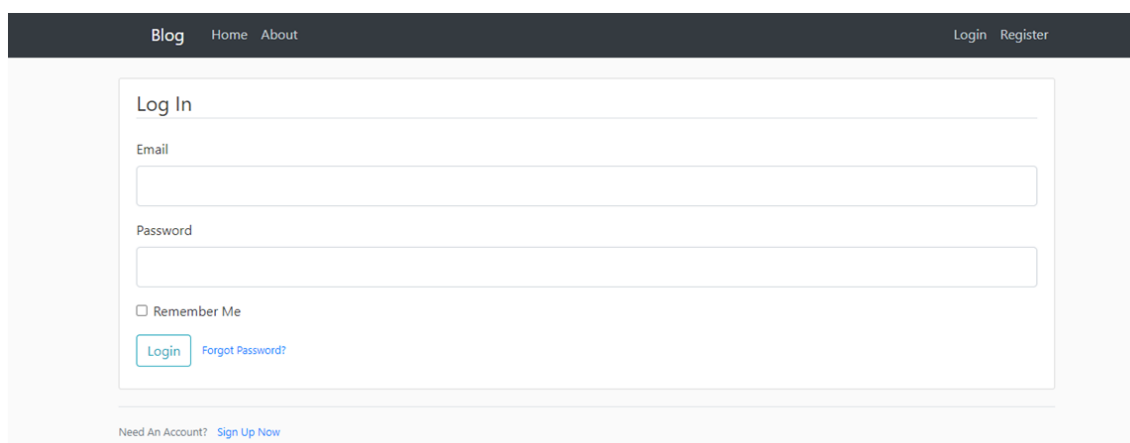
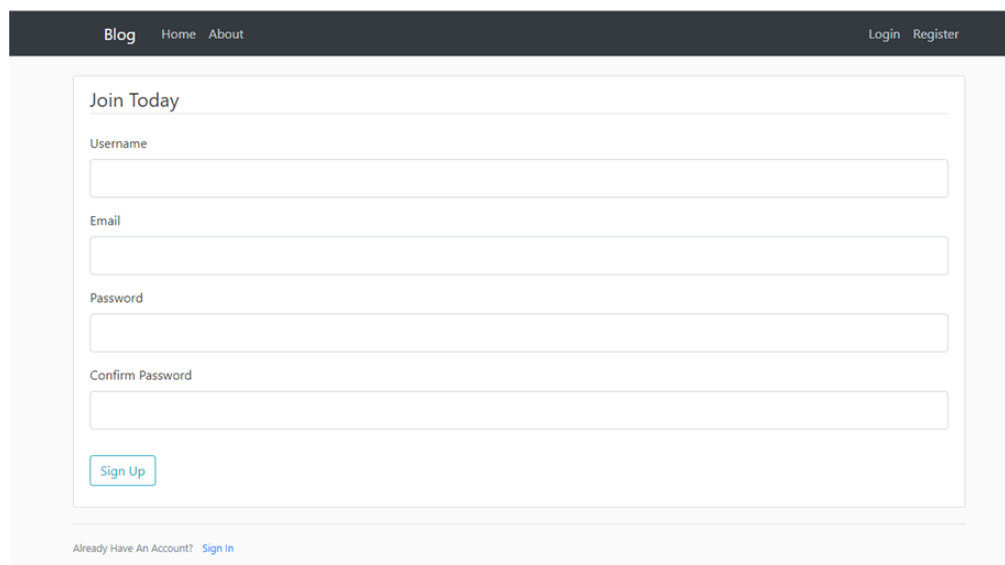


Рисунок 4.2 – Сторінка авторизації у ПЗ багатокористувацького блогінгу

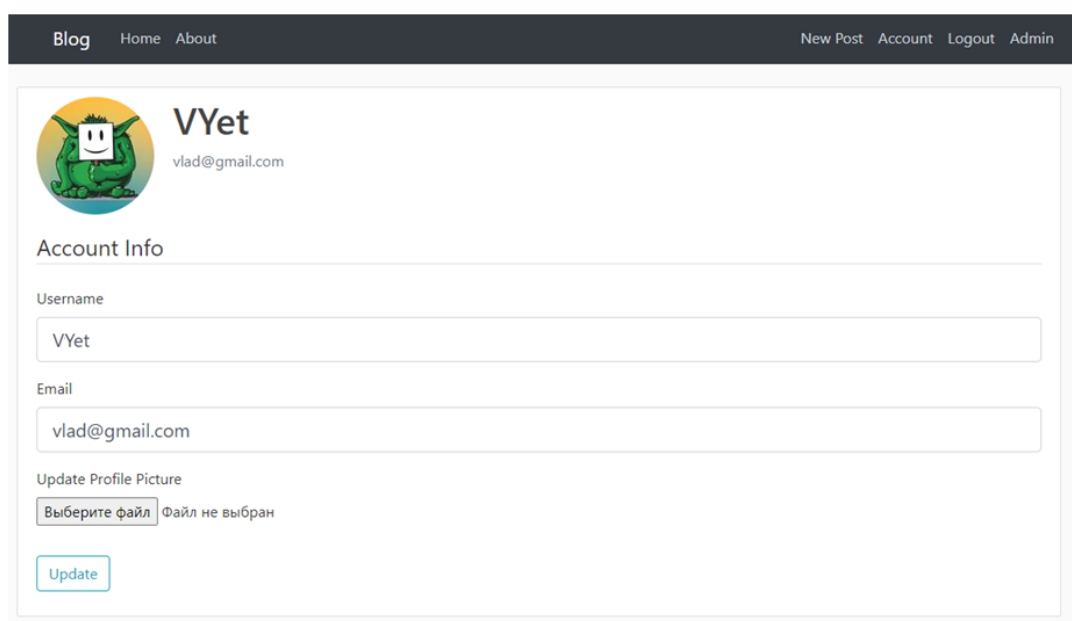
Для реєстрації (рис. 4.3) необхідно ввести ім'я користувача, його електронну пошту та пароль.



The image shows a registration form titled "Join Today". It is part of a web application with a dark header containing links for "Blog", "Home", "About", "Login", and "Register". The form itself is white and contains four input fields: "Username", "Email", "Password", and "Confirm Password". Below these fields is a "Sign Up" button. At the bottom of the form, there is a link that says "Already Have An Account? Sign In".

