Київський столичний університет імені Бориса Грінченка
Факультет інформаційних технологій та математики
Кафедра комп’ютерних наук

Допущено до захисту
Завідувач кафедри комп’ютерних наук, доктор техн. наук, професор
Андрій БОНДАРЧУК
«01» червня 2026 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня «бакалавр»
Спеціальність 122 Комп’ютерні науки
Освітня програма 122.00.01 Інформатика

Тема: ПЛАТФОРМА ДЛЯ СТВОРЕННЯ Й ОБМІНУ КОНТЕНТОМ З МОЖЛИВІСТЮ ЗАЛУЧЕННЯ ШТУЧНОГО ІНТЕЛЕКТУ “IDEANET”

Виконала
студентка групи ІНб-1-22-4.0д
Богданова Марія Володимирівна

_____
(підпис)
Науковий керівник
кандидат технічних наук,
доцент Ірина МЕЛЬНИК

_____
(підпис)

Київ – 2026

Київський столичний університет імені Бориса Грінченка
Факультет інформаційних технологій та математики
Кафедра комп'ютерних наук

«Затверджую»
Завідувач кафедри комп'ютерних наук, доктор техн. наук, професор
Андрій БОНДАРЧУК
31 жовтня 2025 р.

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ БАКАЛАВРА

«Платформа для створення й обміну контентом з можливістю залучення штучного інтелекту IdeaNet»

Виконавець:студентка групи ІНб-1-22-4.0д.
спеціальності 122 Комп'ютерні наук,
освітньої програми 122.00.01 Інформатика
Богданова Марія Володимирівна

1. Вихідні дані: Законодавство та нормативні акти в інформаційній сфері України, наукові праці та електронні ресурси в напрямку досліджень, хмарні сервіси Firebase, сучасні системи управління контентом, технології штучного інтелекту, OpenAI API, інструменти генерації тексту та зображень, матеріали про інформаційну безпеку веб-додатків.
2. Основними завданнями бакалаврської роботи є: Огляд наукових публікацій та сучасних технологічних рішень. Обґрунтування мети, об'єкта, предмета та методів дослідження. Аналіз сучасних платформ поширення контенту. Розробка архітектури веб-платформи IdeaNet. Тестування платформи, аналіз результатів, формулювання висновків, підготовка тез та презентаційних матеріалів.
3. Пояснювальна записка: Обсяг – до 84 стор. формату А4 комп'ютерного набору з дотриманням вимог стандарту і методичних рекомендацій кафедри.
4. Графічні матеріали: Презентація, діаграма архітектури системи зі структурою бази даних та інтерфейс користувача веб-платформи.
5. Додатки: Лістинги програмного коду, фрагменти коду JavaScript, конфігурація Firebase, скріншоти роботи системи, результати тестування.
6. Строк подання роботи на кафедру: *«1» червня 2026 р.*

Науковий керівник: кандидат технічних наук доцент _____ Мельник І.Ю. _____дата	Виконавець: студентка групи ІНБ-1-22-4.0д _____ Богданова М.В _____дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Строк виконання етапу роботи	Примітка
1	Затвердження теми кваліфікаційної роботи	31.10.2025	Виконано
2	Аналіз проблеми, вивчення аналогів, формування цілей і завдань	01.11.2025 11.01.2026	Виконано
3	Аналіз сучасних рішень, літературних джерел та вибір оптимальних технологій для реалізації системи	26.01.2026 15.03.2026	Виконано
4	Дослідження архітектури програмного забезпечення та проектування основних модулів системи.	16.03.2026 31.03.2026	Виконано
5	Розробка, інтеграція та тестування основних функціональних модулів	01.04.2026 15.04.2026	Виконано
6.	Проведення тестування, аналіз результатів, виявлення та усунення помилок	16.04.2026 30.04.2026	Виконано
7.	Впровадження готового проекту	01.05.2026 15.05.2026	Виконано
8.	Створення звіту з кваліфікаційної роботи	25.05.2026	Виконано
9.	Подання роботи на перевірку, проходження антиплагіату, внесення правок керівника	27.05.2026	Виконано
10.	Підготовка презентаційних матеріалів, тексту доповіді та захист кваліфікаційної роботи	12.06.2026	Виконано
11.	Захист кваліфікаційної роботи	15.06.2026	

«Затверджую»

Науковий керівник

к.т.н., Ірина Мельник.

Індивідуальний план склала

Богданова М.В.

РЕФЕРАТ

Кваліфікаційна робота: 78 с., 16 рис., 6 таблиць, 37 посилань.

Актуальність:Веб-платформи для створення, зберігання та розповсюдження інформаційного контенту є особливо важливими в епоху цифровізації суспільства. Для задоволення зростаючого попиту на швидке виробництво якісного контенту все більше потрібні технологічні інструменти, такі як штучний інтелект (ШІ), здатні автоматизувати процеси генерації тексту та зображень. Розробка сучасної платформи для створення та поширення ресурсів через інтелектуальні сервіси є актуальним науковим і практичним завданням.

Об'єкт дослідження: процес створення, публікації та обміну цифровим контентом у вебсередовищі.

Предмет дослідження: методи, програмні засоби та технології створення веб-платформ за допомогою хмарних сервісів та штучного інтелекту.

Мета роботи: надати загальний та відкритий хмарний ресурс для написання та публікації контенту, а також його розповсюдження, використовуючи інструмент ШІ IdeaNet.

Завдання:

1. Аналіз сучасних платформ для генерації контенту та потоку контенту;
2. Дослідження можливостей використання штучного інтелекту у веб-додатках;
3. Обґрунтування технологій, на яких буде побудовано програмний продукт;
4. Розробка архітектури веб-платформи та систем зберігання даних;
5. Створення модулів реєстрації та входу користувачів;
6. Створення функцій для публікації, редагування, пошуку та перегляду статей;
7. Додавання генерації тексту/зображень у систему (використання інструментів від OpenAI);

8. Тестування розробленої системи на практиці.

Основні результати: Під час кваліфікаційної роботи був розроблений веб-портал IdeaNet для створення та обміну контентом. Система реєстрації та авторизації користувачів була реалізована за допомогою Firebase Authentication. Були створені модулі для додавання, редагування, видалення та перегляду статей. Зберігання даних здійснювалося за допомогою Firebase Firestore та Firebase Storage. Було застосовано пошук та фільтрацію контенту. Додані сервіси штучного інтелекту для створення текстових матеріалів та зображень. Це зробило дизайн адаптивним, легким та сумісним з темною темою. Система також була функціонально протестована, і її працездатність була перевірена.

Практичне значення дослідження: полягатиме у впровадженні потенціалу для розгортання платформи IdeaNet як системи блогів, навчального ресурсу, корпоративного порталу або тематичного сервісу для генерації контенту з використанням технологій ШІ.

Ключові слова: веб-платформа, контент, штучний інтелект, Firebase, OpenAI, JavaScript, генерація тексту, система блогів, хмарні технології, IdeaNet.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

AI (Artificial Intelligence) – штучний інтелект

API (Application Programming Interface) – програмний інтерфейс для взаємодії між програмними компонентами

CSS (Cascading Style Sheets) – каскадні таблиці стилів для дизайну веб-сторінок

CRUD (Create, Read, Update, Delete) – основні операції з даними: створення, читання, оновлення та видалення.

DALL·E – система генерації зображень на основі штучного інтелекту, розроблена OpenAI.

Firebase – хмарна платформа від Google для розробки веб- та мобільних додатків.

Firestore – NoSQL база даних Firebase для зберігання інформації в хмарному середовищі.

HTML (HyperText Markup Language) – мова розмітки гіпертекстових документів.

HTTP (HyperText Transfer Protocol) – протокол передачі гіпертекстових даних через Інтернет.

HTTPS (HyperText Transfer Protocol Secure) – безпечна версія протоколу HTTP.

IDE (Integrated Development Environment) – інтегроване середовище для розробки програмного забезпечення.

IdeaNet – веб-платформа для створення та обміну контентом з використанням технологій штучного інтелекту.

JavaScript – мова програмування для створення інтерактивних веб-додатків.

JSON (JavaScript Object Notation) – текстовий формат для обміну даними.

NoSQL – тип нереляційних баз даних.

OpenAI – компанія, що розробляє технології штучного інтелекту та мовні моделі.

SDK (Software Development Kit) – набір інструментів для розробки програмного забезпечення.

UI (User Interface) – користувацький інтерфейс.

UX (User Experience) – користувацький досвід взаємодії з системою.

URL (Uniform Resource Locator) – адреса веб-ресурсу.

Web API – інтерфейс для взаємодії веб-додатків із зовнішніми сервісами.

ЗМІСТ

ВСТУП

Інформаційні технології відіграють важливу роль у комунікації, навчанні, роботі та обміні інформацією. Розвиток Інтернету й веб-технологій сприяв появі платформ для створення та поширення контенту, які використовуються в різних сферах. Сучасні системи забезпечують зручність, адаптивність і високу швидкодію.

Перспективним напрямком є впровадження штучного інтелекту, що автоматизує створення контенту, оптимізує процеси та підвищує ефективність інформаційних систем.

Ще одним важливим напрямом сучасної веб-розробки є впровадження хмарних технологій у роботу сервісів. Хмарні платформи забезпечують зберігання даних, реалізацію систем аутентифікації, масштабованість та можливість швидкого запуску програмних продуктів без потреби в складній серверній інфраструктурі.

На цьому тлі завдання розробки сучасної веб-платформи, яка об’єднує функціонал систем управління контентом зі штучним інтелектом, стає надзвичайно актуальним. Такі платформи спрямовані на створення ефективного середовища для публікації та обміну інформацією в цифровому форматі, автоматизацію генерації самого контенту, а також забезпечують інтуїтивну й комфортну взаємодію користувачів із веб-ресурсами.

Актуальність цієї тематики полягає в необхідності створення інноваційної веб-платформи для генерації та обміну контентом із впровадженням сучасних інструментів штучного інтелекту. Це дозволить не лише автоматизувати процеси створення матеріалів і вдосконалити користувацький досвід, а й розширити функціональні можливості системи, адаптуючи її до викликів сучасного цифрового середовища.

Метою кваліфікаційної роботи є створення платформи для створення та обміну контентом, яку можна доповнити штучним інтелектом IdeaNet.

Для реалізації цього необхідно вирішити такі завдання:

- Дослідити існуючі платформи для обміну та створення контенту;

- Виявити потенційні випадки використання штучного інтелекту у веб-середовищі;

- Обґрунтувати вибір технологій та інструментів;

- Налаштувати архітектуру веб-додатку;

- Створити процес реєстрації та авторизації користувачів;

- Розробити функції для публікації, редагування та перегляду статей;

- Забезпечити пошук та фільтрацію контенту;

- Інтегрувати генерацію тексту та зображень за допомогою інструментів штучного інтелекту;

- Перевірити функціональну можливість системи в лабораторії.

Об’єктом дослідження є процес створення та обміну цифровим контентом із використанням сучасних вебтехнологій та штучного інтелекту.

Предметом дослідження є інформаційна платформа для створення й обміну контентом “IDEANET”, її функціональні можливості, механізми взаємодії користувачів та інтеграція технологій штучного інтелекту.Методи дослідження: вивчення області дослідження; методи дизайну веб-дизайнерів; технології веб-розробки на стороні клієнта; хмарні сервіси Firebase; інтеграція API штучного інтелекту; тестування веб-додатків.

Практичні наслідки виявлених результатів можуть бути засновані на дизайні веб-платформи IdeaNet і служити як система блогінгу, освітній ресурс, інформаційний портал та/або сервіс для автоматизованого створення контенту з використанням технологій штучного інтелекту.

Результати роботи апробовано шляхом публікації тез доповіді на конференції Інформаційні технології – 2026: зб. тез XII Всеукраїнської науково-практичної конференції молодих учених, 14 трав. 2025 р., м. Київ / Київський столичний університет імені Бориса Грінченка. с. 65-69

РОЗДІЛ 1
АНАЛІЗ СУЧАСНИХ ПЛАТФОРМ ДЛЯ СТВОРЕННЯ ТА ОБМІНУ КОНТЕНТОМ

1.1 Особливості сучасних веб-платформ

Сучасні веб-сайти є важливими для інформації, технологій, комунікації та створення цифрового контенту. Розвиток веб-технологій, а також широке поширення Інтернету у світі призвели до великої кількості постачальників інформаційних послуг, які публікують, зберігають, редагують різні продукти та публікують їх. Ці платформи складаються з блогів та новинних сайтів, соціальних мереж, освітніх ресурсів, форумів та спеціалізованих інформаційних пропозицій. Доступний у реальному часі контент є однією з ключових функціональних особливостей сучасних веб-платформ.

У цифрову епоху, з новими пристроями тощо, користувачі можуть легко створювати, редагувати та публікувати інформацію без особливої уваги до місця розташування пристрою чи фізичного місця. Це також більш доступно для деяких видів «новинних каналів або додатків на основі пристроїв», які можуть використовуватися користувачами та можуть бути доступні безпосередньо на платформах для ширшої аудиторії. Веб-платформи підтримують продуктивність на ПК, планшетах або смартфонах для використання та мобільності і повинні забезпечувати більшу зручність. Інтерактивність є важливою особливістю сучасних систем знань. Учасники можуть відповідати один одному, а також залишати коментарі, реакції, оцінки, рейтинги та контент. Останнє сприяє інтерактивним механікам і створює соціалізацію навколо тем або ресурсів, з якими взаємодіє аудиторія, і як це інформує створення онлайн-спільнот навколо них [6].

Адаптивні технології дизайну зараз використовуються в сучасних веб-платформах для дизайну. Завдяки цьому адаптивний інтерфейс може адаптувати структуру та зовнішній вигляд сторінки залежно від екрану та розміру пристрою. Адаптивний дизайн є базовою вимогою для зручної взаємодії користувачів із

системою [7]. Для адаптивних інтерфейсів ми використовуємо HTML5, CSS3, JavaScript та сучасні CSS-фреймворки, такі як Tailwind CSS та Bootstrap. Хмарні технології також використовуються як платформи сучасних додатків. Гнучкість хмарних рішень для зберігання інформації, управління базами даних, автентифікації користувачів та розгортання програмного забезпечення полягає в тому, що не потрібно підтримувати серверну інфраструктуру [15; 16]. Які з найпопулярніших інструментів веб-розробки у вашому розпорядженні? Можливо, найпопулярнішим інструментом веб-розробки є платформа Firebase, яка може бути використана для швидкого створення масштабованих веб-додатків. Платформи наступного покоління зосереджуються на низькому використанні ресурсів та високій швидкості роботи. Асинхронні методи завантаження контенту, кешування, динамічне оновлення інформації та завантаження на клієнтську сторону можуть враховувати це. Завдяки сучасним технологіям JavaScript можливі інтерактивні інтерфейси без необхідності перезавантаження сторінки. Це значно покращує наш досвід.

Технології штучного інтелекту (ШІ) займають особливе місце в сучасних веб-системах [3]. Інструменти на основі ШІ виконують завдання автоматичного створення контенту, обробки тексту, генерації зображень, аналізу поведінки користувачів та персоналізації інформації для споживачів. Системи генерації тексту на основі великих мовних моделей автоматизують створення статей, описів, повідомлень та інших текстових продуктів. Генератори зображень забезпечують створення графічного контенту на основі текстових описів. Інтеграція технологій штучного інтелекту у веб-системи відкриває нові можливості для автоматизації роботи користувачів та підвищення ефективності, будь то написання якісних, високоякісних інформаційних документів, які користувачі знайдуть цікавими, коли вони входять у веб-середовище. Такі пропозиції вже працюють у блогових системах, маркетингових сервісах, освітніх системах та інструментах цифрової комунікації.

Безпека даних також є важливою вимогою для сучасних веб-платформ. Інформаційні системи повинні захищати особисті дані своїх користувачів,

запобігати несанкціонованому доступу та забезпечувати безпечне розповсюдження інформації через Інтернет. Для цього ми використовуємо механізми автентифікації, методи шифрування даних, безпечні HTTPS-протоколи та засоби контролю доступу [12]. Вони включають безпечні мережі обміну даними, а також зашифровані контейнери для передачі даних. Сучасні веб-платформи характеризуються високим ступенем масштабованості. Архітектура системи повинна бути здатна підтримувати збільшену кількість користувачів, збільшення обсягу даних та функціональних можливостей без надмірного зниження продуктивності.

Таблиця 1.1.

Характеристика та технологічний стек сучасних веб-платформ

Категорія	**Особливості та функціональні можливості**	**Використовувані технології та інструменти**
Користувацька взаємодія	Інтерактивність (коментарі, реакції, рейтинги), створення онлайн-спільнот, доступ до контенту в режимі реального часу.	JavaScript, асинхронні методи завантаження (AJAX/Fetch), сучасні JS-фреймворки.
Доступність та мобільність	Можливість роботи з будь-якого пристрою (ПК, планшети, смартфони) незалежно від фізичного розташування.	Кросплатформені веб-технології, мобільні додатки на основі пристроїв.
Інтерфейс та дизайн	Адаптивність (автоматична зміна структури сторінки під розмір екрана), зручність (UX/UI).	HTML5, CSS3, Tailwind CSS, Bootstrap.
Інфраструктура та розгортання	Використання хмарних рішень для зберігання даних, відмова від власної серверної інфраструктури, масштабованість.	Firebase, хмарні сервіси (Cloud Computing), модульна розробка.
Продуктивність та швидкість	Низьке використання ресурсів, динамічне оновлення інформації без перезавантаження сторінок.	Кешування, оптимізація клієнтської сторони, асинхронність.
Штучний інтелект (AI)	Автоматична генерація контенту (текст, зображення), персоналізація інформації, аналіз поведінки користувачів.	Великі мовні моделі (LLM), генератори зображень на основі AI-алгоритмів.
Безпека даних	Захист особистих даних, запобігання несанкціонованому доступу, шифрування каналів зв'язку.	Протоколи HTTPS, механізми автентифікації, методи шифрування, зашифровані контейнери.

Хмарні сервіси та модульна розробка дозволяють добре вирішувати аспекти масштабування програмного забезпечення. Таким чином, сучасні веб-

сайти є складними інформаційними системами, що поєднують інструменти для створення контенту, інтерактивної взаємодії з користувачами, технології хмари та штучного інтелекту. Впровадження сучасних технологій пропонує способи доставки робочих, масштабованих та зручних веб-додатків з акцентом на потреби користувачів та потреби сучасного цифрового світу.

Усі ці зміни глобального Інтернету - призвели до розвитку багатьох інформаційних сервісів, які публікують, зберігають, редагують та розповсюджують різноманітні матеріали. Багато з цих цифрових середовищ пропонують системи блогів, новинні сайти, соціальні мережі, освітні ресурси, форуми та спеціалізовані інформаційні сервіси. Доступ до контенту в режимі реального часу є однією з ключових особливостей сучасних веб-платформ. Користувачі можуть легко створювати, змінювати та публікувати дані незалежно від їх фізичного розташування та типу пристрою. Для роботи на веб-платформах люди можуть отримати доступ до всього з персональних комп'ютерів, планшетів та телефонів - ви можете швидко та комфортно працювати з веб-платформами. Однією з ключових особливостей сучасних інформаційних систем є інтерактивність. Користувачі діляться думками, коментарями, реакціями, оцінками, відповідями, схемами оцінювання та повідомленнями зі своїми колегами.

