МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ
Кафедра програмної інженерії

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

на тему: **«РОЗРОБКА ВЕБПЛАТФОРМИ ДЛЯ УПРАВЛІННЯ БІЗНЕС-ДОДАТКАМИ З ВИКОРИСТАННЯМ NEXT.JS ТА ШТУЧНОГО ІНТЕЛЕКТУ»**

Виконав: студент 4 курсу, групи 8.1214-іпз-1
спеціальності 121 інженерія програмного забезпечення
(шифр і назва спеціальності)
освітньої програми інженерія програмного забезпечення
(назва освітньої програми)
М.О. Малоок
(ініціали та прізвище)

Керівник доцент кафедри програмної інженерії, к.ф.-м.н. Мильцев О.М.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Рецензент завідувач кафедри фундаментальної та прикладної математики, професор, д.т.н. Гребенюк С.М.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Запоріжжя – 2025

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет математичний

Кафедра програмної інженерії

Рівень вищої освіти магістр

Спеціальність 121 інженерія програмного забезпечення
(шифр і назва)

Освітня програма інженерія програмного забезпечення

ЗАТВЕРДЖУЮ
Завідувач кафедри програмної інженерії, к.ф.-м.н., доцент

_____ Лісняк А.О.
(підпис)

“ ____ ” _____ 2025 р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

Малооку Микиті Олександровичу
(прізвище, ім’я та по-батькові)

1. Тема роботи Розробка вебплатформи для управління бізнес-додатками з використанням Next.js та штучного інтелекту

керівник роботи Мильцев Олександр Михайлович, к.ф.-м.н.
(прізвище, ім’я та по-батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від « 28 » лютого 2025 року № 326-с

2. Строк подання студентом роботи 04.12.2025 р.

3. Вихідні дані до роботи 1. Постановка задачі.
2. Перелік літератури.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Основні теоретичні відомості.
2. Аналіз вимог до програмного застосунку.
3. Створення програмного застосунок з інтеграцією штучного інтелекту.
4. Тестування програмного застосунку.

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)
презентація за темою доповіді

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання 07.03.2025 р.

К А Л Е Н Д А Р Н И Й П Л А Н

№	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Розробка плану роботи.	14.03.2025	
2.	Збір вихідних даних.	11.04.2025	
3.	Обробка методичних та теоретичних	16.05.2025	
	джерел.		
4.	Розробка першого та другого розділу.	25.08.2025	
5.	Розробка третього та четвертого розділу.	24.11.2025	
6.	Оформлення та нормоконтроль	01.12.2025	
	кваліфікаційної роботи магістра.		
7.	Захист кваліфікаційної роботи.	18.12.2025	

Студент _____ М.О. Малоок
(підпис) (ініціали та прізвище)

Керівник роботи _____ О.М. Мильцев
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

Нормоконтролер _____ А.В. Столярова
(підпис) (ініціали та прізвище)

РЕФЕРАТ

Кваліфікаційна робота магістра «Розробка вебплатформи для управління бізнес-додатками з використанням Next.js та штучного інтелекту»: 78 с., 33 рис., 1 табл., 31 джерело.

АВТЕНТИФІКАЦІЯ, БАЗА ДАНИХ, ГЕЙМІФІКАЦІЯ, СИСТЕМА УПРАВЛІННЯ КОНТЕНТОМ, МОДУЛЬНА АРХІТЕКТУРА, ШТУЧНИЙ ІНТЕЛЕКТ, WEB-ПЛАТФОРМА, WEB-РОЗРОБКА, API, BUSINESS PLATFORM, MONGODB, NEXT.JS, SAAS.

Об’єкт дослідження – процес проєктування та реалізації універсальних вебплатформ для створення та управління бізнес-додатками.

Мета роботи: створити масштабовану модульну вебплатформу для бізнес-додатків із використанням Next.js та інструментів штучного інтелекту, а також розробити на її основі прикладний бізнес-додаток – планувальник контенту для соціальних мереж.

Методи дослідження – аналіз сучасних вебтехнологій і платформ планування, моделювання архітектури, проєктування структури даних, реалізація фронтенд і бекенд компонент, інтеграція функцій AI, тестування продуктивності та оцінка якості системи.

У кваліфікаційній роботі проведено аналіз існуючих рішень для бізнес-автоматизації та платформ управління контентом, визначено вимоги до універсальної платформи, сформовано архітектуру модульної системи з підтримкою багатофункціонального ядра. Реалізовано вебплатформу з авторизацією, керуванням користувачами, конфігурацією бізнес-додатків, структурою бази даних та інтегрованими можливостями штучного інтелекту. На основі цієї платформи створено бізнес-додаток – планувальник контенту, що включає генерацію контенту за допомогою ШІ, систему рівнів і винагород та інструменти для управління публікаціями.

SUMMARY

Master's qualifying paper "Development of a Web Platform for Managing Business Applications Using Next.js and Artificial Intelligence": 78 pages, 33 figures, 1 table, 31 references.

AUTHENTICATION, DATABASE, GAMIFICATION, CONTENT MANAGEMENT SYSTEM, MODULAR ARCHITECTURE, ARTIFICIAL INTELLIGENCE, WEB DEVELOPMENT, WEB PLATFORM, API, BUSINESS PLATFORM, MONGODB, NEXT.JS, SAAS, ARTIFICIAL INTELLIGENCE.

The object of the study is the process of designing and implementing universal web platforms for creating and managing business applications.

The aim of the study is to develop a scalable modular web platform for business applications using Next.js and artificial intelligence tools, as well as to build a practical business application based on this platform – a content planner for social networks.

The research method includes analysis of modern web technologies and planning platforms, system architecture modeling, data structure design, implementation of frontend and backend components, AI functionality integration, performance testing, and system quality evaluation.