Рисунок 4.3 – Сторінка реєстрації у програмі багатокористувацького блогінгу

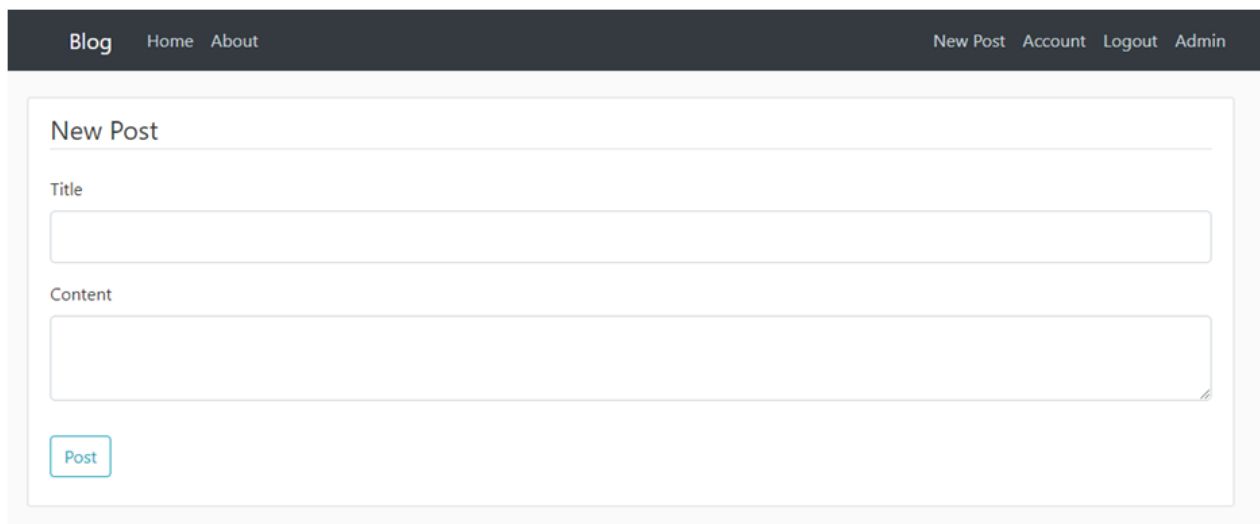
Після авторизації користувач може перейти на сторінку свого профілю (рис. 4.4) за допомогою меню Account, де він має можливість відредагувати інформацію про свою електронну пошту, ім'я та зображення у системі.



The image shows a user profile page titled "Account Info". It is part of a web application with a dark header containing links for "Blog", "Home", "About", "New Post", "Account", "Logout", and "Admin". The profile section includes a circular profile picture of a green alien-like character, the username "VYet", and the email address "vlad@gmail.com". Below this, there is a section for updating the profile picture, which includes a file selection button labeled "Выберите файл" (Choose file) and a status "Файл не выбран" (File not selected). At the bottom of this section is an "Update" button.

Рисунок 4.4 – Сторінка профілю користувача

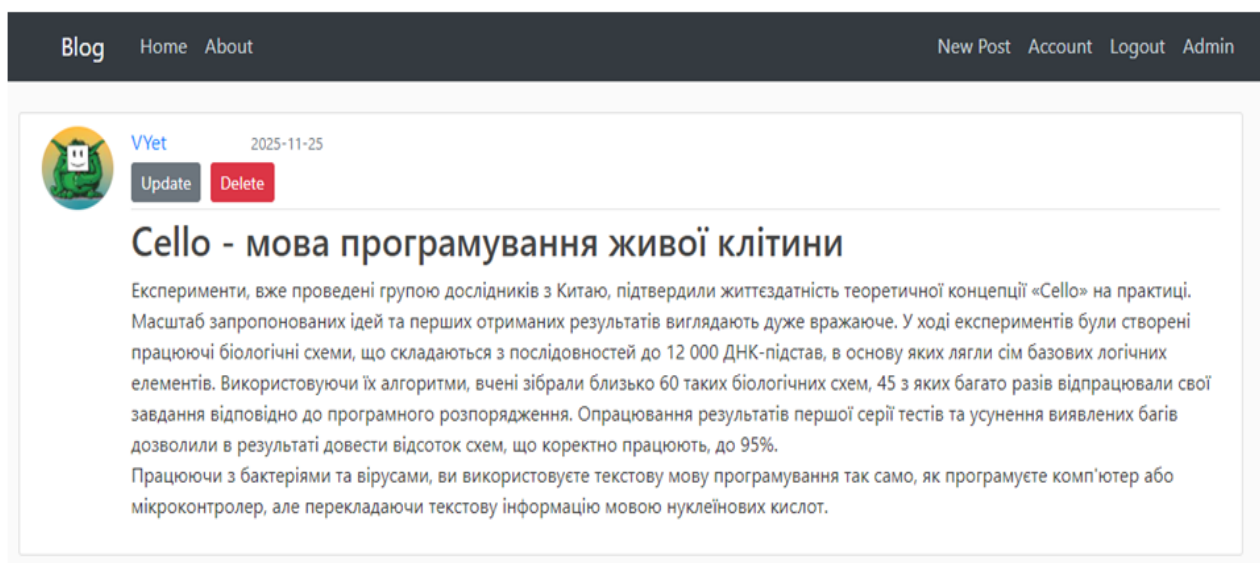
Для створення нового посту у блозі (рис. 4.5) необхідно обрати меню New Post, ввести назву посту, його зміст та натиснути кнопку Post для публікації матеріалів.



The screenshot shows a web interface with a dark header bar containing navigation links: 'Blog', 'Home', 'About', 'New Post', 'Account', 'Logout', and 'Admin'. The 'New Post' link is highlighted. Below the header is a form titled 'New Post'. It contains two input fields: 'Title' and 'Content'. The 'Content' field is a larger text area. At the bottom left of the form is a blue button labeled 'Post'.

Рисунок 4.5 – Сторінка створення нового посту у блозі

Зареєстрований користувач має можливості редагування власних постів за допомогою кнопки Update. Видалення посту виконується за допомогою кнопки Delete (рис. 4.6).



The screenshot shows a web interface with a dark header bar containing navigation links: 'Blog', 'Home', 'About', 'New Post', 'Account', 'Logout', and 'Admin'. The 'New Post' link is highlighted. Below the header is a post editing form. At the top left of the form is a circular profile picture of a green cartoon character. To its right is the username 'VYet' and the date '2025-11-25'. Below the profile picture are two buttons: 'Update' (blue) and 'Delete' (red). The main title of the post is 'Cello - мова програмування живої клітини'. Below the title is a paragraph of text: 'Експерименти, вже проведені групою дослідників з Китаю, підтвердили життєздатність теоретичної концепції «Cello» на практиці. Масштаб запропонованих ідей та перших отриманих результатів виглядають дуже вражаюче. У ході експериментів були створені працюючі біологічні схеми, що складаються з послідовностей до 12 000 ДНК-підстав, в основу яких лягли сім базових логічних елементів. Використовуючи їх алгоритми, вчені зібрали близько 60 таких біологічних схем, 45 з яких багато разів відпрацювали свої завдання відповідно до програмного розпорядження. Опрацювання результатів першої серії тестів та усунення виявлених багів дозволили в результаті довести відсоток схем, що коректно працюють, до 95%. Працюючи з бактеріями та вірусами, ви використовуєте текстову мову програмування так само, як програмуєте комп'ютер або мікроконтролер, але перекладаючи текстову інформацію мовою нуклеїнових кислот.'