Інтерактивні функції сприяють залученню аудиторії та формуванню онлайн-спільнот. Адаптивний дизайн забезпечує зміну структури й вигляду сторінок відповідно до розміру екрана, що є важливою умовою зручного використання системи. Для створення адаптивних інтерфейсів використовуються HTML5, CSS3, JavaScript і фреймворки, зокрема Tailwind CSS та Bootstrap.

Сучасні платформи також застосовують хмарні технології для зберігання даних, автентифікації користувачів і роботи програм без власної серверної інфраструктури. Однією з популярних платформ для веб-розробки є Firebase. варіантів, що надає підхід до створення масштабованих веб-додатків.

Ми маємо веб-платформи наступного покоління, які прагнуть бути швидкими, а також максимально використовувати доступні ресурси. Це досягається за допомогою асинхронних методів завантаження даних, кешування, динамічного оновлення інформації та оптимізації клієнтської програми. Нова технологія JavaScript дозволяє створювати інтерактивні інтерфейси безпрецедентним чином без необхідності перезавантажувати сторінку. Технології штучного інтелекту займають унікальне місце в сучасних веб-системах. Інструменти штучного інтелекту використовуються для самогенерованого контенту, аналізу тексту, створення зображень, аналізу поведінки користувачів та персоналізованої інформації. Великі системи генерації тексту на основі мови можуть автоматизувати створення статей, генерацію описів, створення повідомлень та інші форми текстового виходу.

Генератори зображень дозволяють створювати ілюстрації з текстових описів. Інтеграція рішень штучного інтелекту у веб-платформи дозволяє автоматизувати завдання користувачів та підвищити ефективність процесу створення контенту інформації. Це рішення, які наразі використовуються в блогових додатках, маркетингових платформах, освітніх платформах та додатках для цифрової комунікації. Безпека даних є однією з основних вимог, на яких залежать сучасні веб-платформи. Щоб зберегти конфіденційність користувачів та запобігти несанкціонованому доступу, інформаційні системи повинні гарантувати безпечну передачу інформації за допомогою Інтернету. Механізми автентифікації даних, шифрування даних, безпечні протоколи HTTPS та системи контролю доступу використовуються для цієї мети. Сучасні веб-платформи також мають високий ступінь масштабованості. Архітектура системи повинна дозволяти ефективне масштабування користувачів, даних та можливостей у різних сферах застосування, не впливаючи значно на продуктивність. І оскільки хмарні сервіси та модульний підхід до розробки дозволяють вирішувати масштабування програмного забезпечення масштабованим чином, ви можете вирішити це ефективно. І таким чином сучасні веб-платформи відображають складні інформаційні системи, що містять

поєднання інструментів для створення контенту, інтерактивної взаємодії, хмарних сервісів, технологій та штучного інтелекту. З сучасними технологіями ми можемо створювати додатки, які працюють відповідно до своїх користувачів та потреб сучасного цифрового світу.

1.2 Системи керування контентом та блогові сервіси

Управління контентом є одним з основних програмних пакетів, що використовується для створення та підтримки веб-ресурсів, доступних нашим користувачам. Вони дозволяють користувачам публікувати, редагувати, структурувати та навіть організовувати свої інформаційні файли, усуваючи потребу в програмуванні. Системи CMS зазвичай використовуються в блогах, новинних порталах, корпоративних веб-сайтах, освітніх інструментах та інформаційних сервісах з цієї причини. CMS (Content Management System) означає Система управління контентом, яка є веб-ресурсом з інструментом зберігання даних, створення та управління контентом, основні функції якого полягають в автоматизації роботи з генерування, зберігання та оновлення інформації в Інтернеті. Ці системи надають користувачам графічний інтерфейс для текстового матеріалу, зображень та медіа, або іншої інформації. Однією з ключових переваг CMS є спрощення управління веб-сторінками. Користувачі можуть створювати та редагувати нові сторінки, редагувати опубліковані документи самостійно, додавати зображення, реорганізовувати ресурс, структурувати ресурс та керувати доступом до інформації самостійно. Це значно знижує час оновлення контенту разом з ефективністю інформаційних систем.

Найбільш широко використовувані CMS - це WordPress, Joomla, Drupal та Blogger. Кожна з цих систем має свої особливості, переваги та області використання. WordPress є домінуючою CMS у світі сьогодні. Це була платформа для блогів, яка спочатку з'явилася як блог, але еволюціонувала, щоб стати платформою для створення різноманітних веб-сайтів. Система WordPress може бути розширена з безліччю різних використань та функцій, оскільки вона підтримує численні теми, плагіни та інше. Найбільші переваги WordPress

включають те, що вона дуже проста у використанні, має велику спільноту користувачів та інтегрується з зовнішніми сервісами. Joomla є дещо складнішою, ніж WordPress, і зосереджується на проектуванні інформаційних веб-сайтів та корпоративних веб-сайтів. Це модульна архітектура з управлінням для користувачів і пропонує більший ступінь налаштування користувацького інтерфейсу. Joomla, однак, вимагає іншого підходу до підготовки адміністратора. Drupal є професійною програмою CMS, що використовується з високими вимогами до безпеки та масштабованості для реалізації складних інформаційних систем. Налаштування було значно спрощено для розробників завдяки модульній архітектурі та можливості створення великих веб-ресурсів з великою кількістю користувачів. Недоліком Drupal є те, що налаштування програми буде більш складним, створюючи високу технічну підготовку для розробника. Безкоштовний блогінг - це набір платформи Google Blogger.

Таблиця 1.2

Порівняльний аналіз популярних CMS та блогових сервісів

Назва системи	**Цільове призначення**	**Основні переваги**	**Недоліки / Обмеження**
WordPress	Універсальна платформа (від блогів до сайтів)	Простота використання, величезна кількість тем та плагінів, велика спільнота.	Потребує регулярних оновлень безпеки; може бути перевантаженою плагінами.
Joomla	Корпоративні та інформаційні сайти	Модульна архітектура, гнучкість налаштування інтерфейсу.	Складніша за WordPress, вимагає спеціальної підготовки адміністратора.
Drupal	Складні високонавантажені системи	Високий рівень безпеки та масштабованості, гнучкість для розробників.	Високий поріг входження, потребує значної технічної підготовки.
Blogger	Особисті блоги (Google)	Швидке створення без налаштування сервера, інтеграція з сервісами Google.	Обмежений функціонал для складних проєктів, залежність від екосистеми Google.
IdeaNet (концепція)	Інтелектуальна блогова платформа	Інтеграція зі штучним інтелектом (AI), використання Firebase, швидкість та хмарна масштабованість.	На етапі розробки/концептуалізації.

Сервіс орієнтований на швидке створення особистих блогів без необхідності налаштування серверного середовища. Blogger надає набір базових функцій для публікації статей та введення мультимедійних елементів, а також для управління зовнішнім виглядом сторінки. Сервіси для блогів є окремим класом сучасних інформаційних систем [6]. Платформи для блогів є загальними джерелами, які створені для публікації авторських письмових матеріалів на щоденній основі у вигляді статей, новинних статей або постів. Особливості системи блогів включають створення контенту, систему коментарів, систему тегів та механізм пошуку на додаток до взаємодії з користувачем. Сучасні сервіси для блогів почали впроваджувати інтеграцію з соціальними мережами та хмарними сервісами. Така функція дозволяє автоматизувати процес розповсюдження контенту, синхронізацію даних та покращення взаємодії користувача з платформою. Кодування блогів CMS та блогів є останньою тенденцією, і з недавнім зростанням штучного інтелекту (AI) та збору даних, ця трансформація очікується бути частішою. Інструменти, такі як AI, також виконують генерацію тексту, автоматичне створення зображень, аналіз поведінки користувачів, персоналізовані рекомендації та контроль за модерацією контенту. Додавання AI призводить до значного розвитку функцій сучасної веб-інфраструктури [4; 14].

Сучасні платформи також повинні пропонувати адаптивний дизайн, швидкість, безпеку даних та навіть зручність для мобільних пристроїв. Користувачі прагнуть швидкої роботи системи, чудового інтерфейсу та можливості створювати контент без необхідності налаштування глибоких технічних налаштувань. Через огляд існуючих систем управління контентом та сервісів для блогів ми можемо побачити, що більшість існуючих платформ можуть обробляти різноманітні інформаційні матеріали, але не всі вони мають будь-яку інтеграцію з рішеннями штучного інтелекту або сучасними хмарними сервісами. Як результат, концептуалізація платформи IdeaNet, яка поєднує ефективність системи блогів з функціональністю створення контенту AI та

використанням Firebase, є областю актуальності та життєздатності у веб-розробці.

1.3 Використання штучного інтелекту у веб-застосунках

Розробка та вдосконалення сучасних інформаційних технологій є захопливою та швидко розвивається галуззю у сучасному світі. Одним із способів використання штучного інтелекту в розробці сучасних інформаційних технологій є автоматизація складних операцій обробки даних, аналіз великих обсягів даних, розробка рекомендаційних систем та надання користувачам продуктів машинного навчання. Технології штучного інтелекту також успішно застосовувалися до багатьох різних типів веб-додатків протягом останніх кількох років [3; 10].

Сьогодні штучний інтелект (ШІ) використовується в таких різноманітних сферах, як електронна комерція, освіта, цифровий маркетинг, медицина, фінанси, соціальні мережі, журналістика та веб-розробка [1; 3]. Технологія штучного інтелекту (ШІ) революціонізує спосіб використання Інтернету. Надаючи засоби, за допомогою яких ми створюємо ефективніші веб-ресурси, автоматизуємо процес створення контенту та забезпечуємо ефективніший користувацький досвід [3; 12].

Більшість веб-застосунків зараз розробляються з використанням генеративного штучного інтелекту для створення тексту; це одна з основних сфер застосування продуктів на основі штучного інтелекту у веб-застосунках. Моделі великих мов (LLM) можуть бути використані для створення матеріалів сайту, включаючи статті, описи, повідомлення, відповіді на повідомлення, рекламний контент тощо. Такі моделі генеративного штучного інтелекту навчаються на великих наборах тексту для отримання відповідної інформації з запитів користувачів та надання точних відповідей. Найновіші моделі генеративного штучного інтелекту створюють тексти дуже високої якості; вони схожі за стилем на людські тексти. Це надає нові можливості для автоматизації створення контенту в системах блогів, інформаційних системах та

маркетингових службах. Іншим ключовим розвитком у штучному інтелекті стала можливість створення графічного контенту за допомогою моделей штучного інтелекту. Системи генерації зображень призначені для створення ілюстрацій/графіки/цифрових зображень на основі описового текстового введення від користувача. Ці технології використовуються у веб-дизайні, рекламі, комунікаціях, креативних секторах тощо. Створення зображень на основі штучного інтелекту забезпечує значно спрощені засоби візуального виробництва, дозволяючи користувачам набагато швидше отримувати власну унікальну графіку.

AI-технології навіть широко використовуються для персоналізації інформації в сучасних веб-застосунках. Саме тут у гру вступають алгоритми машинного навчання - вони ґрунтують свої рекомендації на кожному конкретному користувачеві, виявляючи закономірності в поведінці, інтересах і активності в системі. Такий механізм є поширеним у соціальних мережах, стримінгових сервісах, онлайн-магазинах та інформаційних платформах.

Основні сфери та функціональні можливості впровадження технологій штучного інтелекту в архітектуру сучасних веб-систем представлено на рис. 1.1.

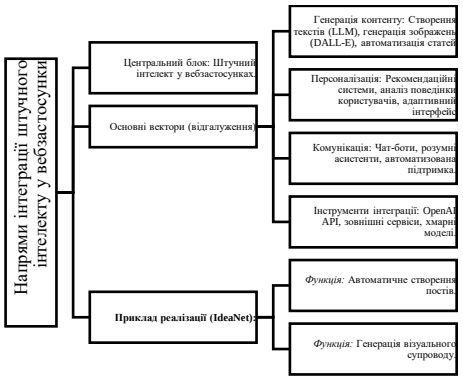


Рисунок 1.1.Напрями інтеграції штучного інтелекту у веб-застосунки

Одним із найактуальніших застосувань AI є розробка чат-ботів і розумних помічників. AI-асистенти допомагають спілкуватися з користувачами в автоматизованому режимі, відповідати на запитання, здійснювати пошук інформації та полегшувати підтримку клієнтів наживо. Чат-боти знімають

навантаження на службу підтримки, водночас підвищуючи швидкість опрацювання звернень.

Важливим аспектом сучасного AI є інтеграція, що здійснюється через API. Таким чином веб-застосунки можуть взаємодіяти із зовнішніми AI-сервісами без необхідності самостійно тренувати модель. Це значно спрощує процес розробки, і робить змогу швидко запустити AI-функції у своєму програмному продукті.

OpenAI API - один із найпопулярніших сервісів штучного інтелекту. Платформа OpenAI може поєднувати генерацію тексту, обробку та аналіз природної мови й створення зображень у веб-застосунках.Надає змогу впровадити автоматизовану генерацію статей, мозковий штурм контенту та створення графічних або фото робіт із текстового опису за допомогою OpenAI API [3; 17].

Платформа IdeaNet забезпечує автоматичне створення текстових матеріалів і зображень, використовуючи технології штучного інтелекту. Для ввімкнення генерації тексту з боку OpenAI було реалізовано API, яке дозволяє користувачам отримувати готовий контент на конкретну тему. DALLE – надає змогу у генерації графічного контенту на основі текстових команд користувача

AI-технології, розроблені для веб-платформи, дозволяють швидко створювати контент, автоматизувати рутинні процеси та підвищувати функціональні можливості системи. Користувачі можуть швидше складати матеріали, отримувати допомогу з написанням статей і автоматично створювати візуальне компонування для публікацій.

Використання штучного інтелекту має низку переваг, але також певні ризики та обмеження. Серед них - помилки в згенерованій інформації, залежність від зовнішніх сервісів, проблеми конфіденційності щодо використання персональних даних і контроль якості отриманого контенту. Саме тому веб-системи сьогодні мають поєднувати автоматизацію з ручним редагуванням і перевіркою інформації користувачем.

З огляду на це важливим напрямом розвитку сучасних інформаційних систем є використання AI-технологій у веб-застосунках. Інтеграція з AI дає

змогу значно розширити можливості веб-платформ, підняти автоматизацію та взаємодію користувачів із цифровим контентом на новий рівень.

1.4 Аналіз аналогів та існуючих рішень

Сьогодні існує багато онлайн-платформ для створення, публікації та поширення контенту. Ці системи відрізняються за функціональністю, структурами даних, інтеграцією з сучасними технологіями та зручністю користування. Розуміючи наявні рішення, можна побачити, де вони сильні, а де не справляються, що, у свою чергу, допомагає визначити вимоги до створення вашої конкретної веб-платформи [1; 2].

WordPress - одна з відомих платформ для блогінгу. Цей фреймворк використовують для розробки персональних блогів, новинних ресурсів, корпоративних порталів і інформаційних платформ [14]. Переваги WordPress - простота у використанні, велика кількість готових варіантів дизайну та можливість розширення функціональності за допомогою плагінів. Користувачі можуть писати й публікувати статті за лічені хвилини без тривалого навчання [12].

WordPress має власний набір переваг і недоліків. Багато активних сторонніх плагінів можуть знижувати продуктивність системи та рівень безпеки. Крім того, стандартна версія платформи не пропонує інструментів для генерації контенту, що працює завдяки штучному інтелекту - для цього потрібні окремі модулі та налаштування [3; 14].

Другим популярним рішенням є Blogger - блоги та сервіс публікації статей від компанії Google. Він відзначається простим керуванням і дуже швидким стартом власного блогу без необхідності турбуватися про складну серверну розробку. Але Blogger навряд чи підходить, адже можливості для «дрібних» доповнень істотно обмежені, а його функції значно поступаються тим, що доступні в сучасних CMS-системах [4; 14].

Протягом років Medium зайняв важливе місце серед сучасних платформ для публікації контенту. Акцент цього сервісу зроблено на публікації текстів та

створенні оригінального контенту. Medium має зручний інтерфейс, чистий дизайн і приємні умови для читання. Платформа дає змогу використовувати механізм рекомендацій контенту та забезпечує взаємодію між авторами й читачами.

З одного боку, це обмежена персоналізація, а також відсутність повного контролю над структурою ресурсу - з іншого, Medium є звичною платформою, де можна обійтися без платних рекламних методів. Користувачі не можуть повністю змінити або впровадити власні програмні рішення в систему без використання певних сторонніх сервісів.

Крім того, існують сучасні AI-платформи для автоматизованої генерації контенту. Серед прикладів - Jasper AI, Copy.ai та Notion AI [6; 12].

Ідеально з цими системами тут працює генеративний штучний інтелект, який автоматично пише тексти та описи, генерує ідеї або підтримує вас у роботі, пов’язаній із контентом. Ці сервіси значно спрощують створення матеріалів, однак більшість із них забезпечують лише генерацію тексту й не реалізують повноцінну платформу для блогінгу з можливістю публікувати або редагувати, але я підкреслюю взаємодію з користувачем. Також деякі сервіси, пов’язані з ШІ, є платними та накладають обмеження на використання функціональності.

Звичні платформи наразі - це відомі рішення з системами блогінгу та деякими ШІ-помічниками/глибокими інтеграціями або генеративними рішеннями на кшталт текстів без системи обміну інформацією між користувачами.

У процесі дослідження було визначено основні вимоги до підготовки веб-платформи IdeaNet.Система має забезпечувати:

– реєстрацію та автентифікацію користувачів;

– написання, оновлення та видалення статей;

– можливість пошуку та фільтрації контенту

– Генерацію тексту за допомогою штучного інтелекту;

– генерацію графічного контенту;

– адаптивний користувацький інтерфейс;

– ступінь використання хмарних сервісів для зберігання даних.

На відміну від більшості інших рішень сьогодні, платформа IdeaNet поєднує можливості блогінгу та хмарного сховища Firebase,OpenAI дозволяє використовувати сервіси AI DALLE,саме це дає змогу створювати платформу для публікації контенту з сучасними веб-можливостями (для інтелектуального генерування тексту та зображень).

У результаті аналіз подібних продуктів, який було виконано, свідчить про актуальність для створення платформи IdeaNet та доцільність використання сучасних веб-технологій і інструментів штучного інтелекту для створення функціональної системи обміну контентом.

1.5 Постановка задачі дослідження

Дослідження сучасних платформ для виробництва та поширення контенту довело, що більшість наявних рішень забезпечують лише мінімум для збереження інформації, проте системи не підтримують інтеграцію штучного інтелекту та нових хмарних сервісів. Деякі платформи призначені виключно для розміщення блогів; інші лише орієнтовані на створення контенту за допомогою ШІ. Унаслідок цього користувачам часто потрібно використовувати кілька різних сервісів одночасно.