This work examines existing solutions for business automation and content management, identifies the requirements for a universal SaaS platform, and develops the architecture of a modular system with a multifunctional core. A fully functional web platform has been implemented, featuring authentication, user management, application configuration, database structure, and integrated AI capabilities. Based on this platform, a business application – a content planner – has been developed. It includes AI-assisted content generation, a system of levels and rewards, and tools for managing publication schedules.

ЗМІСТ

Завдання на кваліфікаційну роботу……..……..2
Реферат 4
Summary 5
Вступ..... 8
1 Аналіз існуючих рішень і технологій 10
1.1 Огляд сучасних рішень для управління бізнес-процесами (CRM, ERP, SaaS) ……..……..10
1.2 Технології веброзробки: порівняння Next.js, React, Angular, Vue.js . 11
1.3 Використання штучного інтелекту в сучасних вебплатформах 14
1.4 Інтеграція штучного інтелекту у вебплатформи 16
1.5 Платформи для планування та управління контентом у соціальних мережах (Buffer, Hootsuite, Later, Metricool)..... 18
1.6 Аналіз вимог до універсальної платформи для бізнес-додатків..... 19
2 Проєктування вебплатформи 23
2.1 Функціональні та нефункціональні вимоги до системи 23
2.2 Архітектура системи (Frontend, Backend, AI, база даних, API) 24
2.3 Модель даних (на основі MongoDB, Mongoose, Prisma) 27
2.4 Безпека, автентифікація, права доступу (NextAuth, JWT)..... 28
2.5 Проєктування модульної платформи для створення бізнес-додатків 30
2.6 Проєктування бізнес-додатку «Планувальник контенту» на основі платформи..... 33
2.7 Проєктування системи гейміфікації..... 35
3 Реалізація проєкту 38
3.1 Вибір і налаштування стеку технологій (Next.js, React, MongoDB, Prisma) ……38
3.2 Реалізація універсальної вебплатформи для бізнес-додатків..... 39
3.3 Реалізація бізнес-додатку «Планувальник контенту»..... 46

3.4 Інтеграція AI для генерації та оптимізації контенту 61
3.5 Реалізація системи гейміфікації 64
4 Тестування та оцінка якості 67
4.1 Методологія тестування 67
4.2 Тестування базової SaaS-платформи 68
4.3 Тестування функцій бізнес-додатку (планувальника) 69
4.4 Тестування продуктивності 71
4.5 Оцінка якості розробленої системи 72
Висновки 74
Перелік посилань 76

ВСТУП

У сучасних умовах цифрової трансформації бізнесу вебплатформи стали ключовим інструментом для організації, автоматизації та масштабування бізнес-процесів. Компанії все частіше використовують програмні рішення не лише для управління клієнтськими відносинами (CRM) або ресурсами підприємства (ERP), а й для комплексної взаємодії з користувачами, керування фінансами, аналітики, комунікацій, підтримки та інших аспектів діяльності. Зростаюча популярність моделей програмного забезпечення як сервісу (SaaS) також сприяє активному впровадженню гнучких, доступних і масштабованих вебплатформ, що не потребують розгортання інфраструктури на стороні замовника.

Водночас потреба в персоналізованих і розумних рішеннях спричиняє інтеграцію технологій штучного інтелекту (ШІ) у такі платформи. ШІ-сервіси, зокрема машинне навчання, обробка природної мови або інтелектуальні рекомендаційні системи, дозволяють значно підвищити ефективність управління даними, автоматизувати рутинні процеси та покращити взаємодію з користувачами. Вебплатформи нового покоління мають враховувати ці можливості на рівні своєї архітектури, підтримувати масштабування, адаптивний інтерфейс та забезпечувати безпеку на всіх етапах обробки даних.

Актуальність теми обумовлена зростаючим попитом на інтелектуальні SaaS-рішення для бізнесу, які поєднують сучасні фронтенд і бекенд технології, безпечну архітектуру, інтеграцію з API сторонніх сервісів та підтримку ШІ-функціональності. Розробка таких систем потребує поєднання знань з галузі веброзробки, інженерії програмного забезпечення, штучного інтелекту та інформаційної безпеки.

Метою кваліфікаційної роботи є проєктування та реалізація вебплатформи для управління бізнес-додатками з використанням фреймворку Next.js та технологій штучного інтелекту. Запропоноване рішення має

забезпечити гнучкість у керуванні функціональністю вебдодатків, підтримку масштабованої архітектури, зручний інтерфейс для адміністраторів та користувачів, а також механізми безпечної автентифікації та контролю доступу.

Об’єктом дослідження є процес побудови багатофункціональних вебплатформ з відкритою структурою, що дозволяє інтегрувати нові модулі та сервіси. Предметом дослідження є архітектурні рішення, засоби реалізації фронтенд і бекенд частини, використання бази даних, побудова API та впровадження інтелектуальних можливостей у межах платформи.

У першому розділі роботи розглянуто теоретичні аспекти: існуючі рішення для автоматизації бізнесу (CRM, ERP, SaaS), аналіз популярних вебтехнологій (React, Next.js, Angular, Vue.js), можливості інтеграції ШІ та обґрунтування вибору технологічного стеку. Другий розділ присвячено проєктуванню системи: визначено функціональні та нефункціональні вимоги, описано архітектуру платформи, модель даних, принципи безпеки та доступу. У третьому розділі наведено реалізацію прототипу вебплатформи, описано основні функції, інтерфейси та особливості ШІ-модуля. Також проведено тестування системи, оцінку її продуктивності та перспектив подальшого розвитку.

Таким чином, дана робота спрямована на дослідження та практичну реалізацію сучасної вебплатформи, що поєднує гнучку інфраструктуру, масштабовану архітектуру та функції штучного інтелекту, орієнтованої на потреби бізнесу в умовах цифрової трансформації.