Рисунок 4.6 – Сторінка редагування посту

Користувач з правами адміністратора має доступ до сторінки з таблицею користувачів (рис. 4.7), де існує можливість створення, редагування та видалення необхідної інформації про користувачів програми.

Blog - Admin Home User Post					
List (2)		Create	With selected ▾		
☐		Username	Email	Image File	Password
☐	 	VYet	vlad@gmail.com	deb.jpg	12\$kvSGFA
☐	 	testUser	test@test.ua	def.jpg	12\$/8UI5J2

Рисунок 4.7 – Сторінка адміністратора з таблицею користувачів

4.5 Тестування програмного забезпечення для багатокористувацького блогінгу

Виконано тестування програмного забезпечення для багатокористувацького блогінгу.

Виявлено, що програма для багатокористувацького блогінгу функціонує правильно та злагоджено. Вона реалізує всі функціональні вимоги та успішно виконує свою основну задачу підтримки процесу організації харчування домашніх тварин.

Розроблене програмне забезпечення дозволяє забезпечувати автоматизацію процесів, пов'язаних з підтримкою організації харчування домашніх тварин.

4.6 Висновки за розділом 4

Описано програмне забезпечення для багатокористувацького блогінгу. Виконано тестування розробленого програмного забезпечення для

багатокористувацького блогінгу. Результати тестування програмного забезпечення показали, що розроблена програма дозволяє забезпечити автоматизацію процесів, пов'язаних з підтримкою організації харчування домашніх тварин.

ВИСНОВКИ

В ході виконання дипломної кваліфікаційної роботи магістра було проаналізовано та досліджено процес розробки програмного забезпечення для багатокористувацького блогінгу.

Визначено, що блог являє собою Інтернет ресурс, призначений для поширення авторського контенту, що може охоплювати особисті міркування, аналітичні матеріали, навчальні тексти або інші тематичні публікації. На відміну від звичайних вебсайтів, блоги зазвичай оновлюються регулярно, а матеріали подаються у зворотному хронологічному порядку, де першим відображається найсвіжіший допис. Програмне забезпечення для ведення блогів являє собою інструмент, призначений для спрощення процесу створення, упорядкування та поширення публікацій у мережі. Таке ПЗ дозволяє зосередитися на змісті матеріалів, усуваючи потребу у складних технічних діях, пов'язаних з налаштуванням та підтримкою вебресурсу.

За результатами проведеного аналізу зроблено висновок, що у наш час існує досить багато програмних засобів для ведення блогів. Проте деякі програмні засоби для ведення блогів можуть мати підвищену складність адміністрування, оскільки забезпечення узгодженої взаємодії між численними користувачами потребує чітко налаштованих механізмів контролю доступу, модерації матеріалів та запобігання конфліктам при одночасному редагуванні. У разі відсутності належної оптимізації інтерфейсу та внутрішньої архітектури програми зростає ймовірність виникнення помилок під час публікації або оновлення вмісту блогу. Крім того, використання готових платформ із закритим вихідним кодом може обмежувати можливості налаштування функціоналу під специфічні потреби, що ускладнює інтеграцію з іншими інформаційними системами та сервісами. Тому актуальною є розробка програмного забезпечення для багатокористувацького блогінгу.

Сформульовано функціональні вимоги до програмного забезпечення для багатокористувацького блогінгу.

Для реалізації програмного забезпечення для багатокористувацького блогінгу обрано мову програмування Python, що обумовлено поєднанням доступності синтаксису, розвиненої інфраструктури інструментів і бібліотек, а також високого рівня інтеграційних можливостей. Використання готових модулів дає змогу суттєво скоротити час реалізації ключових функцій, зокрема управління користувачами, обробки HTTP-запитів, взаємодії з базами даних та впровадження механізмів безпеки.

Для створення програмного забезпечення для багатокористувацького блогінгу обрано фреймворк Flask, що забезпечує високу продуктивність, надійність та гнучкість ПЗ, а також є надійною платформою для розробки сучасних програмних систем для підтримки блогів з багатокористувацьким доступом, високим рівнем безпеки та зручністю експлуатації.

Новизна роботи полягає у тому, що створено модель взаємодії користувача з програмним забезпеченням для багатокористувацького блогінгу, яка подана у вигляді діаграми прецедентів, що відображає ролі та їхні взаємозв'язки з функціональними можливостями програми, демонструючи розподіл прав доступу та специфіку взаємодії кожного типу користувачів із програмним забезпеченням.

Практична цінність роботи полягає у тому, що розроблено програмне забезпечення для багатокористувацького блогінгу.

Запропоновано структуру програмного забезпечення для багатокористувацького блогінгу. Визначено, що основними класами програмного забезпечення для багатокористувацького блогінгу є такі: Блог, Контент (статті), Користувач, Автор контенту, Адміністратор. Розроблено діаграму компонентів програмного забезпечення для багатокористувацького блогінгу, що відображає основні структурні елементи системи та канали взаємодії між ними.

Функціонування програмного забезпечення для багатокористувацького блогінгу подано за допомогою діаграми станів, яка відображає послідовність

переходів між ключовими етапами роботи користувача під час публікації матеріалу.

Визначено, що архітектура програмного забезпечення багатокористувацького блогінгу побудована з використанням шаблону Model-View-Controller, що забезпечує чітке розділення функціональних складових системи.

Виконано проєктування інтерфейсу взаємодії користувача з програмним забезпеченням для багатокористувацького блогінгу.