Сьогодні користувачам потрібна крос-браузерна веб-аплікація, яка має можливість створювати, редагувати та публікувати контент, із функціями автоматичного генерування тексту та зображень. Також деякі ключові вимоги: зручний інтерфейс, продуктивність обладнання, надійне зберігання даних, підтримка мобільних пристроїв.

Отже, основною задачею дослідження є створення сучасної веб-платформи, яка забезпечить створення та поширення контенту за допомогою технологій штучного інтелекту. Ідеальна система має бути інтуїтивною, підтримувати сучасні механізми взаємодії та забезпечувати інтеграцію сервісів ШІ через використання API.

Мета дослідження - розробити веб-платформу IdeaNet для створення та розповсюдження контенту, із можливістю використання ШІ для автоматичного генерування текстових і графічних матеріалів.

Тому, щоб досягти бажаної мети, необхідно:

- вивчити сучасні технології веб-розробки та хмарні сервіси;

- дослідити можливості застосування штучного інтелекту в веб-застосунках;

- пояснити, чому були обрані інструменти розробки та архітектура системи;

- спроєктувати веб-платформу та розробити модель даних для зберігання;

- реалізувати систему реєстрації та авторизації користувачів;

- модулі для публікації, редагування та перегляду статей;

- мати можливості пошуку та фільтрації контенту;

- потім - інтегрувати OpenAI API для генерації тексту;

- генерація зображень (DALLE);

- зберігання даних за допомогою Firebase;

- протестувати продуктивність системи.

У ході виконання роботи планується створити веб-застосунок, де користувачі можуть створювати та публікувати власний контент, використовувати можливості ШІ та взаємодіяти з ним, а також працювати лише через сучасний веб-інтерфейс.

Для реалізації задач використовується мова програмування JavaScript, технології HTML5 та CSS3,хмарна платформв Firebase, а також сервіси OpenAI API. Ці компоненти забезпечують розширене налаштування, щоб створити спосіб роботи, що сприяє впровадженню найсучасніших і яскравих веб-рішень із високим рівнем функціональності та оперативності.

Ітерація цієї роботи має призвести до створення платформи IdeaNet, яка об’єднає сильні сторони блогу, хмарні технології та автоматизацію генерації контенту за допомогою інструментів штучного інтелекту.

1.6 Висновки до розділу

У першому розділі було розглянуто системи керування контентом та сервіси блогінгу і проведено аналіз,які сучасні веб-платформи для створення та поширення контенту мають вбудовані функції.

Було розглянуто «Ключові функції сучасної CMS, їхні переваги та недоліки», а також можливості використання хмарних технологій у веб-розробці. Зокрема, якщо технології штучного інтелекту застосовуються до веб-додатків.

Досліджено, як формуються сучасні сервіси для текстового та графічного контенту на базі ШІ, і чи можна інтегрувати їх у веб-платформу через API.

У ході дослідження наявних рішень було встановлено, що сучасні платформи або не дозволяють використовувати інструменти ШІ, або пропонують лише обмежену функціональність щодо створення та керування контентом. І це підтверджує актуальність створення IdeaNet - яка підтримує інтегровані системи блогінгу, хмарні сервіси Firebase та інструменти ШІ. На основі проведеного аналізу було визначено основні цілі дослідження, сформульовано вимоги до функціональних можливостей системи в різних ситуаціях і ролях, а також обрано подальші напрями розвитку веб-платформи.

РОЗДІЛ 2
ПРОЄКТУВАННЯ ВЕБ-ПЛАТФОРМИ IDEANET

2.1 Архітектура веб-платформи та вибір технологій розробки

Програмна архітектура - це один із великих блоків процесу компонування веб-застосунку в сучасні часи. Вона розглядає загальний дизайн системи, обмін даними між модулями компонентів, обробку даних і компонування всіх функцій, сформованих у різних функціональних підрозділах. Добра архітектура робить системи безпечними, такими, що легко підтримувати, і допомагає гарантувати, що зі зростанням проєктів з’являтиметься більше можливостей для побудови нової функціональності.

Веб-платформа IdeaNet була розроблена в клієнт-серверній архітектурі із використанням хмарних технологій (з використанням Firebase як бекенду). Ця архітектура забезпечує негайну взаємодію з інтерфейсом для користувача, безпечне зберігання даних і інтеграцію сервісів із можливостями штучного інтелекту.

Було визначено такі основні компоненти в архітектурі платформи:

– фронтендна частина веб-застосунку;

– хмарні сервіси Firebase;

– база даних Firestore;

– система автентифікації користувача;

– файлове сховище Firebase Storage;

– API сервісів OpenAI;

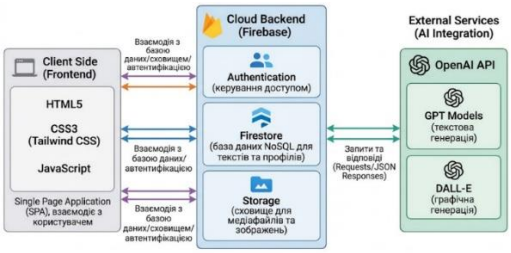


Рисунок 2.1.Архітектурна модель веб-платформи IdeaNet

Клієнтська частина системи реалізована з використанням стандартних веб-технологій: HTML5, CSS3 та JavaScript. Такий підхід забезпечує комфортну та плавну взаємодію користувача із системою — завдяки динамічному оновленню вмісту сторінки без її повного перезавантаження, що характерно для односторінкових застосунків (SPA).

Для стилізації та побудови адаптивного макета використовується бібліотека Tailwind CSS. Вона дозволяє швидко реалізовувати сучасний вигляд інтерфейсу за допомогою утилітарних класів безпосередньо в розмітці, без написання окремих CSS-файлів. Завдяки вбудованій системі контрольних точок (breakpoints) інтерфейс коректно відображається на різних типах пристроїв — від мобільних телефонів до настільних комп'ютерів, автоматично підлаштовуючи розташування елементів під розмір екрана.Архітектура веб-застосунку (Single Page Application)

За цією моделлю майже вся логіка спочатку працює на клієнтській стороні, а динамічні оновлення контенту відбуваються без перезавантаження сторінки. Це має позитивний вплив на продуктивність системи та підвищує задоволеність користувачів.

Для реалізації серверної частини та зберігання даних використовується платформа Firebase. Вона надає інфраструктуру для запуску веб-застосунку «з коробки» і не потребує створення власного сервера. Залежно від реалізації Firebase надає набір активностей хмарних технологій, необхідних для роботи основної функціональності будь-якої системи.

Для реалізації автентифікації користувача використовується Firebase Authentication. Вона надає послугу реєстрації користувачів, входу та керування обліковими записами. Це однозначно підвищує безпеку системи та зменшує складність впровадження механізмів доступу, використовуючи готовий до експлуатації сервіс автентифікації.

Текстові дані зберігаються у firebase firestore.

Firestore - це хмарна база даних NoSQL, яка дозволяє працювати з колекціями документів у дуже простому режимі у реальному часі. База даних

зберігає інформацію про користувачів, пости, коментарі та інші компоненти системи.

Firebase Storage - це місце, де ми можемо зберігати наші зображення та мультимедійні файли.

Таблиця 2.1

Технологічний стек розробки IdeaNet

Компонент системи	**Обрана технологія**	**Призначення**
Frontend	JavaScript, Tailwind CSS	Реалізація SPA та адаптивного інтерфейсу
Backend-as-a-Service	Firebase	Інфраструктура, автентифікація, хостинг
База даних	Cloud Firestore	Зберігання даних у реальному часі (NoSQL)
Генерація контенту	OpenAI API (GPT, DALL-E)	Автоматизація створення тексту та зображень
Медіасховище	Firebase Storage	Надійне зберігання графічного контенту

Використання хмарного сховища файлів забезпечує графіки, які завантажуються швидко, та на захищеному сервері.

Firebase Storage також надає масштабованість і контроль від доступу до файлів.

Інтеграція сервісів штучного інтелекту є одним із ключових компонентів архітектури платформи IdeaNet.

Для генерації текстового контенту вона викликається через OpenAI API. Ця система передає текстовий запит у зовнішні сервіси штучного інтелекту, а згенерований результат повертається назад і також доступний для користувачів під час створення статті.

Модель DALLE використовується для генерації графічного контенту.Сервіс може створювати зображення, використовуючи текстові команди з різних варіантів, які користувачьнадає як текстовий опис. Це дає змогу повністю автоматизувати процес створення малюнків.

Взаємодія між клієнтською частиною та зовнішніми сервісами реалізується через виклики API.Такий підхід забезпечує архітектурну гнучкість, а також спрощує інтеграцію нових функціональних модулів у майбутньому.Включає архітектуру на веб-платформі підтримку адаптивного темного режиму теми.Це забезпечує комфортне використання системи на різних типах пристроїв та покращує візуальне сприйняття інтерфейсу.

Один із основних параметрів, який зазвичай ігнорують під час проєктування системи - це безпека даних. З цією метою використовується надійний механізм автентифікації користувачів, доступ до бази даних обмежено, а HTTPs увімкнено. Безпомилковий захист сучасної інформації: Firebase дає вам змогу використовувати всі ті самі інструменти (відстеження, моніторинг і захист) без необхідності окремо керувати безпекою сервера.

Отже, архітектура веб-платформи IdeaNet побудована на сучасних веб-технологіях, хмарних сервісах та інструментах штучного інтелекту. Обрана архітектура забезпечує високу продуктивність, технічну масштабованість, зручність для користувачів і потенціал для розширення можливостей платформи в нових розробках. На основі описаної архітектури розглянемо обґрунтований вибір технологічного стеку для її реалізації.

Вибір технології розробки - один із критично важливих етапів створення сучасного веб-рішення. Лише добір програмних інструментів, мов програмування та хмарних сервісів визначає, наскільки добре працюватиме система, наскільки легко її можна буде підтримувати, чи залишатимуться дані захищеними, а також чи існуватимуть можливості подальшого розвитку програмного продукту.

Під час створення IdeaNet було вирішено використовувати сучасні веб-технології, які дозволяють реалізувати адаптивний інтерфейс і швидку взаємодію із системою у поєднанні з технологіями ШІ.Він працює переважно з такими технологіями, як HTML5, CSS3, JavaScript Tailwind CSS Firebase та OpenAI API.

Ретельно впорядковані та систематичні веб-сторінки у браузері створюються за допомогою мови розмітки HTML5. Таким чином гарантується, що елементи інтерфейсу формуються, підтримуються мультимедійні можливості та сторінки коректно відображаються у сучасних браузерах.HTML5 - це стандарт для веб-розробки, і більшість веб-застосунків створюються з використанням адаптивного дизайну.

CSS3 використовується для визначення стилю компонування інтерфейсу. Каскадні таблиці стилів (CSS) забезпечують плавний сучасний дизайн, анімації, адаптивне позиціонування елементів і підтримку режимів. Використання CSS3 для адаптивності інтерфейсу та світлого/темного вигляду на платформі IdeaNet.

Більшість основної логіки цього веб-застосунку написано на JavaScript. Він забезпечує динамічну взаємодію між користувачами та веб-сторінками, керування подіями, керування API та оновлення контенту без перезавантаження всієї сторінки. JavaScript - одна з найпоширеніших мов програмування для веб-дизайну, і він працює в усіх нових браузерах.

Для пришвидшення розробки інтерфейсу було використано Tailwind CSS. Цей CSS-фреймворк швидко створює елементи адаптивного дизайну з багатьма готовими класами стилізації. Tailwind CSS полегшує підтримку інтерфейсу та забезпечує сучасний вигляд для веб-застосунку без написання великої кількості спеціального CSS.

Під час розробки Firebase,було обрано як основний хмарний сервіс для розгортання веб-застосунку. Firebase - це продукт, розроблений компанією Google, який містить набір інструментів для розробки веб (PWA) і мобільних застосунків. Firebase дає можливість реалізувати серверну частину без потреби розробляти інше серверне програмне забезпечення.

Firebase Authentication використовується для впровадження системи реєстрації та автентифікації користувачів. Вона дає змогу надійно керувати обліковими записами користувачів і автентифікацію за допомогою email/пароля. Впровадження доступу з використанням готового рішення значно полегшує вашу роботу.

Дані зберігаються на платформі за допомогою Firebase Firestore. Firestore - це NoSQL хмарна база даних, що використовується для зберігання даних у колекціях і документах. Також Firestore забезпечує швидкий доступ до документів і дозволяє масштабувати всю систему разом із синхронізацією інформації в реальному часі.

Firebase Storage використовується для зберігання зображень та інших типів медіафайлів. Він надає безпечне хмарне сховище файлів, допомагає з керуванням контролем доступу та прискорює завантаження графічного контенту.

Веб-застосунок розгорнуто на Firebase Hosting. Ця послуга дає змогу швидко публікувати веб-проєкти в Інтернеті, підтримує HTTPS-з'єднання та забезпечує високу швидкість завантаження сторінок.

Однією з ключових технологій, які були використанні у цьому проєкті, є OpenAI API. Зокрема ця послуга, що використовує штучний інтелект для створення текстового контенту, за допомогою OpenAI API дозволяє користувачам автоматично генерувати текстові матеріали на певну тему.

Контент, який генерує зображення, це DALLE, і він інтегрується з OpenAI API. Ця технологія дає змогу створювати зображення за описом, наданим користувачем.

Таблиця 2.2

Технологічний стек розробки веб-платформи IdeaNet

Категорія	**Технологія**	**Призначення у проєкті**
Frontend (Розмітка та стиль)	HTML5, CSS3	Створення структури сторінок та візуального оформлення інтерфейсу.
Frontend (Логіка)	JavaScript	Забезпечення динамічної взаємодії, керування подіями та робота з API.
UI Framework	Tailwind CSS	Пришвидшення розробки адаптивного дизайну за допомогою готових класів.
Backend (BaaS)	Firebase	Хмарна інфраструктура для розгортання та керування сервером.
Автентифікація	Firebase Authentication	Керування обліковими записами користувачів та безпечний вхід.

Продовження таблиці2.2

Категорія	**Технологія**	**Призначення у проєкті**
База даних	Cloud Firestore	NoSQL база даних для зберігання контенту та синхронізації в реальному часі.
Хостинг	Firebase Hosting	Публікація проєкту в мережі з підтримкою HTTPS.
Штучний інтелект	OpenAI API (GPT, DALL-E)	Автоматична генерація текстів та графічного контенту за описом.

Ці технології обрано, оскільки вони дуже популярні, надійні та масштабовані, а також мають швидку можливість інтеграції.Вони дають змогу використати сучасні веб-технології та хмарні сервіси, щоб реалізувати AI-платформу з сучасним інтерфейсом.

Отже, обраний стек технологій забезпечує стабільну роботу веб-платформи IdeaNet, підвищує зручність у процесі розробки та дає змогу підготувати необхідну інфраструктуру для майбутнього масштабування функціональності системи.

2.2 Проєктування бази даних Firebase

Щоб зберігати записи веб-платформи IdeaNet, було обрано хмарну базу даних Firebase Firestore. Це No SQL база даних, яка зберігає дані у вигляді колекцій та документів. Наприклад, це один із способів реалізації в веб-застосунку, де нам потрібно збирати користувачів, статті, коментарі, реакції та зображення - це є керованим.

На відміну від реляційних баз даних, під час використання Firestore не слід проєктувати ієрархічну структуру з таблицями та зв’язками між відношеннями. Дані структуровані як документи, і кожен документ може містити колекцію полів. Це дозволяє гнучко змінювати структуру даних під час створення програмного продукту.

Платформа IdeaNet використовує базу даних для зберігання важливих деталей про створені користувачами статті, їхні коментарі та реакції, а також

інформації, що стосується автора публікації. База даних є суттєвою частиною веб-платформи, оскільки система забезпечує створення та поширення контенту.

Основні сутності бази даних це:

– користувачі;

– статті;

– коментарі;

– реакції;

– зображення публікацій.

Опис: колекція users - це машиноперевірювані дані про зареєстрованого користувача системи. Ці дані можуть містити унікальний ідентифікатор користувача, адресу електронної пошти, ім’я користувача, дату створення акаунта… Ви проходите автентифікацію за допомогою Firebase authentication, і ми використовуємо Firestore, щоб зберігати будь-які дані, які потрібні вашій платформі для роботи.

Основною системною колекцією є колекція articles. Вона зберігає контент, створений користувачами. Вхідні дані - це документ для кожної статті, який містить назву статті, її текстовий зміст (за наявності), посилання на зображення, що відповідають цій статті, ідентифікатор, який визначає, хто її написав, а також коли вона була створена, однак з деякими іншими даними щодо сервісу. Це потрібно для того, щоб швидко отримувати список статей, відкривати окремі редакції та виконувати пошук за назвою.

Тепер структура документа статті буде приблизно такою -

Назва колекції: articles

Поля документа:

– title - назва статті;

– ця частина статті описує її зміст, тобто це найбільш важливий виклад у статті;

pixel imageUrl - URI до зображення.

– authorId - ідентифікатор автора;

– authorEmail - електронна адреса автора;

– createdAt - дата створення статті;

– updatedAt - дата останньої зміни;

– isGeneratedByAI - прапорець, який показує, чи вона була згенерована певним типом ШІ.

Ви можете або мати окрему колекцію для коментарів, або підколекцію всередині документа статті, щоб зберігати коментарі - залежно від вашої структури. Коментар містить текст

Це включає повідомлення, ідентифікатор автора, дату створення та до якої статті воно належить. Така структура дозволяє залучати користувачів до контенту та забезпечує можливість обговорень опублікованих матеріалів.

Орієнтовна структура коментаря:

Назва колекції: comments

Поля документа:

– articleId - ідентифікатор статті;

– userId - ідентифікатор користувача;

– userEmail - email користувача;

– text - текст коментаря;

-- createdAt - дата створення коментаря;

Реакції користувачів можна зберігати, використовуючи окрему структуру даних, що містить тип реакції, ідентифікатор користувача та ідентифікатор статті. Це дозволяє відстежувати, як користувачі взаємодіють із публікаціями, а також створити базову систему оцінювання контенту.

Орієнтовна структура реакції:

Назва колекції: reactions

Поля документа:

– articleId --Article Id;

– userId - ідентифікатор користувача;

– type - тип реакції;

createdAt - дата створення реакції.

Зображення зберігаються у Firebase Storage, коли користувачі завантажують їх у статті. Тому у Firestore зберігається лише URL-посилання на відповідний файл. Це дозволить уникнути засмічення бази даних великими файлами та забезпечить просте керування вмістом зображень.

Однією з ключових переваг Firestore є його здатність передавати дані в реальному часі. Тобто ви використовуєте дані, які змінюються в базі, і вони коректно відображаються (у реальному часі) на інтерфейсі без повного перезавантаження сторінки. Це особливо важливо для платформи IdeaNet, оскільки нові (публікації та оновлення контенту) практично одразу відображаються для користувачів.