1 АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ І ТЕХНОЛОГІЙ

1.1 Огляд сучасних рішень для управління бізнес-процесами (CRM, ERP, SaaS)

У сучасних умовах цифрової трансформації бізнесу зростає потреба в ефективному управлінні процесами, ресурсами та взаємодією з клієнтами. Для цього компанії впроваджують спеціалізовані інформаційні системи, серед яких найбільш поширеними є CRM (Customer Relationship Management), ERP (Enterprise Resource Planning) та SaaS (Software as a Service) рішення [1].

CRM-системи орієнтовані на управління відносинами з клієнтами. Вони дозволяють автоматизувати процеси продажів, маркетингу, обслуговування клієнтів, забезпечуючи централізований доступ до інформації про контрагента. Прикладами популярних CRM-платформ є Salesforce, HubSpot, Zoho CRM, які надають широкі можливості для аналітики, комунікацій і персоналізації взаємодії з клієнтами. Такі системи особливо корисні для B2B-компаній, кол-центрів і організацій із розгалуженою мережею контактів [2].

ERP-системи забезпечують комплексне управління ресурсами підприємства, охоплюючи фінанси, логістику, виробництво, управління персоналом та інші функціональні області. ERP-платформи, такі як SAP, Oracle NetSuite, Microsoft Dynamics 365, дозволяють уніфікувати облік і координацію між підрозділами підприємства, підвищуючи прозорість та ефективність управління. ERP-рішення найчастіше використовуються середнім і великим бізнесом, де важлива інтеграція між усіма службами організації [1, 3].

SaaS-платформи – це програмне забезпечення, яке надається як онлайн-сервіс за підпискою. Такий підхід дозволяє користувачам працювати з додатком через браузер без необхідності локального встановлення. Серед переваг SaaS – швидке розгортання, масштабованість, зниження витрат на IT-

інфраструктуру. Популярними SaaS-рішеннями є Google Workspace, Trello, Slack, Shopify. SaaS-модель стала основою для більшості сучасних вебплатформ, зокрема у сфері управління бізнес-додатками [1].

В умовах високої конкуренції на ринку програмного забезпечення важливою тенденцією стало об’єднання функціональностей CRM, ERP та SaaS у єдині гнучкі платформи, які дозволяють налаштовувати інтерфейси, логіку бізнес-процесів та аналітику під потреби конкретної організації. Крім того, дедалі більше рішень інтегрують елементи штучного інтелекту для автоматизації задач, прогнозування тенденцій, персоналізації досвіду користувача [1].

На цьому тлі актуальним є створення універсальної вебплатформи, що поєднує в собі можливості управління додатками, користувачами, інтеграції сторонніх сервісів та використання AI-технологій. Саме таку концепцію закладено в розробку, яка розглядається в межах даної кваліфікаційної роботи.

1.2 Технології веброзробки: порівняння Next.js, React, Angular, Vue.js

У сучасній веброзробці одним із ключових рішень є вибір технологічного стеку, від якого залежить масштабованість, продуктивність, зручність підтримки та швидкість розробки програмного забезпечення. Серед найпопулярніших інструментів для створення вебінтерфейсів виділяються React, Angular, Vue.js та побудовані на їх основі фреймворки, зокрема Next.js. Кожна з цих технологій має свої особливості, переваги та сфери застосування.

React – це JavaScript-бібліотека, розроблена компанією Meta (раніше Facebook), яка дозволяє створювати інтерфейси користувача на основі компонентної архітектури. React підтримує односторінкові додатки (SPA), віртуальний DOM для високої продуктивності та має широку екосистему сторонніх бібліотек.

Основні переваги React:

– простота у вивченні та використанні;
– висока гнучкість;
– активна спільнота та багатий набір готових рішень [4].

Однак React сам по собі не є повноцінним фреймворком, тому для побудови повного стеку розробник має самостійно вибирати рішення для маршрутизації, роботи з даними, SEO тощо.

Angular – це повноцінний фронтенд-фреймворк від Google, побудований на TypeScript. Він забезпечує повний набір інструментів для створення масштабованих вебдодатків: від вбудованої маршрутизації до форм, валідації, DI (dependency injection) та RxJS для реактивного програмування. Angular орієнтований на великі проєкти з чіткою структурою коду. Переваги Angular:

– сувора структура та стандартизація;
– підтримка великих команд розробників;
– висока безпека та підтримка google [5].

Серед недоліків – висока складність входження та велика кількість шаблонного коду, що може сповільнювати розробку в невеликих проєктах.

Vue.js – це прогресивний фреймворк, що поєднує кращі ідеї з React і Angular, залишаючись при цьому простим та інтуїтивно зрозумілим. Він має низький поріг входу, підтримує компонентний підхід, реактивність, а також має власну систему маршрутизації (Vue Router) і управління станом (Vuex/Pinia). Переваги Vue.js:

– простота інтеграції;
– гнучкість у налаштуванні;
– компактність і висока продуктивність [6].

Vue широко використовується в Азії, особливо в Китаї, та має стабільну підтримку спільноти, хоча дещо поступається React і Angular за популярністю в глобальному масштабі.

Next.js – це фреймворк для React, розроблений компанією Vercel, що дозволяє створювати як SPA, так і серверно-рендерені (SSR) додатки. Він

поєднує простоту React із потужним інструментарієм для SEO, маршрутизації, генерації статичних сторінок (SSG), API-роутів та оптимізації продуктивності.

Переваги Next.js:

– підтримка SSR та SSG – важливо для SEO;
– інтегрована маршрутизація та підтримка API;
– висока продуктивність та автоматична оптимізація сторінок;
– підтримка модульної структури, TypeScript, middleware та інтернаціоналізації [7].

Next.js ідеально підходить для побудови SaaS-платформ, маркетплейсів, блогів і корпоративних сайтів, де важлива швидкість завантаження, адаптивність та SEO-оптимізація.