Виконано тестування розробленого програмного забезпечення для багатокористувацького блогінгу. Результати тестування програмного забезпечення показали, що розроблена програма дозволяє забезпечити автоматизацію процесів, пов'язаних з підтримкою організації харчування домашніх тварин.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. What Is Blogging In 2025? Definition, Pros & Cons Explained [Electronic resource]. – Access mode: <https://bloggerspassion.com/what-is-blogging/>.
2. What is Blog Software and How It Works [Electronic resource]. – Access mode: <https://blog.9cv9.com/what-is-blog-software-and-how-it-works/>.
3. What Is A Blogging Software? Choosing The Right Platform For Your Online Presence [Electronic resource]. – Access mode: <https://keywordjuice.com/blog/what-is-a-blogging-software/>.
4. How to Start A Blog [Electronic resource]. – Access mode: <https://www.theblogstarter.com/>.
5. Where to Start a Blog? 5 Best Blogging Platforms [Electronic resource]. – Access mode: <https://jetpack.com/resources/best-blogging-platforms/>.
6. What is a blog? Definition, blog types, and benefits explained [Electronic resource]. – Access mode: <https://www.hostinger.com/tutorials/what-is-a-blog>.
7. Best Blogging Software Options to Consider [Electronic resource]. – Access mode: <https://www.ryrob.com/best-blogging-software/>.
8. Clean and Simple Open-Source Blogging Platforms for Your Developer Blog [Electronic resource]. – Access mode: <https://www.codementor.io/@lovellifuad/15-clean-and-simple-open-source-blogging-platforms-for-your-developer-blog-qdrlfw6>.
9. Best Free Blogging Platforms [Electronic resource]. – Access mode: <https://www.forbes.com/advisor/business/software/best-free-blogging-platform/>.
10. Best Blogging Platforms [Electronic resource]. – Access mode: <https://www.softwaresuggest.com/blog-platform>.
11. Best Blogging Tools for Growth [Electronic resource]. – Access mode: <https://www.semrush.com/blog/blogging-tools/>.
12. The 17 Best Blogging Platforms [Electronic resource]. – Access mode: <https://blog.hubspot.com/marketing/best-blogging-platform>.

13. Tumblr [Electronic resource]. – Access mode: <https://www.tumblr.com/>.
14. Medium [Electronic resource]. – Access mode: [https://medium.com /](https://medium.com/).
15. Write.as [Electronic resource]. – Access mode: <https://write.as/>.
16. Python. Browse the docs online or download a copy of your own [Electronic resource]. – Access mode: <https://www.python.org/doc/>.
17. Python Variables [Electronic resource]. – Access mode: https://www.w3schools.com/python/python_variables.asp.
18. Flask [Electronic resource]. – Access mode: <https://flask.palletsprojects.com/en/stable/>.
19. Flask [Electronic resource]. – Access mode: <https://pypi.org/project/Flask/>.

ДОДАТОК А
Технічне завдання

Вступ

Програмне забезпечення може використовуватися для багатокористувацького блогінгу.

A.1 Підстава для розробки

Підставою для розробки є завдання на дипломну кваліфікаційну роботу на тему «Дослідження та програмна реалізація застосунку для багатокористувацького блогінгу», затверджене наказом Національного університету «Запорізька політехніка» № 447 від 30 вересня 2025 р.

A.2 Призначення розробки

Програмний продукт призначений для багатокористувацького блогінгу.

A.3 Основні вимоги до програми, що розробляється

A.3.1 Вимоги до функціональних характеристик

Програмне забезпечення для багатокористувацького блогінгу забезпечує виконання таких функцій:

- підтримка можливості відображення публікацій у блозі;
- підтримка можливості роботи з програмою різних користувачів;
- можливість редагування інформації користувача в особистому профілі;
- можливість створення нових постів у блозі;
- підтримка можливості редагування та видалення власних постів користувача;
- підтримка можливості адміністратора створення, редагування та видалення інформації про користувачів програми.

А.3.2 Вимоги до інтерфейсу програми

Інтерфейс програмного забезпечення для багатокористувацького блогінгу повинен бути зручним для користувачів. Повинна бути забезпечена можливість роботи в візуальному режимі.

А.3.3 Вимоги до надійності

Програмне забезпечення для багатокористувацького блогінгу повинно забезпечити надійне функціонування.

А.3.4 Умови експлуатації

Для експлуатації програмного забезпечення для багатокористувацького блогінгу необхідна наявність персонального комп'ютера.

А.3.5 Вимоги до складу та параметрів технічних засобів

Для експлуатації програмного забезпечення для багатокористувацького блогінгу необхідно таке апаратне та програмне забезпечення:

- процесор з тактовою частотою не нижче 2,0 ГГц, що забезпечує достатню обчислювальну потужність для обробки запитів від декількох користувачів одночасно;
- обсяг оперативної пам'яті має становити щонайменше 4 ГБ, що дозволяє підтримувати стабільну роботу серверних процесів та клієнтських з'єднань;
- накопичувач із вільним простором не менше 20 ГБ;

– стабільне з'єднання з мережею Інтернет із пропускною здатністю від 10 Мбіт/с для швидкого обміну інформацією між клієнтами та сервером.

Програмна складова повинна включати операційну систему (Linux, Windows або macOS у підтримуваних версіях), вебсервер, інтерпретатор Python версії 3.8 чи новішої, а також систему управління базами даних. Для коректної роботи інтерфейсу користувача потрібен браузер.

А.3.6 Вимоги до маркування і пакування

Програма для багатокористувацького блогінгу може бути записана на будь-якому носії інформації.

На пакуванні повинна бути назва програми – «Програмне забезпечення для багатокористувацького блогінгу».

А.4 Вхідні дані до роботи

Вхідними даними до програмного забезпечення для багатокористувацького блогінгу є інформація про користувача, про блог, пости у блозі.

Вихідними даними програмного забезпечення для багатокористувацького блогінгу є вебсторінки з постами користувачів, вебсторінка з профілем користувача та ін.

ДОДАТОК Б
Фрагмент тексту програми

```

def new_pst():
    frm = PstFrm()
    if frm.vldton_sbmt():
        pst = Pst(title=frm.title.data, content=frm.content.data,
avtor=krntkrstv)
        datbas.session.add(pst)
        datbas.session.commit()
        return redirect(url_for('main.home'))
    return render_template('stvornpst.html', title='New Pst',
                           frm=frm, legend='New Pst')

@psts.route("/pst/<int:pst_id>")
def pst(pst_id):
    pst = Pst.query.get_or_404(pst_id)
    return render_template('pst.html', title=pst.title, pst=pst)

@psts.route("/pst/<int:pst_id>/update", methods=['GET', 'PST'])
@lgnrequired
def update_pst(pst_id):
    pst = Pst.query.get_or_404(pst_id)
    if pst.avtor != krntkrstv:
        abort(403)
    frm = PstFrm()
    if frm.vldton_sbmt():
        pst.title = frm.title.data
        pst.content = frm.content.data
        datbas.session.commit()
        return redirect(url_for('psts.pst', pst_id=pst.id))
    elif rqvst.method == 'GET':
        frm.title.data = pst.title
        frm.content.data = pst.content
    return render_template('stvornpst.html', title='Update Pst',
                           frm=frm, legend='Update Pst')

@psts.route("/pst/<int:pst_id>/delete", methods=['PST'])
@lgnrequired
def delete_pst(pst_id):
    pst = Pst.query.get_or_404(pst_id)
    if pst.avtor != krntkrstv:
        abort(403)
    datbas.session.delete(pst)
    datbas.session.commit()
    return redirect(url_for('main.home'))

class PstFrm(FlaskFrm):