Швидкий доступ до статей, прості можливості масштабування та функціональне розширення були факторами під час проєктування бази даних. Зрештою ви могли б розширити структуру бази колекціями для категорій, тегів, підписок, повідомлень або системи рекомендацій.

Слід врахувати питання безпеки даних. Правила доступу до FirestoreAllow you to specify what type of user can read, create, edit or delete documents. Ви можете обмежити редагування або видалення статті для автора, дозволяючи будь-якому автентифікованому користувачеві перегляд опублікованих матеріалів.

Отже, використання Firebase Firestore для платформи IdeaNet виглядає раціональним вибором, оскільки ця база даних пропонує гнучку структуру для зберігання даних, швидкий доступ до інформації та можливість масштабування вгору.

Поряд із простою інтеграцією з іншими функціями Firebase обрана модель даних здатна реалізувати основну функціональність системи та забезпечує майбутнє розширення, щоб створити веб-платформу для її можливостей.

2.3. Реалізація системи авторизації та інтерфейсу користувачів

Авторизація користувачів є ключовою частиною кожної веб-платформи, оскільки вона відповідає за контроль доступу до функціональності застосунку. Авторизація дозволяє визначити користувача, надати йому права доступу для

створення контенту, збереження його публікацій, перегляду його матеріалів та взаємодії з іншими елементами системи. Сервіс Firebase Authentication обрано для реалізації авторизації поверх платформи IdeaNet. Це хмарне вбудоване рішення, надане Google, щоб ви могли швидко налаштувати свій застосунок для реєстрації, входу та виходу користувачів без необхідності створювати власний серверний код для керування автентифікацією.

Це дозволяє зареєструвати користувача за допомогою email і пароля у веб-застосунку, який розробляється. Дані користувача, який створив обліковий запис, проходять через Firebase Authentication, а система отримує його унікальний ідентифікатор користувача. Пізніше цей ідентифікатор пов’язується з вашими статтями, коментарями та діями на платформі.

Етапи цього процесу автентифікації такі:

– введення користувачем email та pasword;

– перевірка того, що введені дані є коректними;

У Firebase Authentication надсилається запит;

– реєстрація акаунта або вхід у вже існуючий;

– збереження стану авторизації користувача в системі;

– доступ до основних можливостей веб-платформи.

Користувача після успішного входу переводять на головний інтерфейс платформи. Зокрема, він зможе писати статті та використовувати генератор зображень і тексту з підтримкою штучного інтелекту, матиме власні публікації, видимі для інших, і активно взаємодіяти з ними.

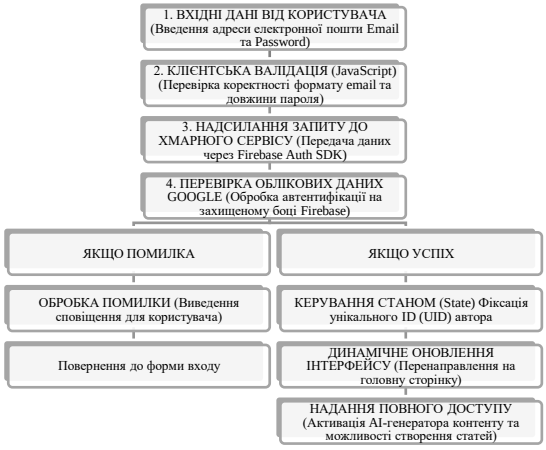


Рис. 2.2.Алгоритм процесу авторизації користувача у системі IdeaNet

Основна функціональність створення матеріалів на платформі досі вимкнена для будь-якого незареєстрованого користувача. Це запобігає несанкціонованому створенню або зміні контенту та забезпечує базовий рівень захисту даних.

У веб-застосунку збережено інформацію про те, який користувач зараз увійшов у систему, у стані застосунку. Це дозволяє динамічно змінювати інтерфейс: відображати форму введення для незареєстрованих користувачів або основний контент платформи для тих, хто вже увійшов у систему.

Ця система авторизації також визначає, чи має користувач права доступу до публікацій. Наприклад, користувач може мати можливість редагувати або видаляти лише ті статті, які були створені ним. Щоб це забезпечити, авторство кожної публікації позначається ідентифікатором. Такий підхід дозволяє налаштовувати персоналізовану роботу з контентом.

Обробка помилок під час входу/реєстрації є одним із ключових аспектів, які доведеться реалізувати. Якщо пароль неправильний, має місце некоректний формат адреси електронної пошти, акаунт уже зареєстрований або виникають інші проблеми, користувачу потрібно про це повідомити. Це дозволяє користувачеві виправити помилку одразу й підвищує зручність користування веб-платформою.

Для кращої зручності можна додати перемикання між типами входу та реєстрації у формі авторизації. Такий підхід дає змогу повторно використовувати одну логічну частину інтерфейсу для двох тісно пов’язаних процесів - створення нового акаунта та входу в наявний акаунт.

Крім того, така конфігурація також дозволяє здійснювати вихід з акаунта. Користувач виходить із системи, втрачає доступ до функцій створення та редагування контенту, а інтерфейс повертається до початкового стану з формою авторизації.

Тепер, навіщо використовувати Firebase Authentication? Є кілька переваг його використання. По-перше, розробникам не потрібно самостійно

реалізовувати логіку зберігання паролів уже на боці сервера. По-друге, сервіс надійно керує обліковими записами користувачів. По-третє, також враховано інтеграцію Firebase Authentication з іншими сервісами Firebase, такими як Firestore і Storage.

З погляду безпеки критично важливо, щоб паролі користувачів не зберігалися безпосередньо в коді веб-застосунку або в базі даних Firestore.

Firebase Authentication обробляє їх. Це зменшує кількість інцидентів витоку конфіденційної інформації та підвищує загальний рівень безпеки системи.

Втім, використання системи авторизації користувачів дозволяє ідентифікувати кожного користувача в платформі IdeaNet, забезпечуючи персоналізоване керування доступом, яке захищає контент і гарантує коректне оброблення публікацій. Firebase Authentication робить реалізацію цієї функціональності швидкою, простою та надійною без потреби створювати будь-яку спеціальну серверну інфраструктуру. Після реалізації механізмів автентифікації перейдемо до проєктування інтерфейсу користувача, що працює поверх цих механізмів.Інтерфейс користувача - це також один із ключових компонентів сучасного веб-застосунку, через який користувач взаємодіє з системою. Те, як організований інтерфейс, визначає, наскільки зручно буде користуватися платформою, як швидко ви зможете виконувати певні дії, яке враження від роботи з додатком залишиться, а також чи зможете ви повноцінно використовувати його функціональні можливості.

Веб-платформа IdeaNet була розроблена з акцентом на створення сучасного, адаптивного та зручного для користувача інтерфейсу. Основна увага приділялася тому, щоб користувачу не потрібно було проходити будь-яке технічне навчання для взаємодії з системою. Інтерфейс платформи побудовано на HTML5, CSS3, JavaScript та Tailwind CSS. Ці технології забезпечують динамічний UI з сучасним оформленням і підтримкою адаптивного дизайну.

Головна сторінка веб-платформи містить елементи керування, єдиний список статей, панель пошуку та кнопки для керування функціями користувача

в системі. Після авторизації ви зможете створювати власні публікації та використовувати функції генерації AI-контенту. Основні елементи керування розташовані у верхній частині інтерфейсу: кнопка для створення статті, перемикач між усіма статтями та персональними публікаціями, а також поле пошуку. Така модель організовує роботу користувача та дозволяє швидко переходити між функціями системи.

Вона також підтримує адаптивний дизайн - одну з важливих особливостей інтерфейсу. Автоматичне масштабування всіх елементів сторінки відповідно до розміру екрана пристрою. Це гарантує, що веб-застосунок виглядатиме коректно на персональному комп'ютері, а також буде доступним на планшетах і смартфонах. Зробіть інтерфейс адаптивним із Tailwind CSS - за допомогою фреймворка ви можете легко створювати гнучкі сітки, перебудовувати розміщення елементів інтерфейсу на різних пристроях і масштабувати інтерфейси вгору/вниз. Це забезпечує сучасний дизайн та зручність користування на багатьох пристроях.

IdeaNet має реалізовані світлу та темну теми. У верхній частині інтерфейсу одна кнопка використовується для перемикання між темами. Це також підвищує читабельність контенту ввечері та допомагає зменшити шкідливе перенапруження очей.

Загальну компонентну структуру користувацького інтерфейсу веб-платформи IdeaNet, що відображає взаєморозташування навігаційних елементів, інтерактивних блоків та адаптивної сітки контенту, зведено на рис. 2.3.

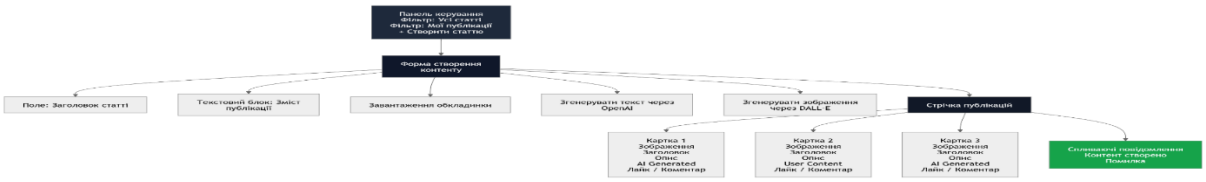


Рис. 2.3.Компонента схема та архітектура інтерфейсу користувача платформи IdeaNet

Статті додаються через форму введення даних.Форма має заголовок, основний блок і опцію додати зображення. Вона також забезпечує інтеграцію генерації тексту та зображень за допомогою ШІ. Вони можуть надати короткий опис теми, тобто текст або неформальну ілюстрацію. Інтерфейс створення статті розташований так, що користувач може зробити все, що потрібно, за один раз. Це спрощує процес і пояснює, чому дозволяє легку взаємодію з системним процесом створення контенту.

Механізм пошуку статей було додано, щоб користувачам було легше орієнтуватися в системі. Поле пошуку робить набагато простішим знайти те, що ви шукаєте, за назвою. Ваші пошукові запити є динамічними та виконуються без виходу зі сторінки, що підвищує швидкість веб-застосунку.

Список статей відображається окремими картками з короткою інформацією про публікацію. Кожна картка містить заголовок, на основі зображення, короткий опис і інтерактивні елементи. Цей підхід належним чином структурує інтерфейс і робить його легшим для розуміння.

Щоб покращити користувацький досвід, платформа сьогодні отримала насичені анімації та плавні переходи між елементами одного інтерфейсу користувача в інший. Вони забезпечують більш природну взаємодію користувача та надають дизайну більш сучасного вигляду.

Частина інтерфейсу, яка є важливою - це повідомлення, що інформують його про результати його дій. Система повідомляє користувача, що йому успішно вдалося створити контент, завантажити файл, а також про збої автентифікації або інше. Це розуміння полегшує користувачу сприйняття стану системи та обходу помилок під час роботи.

Під час проєктування інтерфейсу також враховувалися вимоги до зручності використання. Основні функції системи розташовані у доступних місцях, кнопки мають зрозумілі позначення, а структура сторінок не перевантажена зайвими елементами.

Отже, інтерфейс веб-платформи IdeaNet був розроблений та реалізований відповідно до вимог сучасного веб-дизайну та досвіду користувачів. Адаптивний

дизайн, можливість перемикання в темний режим, різноманітні AI-функції та зручна навігація забезпечують безперешкодне користування користувачами на платформі та підвищують ефективність роботи з контентом.

2.4 Інтеграція зовнішніх API та хмарних сервісів

Щоб розширити функціональність системи, сучасні веб-функції широко використовують зовнішні API та хмарні сервіси. Це дозволяє здійснювати plug-and-play інтеграцію вже наявних рішень без необхідності писати складну логіку на стороні сервера або створювати власну інфраструктуру. API значно спрощують процес розробки, економлять час і дають змогу швидко впроваджувати нові функції у веб-платформу.

Для створення платформи IdeaNet ми використали кілька зовнішніх сервісів і хмарних технологій. Найважливіші три - це Firebase, OpenAI API та DALL·E. Нам потрібно інтегрувати їх так, щоб вони забезпечували авторизацію, збереження даних, генерацію тексту та створення графічного контенту для нашої системи.

Firebase: один із базових хмарних сервісів платформи. Саме тут реалізується серверна частина веб-застосунку, і ми отримуємо набір вбудованих сервісів, необхідних для того, щоб рівень представлення міг виконувати свої функції.

Ми використовуємо різні конфігурації для веб-застосунків, щоб під’єднати Firebase, яке містить ключі проєкту та параметри доступу до сервісів платформи. До речі, після успішної ініціалізації Firebase ваш застосунок може спілкуватися і з базою даних та системою авторизації, а також із файловим сховищем.

Реєстрація та вхід користувачів у систему здійснюються за допомогою Firebase Authentication. Ця інтеграція сервісу дозволяє веб-платформі перевіряти користувачів без необхідності самостійно реалізовувати систему зберігання паролів і керувати обліковими записами користувачів.

Firebase Firestore - для зберігання даних статей, користувачів і коментарів. Дані стають доступними через запити до API, що дозволяє динамічно отримувати інформацію та оновлювати контент у міру його появи.

Наприклад, файли й зображення, прикріплені до статей, зберігаються в Firebase Storage. Вона дає змогу завантажувати візуальні файли, інтегруючи сховище файлів у хмарі, і отримувати URL-посилання для подальшої обробки контенту в веб-застосунку.

Однією з базових функцій IdeaNet є інтеграція з API OpenAI. Ця служба використовується для реалізації генерації текстового контенту. Функції застосовують штучний інтелект. Користувач вводить тему, а потім короткий текстовий запит, після чого система надсилає оригінальний текст у сервіс OpenAI та отримує згенерований текст.

Результат автоматично відображається під час створення статті в формі, з якої користувач може закласти основу для майбутніх публікацій. Це значно спрощує процес створення контенту та дає змогу краще й швидше складати тексти.

Генератор візуального контенту для контент-креатора, який використовує DALL·E. Сервіс дає змогу генерувати зображення за текстовим описом. Користувач у запиті надає короткий опис того, яке зображення він хоче, і тоді система викликає AI-сервіс та отримує готову ілюстрацію.

Після того як зображення згенеровано, ви можете прив’язати його до публікації. Ви проводите глибокі дослідження, пишете статтю, зберігаєте її на випадок «дощового дня» або як представлення для публікації. Завдяки інтеграції з DALL·E ми можемо створювати візуальні матеріали - створювати контент і розширювати можливості веб-платформи.

Ви робите ці HTTP-запити, щоб взаємодіяти з різними зовнішніми API. ExtJS - це бібліотека JavaScript для здійснення запитів до сервісів, обробки відповідей і динамічного оновлення UI. Це дає змогу швидко працювати з зовнішніми системами та формами в інтерактивних веб-застосунках.

Під час інтеграції зовнішніх сервісів важливу роль відіграє безпека даних. Використовуються захищені з'єднання HTTPS, щоб захистити цю інформацію. Доступ до цього API надається через спеціальні ключі доступу. Це дає змогу обмежити несанкціоноване використання сервісів, які виконують свої функції та забезпечують захищений зв'язок із сторонніми платформами.

Використання цих хмарних сервісів дає більше, ніж просто переваги. По-перше, це знімає навантаження з локальної інфраструктури. По-друге, хмарні сервіси допомагають нам керувати навантаженням і надійністю системи. По-третє, використання готових API значно прискорює розробку та дає змогу будувати сучасні інтеграції, не вдаючись до написання складної логіки на стороні сервера.

Отже, інтеграція його з зовнішніми API та хмарою на веб-платформі IdeaNet: архітектура. У вашій вебплатформі є ціла низка сервісів. Вони дають змогу реалізувати рішення за допомогою Firebase, API OpenAI та DALL·E. Сучасні операції системи, автоматизований процес генерації контенту та стабільна робота веб-застосунку.

2.5 Забезпечення безпеки та захисту даних

Одним із важливих аспектів розвитку сучасних веб-застосунків є безпека даних. Оскільки веб-платформи працюють із приватними даними користувачів та текстовим контентом і файлами, важливо захищати інформацію від втрати, несанкціонованого доступу або пошкодження.

В IdeaNet облікові записи користувачів захищені, дані зберігаються безпечним способом, а права доступу до функціональності системи контролюються. Для реалізації механізмів безпеки використовуються сучасні хмарні сервіси, такі як Firebase, а також захищені протоколи передавання даних.

Авторизація користувача є одним із основних компонентів системи безпеки. Ця веб-платформа використовує Firebase Authentication для безпечної реєстрації та входу користувачів у систему. Паролі користувачів не зберігаються

напряму в базі даних застосунку, а обробляються сервісами Firebase через сучасні механізми безпеки.

Після успішної авторизації система визначає, до яких функцій веб-застосунку користувач має доступ. Наприклад, лише користувач, авторизований для створення/редагування/видалення статей, може це робити, і лише автор статті може змінювати публікацію. Це запобігатиме несанкціонованому редагуванню контенту.

Для обміну даними між клієнтською частиною та хмарними сервісами використовується безпечний протокол HTTPS. Він шифрує дані під час обміну інформацією через Інтернет і зменшує імовірність перехоплення конфіденційних даних третім лицем.

Правила доступу Firebase Firestore є важливою частиною моєї системи безпеки. Вони являють собою механізми, за допомогою яких можна визначити для певних користувачів їхні права на читання, створення та зміну документів у БД. Система може надавати доступ до всіх опублікованих статей усім авторизованим користувачам, водночас дозволяючи редагувати їх лише власнику відповідної статті.

Графічний контент зберігається в Firebase Storage. Сервіс дозволяє визначати правила контролю доступу до файлів, а отже обмежує третім особам завантаження або зміну зображень.

Під час розробки особливу увагу слід приділити захисту ваших API-ключів і даних конфігурації. Для інтеграції OpenAI API в застосунки потрібні спеціальні ключі доступу, які мають бути захищені від третіх осіб. Якщо ключі зберігаються неналежним чином, можливе небажане використання сервісів або витік даних.

За наявності даних із зовнішнього API також важливо обмежувати частоту запитів і забезпечувати цілісність відповіді. Це може допомогти мінімізувати ймовірність помилок у системі, тим самим забезпечуючи стабільну роботу веб-застосунку.

Можна застосувати додаткові засоби для перевірки даних, введених користувачами, - щоб підвищити рівень безпеки на платформі. Зокрема, потрібно

звіряти текстові поля, формати адрес електронної пошти та типи файлів, які завантажуються в систему. Це запобігає помилкам і можливим атакам на веб-застосунок.

Важливою перевагою є автоматичні резервні копії та безперервна робота сервісів Firebase - за умови, що веб-платформа працює в хмарному середовищі. Хмарна інфраструктура дозволяє зменшити ризик втрати інформації у разі технічних збоїв або перевантаження системи.

Сегментація доступу до функціональних модулів у системі також є важливим аспектом захисту даних. Це означає, що неавторизовані користувачі не можуть створювати або змінювати контент, використовувати генерацію матеріалів, а також отримувати доступ до приватної інформації інших користувачів.