Далі наведена порівняльна характеристика вищезазначених інструментів (див. таб. 1.1).

Таблиця 1.1 – Порівняльна таблиця інструментів для створення вебінтерфейсів

Характеристика	**React**	**Angular**	**Vue.js**	**Next.js (React-based)**
Тип	Бібліотека	Фреймворк	Фреймворк	Фреймворк
Мова	JavaScript	TypeScript	JavaScript	JavaScript / TypeScript
SEO	Обмежене	Складне	Складне	Вбудоване
Гнучкість	Висока	Низька	Висока	Висока
Поріг входу	Середній	Високий	Низький	Середній
Продуктивність	Висока	Висока	Висока	Дуже висока
Підтримка AI/API	Обмежене	Через модулі	Через модулі	Вбудовані API-роути

З урахуванням цілей даної роботи – створення сучасної, масштабованої та SEO-оптимізованої вебплатформи з можливістю інтеграції штучного інтелекту – Next.js є найбільш доцільним вибором. Його гнучкість, продуктивність, розширюваність та активна підтримка роблять його оптимальним фреймворком для розробки бізнес-орієнтованих вебдодатків нового покоління.

1.3 Використання штучного інтелекту в сучасних вебплатформах

Штучний інтелект (ШІ) поступово стає невід'ємною частиною сучасних вебплатформ, трансформуючи спосіб взаємодії з користувачами, автоматизуючи бізнес-процеси та забезпечуючи персоналізований досвід. Впровадження інтелектуальних функцій дозволяє платформам не лише аналізувати дані в режимі реального часу, але й передбачати поведінку користувачів, оптимізувати робочі процеси та приймати обґрунтовані рішення без участі людини.

Основні напрямки використання ШІ у вебплатформах:

а) персоналізація контенту;

б) обробка природної мови (NLP);

в) аналітика та прогнозування;

г) системи рекомендацій;

д) автоматизація та оптимізація бізнес-процесів;

е) безпека та виявлення загроз [8].

Тепер детальніше про кожен з пунктів.

За допомогою алгоритмів машинного навчання можна аналізувати дії користувача (пошукові запити, перегляди, кліки) та автоматично формувати персоналізовані рекомендації. Наприклад, e-commerce платформи пропонують товари, які з високою ймовірністю зацікавлять конкретного користувача [8].

Використання моделей обробки природної мови дозволяє реалізовувати функціонал інтелектуальних чат-ботів, пошукових систем, автозаповнення форм, генерації текстів, машинного перекладу та аналізу відгуків користувачів. Сучасні API, такі як OpenAI GPT, Google Cloud NLP або Microsoft Azure Cognitive Services, забезпечують потужний інструментарій для інтеграції NLP у вебдодатки [9].

ШІ-моделі здатні аналізувати великі масиви даних для виявлення трендів, аномалій і створення прогнозів. Це особливо важливо для платформ

управління бізнес-додатками, де потрібно передбачити фінансові показники, поведінку клієнтів або ефективність маркетингових кампаній [8].

Системи рекомендацій широко використовуються у вебплатформах для підвищення конверсій і утримання користувачів. Вони базуються на колаборативній фільтрації, кластеризації користувачів або контенту, що дозволяє пропонувати товари, сервіси або функціональність відповідно до інтересів.

Інтелектуальні алгоритми можуть автоматично розподіляти завдання, визначати пріоритети, виявляти неефективність і навіть приймати управлінські рішення. Наприклад, у CRM-системах ШІ допомагає визначати найбільш перспективних лідів або оптимізувати комунікації з клієнтами.

ШІ-технології також використовуються для підвищення кібербезпеки вебплатформ – від виявлення підозрілої активності до блокування автоматичних атак. Алгоритми виявлення аномалій, поведінковий аналіз та багатофакторна автентифікація, що адаптується до користувача, значно підвищують рівень захисту [9].

Інтеграція ШІ у вебплатформи може відбуватися як через підключення зовнішніх API, так і через впровадження власних моделей машинного навчання. У випадку використання Next.js, можливості інтеграції є широкими завдяки:

- API-роутам для побудови бекенд-логіки (наприклад, обробка запитів до AI-сервісів);
- Server Actions у новому App Router для обробки форм та взаємодії з ШІ;
- Streaming і SSR/ISR для роботи з великими мовними моделями у реальному часі [10].

Використання Node.js-бібліотек для локального машинного навчання або доступу до сторонніх сервісів (наприклад, openai, tensorflow.js, langchain, huggingface тощо).

Інтеграція ШІ у вебплатформи відкриває нові можливості для автоматизації, персоналізації та інтелектуального аналізу. Вона підвищує

цінність платформи для кінцевих користувачів і бізнесу загалом. У контексті даної роботи використання ШІ стане ключовим елементом для реалізації інтелектуальних сервісів – аналітики, генерації звітів, підтримки клієнтів та прийняття рішень у режимі реального часу.

1.4 Інтеграція штучного інтелекту у вебплатформи

Інтеграція ШІ у вебплатформи відкриває нові можливості для автоматизації, персоналізації та інтелектуального аналізу. Вона підвищує цінність платформи для кінцевих користувачів і бізнесу загалом. У контексті даної роботи використання ШІ стане ключовим елементом для реалізації інтелектуальних сервісів – аналітики, генерації звітів, підтримки клієнтів та прийняття рішень у режимі реального часу [8].

На основі проведеного аналізу сучасних рішень для управління бізнес-додатками (CRM, ERP, SaaS), технологій веброзробки та способів інтеграції штучного інтелекту, можна сформулювати обґрунтований підхід до вибору архітектури та стеку технологій для розробки вебплатформи.