```

```

title = StringField('Title', valdtrs=[DataRequired()])
content = TextAreaField('Content', valdtrs=[DataRequired()])
sbmt = SbmtField('Pst')

krstvs = Blueprint('krstvs', __name__)

@krstvs.route("/regstr", methods=['GET', 'PST'])
def regstr():
    if krntkrstv.isavtrsd:
        return redirect(url_for('main.home'))
    frm = RgstrtnFrm()
    if frm.vldton_sbmt():
        hshdpswr =
bcrpt.generate_pswrd_hash(frm.pswrd.data).decode('utf-8')
        krstv = Krstv(krstvname=frm.krstvname.data,
elpst=frm.elpst.data, pswrd=hshdpswr)
        datbas.session.add(krstv)
        datbas.session.commit()
        return redirect(url_for('krstvs.login'))
    return render_template('regstr.html', title='Regstr',
frm=frm)

@krstvs.route("/login", methods=['GET', 'PST'])
def login():
    if krntkrstv.isavtrsd:
        return redirect(url_for('main.home'))
    frm = LoginFrm()
    if frm.vldton_sbmt():
        krstv =
Krstv.query.filter_by(elpst=frm.elpst.data).first()
        if krstv and bcrpt.check_pswrd_hash(krstv.pswrd,
frm.pswrd.data):
            lgnkrstv(krstv, rmbr=frm.rmbr.data)
            nxtpage = rqvst.args.get('next')
            return redirect(nxtpage) if nxtpage else
redirect(url_for('main.home'))
        else:
            return render_template('login.html', title='Login', frm=frm)

@krstvs.route("/lgout")
def lgout():
    lgout_krstv()
    return redirect(url_for('main.home'))

```

```

@krstvs.route("/aknt", methods=['GET', 'POST'])
@loginrequired
def aknt():
    frm = UpdateAkntFrm()
    if frm.vldton_sbmt():
        if frm.picture.data:
            picture_file = save_picture(frm.picture.data)
            krntkrstv.image_file = picture_file
            krntkrstv.krstvname = frm.krstvname.data
            krntkrstv.elpst = frm.elpst.data
            datbas.session.commit()
            return redirect(url_for('krstvs.aknt'))
        elif rqvst.method == 'GET':
            frm.krstvname.data = krntkrstv.krstvname
            frm.elpst.data = krntkrstv.elpst
            image_file = url_for('static', filename='profile_pics/' +
krntkrstv.image_file)
            return render_template('aknt.html', title='Aknt',
                                   image_file=image_file, frm=frm)
@krstvs.route("/krstv/<string:krstvname>")
def krstv_psts(krstvname):
    page = rqvst.args.get('page', 1, type=int)
    krstv =
    Krstv.query.filter_by(krstvname=krstvname).first_or_404()
    psts = Pst.query.filter_by(avtor=krstv)\
        .order_by(Pst.date_psted.desc())\
        .paginate(page=page, per_page=5)
    return render_template('krstv_psts.html', psts=psts,
krstv=krstv)

@krstvs.route("/rstpswrd", methods=['GET', 'POST'])
def rstqvst():
    if krntkrstv.isavtrsd:
        return redirect(url_for('main.home'))
    frm = RqvstResetFrm()
    if frm.vldton_sbmt():
        krstv =
    Krstv.query.filter_by(elpst=frm.elpst.data).first()
    return redirect(url_for('krstvs.login'))
    return render_template('rstqvst.html', title='Reset Pswrd',
frm=frm)

@krstvs.route("/rstpswrd/<token>", methods=['GET', 'POST'])
def rsttoken(token):
    if krntkrstv.isavtrsd:
        return redirect(url_for('main.home'))
    krstv = Krstv.verify_rsttoken(token)

```

```

    if krstv is None:
        return redirect(url_for('krstvs.rstrqvst'))
    frm = ResetPswrdFrm()
    if frm.vldton_sbmt():
        hshdpswrd =
bcrpt.generate_pswrd_hash(frm.pswrd.data).decode('utf-8')
        krstv.pswrd = hshdpswrd
        datbas.session.commit()
        return redirect(url_for('krstvs.login'))
    return render_template('rsttoken.html', title='Reset Pswrd',
frm=frm)

class RgstrtnFrm(FlaskFrm):
    krstvname = StringField('Krstvname', valdtrs=[DataRequired(),
Length(min=2, max=20)])
    elpst = StringField('Elpst',
                        valdtrs=[DataRequired(), Elpst()])
    pswrd = PswrdField('Pswrd', valdtrs=[DataRequired()])
    confirm_pswrd = PswrdField('Confirm Pswrd',
                                valdtrs=[DataRequired(),
EqualTo('pswrd')])
    sbmt = SbmtField('Sign Up')

    def vldtkrstvname(self, krstvname):
        krstv =
Krstv.query.filter_by(krstvname=krstvname.data).first()
        if krstv:
            raise ValidationError('Pomyłka')

    def vldtelpst(self, elpst):
        krstv = Krstv.query.filter_by(elpst=elpst.data).first()
        if krstv:
            raise ValidationError('Pomyłka.')

class LoginFrm(FlaskFrm):
    elpst = StringField('Elpst',
                        valdtrs=[DataRequired(), Elpst()])
    pswrd = PswrdField('Pswrd', valdtrs=[DataRequired()])
    rmbt = BooleanField('Rmbt Me')
    sbmt = SbmtField('Login')

class UpdateAkntFrm(FlaskFrm):
    krstvname = StringField('Krstvname',
                            valdtrs=[DataRequired(), Length(min=2,
max=20)])