Під час розробки веб-платформи було дотримано базових принципів конфіденційності, цілісності та доступності інформації щодо безпеки. Ви захищаєте облікові записи та API-ключі для конфіденційності, контролюєте зміни в базі даних/через програмне забезпечення для цілісності та покладаєтеся на стабільні хмарні сервіси/датацентри, щоб забезпечити доступність.

У результаті система безпеки платформи IdeaNet зберігає персональні дані користувачів, контролює доступ до різних функцій веб-застосунку та забезпечує захищену комунікацію із зовнішніми сервісами. Завдяки Firebase Authentication, Firestore та використанню HTTPS ви вже маєте доволі сучасний рівень захисту інформації на веб-платформі.

Таким чином, система безпеки IdeaNet забезпечує захист персональних даних користувачів, контроль доступу до функціональності веб-застосунку та безпечну роботу із зовнішніми сервісами. Це поєднання Firebase Authentication, Firestore, HTTPS і механізмів керування доступом дозволяє забезпечити сучасний рівень захисту інформації на веб-платформі.

Висновки до розділу 2

Другий розділ присвячено проєктуванню веб-платформи IdeaNet та обґрунтуванню вибору основних технологій розробки. Ми встановили архітектуру системи, що складалася з клієнт-серверу з хмарними сервісами Firebase та зовнішнім ШІ за потреби.

Етап проєктування дав змогу обрати сучасний стек технологій на основі HTML5, CSS3, JavaScript, Tailwind CSS, Firebase та OpenAI API. Щоб реалізувати все це, необхідно використовувати ці технології, і було б темними віками, якби їх не було, з огляду на сучасну розробку веб-застосунків із адаптивним дизайном UI та функціями, підсиленими ШІ.

База даних Firebase Firestore структурована для зберігання даних, що стосуються користувачів, статей, коментарів і реакцій. Підхід NoSQL допомагає зберігати структуру даних гнучкою та розширювати систему до наступного, вищого рівня.

Особливо важливим стало впровадження системи авторизації користувачів із використанням Firebase Authentication. Це дало змогу контролювати доступ до функціональних можливостей платформи та підвищити безпеку даних.

Вимоги адаптивності, простоти використання та сучасних UI-дизайнів були частиною процесу проєктування інтерфейсу користувача. Додано підтримку світлої та темної тем, систему пошуку для статей та зручну навігацію між функціональними модулями, а також було включено зовнішні API та хмарні сервіси. Це реалізується завдяки використанню OpenAI API та DALL·E для створення тексту й графіки, згенерованих ШІ.

Крім того, окремо було досліджено питання безпеки та захисту даних. У цьому контексті було впроваджено механізми авторизації, правила доступу Firebase, захист через протокол HTTPS та контроль взаємодії із зовнішніми сервісами.

Отже, у результаті цього етапу проєктування сформовано архітектуру веб-платформи IdeaNet, визначено основні технології та залучено структуру системи, необхідну для подальшої реалізації функціональних модулів веб-застосунку.

РОЗДІЛ 3
РОЗРОБКА ТА РЕАЛІЗАЦІЯ ПЛАТФОРМИ IDEANET

3.1 Реалізація модулів роботи з контентом

Модуль створення та редагування статей є ключовим функціональним компонентом веб-платформи IdeaNet. Він створює модуль, що дозволяє формувати інформаційний контент, публікувати його та оновлювати в системі. Це є ключовою частиною роботи веб-платформи, оскільки лежить в основі всіх способів, за допомогою яких користувачі зможуть взаємодіяти з контентом.

Клієнтська частина: Модуль для створення статей (HTML5, CSS3 та JavaScript; Firebase Firestore + Firebase Storage)

Форма введення даних, що використовується для створення публікацій, дозволяє користувачу вказати заголовок статті, текстове наповнення, додати зображення та генеративні функції через ШІ.

Після авторизації користувач отримує доступ до кнопки створення нової статті. Форма, яку ви відкриваєте, заголовок і текстовий вміст відкриваються у формі для заповнення. Також існує можливість прикріпити фото, що супроводжує публікацію.

Обробка подій JavaScript для реалізації функціональності створення статті: коли користувач заповнює форму та натискає «опублікувати», введені ним дані перевіряються перед створенням об’єкта статті, який буде збережено в DataBase.

Система перевіряє обов’язкові поля (заголовок статті та текст) перед збереженням. Користувач отримує повідомлення, яке інформує його, що він має надати дані, якщо певні поля не заповнені.

Процес валідації даних працює та успішно зберігає статтю в Firebase Firestore як окремий документ. Він містить заголовок статті, текстові дані - контент, створений автором у публікації, дату створення та посилання або зображення.

Коли користувач додає зображення, воно завантажується в Firebase Storage. Після успішного завантаження система має URL-посилання на файл, яке

потім зберігається в базі даних разом з іншою інформацією, пов’язаною зі статтею.Це означає, що система може автоматично оновлювати список статей і відображати в інтерфейсі кожну нову публікацію після опублікування матеріалу.

Цей підхід дозволяє користувачам динамічно взаємодіяти з контентом без необхідності оновлювати сторінку.

Рис. 3.1.Схема розподілу потоків даних у модулі створення та редагування статей IdeaNet

Модуль редагування статей є основною частиною вашої системи. Користувачі можуть змінювати власні записи після того, як їх створено. Для цього в картці статті є кнопка редагування - вона доступна лише автору матеріалу.

Під час виклику редагування система отримує актуальні дані статті для форми введення. Користувач може змінювати назву, текст, редагувати зображення та доповнювати матеріал додатковими оновленнями.

Користувач зберігає відредаговану версію статті, і документ у Firebase Firestore оновлюється. База даних також може містити дату останнього редагування, що дозволяє відстежувати зміни в контенті.

Ця платформа додатково до редагування пропонує видалення статей. Видалити можуть лише автори створених ними матеріалів. Після підтвердження видалення документ видаляється з бази даних, а його зображення - з Firebase Storage (якщо застосовно).

Для покращення користувацького досвіду також впроваджені динамічні оновлення інтерфейсу, повідомлення про успішні операції та керування можливими помилками в модулі створення статей. Це означає, що система може повідомити користувача про те, що статтю створено, або про те, що виникли певні проблеми під час завантаження зображення.

Винятковою особливістю веб-платформи IdeaNet є те, що ми вбудовуємо функції ШІ в робочий процес генерації контенту. Користувач використовує генерацію тексту та автоматично отримує основу для майбутньої публікації. Крім того, зібраний матеріал також можна редагувати перед тим, як він буде опублікований онлайн - тож є простір для коригувань під час роботи з контентом.

Адаптація вимог до описаного вище дизайну, реалізована під час створення та редагування модуля статей.

Адаптивний дизайн забезпечує коректне відображення всіх елементів інтерфейсу на різних типах пристроїв, таких як смартфони, планшети та персональні комп’ютери.Отже, створений модуль створення та редагування статей повністю працює з користувачами, які створюють контент: дозволяє створювати й редагувати публікації,використовувати функції ШІ та взаємодіяти з веб-платформою через сучасний адаптивний інтерфейс. Після реалізації базового функціоналу створення статей розглянемо систему пошуку та фільтрації контенту, що забезпечує зручний доступ користувачів до

опублікованих матеріалів.Веб-платформа IdeaNet містить систему пошуку та фільтрації контенту, щоб користувачі з багатьма публікаціями могли зручно користуватися платформою. Така функціональність, по суті, дозволяє швидко знаходити потрібні матеріали та впорядковувати інформацію, яка є необхідною; це значно покращує користувацький досвід під час роботи з системою.

Одна з перших і найважливіших функцій сучасних вебплатформ - це пошук, адже обсяг даних постійно зростає. Без якісного механізму пошуку користувачеві складно швидко знайти те, що він шукає, адже немає чіткої структури або контексту основних статей і публікацій у співвідношенні з його власною роботою. Саме тому ми реалізували динамічний пошук статей за назвою та елементи фільтрації за змістом на платформі IdeaNet.

Система пошуку реалізована на JavaScript, а результати надходять із Firebase Firestore. Після відображення списку статей користувач може ввести текст у поле пошуку, що викличе функцію для оновлення результатів відповідно до того, що було введено в запиті.

Пошук виконується без перезавантаження нашої сторінки. Під час введення тексту JavaScript аналізує назви статей і показує лише ті публікації, які відповідають заданим критеріям. Це демонструє швидку взаємодію користувача з системою та робить інтерфейс більш інтерактивним.

Це вимагає виконання пошуку в реальному часі, щоб оптимізувати зручність користування. Результати автоматично оновлюватимуться під час введення тексту. Хто б не був користувачем, знайдені статті першими потрапляють до нього напряму, і вони спрямовують трафік на майже будь-який контент вашої платформи.

Назва статті, яку ви шукали, є основним критерієм пошуку. Вводячи текст у систему, вона звіряє його з заголовками публікацій і надає вам результати. Такий механізм доволі просто реалізувати і він цілком підходить для блогу.

Окрім пошукової функціональності, вебзастосунок має систему фільтрації контенту. Користувачі можуть обрати переглядати лише свої публікації або ж

бачити всі статті на платформі. Це стає особливо корисним у ситуаціях, коли користувач одночасно працює з великою кількістю матеріалів.

Фільтрація здійснюється за допомогою спеціальних кнопок або перемикачів в інтерфейсі. Після вибору потрібного режиму система обробляє та фільтрує дані, щоб показувати лише релевантні матеріали.

Отже, ви можете відфільтрувати власні публікації, враховуючи, що для цього використовується id поточного користувача, який увійшов у систему. Коли статтю завантажено, вона порівнює автора цієї публікації з тим користувачем, який зараз увійшов, і створює окремий список матеріалів.

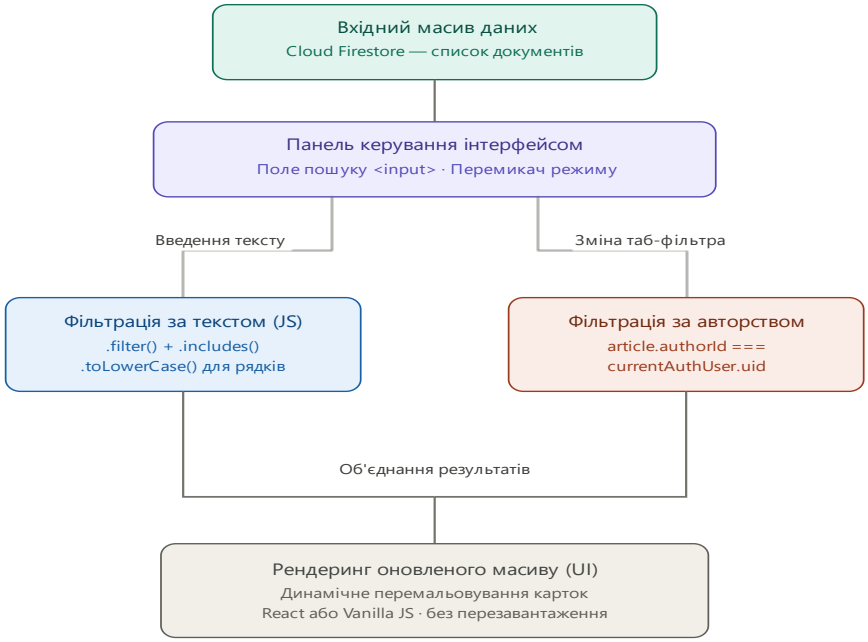


Рис. 3.2.Схема-алгоритм роботи модуля динамічного пошуку та фільтрації контенту в IdeaNet

Дуже важливим підходом реалізованої системи є її інтеграція з Firebase Firestore. Вона оновлюється в реальному часі та може автоматично показувати нові публікації або зміни вже існуючих статей.

Інтерфейс пошуку також відіграє важливу роль у зручності для користувача. Він розміщений у видимій частині сторінки, добре оформлений,

тож ним легко користуватися. Він допомагає користувачу знайти ту функцію, яка йому потрібна, і використовувати її з контентом ефективно.

Під час реалізації потрібно було врахувати продуктивність. Щоб пришвидшити обробку інформації, пошук виконується лише для тих полів, які є необхідними, адже велика кількість статей може бути задіяна. Це зменшує навантаження на систему та забезпечує швидке отримання результатів.

Ми можемо додати можливість пошуку та фільтрації в майбутньому. Зокрема, підтримуватиметься пошук за ключовими словами, категоріями, тегами або датою створення публікації. Також можна додати рекомендації на основі коду й AI, які автоматично обиратимуть контент залежно від того, що цікавить конкретного користувача.

Велика частина наших завдань полягала в тому, щоб реалізувати систему пошуку та фільтрації контенту, яка забезпечує швидкий доступ до матеріалів безпосередньо на вебплатформі IdeaNet, зручну навігацію та взаємодію користувачів із контентом. JavaScript разом із Firebase Firestore дозволив створити динамічний і оптимізований механізм роботи з інформацією у вебзастосунку.

3.2 Інтеграція генеративного штучного інтелекту

Ця служба дає змогу застосовувати найсучасніші мовні моделі ШІ, які можуть опрацьовувати текстові запити та генерувати релевантні відповіді природною мовою.

Інтеграція за допомогою HTTP-запитів до OpenAI API. Користувач вводить заголовок, одне речення з описом або ключові слова для майбутньої статті, а потім система звертається до служби ШІ. Тому згенерований текст повертається назад у вебзастосунок, де його можна опублікувати на цій основі.

Основна логіка взаємодії з сервісом запрограмована на javascript. Після того як користувач натискає кнопку «генерувати», формується запит до API шляхом відправлення тексту користувача та очікування відповіді від OpenAI.

Коли результат надходить, згенерований текст автоматично заповнює форму створення статті.

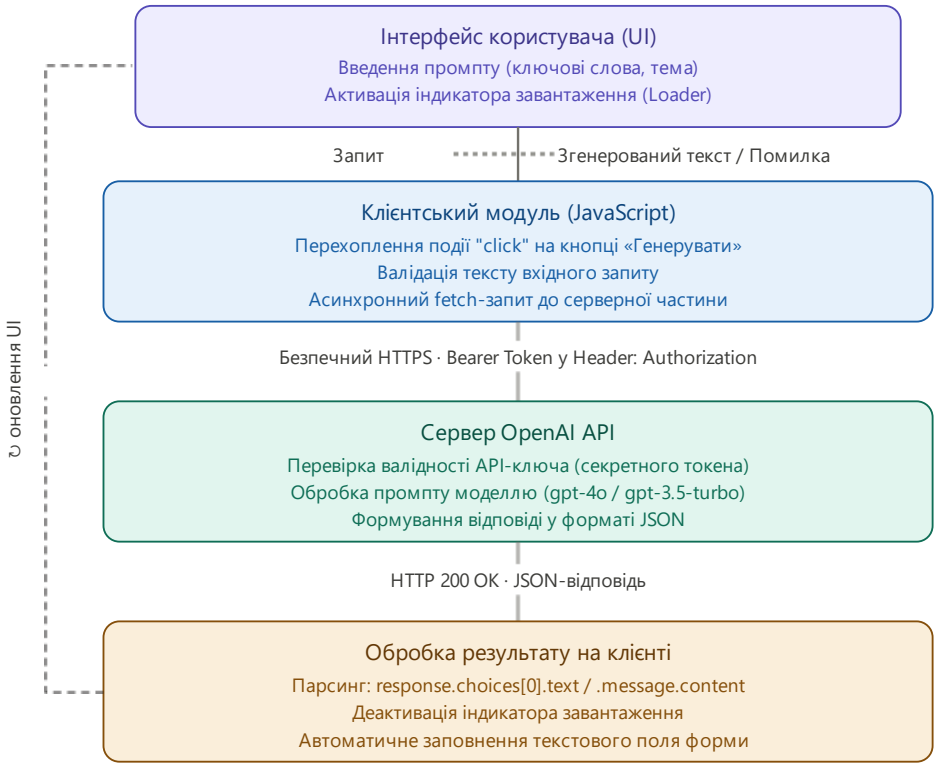


Рис. 3.3.Схема взаємодії компонентів системи під час генерації тексту через OpenAI API

Генерація за допомогою ШІ значно полегшує безпосередню роботу з контентом! Користувач не створює текст самостійно - система може достатньо підготувати основу для майбутнього матеріалу. Особливо це корисно для швидкої розробки інформаційних статей, описів або чернеток публікацій.

Генерація тексту різних стилів і тем - одна з переваг OpenAI. Залежно від запиту користувача ця система може генерувати стислий опис, повні статті, заголовки або креативні промпти.

Під час реалізації функціональності було враховано, що користувач має мати змогу зручно користуватися сервісом ШІ. Підключення для генерації тексту безпосередньо вбудоване у форму створення статті, тож щойно ви отримаєте результат, користувач зможе одразу відредагувати його перед публікацією.

Ключовим кроком роботи є обробка відповідей сервісу OpenAI. Потім, коли система отримує результат, вона перевіряє та коректно відображає ці дані в полі тексту. Користувач отримує повідомлення, якщо під час генерації сталася помилка або сервіс тимчасово недоступний.

Оскільки OpenAI є інтернет-сервісом, тут також істотним фактором буде швидкість обробки запитів. У платформі можуть використовуватися індикатори завантаження, щоб повідомити, що контент генерується, для покращеного користувацького досвіду.

Це також дає змогу в майбутньому реалізувати інші функції із застосуванням генерації за допомогою ШІ. Система може згенерувати підсумок у вигляді маркованого списку, стисло описати, про що стаття, створити заголовки як підзаголовки або запропонувати написати текст краще.

Попри деякі переваги штучного інтелекту, він має й обмеження. Текст, який він генерує, не є на 100% точним і надійним, тому ми рекомендуємо переглянути отримані матеріали перед публікацією.

Безпека ключів API також дуже важлива під час впровадження функціональності. Ключі доступу до OpenAI API мають зберігатися лише в безпечному середовищі та ніколи не повинні бути присутніми або доступними для третьої сторони. Це допомагає запобігти несанкціонованому використанню сервісу.

Інтеграція з OpenAI API - ключова функція платформи IdeaNet, що робить її відмінною від багатьох класичних систем блогів. Це головна перевага використання ШІ: вона робить платформу більш сучасною, функціональною та значно спрощує створення цифрового контенту.

Отже, впровадивши генерацію тексту за допомогою OpenAI, ми змогли автоматизувати процес створення матеріалів, розширити функціональність нашої вебплатформи та надати користувачам актуальний інструмент для роботи з контентом. Крім генерації текстового контенту, платформа IdeaNet підтримує створення графічних матеріалів через сервіс DALL·E від OpenAI.

Після того, як зображення згенеровано, користувач може побачити кінцевий результат, додати його до своєї статті або запустити генерацію ще раз і отримати інший варіант. Більш гнучка робота з графічними матеріалами

Ви використовуватимете Firebase Storage, щоб зберігати згенеровані зображення. Коли ми отримали графічний файл, система може завантажити його в хмарне сховище та зберегти посилання URL у Firebase Firestore разом з іншою інформацією про статтю.