Для реалізації масштабованої, гнучкої та інтерактивної вебплатформи доцільним є застосування компонентно-орієнтованої архітектури з розподілом на клієнтську частину (frontend) – реалізовану у вигляді SPA (Single Page Application) з серверним рендерингом (SSR) для покращення SEO та швидкості завантаження, та серверну частину (backend) – побудовану на API-орієнтованій взаємодії, з підтримкою REST або GraphQL. Базу даних – для зберігання користувацьких даних, бізнес-логіки та налаштувань системи. Мікросервісів/інтеграцій – для підключення зовнішніх сервісів (оплати, email, аналітика, AI) [11].

У межах цієї архітектури рекомендовано використовувати Full Stack-платформу на базі Next.js, яка дозволяє об’єднати frontend і backend у єдиному середовищі розробки, підтримуючи SSR, ISR, API-роути, middleware, а також зручну інтеграцію з базами даних і сторонніми сервісами.

На основі порівняння та потреб платформи, обрано наступний стек технологій:

- Next.js – як основний фреймворк для веброзробки з підтримкою SSR, SEO, маршрутизації, API-роутів і сучасного рендерингу (App Router) [7];
- React – як бібліотека для побудови користувацького інтерфейсу з високим рівнем інтерактивності [4];
- Tailwind CSS – для швидкої та адаптивної верстки без потреби в написанні великої кількості CSS вручну [12];
- Prisma + MongoDB – для ORM-доступу до реляційної або документної бази даних відповідно до бізнес-завдань [13, 14];
- NextAuth – для реалізації системи автентифікації та авторизації з підтримкою OAuth, JWT та кастомних провайдерів [15, 16];
- Stripe – для інтеграції платіжної системи [17];
- Resend – для надсилання системних повідомлень на email [18];
- OpenAI API – для реалізації інтелектуальних функцій: генерації тексту, підтримки користувачів, аналітики [9];
- Recharts – для візуалізації даних та звітів;
- Tiptap – для вбудованого текстового редактора з розширеними можливостями (редагування бізнес-документів, описів тощо).

Перевагами обраного підходу є швидкість розробки завдяки використанню єдиного середовища для frontend і backend. Також гнучкість та масштабованість архітектури, що дозволяє додавати нові модулі, API, сервіси. Важливим є інтегрованість з ШІ, що реалізовано через API та локальні обчислення. Також перевагою є безпека та контроль доступу – завдяки використанню перевірених бібліотек авторизації та шифрування і висока продуктивність і SEO завдяки оптимізація завдяки SSR та оптимізації рендерингу в Next.js [7].

Таким чином, вибрана архітектура платформи на базі Next.js у поєднанні з інструментами штучного інтелекту забезпечує необхідний функціонал,

масштабованість та інтеграційні можливості для створення сучасної вебплатформи з інтелектуальним управлінням бізнес-додатками.

1.5 Платформи для планування та управління контентом у соціальних мережах (Buffer, Hootsuite, Later, Metricool)

У сучасних умовах цифрової економіки ефективна присутність бізнесу у соціальних мережах є одним із ключових факторів конкурентоспроможності. Створення та публікація контенту вимагають значних часових, організаційних та креативних ресурсів, тому на ринку сформувався попит на інструменти, що дозволяють автоматизувати планування, створення, публікацію та аналітику контенту. Серед найбільш популярних рішень виділяються платформи Buffer, Hootsuite, Later та Metricool, які поєднують функціонал контент-планувальника, менеджера соціальних мереж та аналітичної системи.

Buffer є одним із найстаріших сервісів у цій категорії та спеціалізується на простому й ефективному плануванні контенту. Платформа дозволяє створювати черги публікацій, працювати в команді, аналізувати ключові показники ефективності та керувати кількома соціальними мережами з єдиного інтерфейсу. Особливістю Buffer є фокус на мінімалізмі та швидкості роботи: інтерфейс не перевантажений додатковими інструментами, тому підходить для малого бізнесу та індивідуальних користувачів [19].

Hootsuite, на відміну від Buffer, позиціонується як корпоративне рішення. Платформа підтримує понад 20 соціальних мереж, надає розширені можливості моніторингу активності користувачів, аналізу взаємодій, управління рекламними кампаніями та роботи великої SMM-команди. Hootsuite також включає інструменти для реагування на коментарі, вхідні повідомлення та згадки бренду у режимі реального часу, що робить його корисним для великих компаній та агентств [20].

Later орієнтований переважно на візуальні соціальні мережі, такі як Instagram, Pinterest та TikTok. Його ключовою особливістю є зручний

інтерфейс для візуального планування контенту: публікації відображаються у вигляді сітки, що допомагає сформувати естетичний вигляд профілю. Платформа також пропонує інструменти для аналізу аудиторії, підбору хештегів та планування Reels, Stories і каруселей. Завдяки цьому Later є популярним серед брендів, орієнтованих на lifestyle, fashion та creative-індустрію [21].

Metricool є універсальною платформою, яка поєднує функції планування контенту, аналітики сайтів та рекламних кампаній. На відміну від інших сервісів, Metricool надає єдину панель керування для соціальних мереж, сайтів, блогів та рекламних кабінетів (Google Ads, Facebook Ads). Платформа дозволяє аналізувати показники взаємодії, ріст аудиторії, ефективність рекламних витрат та оптимізувати стратегію просування. Додатковою перевагою є можливість генерувати детальні автоматизовані звіти, що робить Metricool зручним для агентств та фрилансерів [22].

Загалом розглянуті платформи вирішують спільне завдання – забезпечення централізованого управління присутністю бренду у соціальних мережах. Однак кожне рішення має власні відмінності у фокусі та цільовій аудиторії. Buffer орієнтується на простоту, Hootsuite – на корпоративний сегмент, Later – на візуальний контент, а Metricool – на універсальність та інтеграцію рекламних інструментів. Вивчення можливостей цих платформ дозволяє визначити ключові вимоги до сучасних систем управління контентом та сформувати основу для розробки власних бізнес-додатків, включно з контент-планувальником, який реалізовано у рамках даної роботи.