```

```

    elpst = StringField('Elpst',
                        valdtrs=[DataRequired(), Elpst()])
    picture = FileField('Update Profile Picture',
valdtrs=[FileAllowed(['jpg', 'png'])])
    sbmt = SbmtdField('Update')

    def vldtkrstvname(self, krstvname):
        if krstvname.data != krntkrstv.krstvname:
            krstv =
Krstv.query.filter_by(krstvname=krstvname.data).first()
            if krstv:
                raise ValidationError('Error.')

    def vldtelpst(self, elpst):
        if elpst.data != krntkrstv.elpst:
            krstv =
Krstv.query.filter_by(elpst=elpst.data).first()
            if krstv:
                raise ValidationError('Error')

class RqvstResetFrm(FlaskFrm):
    elpst = StringField('Elpst',
                        valdtrs=[DataRequired(), Elpst()])
    sbmt = SbmtdField('Rqvst Pswrd Reset')

    def vldtelpst(self, elpst):
        krstv = Krstv.query.filter_by(elpst=elpst.data).first()
        if krstv is None:
            raise ValidationError('Pomyłka.')

class ResetPswrdFrm(FlaskFrm):
    pswrd = PswrdField('Pswrd', valdtrs=[DataRequired()])
    confirm_pswrd = PswrdField('Confirm Pswrd',
                                valdtrs=[DataRequired(),
EqualTo('pswrd')])
    sbmt = SbmtdField('Reset Pswrd')

datbas = SQLAlchemy()
bcrpt = Bcrpt()
lgnmanager = LoginManager()
lgnmanager.lgnview = 'krstvs.login'
lgnmanager.lgnmessage_category = 'info'

def stvornapp(config_class=Config):
    app = Flask(__name__)

```

```

app.config.from_object(config_class)

datbas.init_app(app)
bcrypt.init_app(app)
lgnmanager.init_app(app)

admn = Admn(app,
             name="YettiBlog - Admn",
             template_mode="bootstrap4")
admn.add_view(SecurityBlogModelView(models.Krstv,
datbas.session))
admn.add_view(SecurityBlogModelView(models.Pst,
datbas.session))
return app

class Krstv(datbas.Model, KrstvMixin):
    id = datbas.Column(datbas.Integer, primary_key=True)
    krstvname = datbas.Column(datbas.String(20), unique=True,
nullable=False)
    elpst = datbas.Column(datbas.String(120), unique=True,
nullable=False)
    image_file = datbas.Column(datbas.String(20), nullable=False,
default='default.jpg')
    pswrd = datbas.Column(datbas.String(60), nullable=False)
    psts = datbas.relationship('Pst', backref='avtor', lazy=True)

    def gtrsttoken(self, expires_sec=1800):
        s = Serializer(krntapp.config['SECRET_KEY'], expires_sec)
        return s.dumps({'krstv_id': self.id}).decode('utf-8')

    def verify_rsttoken(token):
        s = Serializer(krntapp.config['SECRET_KEY'])
        try:
            krstv_id = s.loads(token)['krstv_id']
        except:
            return None
        return Krstv.query.get(krstv_id)

    def __repr__(self):
        return f"Krstv('{self.krstvname}', '{self.elpst}',
'{self.image_file}')"

class Pst(datbas.Model):
    id = datbas.Column(datbas.Integer, primary_key=True)
    title = datbas.Column(datbas.String(100), nullable=False)
    date_psted = datbas.Column(datbas.DateTime, nullable=False,
default=datetime.utcnow)

```

```
        content = datbas.Column(datbas.Text, nullable=False)
        krstv_id = datbas.Column(datbas.Integer,
datbas.ForeignKey('krstv.id'), nullable=False)

    def __repr__(self):
        return f"Pst('{self.title}', '{self.date_psted}')
```

ДОДАТОК В
Слайди презентації

Міністерство освіти і науки України
Національний університет «Запорізька політехніка»
Кафедра програмних засобів

**Дослідження та реалізація сервісу
багатокористувацького блогінгу**

Виконав: ст. гр. КНТ-124м

Данило ЗІНОВ'ЄВ

Керівник:
к. т. н., доцент

Матвій ІЛЬЯШЕНКО

Рисунок В.1 – Слайд 1

Об'єкт, предмет та мета роботи

Об'єкт дослідження – процес розробки програмного забезпечення для багатокористувацького блогінгу.

Предмет дослідження – програмні засоби для багатокористувацького блогінгу.

Метою роботи є дослідження та розробка програмного забезпечення для багатокористувацького блогінгу.

2

Рисунок В.2 – Слайд 2

Завдання роботи

Для досягнення поставленої мети у кваліфікаційній роботі магістра необхідно розв'язати такі задачі:

- виконати аналіз предметної області та програмних засобів для ведення блогів;
- розробити модель взаємодії користувача з програмним забезпеченням для багатокористувацького блогінгу;
- здійснити проектування програмного забезпечення для багатокористувацького блогінгу;
- створити програмне забезпечення для багатокористувацького блогінгу;
- виконати тестування розробленого програмного забезпечення для багатокористувацького блогінгу.

3

Рисунок В.3 – Слайд 3

Порівняння програмного забезпечення для багатокористувацького блогінгу

Критерій порівняння	Tumblr	Medium	Write.as
Простота використання	+	+	+-
Можливості монетизації	+-	+	-
Налаштування дизайну	+	+-	-
Підтримка медіаконтенту	+	+	+-
Відсутність реклами	+-	+	+
Придатність для командної роботи	-	+-	+

4

Рисунок В.4 – Слайд 4

Порівняння мов програмування

Критерій порівняння мов програмування	Мова програмування		
	Python	PHP	JavaScript
Читабельність коду	+	+-	+-
Розвинена спільнота	+	+	+-
Інтеграція з сучасними технологіями	+	+-	+
Зручність для створення ПЗ для багатокористувацького блогінгу	+	-	-
Безпека	+	+-	+-