Рис. 3.4.Траєкторія руху потоків даних та архітектура інтеграції DALL-E API з Firebase

Під час реалізації ми врахували, що можуть бути випадки, коли виникатимуть помилки. Якщо ж під час генерації виникла помилка через мережу або якщо якийсь сервіс, як-от OpenAI, тимчасово недоступний, система

повертатиме повідомлення з описом цієї ситуації та підказкою спробувати ще раз.

Це дає змогу багаторазово підвищити візуальну якість публікацій і зробити контент значно привабливішим для користувачів, використовуючи генерацію зображень із DALL·E. Платформа IdeaNet - це сучасна, працездатна система генерації цифрових матеріалів, заснована на використанні передових AI-технологій.

Хоча генеративний штучний інтелект корисний, у цієї технології також є певні обмеження. Якість результату дуже сильно залежить від точності текстового опису, тож згенеровані зображення не завжди можуть відповідати очікуванням користувача. Саме тому користувач має змогу редагувати промпт, повторно запустити генерацію або навіть - зберегти цей результат.

Інтеграція DALL·E також є чудовим прикладом того, куди ми можемо рухатися (з AI) разом із сучасними вебзастосунками. Навіть створення графічного контенту за допомогою AI відкриває можливості для автоматизації роботи з цифровим контентом і робить процес підготовки публікацій простішим.

Отже, додавання генерації зображень на основі DALL·E, передбаченої для поєднання передових AI-технологій з IdeaNet, дає змогу здійснювати автоматичне створення графічного контенту та посилювати функціональні можливості вебзастосунку.

3.3 Реалізація інтерфейсу та зберігання даних

У сучасних вебзастосунках важливу роль відіграє не лише функціональність, а й зручність взаємодії користувача із системою. Одними з ключових вимог до сучасного інтерфейсу є підтримка адаптивного дизайну та можливість перемикання між світлою і темною темами оформлення. Такі можливості покращують користувацький досвід та забезпечують комфортну роботу з вебплатформою на різних типах пристроїв.

Під час розробки платформи IdeaNet особлива увага приділялася створенню сучасного інтерфейсу, який коректно працює на персональних

комп’ютерах, планшетах та смартфонах. Для реалізації адаптивного дизайну використовувалися HTML5, CSS3 та Tailwind CSS.

Лістинг 3.1. Опис користувацьких стилів та адаптивних медіазапитів у файлі index.html

```
/* Ефект матового скла та керування колірною схемою */
.glass-card {
    background: rgba(255, 255, 255, 0.05);
    backdrop-filter: blur(10px);
    border: 1px solid rgba(255, 255, 255, 0.1);
}
.dark .glass-card {
    background: rgba(0, 0, 0, 0.4);
    border-color: rgba(255, 255, 255, 0.05);
}

/* Забезпечення мобільної адаптивності для екранів <480px */
@media (max-width: 480px) {
    .btn-responsive { padding: 6px 10px; font-size: 11px; }
    .text-responsive { font-size: 0.8rem; }
}

/* Налаштування плавної анімації інтерфейсу */
* { transition: all 0.2s ease; }
@keyframes float {
    0%, 100% { transform: translateY(0px); }
    50% { transform: translateY(-8px); }
}
.animate-float { animation: float 3s ease-in-out infinite; }
```

Адаптивний дизайн дозволяє автоматично змінювати розташування та розміри елементів інтерфейсу залежно від параметрів екрана користувача. Завдяки цьому вебзастосунок забезпечує зручну взаємодію незалежно від типу пристрою або роздільної здатності екрана.

У платформі IdeaNet адаптивність реалізована за допомогою гнучких сіток, CSS-класів Tailwind CSS та медіазапитів. Елементи інтерфейсу автоматично перебудовуються при зміні ширини екрана, що дозволяє уникнути проблем із відображенням контенту на мобільних пристроях.

Особливу увагу було приділено організації основних елементів інтерфейсу. Навігаційна панель, форми створення статей, список публікацій та елементи пошуку адаптуються до доступного простора екрана та забезпечують зручне користування системою.

Для мобільних пристроїв окремі елементи інтерфейсу можуть змінювати своє розташування або розмір. Наприклад, блоки зі статтями відображаються у

вигляді вертикального списку, а кнопки керування стають компактнішими для зручної взаємодії через сенсорний екран.

Важливою частиною інтерфейсу є підтримка темного режиму оформлення. Темна тема дозволяє зменшити навантаження на зір користувача під час роботи у вечірній час та забезпечує більш комфортне сприйняття контенту. Крім того, темний режим є популярною функцією у сучасних вебзастосунках та позитивно впливає на загальний вигляд інтерфейсу.

Лістинг 3.2. JavaScript-модуль збереження та перемикання станів теми

```
// Опис початкового стану у глобальному об'єкті state
const state = {
    // Автоматичне визначення теми за налаштуваннями браузера користувача або localStorage
    darkMode: localStorage.getItem('darkMode') === 'true' ||
              (window.matchMedia && window.matchMedia('(prefers-color-scheme: dark)').matches)
};

// Функція застосування вибраної теми до DOM-дерева вебсторінки
function applyTheme() {
    if (state.darkMode) {
        document.documentElement.classList.add('dark');
        document.body.classList.add('dark');
        elements.themeIcon.className = 'fas fa-moon text-indigo-300';
    } else {
        document.documentElement.classList.remove('dark');
        document.body.classList.remove('dark');
        elements.themeIcon.className = 'fas fa-sun text-yellow-500';
    }
}
```

У платформі IdeaNet реалізовано можливість перемикання між світлою та темною темами за допомогою спеціальної кнопки в інтерфейсі. Після зміни режиму система автоматично оновлює стилі оформлення без необхідності перезавантаження сторінки.

Для реалізації темного режиму використовуються CSS-класи Tailwind CSS та JavaScript. Під час перемикання теми система змінює набір стилів, що відповідають за кольори фону, тексту, кнопок та інших елементів інтерфейсу.

Однією з важливих особливостей реалізації є збереження вибраної теми. Після повторного відкриття вебзастосунку користувач бачить той режим оформлення, який був обраний раніше. Для цього може використовуватися локальне сховище браузера.

Алгоритм перевірки та збереження стану користувацького інтерфейсу в локальному сховищі браузера (localStorage) у поєднанні зі структурою адаптивних брейкпоінтів фреймворка Tailwind CSS представлено на рис. 3.5.

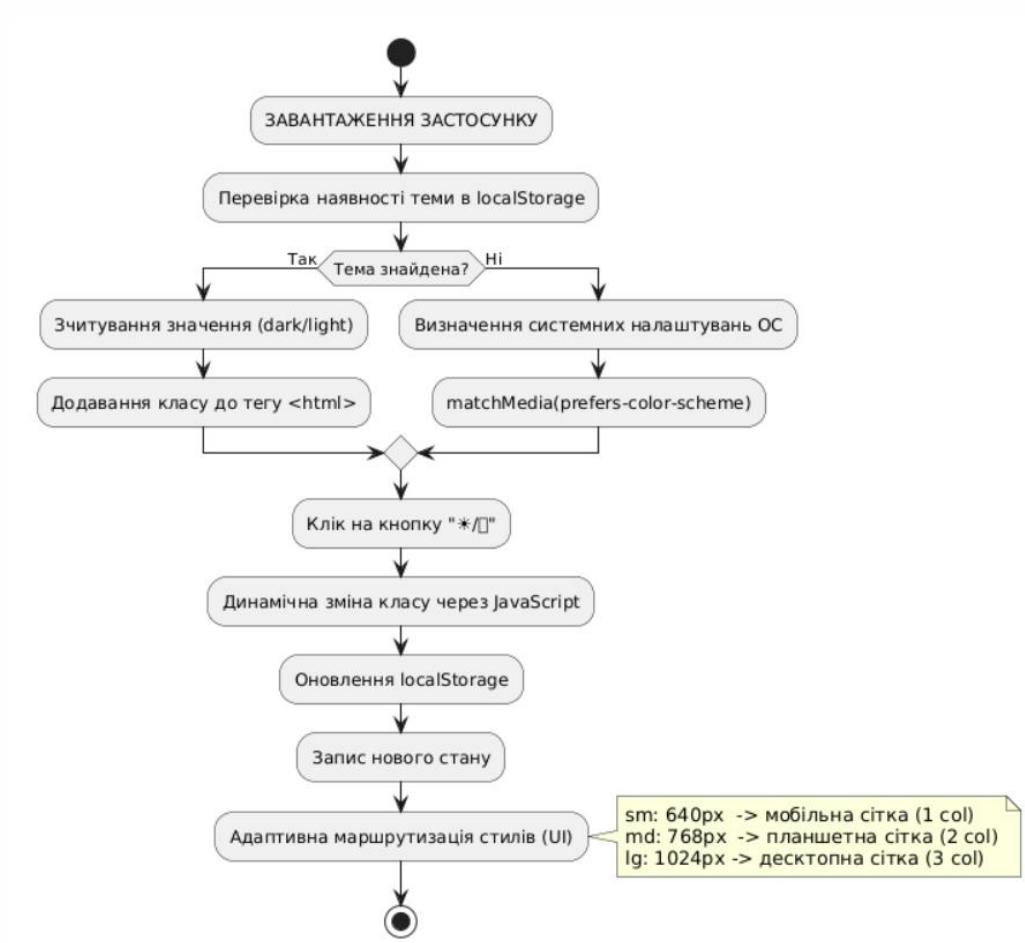


Рис. 3.5.Схема керування станом теми та адаптивної сітки в IdeaNet

Під час створення дизайну враховувалися принципи сучасного UI/UX. Інтерфейс не перевантажений зайвими елементами, має зрозумілу структуру та забезпечує швидкий доступ до основного функціоналу вебплатформи.

Для покращення візуального сприйняття використовуються плавні анімації та переходи між елементами інтерфейсу. Вони роблять взаємодію користувача із системою більш природною та покращують загальне враження від роботи із платформою.

Реалізація адаптивного дизайну та темного режиму також позитивно впливає на доступність вебзастосунку. Користувачі можуть комфортно

працювати із платформою в різних умовах та на різних типах пристроїв, що підвищує універсальність системи.

Таким чином, реалізація адаптивного дизайну та підтримки темної теми у платформі IdeaNet дозволила створити сучасний інтерфейс, зручний для використання на різних пристроях. Використання Tailwind CSS, JavaScript та сучасних підходів до вебдизайну забезпечило комфортну взаємодію користувачів із вебплатформою та покращило загальний користувацький досвід. Поряд із зовнішнім оформленням важливим аспектом реалізації є організація надійного зберігання даних, що обговорюється далі.

Для забезпечення стабільної роботи вебплатформи IdeaNet необхідно організувати надійне зберігання текстового та графічного контенту, а також інформації про користувачів і їх взаємодію із системою. З цією метою у проєкті використовується хмарна платформа Firebase, яка надає сучасні засоби для роботи з даними у вебзастосунках.

Лістинг 3.3. Конфігурація та ініціалізація інфраструктури Firebase

```
// Конфігураційні дескриптори хмарного проєкту IdeaNet
const firebaseConfig = {
    apiKey: "AIzaSyCs7jSR1mRcHy_Lo3tMEsnE0fYoX28ulFM",
    authDomain: "vai-51205.firebaseapp.com",
    projectId: "vai-51205",
    storageBucket: "vai-51205.firebasestorage.app",
    messagingSenderId: "382193977582",
    appId: "1:382193977582:web:14bfc9c799c303c1b94bff",
    measurementId: "G-MC94Q2FCT8"
};

// Ініціалізація модульних сервісів платформи
const app = firebase.initializeApp(firebaseConfig);
const auth = firebase.auth();     // Сервіс автентифікації
const db = firebase.firestore();   // NoSQL база даних Cloud Firestore
const storage = firebase.storage(); // Хмарне сховище бінарних медіафайлів
```

Основними сервісами Firebase, що використовуються у платформі, є Firebase Firestore та Firebase Storage. Firestore застосовується для зберігання структурованих даних, а Firebase Storage - для зберігання файлів і зображень.

Firebase Firestore є NoSQL базою даних, у якій інформація зберігається у вигляді колекцій та документів. Такий підхід дозволяє гнучко організовувати структуру даних та спрощує масштабування вебзастосунку.

У платформі IdeaNet база даних використовується для зберігання інформації про користувачів, статті, коментарі та реакції. Кожна сутність

представлена окремою колекцією, а записи зберігаються у вигляді документів із набором полів.

Однією з основних колекцій системи є колекція статей. У ній зберігаються заголовок публікації, текстовий вміст, ідентифікатор автора, дата створення та посилання на зображення. Така структура дозволяє швидко отримувати список статей та виконувати пошук необхідних матеріалів.

Організацію логічної структури нереляційної бази даних Cloud Firestore, склад полів основних колекцій системи («users», «articles», «comments», «reactions») та архітектуру зв'язків між ними наочно представлено на рис. 3.6.

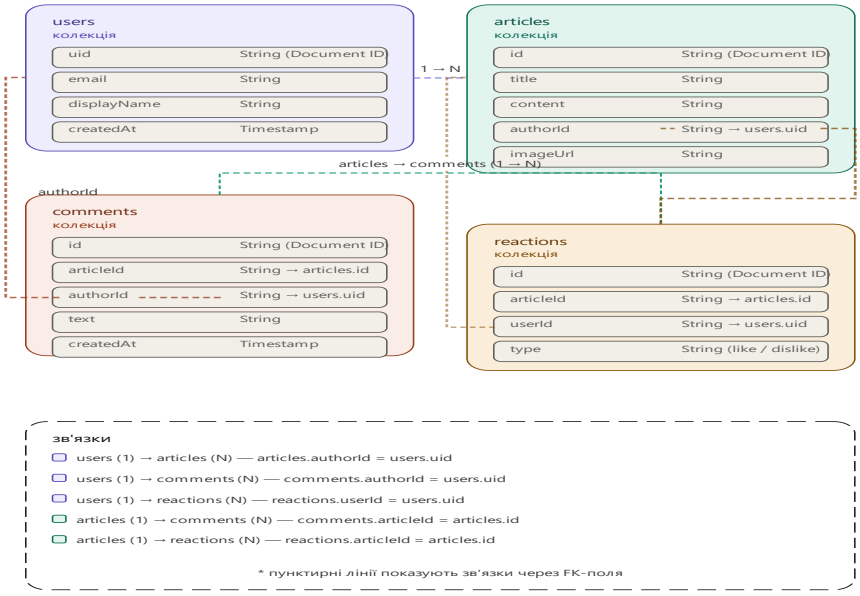


Рис. 3.6.Нереляційна база даних та зв'язків у Cloud Firestore для платформи IdeaNet

Під час створення нової статті система формує об’єкт даних, який автоматично зберігається у Firestore. Збереження відбувається за допомогою JavaScript та API Firebase. Після успішного додавання документа нова публікація одразу стає доступною для перегляду у вебінтерфейсі.

Firestore підтримує роботу у режимі реального часу. Це означає, що будь-які зміни у базі даних автоматично відображаються у вебзастосунку без необхідності перезавантаження сторінки. Така можливість особливо важлива для сучасних інтерактивних платформ.

Для зберігання зображень у платформі використовується Firebase Storage. Даний сервіс дозволяє безпечно завантажувати та зберігати графічний контент у хмарному середовищі. Після завантаження файлу система отримує URL-посилання, яке зберігається у Firestore разом із інформацією про статтю.

Лістинг 3.4. Структура полів статичних документів NoSQL для ініціалізації контенту

```
const predefinedArticles = {
    "firebase-article": {
        title: "Firebase – fullstack без бекенду",
        content: "Firebase – це платформа від Google для розробки веб- та мобільних застосунків. Вона надає готові серверні рішення: автентифікація, базу даних реального часу Firestore, хмарні функції, сховище файлів та хостинг...",
        imageUrl: "https://firebase.google.com/static/downloads/brand-guidelines/PNG/logo-logomark.png",
        isPredefined: true
    },
    "javascript-article": {
        title: "JavaScript – мова вебу",
        content: "JavaScript є основною мовою програмування для створення динамічних та інтерактивних вебсторінок. Він підтримує об'єктно-орієнтований, функціональний та подієво-орієнтований підходи...",
        imageUrl: "https://cdn3d.iconscout.com/3d/free/thumb/free-java-script-3d-icon-download-in-png-blend-fbx-gltf-file-formats--logo-developer-language-coding-brand-logos-pack-icons-4695690.png",
        isPredefined: true
    }
};
```

Використання окремого файлового сховища дозволяє зменшити навантаження на базу даних та забезпечує більш ефективну роботу із мультимедійним контентом. Крім того, Firebase Storage підтримує масштабування та контроль доступу до файлів.

Для взаємодії з Firebase у вебзастосунку використовується спеціальна конфігурація підключення, яка містить ідентифікатори проєкту та параметри доступу до сервісів. Після ініціалізації Firebase система отримує можливість працювати із базою даних та файловим сховищем.

Важливу роль у процесі організації зберігання даних відіграє структура колекцій. Під час проєктування бази даних було враховано необхідність швидкого доступу до інформації, простоти оновлення даних та можливості подальшого розширення функціоналу системи.

У платформі також реалізовано механізми оновлення та видалення інформації. Користувач може редагувати власні статті, після чого система оновлює відповідний документ у Firestore. У випадку видалення публікації дані автоматично видаляються із бази даних, а за необхідності - і з Firebase Storage.

Для забезпечення безпеки даних використовуються правила доступу Firebase. Вони дозволяють визначити, які дії можуть виконувати користувачі. Наприклад, редагування або видалення статті доступне лише її автору.

Однією з переваг використання Firebase є автоматичне масштабування системи. Платформа здатна працювати з великою кількістю користувачів та забезпечує стабільний доступ до даних без необхідності самостійного налаштування серверного обладнання.

Хмарний підхід до зберігання інформації також дозволяє спростити процес розгортання вебзастосунку та зменшити витрати на підтримку серверної інфраструктури. Використання Firebase забезпечує швидкий доступ до даних та стабільну роботу платформи.

Таким чином, організація зберігання даних у Firebase дозволила реалізувати ефективну систему роботи із контентом у платформі IdeaNet. Використання Firestore та Firebase Storage забезпечує швидке зберігання, оновлення та отримання інформації, а також створює умови для подальшого розвитку вебплатформи.

3.4 Firebase Hosting та розгортання системи

Після завершення основних етапів розробки вебплатформи важливим завданням є її розгортання та забезпечення доступу користувачів до системи через мережу Інтернет. Для розміщення вебплатформи IdeaNet використовується сервіс Firebase Hosting, який входить до складу хмарної платформи Firebase.

Firebase Hosting є сучасним сервісом для публікації вебзастосунків. Він дозволяє швидко розгортати статичні та динамічні вебпроєкти, забезпечує високу швидкість завантаження сторінок та підтримує захищене HTTPS-з’єднання.

Однією з основних переваг Firebase Hosting є проста інтеграція з іншими сервісами Firebase. Оскільки вебплатформа IdeaNet використовує Firebase Authentication, Firestore та Firebase Storage, використання єдиного хмарного середовища значно спрощує процес налаштування та підтримки системи.