1.6 Аналіз вимог до універсальної платформи для бізнес-додатків

Універсальні платформи для створення бізнес-додатків стають ключовим інструментом у швидкій розробці цифрових продуктів. Вони дозволяють скоротити час виходу на ринок (time-to-market), стандартизувати базову інфраструктуру та сфокусуватися на розробці унікальних функцій

конкретного застосунку. У рамках даної роботи розглядається така платформа – власнорозроблена кодова основа, яка виступає фундаментом для створення різних видів бізнес-додатків, зокрема планувальника контенту для соціальних мереж. Для визначення ключових вимог до такої універсальної системи необхідно проаналізувати її функціональні, нефункціональні та архітектурні характеристики, а також відповідність сучасним тенденціям у веброзробці [23, 24].

Базовий функціонал універсальної платформи повинен забезпечувати можливість швидкого створення й розгортання масштабованих вебдодатків. До переліку таких можливостей належать:

- автентифікація та авторизація користувачів, включно з підтримкою токенів JWT, OAuth-провайдерів та керуванням ролями;
- управління користувачами та організаціями, що включає створення профілів, редагування, контроль доступу й адміністрування;
- система оплати та білінгу, що підтримує інтеграцію з популярними платіжними сервісами (Stripe, LiqPay) і моделі підписки;
- інструменти для CRUD-операцій, на основі яких створюється логіка взаємодії з даними бізнес-додатку;
- локалізація та інтернаціоналізація, необхідні для підтримки декількох мов інтерфейсу;
- панель адміністратора з моніторингом активності користувачів, керуванням контентом, аналітикою та операційними функціями;
- система email – сповіщень, що дозволяє автоматизувати комунікацію з користувачами;
- базова інтеграція з AI – сервісами, необхідна для побудови інтелектуальних модулів у бізнес-застосунках.

Нефункціональні вимоги. Універсальна платформа має відповідати високим стандартам продуктивності, безпеки та надійності. До основних вимог належать:

- масштабованість, яка дозволяє системі підтримувати зростання кількості користувачів і даних без деградації швидкодії;

- продуктивність, що забезпечується використанням оптимізованих рендеринг-механізмів (Next.js App Router, серверні компонентів, ISR);
- захист даних, включно з шифруванням, захистом сесій, обмеженням доступу та дотриманням вимог GDPR;
- стабільність і толерантність до збоїв, що досягається через використання резервування даних, логування, моніторинг працездатності;
- модульність та розширюваність, які дозволяють відокремлювати різні бізнес-модулі та легко інтегрувати нові функції;
- кросплатформеність, що забезпечує коректну роботу платформи на різних пристроях та операційних системах.

Архітектурні вимоги. Універсальна платформа має ґрунтуватися на сучасних технологіях, здатних забезпечити баланс між швидкістю розробки, стабільністю та можливістю масштабування. Основні вимоги включають:

- використання Next.js як фреймворку для фронтенду та бекенду, що дозволяє поєднати SSR/SSG, API-маршрути та сучасний рендеринг;
- MongoDB як основна база даних, яка забезпечує гнучкість моделі даних та високу продуктивність при роботі з великими структурованими й неструктурованими об’єктами;
- Mongoose або Prisma як ORM/ODM-рішення, які забезпечують типізацію, валідацію, легкість роботи з моделями;
- використання модульної структури директорій, що спрощує організацію бізнес-логіки та підвищує читабельність коду;
- підтримка UI – компонентів на основі Radix UI та Tailwind CSS, що дозволяє створювати адаптивний і доступний інтерфейс.

Аналіз вимог до універсальної платформи демонструє, що вона повинна забезпечувати комплексну екосистему для створення структурованих, масштабованих та функціонально насичених бізнес-додатків. Наявність таких вимог є критичною передумовою для успішної розробки та подальшого

функціонування планувальника контенту, який використовує базову інфраструктуру платформи та розширює її за допомогою унікальних модулів та інтелектуальних рекомендацій.

2 ПРОЄКТУВАННЯ ВЕБПЛАТФОРМИ

2.1 Функціональні та нефункціональні вимоги до системи

Під час розробки вебплатформи для управління бізнес – додатками визначення вимог до системи є критичним етапом, що впливає на якість, стабільність та ефективність майбутнього програмного продукту. Вимоги умовно поділяються на функціональні, які описують основну поведінку та можливості системи, і нефункціональні, що визначають обмеження, характеристики продуктивності та умови експлуатації.

До основних функціональних вимог належить забезпечення повного циклу взаємодії користувача з системою – від реєстрації та автентифікації до управління бізнес-даними, користувачами та інтегрованими додатками. Платформа повинна надавати можливість реєстрації користувачів із різними рівнями доступу, зокрема адміністраторів, менеджерів і звичайних користувачів. Передбачено створення, редагування та видалення бізнес-додатків у межах одного середовища, з можливістю підключення сторонніх сервісів через API. У межах панелі керування користувачі повинні мати змогу переглядати статистику, налаштовувати повідомлення, керувати підписками та здійснювати платежі. Також важливо забезпечити гнучке конфігурування інтерфейсу та підтримку багатомовності для розширення цільової аудиторії.

Щодо нефункціональних вимог, ключовим є досягнення високої доступності системи та гарантії її працездатності за умов високого навантаження. Платформа повинна забезпечувати мінімальний час відгуку та високу швидкість обробки запитів, що є особливо важливим для SaaS-рішень із великою кількістю одночасних користувачів. Безпека даних займає центральне місце серед нефункціональних вимог, що передбачає реалізацію сучасних методів шифрування, захисту автентифікації, перевірки прав доступу та регулярного резервного копіювання даних. Крім того, система

повинна бути масштабованою, тобто здатною до розширення функціональності без істотного впливу на продуктивність або стабільність.