5

Рисунок В.5 – Слайд 5

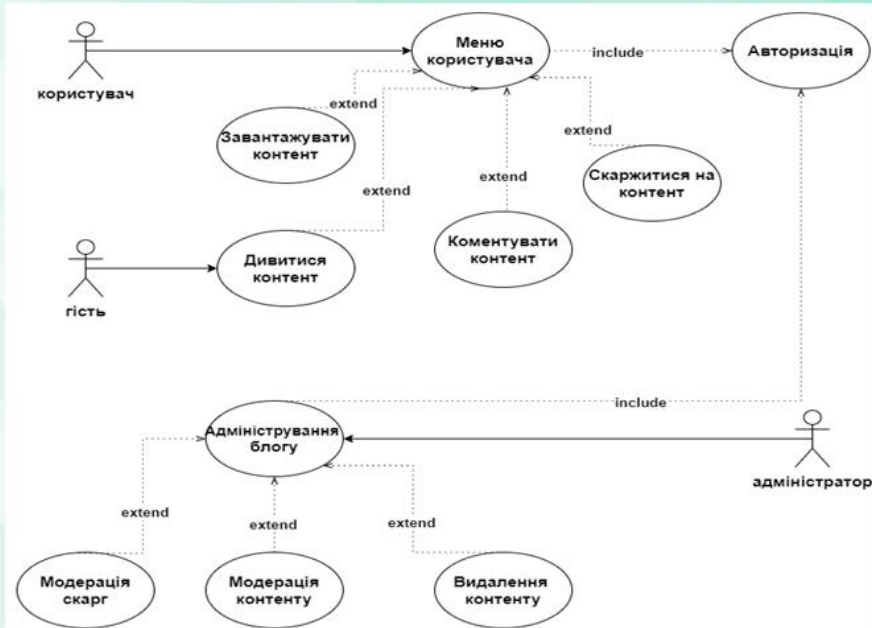
Порівняння фреймворків

Критерій порівняння фреймворків	Фреймворки		
	Flask	Django	Pyramid
Гнучкість архітектури	+	+-	+
Швидкість розробки	+	+-	+-
Легкість налаштування	+	-	+-
Спільнота та підтримка	+	+	+-
Продуктивність для малих і середніх проєктів	+	+-	+-

6

Рисунок В.6 – Слайд 6

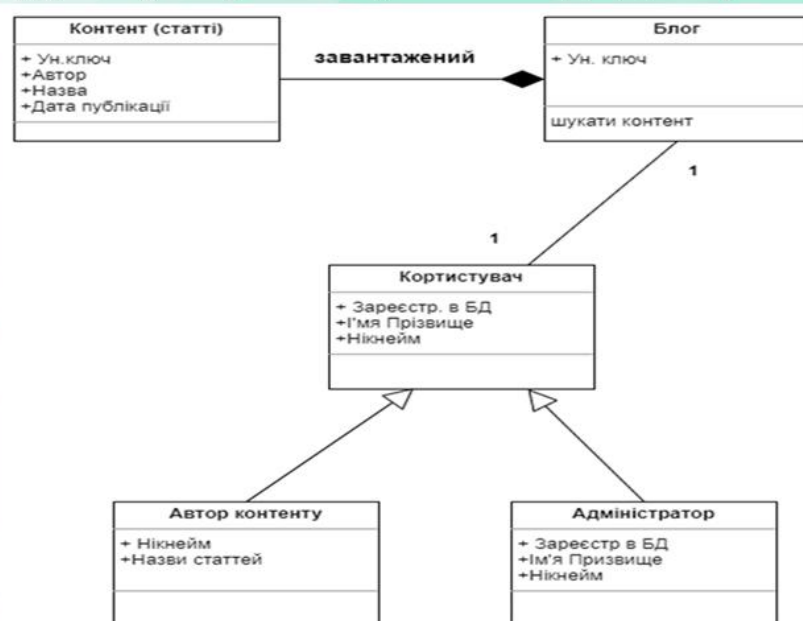
Модель взаємодії користувача з програмою



7

Рисунок В.7 – Слайд 7

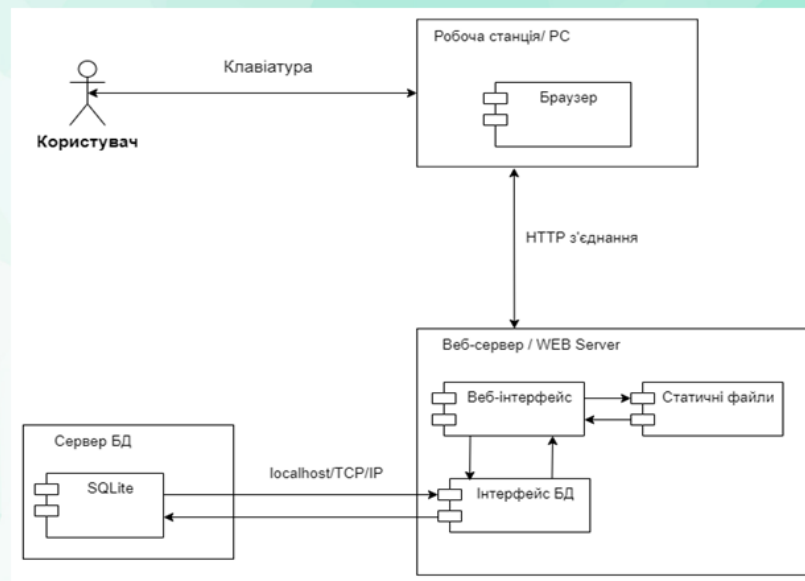
Структура програми у вигляді діаграми класів