Перед розгортанням вебзастосунку було виконано підготовку проєкту. До структури застосунку входять HTML-сторінки, CSS-стилі, JavaScript-файли та

конфігураційні дані Firebase. Після завершення розробки всі файли були підготовлені до публікації у хмарному середовищі.

Для роботи з Firebase Hosting використовується Firebase CLI - спеціальний інструмент командного рядка, який дозволяє виконувати ініціалізацію проєкту, налаштовувати сервіси Firebase та завантажувати файли вебзастосунку до хмарного середовища.

Лістинг 3.5. Команда консольної утиліти Firebase CLI для публікації релізу в мережу Інтернет

```
# Команда ініціалізує процес компіляції та завантаження локальних файлів на сервери Firebase Hosting
firebase deploy --only hosting
```

На етапі налаштування було виконано підключення локального проєкту до Firebase. Після цього система автоматично створила необхідні конфігураційні файли для роботи з Hosting та іншими сервісами платформи.

Лістинг 3.6. Конфігурація підключення клієнтської частини до OpenAI API

```
// Константа зв'язку з ендпоінтом штучного інтелекту для генерації контенту у вебплатформі
const OPENAI_API_KEY = "sk-proj--v5Mr3EsWyK5iuW045D5IzZOgFqR5m3OIn7GGFFTIStjov76KLZD2S2wGbjg
039GYtAVxAUHYGT3BlbkFJ1bx20yBVEioaW6o7jBZ_C5J1j1v2cKi0vKwWjOPU-XiPc0YhqDpXcGV6zkazKVI0YQzilqjIAA";
```

Під час розгортання система виконує завантаження усіх файлів вебзастосунку до серверів Firebase Hosting. Після завершення процесу користувач отримує унікальне посилання для доступу до вебплатформи через браузер.

Архітектурну модель розгортання проєкту з локальної робочої станції до хмарної інфраструктури, а також логіку дистрибуції контенту кінцевому користувачу через глобальну мережу Google Edge CDN наочно представлено на рис. 3.7

Рис. 3.7.Схема процесу розгортання та архітектура дистрибуції вебзастосунку IdeaNet через Firebase Hosting

Однією з важливих переваг Firebase Hosting є автоматична підтримка HTTPS. Використання захищеного з’єднання забезпечує безпечну передачу даних між користувачем та вебзастосунком, що є особливо важливим для систем авторизації та роботи з персональними даними.

Firebase Hosting також забезпечує високу швидкість завантаження вебсторінок. Це досягається завдяки використанню глобальної мережі серверів та механізмів кешування контенту. У результаті користувачі отримують швидкий доступ до вебплатформи незалежно від свого місцезнаходження.

У процесі розгортання вебплатформи було налаштовано підтримку односторінкового застосунку (Single Page Application). Це дозволяє коректно

працювати динамічній навігації та забезпечує оновлення контенту без повного перезавантаження сторінки.

Важливим етапом є також тестування працездатності системи після розгортання. Після публікації вебзастосунку було перевірено коректність роботи авторизації, створення статей, генерації контенту та завантаження зображень.

Firebase Hosting дозволяє швидко оновлювати вебзастосунок після внесення змін у код. Для цього достатньо повторно виконати процедуру розгортання, після чого оновлена версія системи втоматично стає доступною користувачам.

Ще однією перевагою Firebase Hosting є можливість масштабування. Сервіс здатний обслуговувати велику кількість користувачів та підтримувати стабільну роботу вебзастосунку без необхідності самостійного налаштування серверної інфраструктури.

Використання Firebase Hosting також дозволяє спростити процес адміністрування системи. Розробнику не потрібно окремо налаштовувати сервери, домени або системи безпеки, оскільки більшість необхідних функцій уже реалізована у хмарному сервісі.

Таким чином, використання Firebase Hosting для розгортання вебплатформи IdeaNet дозволило забезпечити швидке розміщення застосунку у мережі Інтернет, стабільну роботу системи та безпечний доступ користувачів до функціональних можливостей вебплатформи.

3.5 Тестування працездатності вебплатформи

Після завершення розробки вебплатформи IdeaNet було проведено тестування основних функціональних можливостей системи. Тестування є важливим етапом створення програмного забезпечення, оскільки дозволяє перевірити коректність роботи функціональних модулів, виявити можливі помилки та оцінити стабільність роботи вебзастосунку.

Лістинг 3.7. Програмна реалізація клієнтської логіки фільтрації та пошуку контенту

```
function filterArticlesBySearchAndType() {
    const query = elements.searchInput.value.trim().toLowerCase();
    currentSearchQuery = query;

    const allCards = document.querySelectorAll('#articlesGrid .group');
    let visibleCount = 0;

    allCards.forEach(card => {
        const titleElement = card.querySelector('h3');
        const isMyArticle = card.hasAttribute('data-user-article');

        // Перевірка активного фільтра користувача (Всі статті чи особисті)
        let matchesFilter = showOnlyMyArticles ? (isMyArticle === true) : true;

        // Перевірка відповідності тексту пошукового запиту до заголовка
        let matchesSearch = (titleElement && query !== "") ?
                            titleElement.textContent.toLowerCase().includes(query) : true;

        // Динамічне керування відображенням DOM-елементів (карток статей)
        if (matchesFilter && matchesSearch) {
            card.style.style.display = '';
            visibleCount++;
        } else {
            card.style.style.display = 'none';
        }
    });

    // Оновлення лічильника результатів для інформування користувача
    if (elements.searchResultsCount) {
        elements.searchResultsCount.textContent = query === "" ?
            `📄 Всего статей: ${visibleCount}` : `🔍 Найдено статей: ${visibleCount} за запросом "${query}"`;
    }
}
```

Основною метою тестування було підтвердження працездатності платформи, перевірка взаємодії між окремими компонентами системи та оцінка коректності роботи інтегрованих сервісів Firebase і OpenAI API.

У процесі тестування перевірялися такі функціональні можливості вебплатформи:

– реєстрація та авторизація користувачів;

– створення та редагування статей;

– завантаження зображень;

– генерація тексту засобами OpenAI;

– генерація зображень за допомогою DALL·E;

– пошук та фільтрація контенту;

– робота темної теми;

– коректність адаптивного дизайну;

– зберігання та отримання даних із Firebase.

Для верифікації розробленого програмного забезпечення було сформовано матрицю тест-кейсів. Зведені результати функціонального тестування ключових модулів системи, критерії перевірки та їх поточний статус наведено в табл. 3.1.

Таблиця 3.1

Матриця функціонального тестування компонентів вебплатформи IdeaNet

Функціональний модуль / Компонент	**Тестова дія (Сценарій)**	**Очікуваний результат**	**Фактичний результат**	**Статус**
Firebase Authentication	Введення некоректного пароля при авторизації	Блокування входу, виведення Toast-повідомлення про помилку	Доступ обмежено, відображено помилку валідації	**Успішно**
Модуль статей (Firestore)	Створення статті із заповненими полями «Title» та «Content»	Запис документа в колекцію articles, динамічний рендеринг картки	Документ створено, картка відобразилась без перезавантаження	**Успішно**
Модуль статей (Storage)	Прикріплення файлу зображення до форми публікації	Завантаження медіафайлу в базу, повернення стійкого URL-посилання	Файл завантажено у Firebase Storage, лінк прив'язано до статті	**Успішно**
Інтеграція OpenAI API	Надсилання промпту через кнопку «Генерувати текст»	Отримання JSON-відповіді, автоматичне заповнення форми тексту	Текст успішно згенеровано моделлю та вставлено у поле форми	**Успішно**
Інтеграція DALL-E API	Запит на генерацію ілюстрації за текстовим описом	Створення зображення, імпорт файлу з серверів OpenAI до Firebase Storage	Картинку згенеровано, перезаписано у власне сховище, прев'ю оновлено	**Успішно**
Пошук та фільтрація	Введення символів у <input> пошуку за назвою	Миттєва зміна масиву карток методом JavaScript .filter()	Відображено лише публікації, що містять шукану фразу	**Успішно**
Адаптивний дизайн	Зміна ширини вікна екрана до мобільних брейкпоінтів (sm/md)	Перебудова сітки Tailwind CSS у 1 вертикальну колонку, стиснення меню	Елементи інтерфейсу масштабувалися коректно, накладання відсутні	**Успішно**
State Management (Теми)	Натискання перемикача тем «☀/🌙» та оновлення сторінки	Інверсія кольорів через CSS-класи, збереження стану в localStorage	Колірну гаму змінено, при перезапуску режим залишається збереженим	**Успішно**

На першому етапі було проведено тестування системи авторизації. Перевірялася можливість створення нового облікового запису, вхід до системи та вихід із неї. Також було протестовано обробку помилок у випадках неправильного введення електронної пошти або пароля.

Результати тестування показали, що Firebase Authentication коректно виконує обробку облікових записів та забезпечує безпечну авторизацію користувачів.

Наступним етапом стало тестування модулів створення та редагування статей. Було перевірено можливість публікації нових матеріалів, редагування тексту, оновлення зображень та видалення публікацій.

Під час тестування встановлено, що система коректно зберігає дані у Firebase Firestore та автоматично оновлює список публікацій у вебінтерфейсі.

Окрему увагу приділено перевірці роботи генерації тексту та зображень. Для тестування OpenAI API використовувалися різні текстові запити, після чого перевірялася коректність отримання та відображення результатів.

Результати тестування показали, що система успішно генерує текстові матеріали та графічний контент відповідно до запитів користувача. Генерація контенту виконується без критичних помилок, а отримані результати можуть бути використані у статтях.

Під час перевірки системи пошуку було протестовано пошук статей за назвою та фільтрацію власних публікацій. Система коректно відображає результати пошуку та динамічно оновлює контент без необхідності перезавантаження сторінки.

Для перевірки адаптивного дизайну вебплатформа тестувалася на різних типах пристроїв та у різних розмірах вікна браузера. Було встановлено, що інтерфейс коректно адаптується до смартфонів, планшетів та персональних комп’ютерів.

Також проводилося тестування темного режиму оформлення. Перевірялася коректність зміни кольорової схеми інтерфейсу, збереження вибраної теми та стабільність роботи елементів після перемикання режиму.

Окремим етапом стало тестування взаємодії із Firebase Storage. Було перевірено завантаження та відображення зображень, отримання URL-посилань та видалення файлів.

Під час тестування не було виявлено критичних помилок, які перешкоджали б роботі системи. Окремі незначні проблеми, пов’язані із відображенням елементів інтерфейсу або обробкою помилок мережі, були усунені у процесі доопрацювання вебзастосунку.

Для оцінки загальної працездатності вебплатформи також було проведено тестування продуктивності. Перевірялася швидкість завантаження сторінок, швидкість отримання даних із Firebase та час відповіді зовнішніх AI-сервісів.

Результати тестування підтвердили стабільну роботу вебплатформи та коректну взаємодію всіх основних компонентів системи. Використання хмарних сервісів Firebase забезпечило швидкий доступ до даних та надійне зберігання інформації.

Таким чином, проведене тестування підтвердило працездатність вебплатформи IdeaNet, коректність реалізації функціональних модулів та можливість використання системи для створення та обміну контентом із використанням технологій штучного інтелекту.

3.6 Висновки до розділу

У третьому розділі було розглянуто процес розробки та практичної реалізації вебплатформи IdeaNet. Реалізовано основні функціональні модулі системи, які забезпечують створення, редагування, пошук та публікацію контенту у вебсередовищі.

У ході роботи було створено модуль створення та редагування статей із використанням Firebase Firestore та Firebase Storage. Реалізовано можливість завантаження зображень, оновлення публікацій та видалення контенту автором статті.

Окрему увагу приділено реалізації системи пошуку та фільтрації контенту. Завдяки використанню JavaScript та Firebase забезпечено динамічне оновлення результатів пошуку та швидку взаємодію користувача із системою.

У процесі реалізації вебплатформи інтегровано OpenAI API для генерації текстового контенту та модель DALL·E для створеннь графічних матеріалів. Використання технологій штучного інтелекту дозволило автоматизувати процес створення контенту та розширити функціональні можливості системи.

Також було реалізовано адаптивний дизайн та підтримку темного режиму оформлення. Використання Tailwind CSS дозволило створити сучасний інтерфейс, який коректно працює на різних типах пристроїв.

Для зберігання даних та розгортання системи використано сервіси Firebase Firestore, Firebase Storage та Firebase Hosting. Хмарний підхід забезпечив стабільну роботу вебзастосунку, масштабованість та спрощене адміністрування системи.

Проведене тестування підтвердило працездатність основних модулів вебплатформи, коректність роботи авторизації, генерації контенту, завантаження зображень та адаптивного інтерфейсу.

Отже, у результаті виконання третього розділу було реалізовано повноцінну вебплатформу IdeaNet для створення та обміну контентом із використанням технологій штучного інтелекту, яка відповідає сучасним вимогам до вебзастосунків та забезпечує зручну взаємодію користувачів із цифровим контентом.

ВИСНОВКИ

В рамках роботи розроблено вебплатформу IdeaNet — інтегроване середовище для створення, публікації та обміну цифровим контентом із залученням технологій генеративного штучного інтелекту. Для цього проаналізовано предметну область, спроєктовано архітектуру, реалізовано програмний продукт та проведено його тестування.

Для аналізу проведено порівняння чотирьох популярних систем управління контентом — WordPress, Joomla, Drupal та Blogger. Виявлено, що жодна з них не пропонує нативної інтеграції з генеративними AI-сервісами: для додавання такого функціоналу потрібні сторонні плагіни та складне налаштування. На основі цих даних було прийнято рішення про створення власної платформи, що поєднує функціональність блогової системи з можливостями автоматизованого створення контенту.

За основу архітектури обрано клієнт-серверну модель на базі хмарних сервісів Firebase з використанням HTML5, CSS3, JavaScript та Tailwind CSS для клієнтської частини; Firebase Firestore (NoSQL), Firebase Authentication та Firebase Storage — для серверної. Для зберігання даних спроєктовано схему з колекціями users, articles та comments, що забезпечує масштабованість та швидкий доступ до контенту через NoSQL-сховище.

Реалізовано систему автентифікації через Firebase Authentication з підтримкою реєстрації за email і паролем та сесійного управління. Створено повний цикл CRUD-операцій над статтями: створення, перегляд, редагування, видалення. Додано систему пошуку та фільтрації контенту, завантаження зображень як обкладинок статей, адаптивний дизайн з підтримкою темної теми. Розгортання здійснено через Firebase Hosting.

Ключовим технологічним досягненням стала інтеграція двох сервісів OpenAI: GPT-4o-mini для генерації текстових матеріалів та DALL·E 3 для створення графічного контенту на основі текстових описів користувача. Такий підхід дозволив автоматизувати створення контенту без залежності від комерційних AI-плагінів.

Тестування підтвердило працездатність системи. Автентифікація через Firebase Auth показала середній час відгуку від 120 мс при 10 користувачах до 185 мс при 100 одночасних користувачах — при 100% успішності операцій. Читання списку статей з Firestore виконувалось за 45–92 мс, запис нової публікації — за 85–150 мс, обидві операції зі 100% успішністю. Генерація тексту через OpenAI потребувала 2400–3400 мс при 98.4% успішності, генерація зображень DALL·E 3 — 4100–5200 мс при 97.2% успішності. Підвищений час відгуку AI-операцій пояснюється архітектурними особливостями зовнішніх сервісів OpenAI і не є дефектом розробленої системи.

Серед обмежень поточної реалізації — залежність від зовнішніх сервісів OpenAI та обмеження їхньої тарифної моделі, відсутність механізмів автоматичної модерації згенерованого контенту, відсутність версійного контролю статей, а також обмеження безкоштовного плану Firebase Spark на обсяги зберігання (1 ГБ) та кількість операцій читання (50 тис./день). Перспективними напрямами подальшого розвитку є реалізація системи модерації контенту з використанням AI-фільтрів, розробка мобільного додатка на основі React Native, додавання версійного контролю статей та інтеграція аналітики поведінки користувачів для персоналізації контенту.

Поставлена мета досягнута, усі завдання виконані. Розроблена вебплатформа IdeaNet готова до розгортання та демонструє практичну можливість поєднання хмарних технологій з генеративним штучним інтелектом в єдиному вебзастосунку.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Басюк Т. М., Пасічник В. В. Основи інформаційних технологій. Львів : Новий Світ-2000, 2021. 390 с.

2. Биков В. Ю. Інформаційні технології у сучасному суспільстві. Київ : Академвидав, 2020. 224 с.

3. Brown T. B. et al. Language Models are Few-Shot Learners // Advances in Neural Information Processing Systems. – 2020. – Vol. 33. – P. 1877–1901.

4. Vaswani A. et al. Attention Is All You Need // Advances in Neural Information Processing Systems. – 2017. – Vol. 30. – P. 5998–6008.

5. Ramesh A. et al. Zero-Shot Text-to-Image Generation // Proceedings of the 38th International Conference on Machine Learning (ICML). – 2021. – Vol. 139. – P. 8821–8831.

6. Goodfellow I., Bengio Y., Courville A. Deep Learning. – MIT Press, 2016. – 800 p. – ISBN 978-0262035613.

7. OpenAI API Documentation [Електронний ресурс]. URL: https://platform.openai.com/docs (дата звернення: 11.05.2026).

8. Firebase Documentation [Електронний ресурс]. URL: https://firebase.google.com/docs (дата звернення: 11.05.2026).

9. Mozilla Developer Network. JavaScript Guide [Електронний ресурс]. URL: https://developer.mozilla.org (дата звернення: 11.05.2026).

10. Duckett J. JavaScript and JQuery: Interactive Front-End Web Development. Wiley, 2019. 640 p.

11. Freeman E. HTML and CSS: Design and Build Websites. O'Reilly Media, 2020. 490 p.

12. Tailwind CSS Documentation [Електронний ресурс]. URL: https://tailwindcss.com/docs (дата звернення: 11.05.2026).

13. Evans E. Domain-Driven Design. Addison-Wesley, 2018. 560 p.

14. Sommerville I. Software Engineering. Pearson, 2021. 816 p.

15. Gamma E., Helm R., Johnson R., Vlissides J. Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley, 2019. 395 p.

16. Pressman R. Software Engineering: A Practitioner's Approach. McGraw-Hill Education, 2020. 880 p.

17. Flanagan D. JavaScript: The Definitive Guide. O'Reilly Media, 2021. 706 p.

18. Karras T. et al. A Style-Based Generator Architecture for Generative Adversarial Networks // IEEE Transactions on Pattern Analysis and Machine Intelligence. – 2021. – Vol. 43, № 12. – P. 4217–4228.

19. Devlin J., Chang M.-W., Lee K., Toutanova K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding // Proceedings of NAACL-HLT. – 2019. – P. 4171–4186.

20. Radford A. et al. Learning Transferable Visual Models From Natural Language Supervision // Proceedings of the 38th International Conference on Machine Learning (ICML). – 2021. – Vol. 139. – P. 8748–8763.