Особлива увага приділяється підтримці інтеграції зі штучним інтелектом, що накладає додаткові вимоги до обчислювальних ресурсів та ефективної взаємодії між компонентами системи. Крім того, система повинна мати зрозумілий інтерфейс користувача, адаптований до різних типів пристроїв і розмірів екранів, що сприятиме позитивному користувацькому досвіду.

У сукупності зазначені вимоги формують основу для архітектурного проєктування платформи та визначають напрям її подальшого розвитку.

2.2 Архітектура системи (Frontend, Backend, AI, база даних, API)

Архітектура розроблюваної вебплатформи передбачає побудову багаторівневої системи, яка включає клієнтську та серверну частини, базу даних, модулі штучного інтелекту та набір програмних інтерфейсів (API). Такий підхід дозволяє забезпечити масштабованість, модульність, гнучкість та інтеграцію новітніх технологій, включаючи засоби машинного навчання [25].

Клієнтська частина реалізована з використанням фреймворку Next.js, який базується на React (див. рис. 2.1). Це дозволяє організувати ефективний рендеринг сторінок як на стороні клієнта, так і на сервері, що позитивно впливає на продуктивність системи та оптимізацію для пошукових систем. Архітектура фронтенду враховує потреби багатомовної підтримки, динамічного перемикання тем оформлення, а також забезпечення доступу до функціональності штучного інтелекту через інтерфейси користувача. Крім того, передбачено реалізацію аналітичних панелей з інтерактивними графіками, що забезпечують візуалізацію ключових бізнес-показників [26, 27].

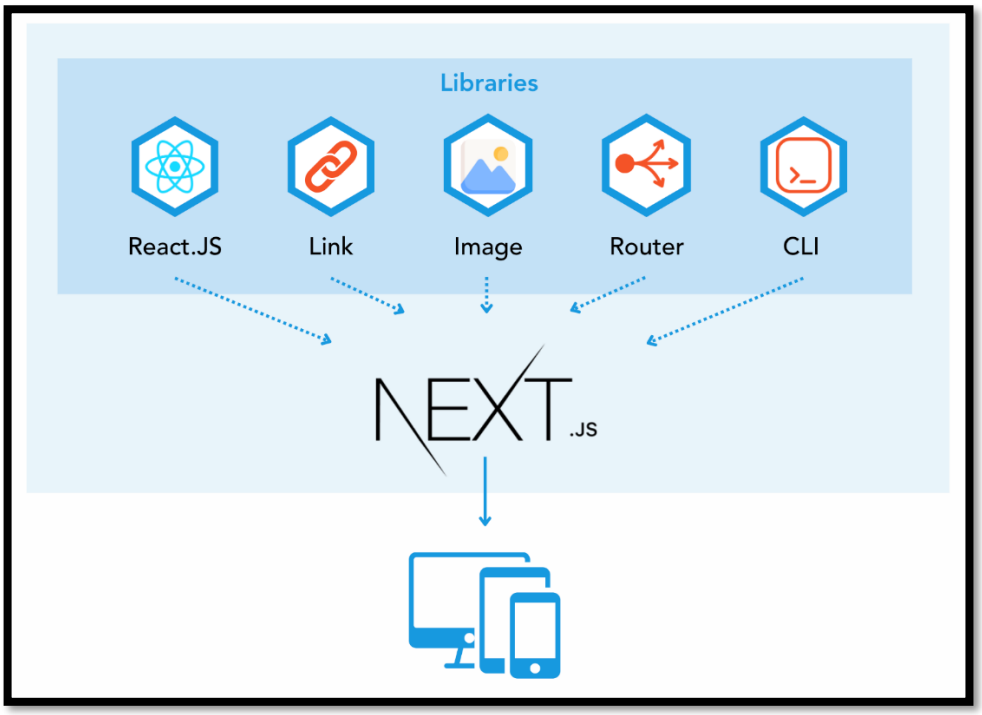


Рисунок 2.1 – Схема реалізації рендерингу сторінки клієнта у фреймворку Next.js

Серверна частина системи, яка також реалізується у межах Next.js або як окремий сервіс, відповідає за обробку запитів, автентифікацію та авторизацію користувачів, керування даними, здійснення транзакцій та взаємодію з модулями штучного інтелекту. Для зберігання та обробки користувацьких сесій, а також інтеграції з платіжними сервісами, таких як Stripe, використовується відповідна серверна логіка, реалізована через API-роути. Надсилання електронних повідомлень здійснюється за допомогою SMTP-протоколу через бібліотеку Resend.

Для зберігання даних система використовує документно-орієнтовану базу даних MongoDB. MongoDB застосовується для зберігання динамічних або слабо структурованих даних, таких як конфігурації, повідомлення або користувацькі налаштування (див. рис. 2.2). Такий підхід дозволяє забезпечуючи одночасно гнучкість та цілісність даних.

Штучний інтелект інтегрується у вигляді окремого сервісу, який виконує завдання генерації текстів, аналізу даних та персоналізації взаємодії з користувачами. Взаємодія з цим сервісом відбувається через програмні

інтерфейси, що забезпечують асинхронний обмін інформацією між клієнтською частиною та AI – модулем. Розміщення інтелектуальних компонентів окремо від основного сервера сприяє підвищенню надійності та дозволяє ефективно масштабувати обчислювальні ресурси за потреби.