8

Рисунок В.8 – Слайд 8

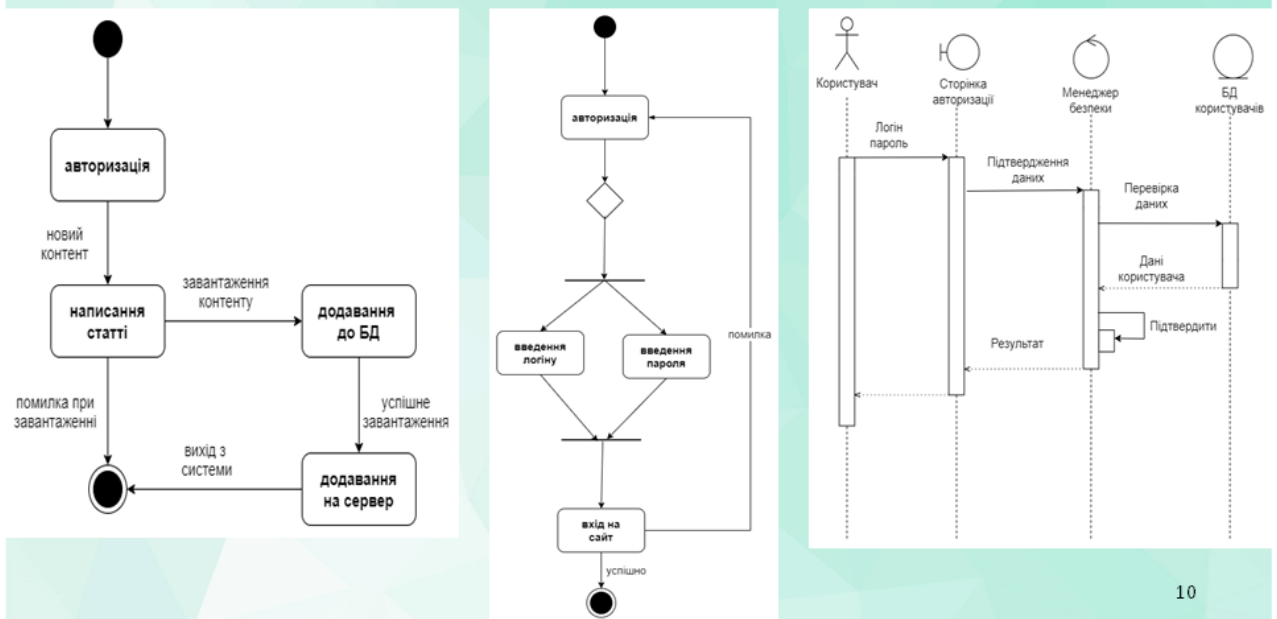
Діаграма компонентів



9

Рисунок В.9 – Слайд 9

Функціонування програмного забезпечення



10

Рисунок В.10 – Слайд 10

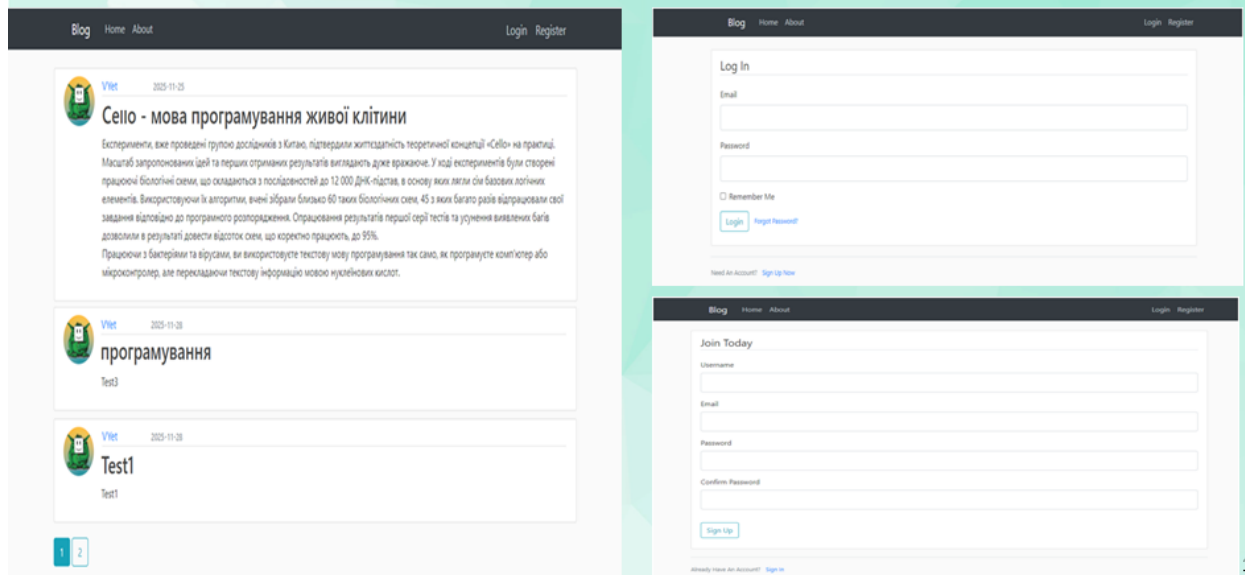
Архітектура програмного забезпечення для багатокористувацького блогінгу



11

Рисунок В.11 – Слайд 11

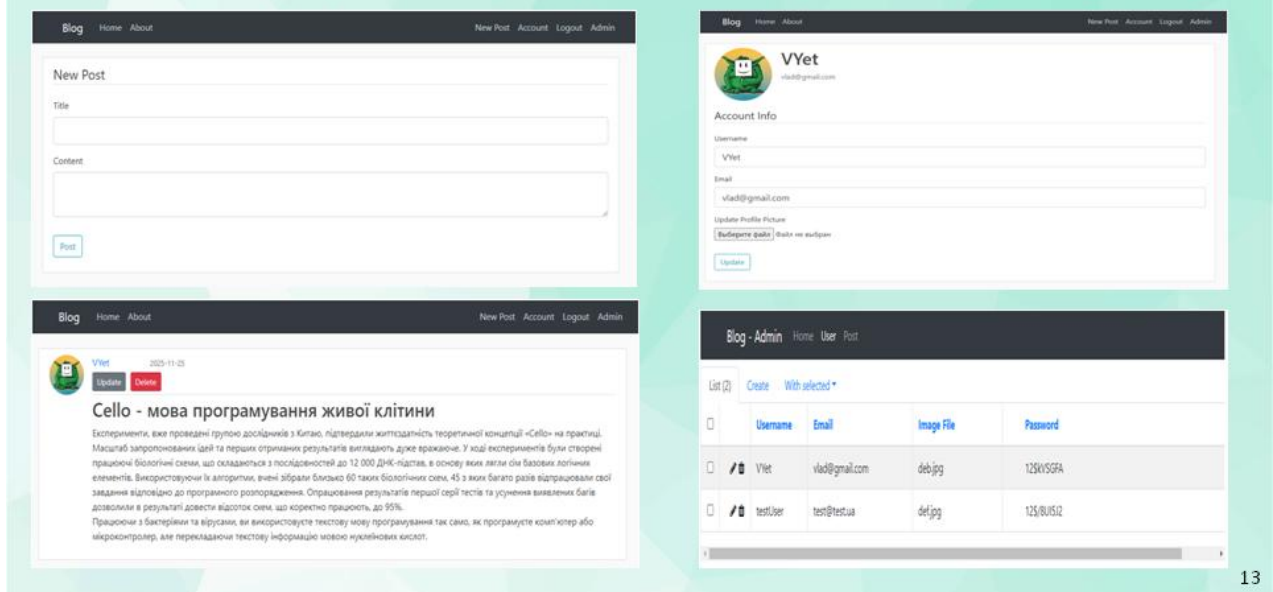
Робота з програмою. Головна сторінка. Авторизація та реєстрація



12

Рисунок В.12 – Слайд 12

Робота з програмою. Додавання та редагування постів. Сторінка профілю користувача. Сторінка адміністратора



13

Рисунок В.13 – Слайд 13

Висновки

В ході виконання дипломної кваліфікаційної роботи магістра було проаналізовано та досліджено процес розробки програмного забезпечення для багатокористувацького блогінгу.

Створено модель взаємодії користувача з програмним забезпеченням для багатокористувацького блогінгу, яка подана у вигляді діаграми прецедентів, що відображає ролі та їхні взаємозв'язки з функціональними можливостями програми, демонструючи розподіл прав доступу та специфіку взаємодії кожного типу користувачів із програмним забезпеченням.

Практична цінність роботи полягає у тому, що розроблено програмне забезпечення для багатокористувацького блогінгу.

Результати тестування програми показали, що вона працює коректно та може використовуватися за призначенням.

14

Рисунок В.14 – Слайд 14