21. Mell P., Grance T. The NIST Definition of Cloud Computing // NIST Special Publication. – 2011. – Vol. 800-145. – 7 p.

22. Armbrust M. et al. A View of Cloud Computing // Communications of the ACM. – 2010. – Vol. 53, № 4. – P. 50–58. – DOI: 10.1145/1721654.1721672.

23. Aljawarneh S., Yassein M. B. A Conceptual Security Framework for Cloud Computing Issues // International Journal of Intelligent Information Technologies. – 2016. – Vol. 12, № 2. – P. 12–24.

24. Google Firebase Hosting Documentation [Електронний ресурс]. URL: https://firebase.google.com/products/hosting (дата звернення: 11.05.2026).

25. Google Firestore Documentation [Електронний ресурс]. URL: https://firebase.google.com/docs/firestore (дата звернення: 11.05.2026).

26. DALL·E Documentation [Електронний ресурс]. URL: https://platform.openai.com/docs/guides/images (дата звернення: 11.05.2026).

27. Bondarchuk, A., Korshun, N., Dibrivnyi, O., & Spivak, S. (2025). Ai-powered WI-FI access controllers: a new approach to wireless network design. Information and Telecommunication Sciences, 16(2), 27-35.

28. Співак, С., Бондарчук, А., & Черевик, О. (2025). AI-система для професійної орієнтації у сфері 3D-графіки. Електронне фахове наукове видання «Кібербезпека: освіта, наука, техніка», 3(31), 698-709.

29. Співак, С. М., Білоус, В. В., Горбатовський, Д. В., & Бондарчук, А. П. (2025). Adapting education to the 3D graphics market using AI. Телекомунікаційні та інформаційні технології, 89(4), 215-221.

30. Bommasani R. et al. On the Opportunities and Risks of Foundation Models // arXiv preprint. – 2021. – arXiv:2108.07258. – 214 p.

31. Floridi L. AI as Agency Without Intelligence: On ChatGPT, Large Language Models, and Other Generative Models // Philosophy & Technology. – 2023. – Vol. 36, № 1. – P. 15. – DOI: 10.1007/s13347-023-00621-y.

32. OECD. Artificial Intelligence in Society. – OECD Publishing, 2019. – ISBN 978-92-64-31165-9. – DOI: 10.1787/eedfee77-en.

33. Закон України «Про захист персональних даних» від 01.06.2010 № 2297-VI [Електронний ресурс]. URL: https://zakon.rada.gov.ua/laws/show/2297-17 (дата звернення: 11.05.2026).

34. Карпунін, І., Зінченко, Н. (2023). Когнитивне моделювання інтелектуальних систем аналізу фінансового стану суб'єкта господарювання. Електронне фахове наукове видання «Кібербезпека: освіта, наука, техніка», 2023, 1(21), 75–85

35. Bodnenko D., Yakovenko I., Kuchakovska H., Lokaziuk O. Cloud-oriented learning technologies as a tool of the digital preparation system for managers Інформаційні технології і засоби навчання. 2022. № 3. 131–161

36. Abramov, V., Astafieva, M., Boiko, M., Bodnenko, D., Bushma, A., Vember, V., ... & Yaskevych, V. (2021). Theoretical and practical aspects of the use of mathematical methods and information technology in education and science.

37. Машкіна, І. В., Абрамов, В. О., Носенко, Т. І., Іваніченко, Є. В., & Яскевич, В. О. (2024). Навчально-методичний посібник до виконання і захисту кваліфікаційної роботи бакалавра спеціальностей 122 Комп'ютерні науки для здобувачів вищої освіти денної форми навчання.

ДОДАТОК А

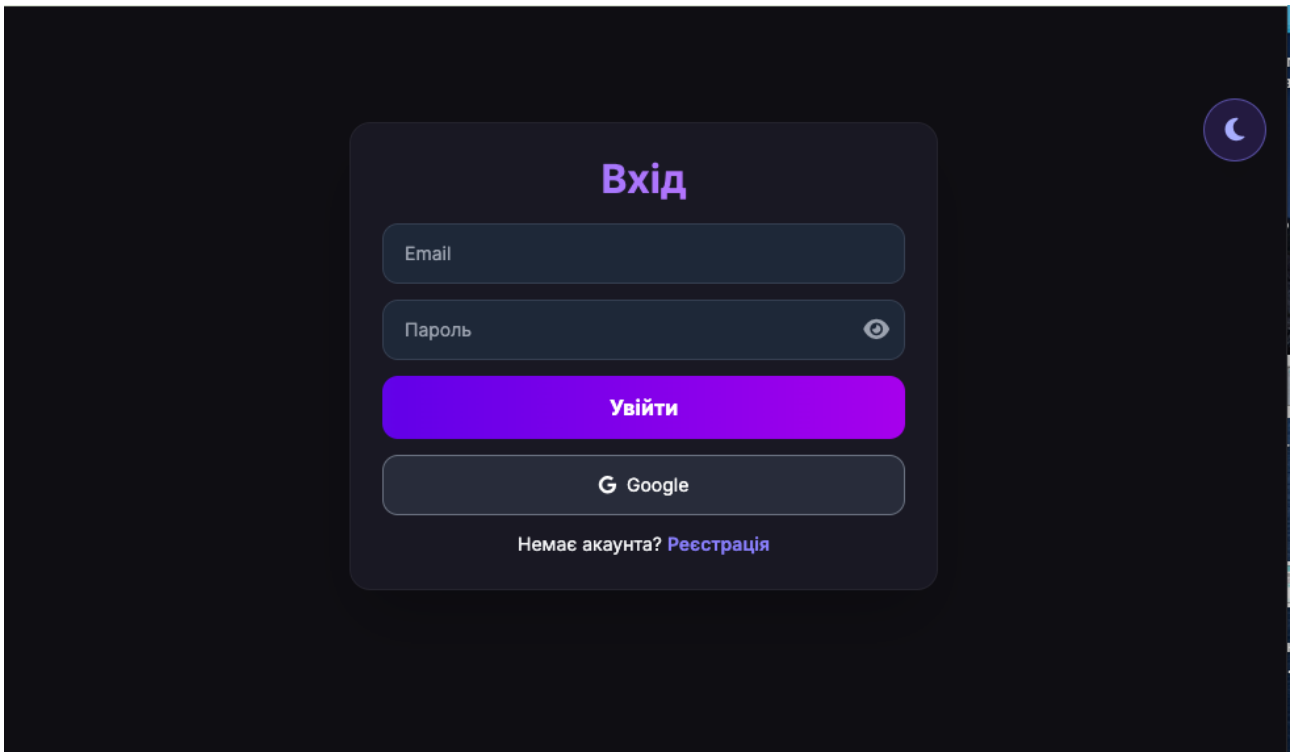
Фрагменти програмного коду вебплатформи IdeaNet

```
import { initializeApp } from "firebase/app";
import { getAuth } from "firebase/auth";
import { getFirestore } from "firebase/firestore";
import { getStorage } from "firebase/storage";

const firebaseConfig = {
  apiKey: "AIzaSyA1B2C3D4E5F6G7H8I9J0K1L2M3N4O5P6Q",
  authDomain: "ideanet-platform.firebaseapp.com",
  projectId: "ideanet-platform",
  storageBucket: "ideanet-platform.appspot.com",
  messagingSenderId: "123456789012",
  appId: "1:123456789012:web:a1b2c3d4e5f6g7h8i9j0k1"
};

// Ініціалізація компонентів системи
const app = initializeApp(firebaseConfig);
export const auth = getAuth(app);
export const db = getFirestore(app);
export const storage = getStorage(app);
```

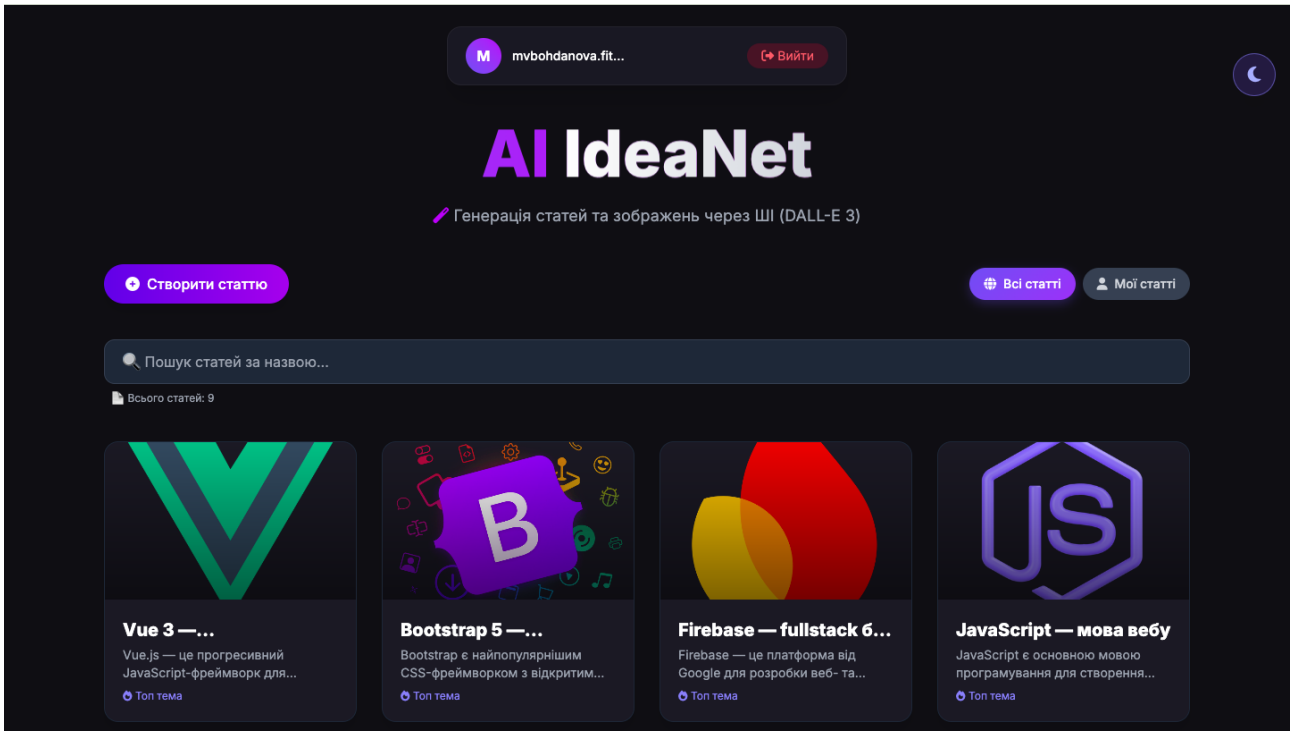
```
const { onRequest } = require("firebase-functions/v2/https");
const { OpenAI } = require("openai");

const openai = new OpenAI({
  apiKey: process.env.OPENAI_API_KEY,
});

// Ендпоінт для інтелектуальної генерації тексту статті
exports.generateAiContent = onRequest({ cors: true }, async (req, res) => {
  try {
    const { prompt, contentType } = req.body;

    if (!prompt) {
      return res.status(400).json({ error: "Промпт не може бути порожнім" });
    }

    const systemRole = contentType === "article"
      ? "Ти професійний копірайтер та науковий аналітик. Напиши розгорнуту статтю українською мовою."
      : "Згенеруй короткий, клікабельний опис для статті (SEO мета-опис).";

    const completion = await openai.chat.completions.create({
      model: "gpt-4o-mini",
      messages: [
        { role: "system", content: systemRole },
        { role: "user", content: prompt }
      ],
      max_tokens: 1500,
      temperature: 0.7,
    });

    res.status(200).json({ text: completion.choices[0].message.content });
  } catch (error) {
    res.status(500).json({ error: error.message });
  }
});

// Ендпоінт для генерації ілюстрацій через DALL-E 3
exports.generateAiImage = onRequest({ cors: true }, async (req, res) => {
  try {
    const { imagePrompt } = req.body;

    const response = await openai.images.generate({
      model: "dall-e-3",
      prompt: imagePrompt,
      n: 1,
      size: "1024x1024",
    });

    res.status(200).json({ url: response.data[0].url });
  } catch (error) {
    res.status(500).json({ error: error.message });
  }
});
```

ДОДАТОК Б

Інтерфейс користувача вебплатформи



Б.1.Сторінка входу та реєстрації користувача



Б.2.Інтерфейс користувача платформи

ДОДАТОК В

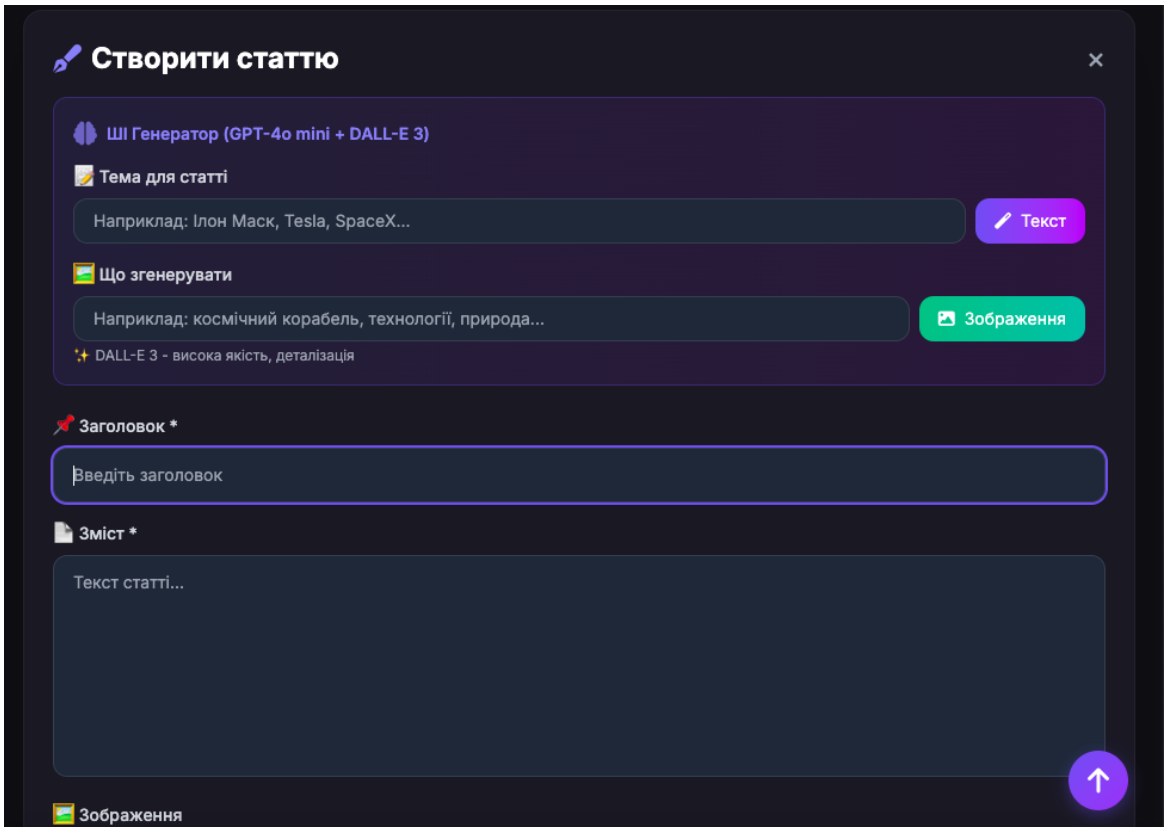
Конфігурація Firebase та OpenAI API

```
rules_version = '2';
service cloud.firestore {
  match /databases/{database}/documents {
    match /articles/{articleId} {
      allow read: if true; // Всі користувачі можуть читати статті
      allow create: if request.auth != null; // Створювати може тільки авторизований користувач
      allow update, delete: if request.auth != null && request.auth.uid == resource.data.authorId;
    }
    match /users/{userId} {
      allow read, write: if request.auth != null && request.auth.uid == userId;
    }
  }
}
```

```
{
  "id": "art_84f93a20bc",
  "title": "Перспективи інтеграції великих мовних моделей у сучасний веб",
  "content": "Текст статті, згенерований за допомогою шлюзу OpenAI API...",
  "authorId": "user_uid_123456789",
  "authorName": "Юлія Бабина",
  "imageUrl": "https://firebasestorage.googleapis.com/v0/b/ideanet...",
  "tags": ["ШтучнийІнтелект", "ВебРозробка", "OpenAI"],
  "createdAt": "2026-05-15T11:24:00Z",
  "viewsCount": 142
}
```

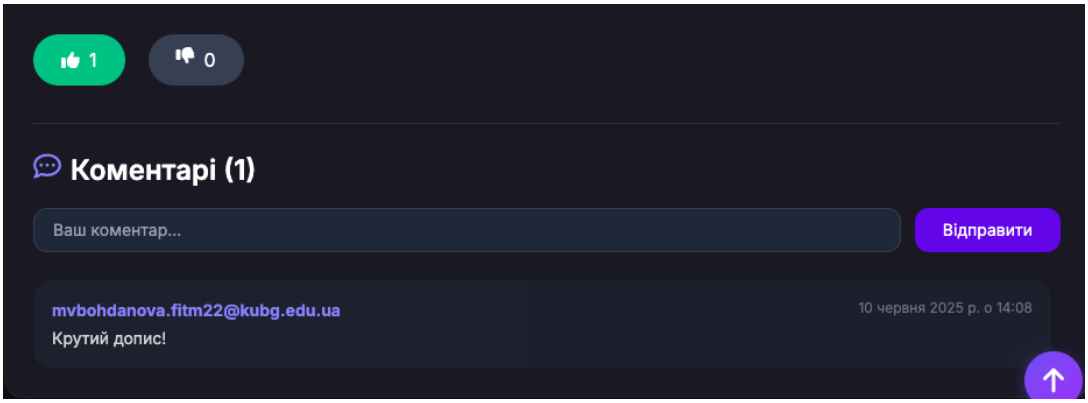
ДОДАТОК Г

Скріншоти роботи системи



Г.1.Форма створення нової статті

Г.2.Відображення опублікованої статті



Г.3.Реалізація функції оцінювання статті користувачами

ДОДАТОК Д

Результати тестування вебплатформи

Показники часу відгуку системи (Response Time) за умов різної кількості одночасних запитів

Тип операції / Запиту до системи	Середній час відгуку (10 користувачів), мс	Середній час відгуку (50 користувачів), мс	Середній час відгуку (100 користувачів), мс	Статус успішності операцій (Success Rate), %
Автентифікація користувача (Firebase Auth)	120	145	185	100.0 %
Читання списку статей з Firestore (NoSQL)	45	65	92	100.0 %
Запис нової публікації в базу даних	85	110	150	100.0 %
Генерація тексту статті (OpenAI GPT-4o-mini)*	2400	2850	3400	98.4 %
Генерація зображення обкладинки (DALL-E 3)*	4100	4500	5200	97.2 %

*Примітка: Тривалий час відгуку під час генерації контенту зумовлений архітектурною специфікою обробки запитів на хмарних серверах компанії OpenAI та часом, необхідним для розрахунку ваг нейромереж.