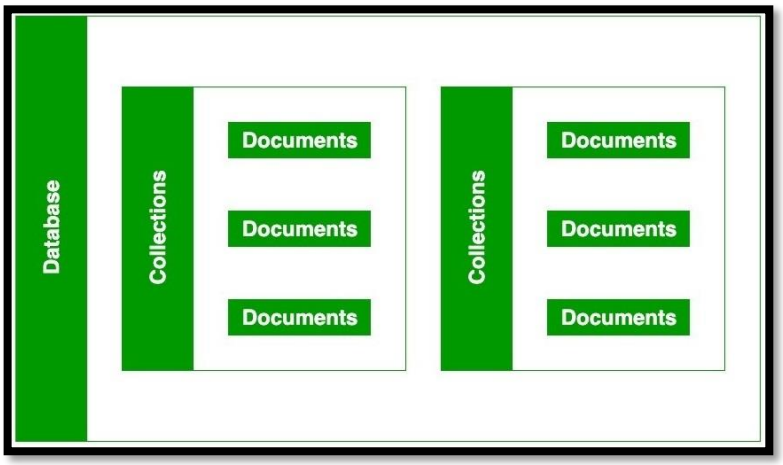


Рисунок 2.2 – Схема реалізації зберігання даних в базі даних MongoDB

Комунікація між окремими компонентами системи реалізується через чітко визначені програмні інтерфейси (див. рис. 2.3).

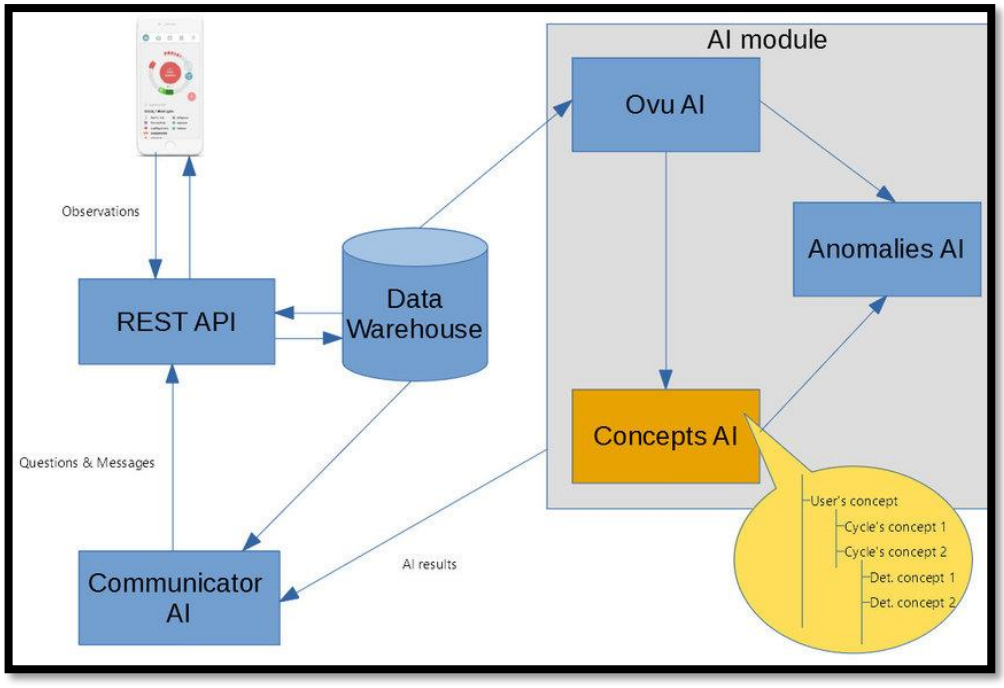


Рисунок 2.3 – Схема реалізації комунікації AI – модуля з компонентами системи через програмні інтерфейси API

API – інтерфейси забезпечують зв’язок між фронтендом, бекендом, базами даних та AI – модулем. Вони також використовуються для обміну даними з мобільними додатками або зовнішніми сервісами. Авторизація при доступі до API здійснюється за допомогою токенів доступу, що забезпечує належний рівень безпеки та контроль над правами доступу.

Загалом, розроблена архітектура відповідає сучасним вимогам до побудови SaaS – платформ, є придатною для масштабування, модернізації та інтеграції додаткових сервісів у майбутньому.

2.3 Модель даних (на основі MongoDB, Mongoose, Prisma)

Під час проєктування вебплатформи для управління бізнес – додатками особливу увагу було приділено вибору моделі даних, яка забезпечує гнучкість, масштабованість і надійність при зберіганні та обробці інформації. З огляду на специфіку завдань системи, у якій поєднуються структуровані транзакційні дані, користувацький контент і конфігураційна інформація, було прийнято рішення використовувати гібридний підхід із залученням як документно-орієнтованої бази даних MongoDB, так і реляційного шару на базі Prisma ORM.

MongoDB є основною системою зберігання для динамічних об’єктів, таких як налаштування користувачів, журнали активності, дані інтерфейсу, тимчасові сесії та кастомізовані модулі, які не мають жорсткої схеми. Такий підхід дає змогу оперативно змінювати структуру документів без необхідності модифікації схеми бази, що значно пришвидшує розробку та дозволяє гнучко реагувати на потреби бізнесу [14].

Для роботи з MongoDB використовується бібліотека Mongoose, яка надає зручний механізм моделювання даних, включаючи валідацію, типізацію та middleware – функціональність. Визначення схем через Mongoose дозволяє зберегти структуру даних у проєкті, не втрачаючи при цьому переваг NoSQL. Наприклад, структура документа користувача може включати особисту

інформацію, список дозволів, параметри інтерфейсу, мітки активності та пов’язані сервіси, що відображають індивідуальний контекст взаємодії з платформою.

Окремо обробляються дані, які вимагають суворої структурованості, консистентності та підтримки транзакцій. До таких даних належать облікові записи, підписки, платіжні транзакції, а також зв’язки між сутностями, що мають критичне значення для ведення бізнес-процесів. Для цього застосовується ORM Prisma, що працює з реляційною базою даних і забезпечує автоматичну генерацію типів, зручний API для взаємодії з базою, а також підтримку міграцій та перевірку цілісності схеми. Використання Prisma також полегшує інтеграцію з Next.js та системами автентифікації, зокрема через адаптер @next-auth/prisma-adapter [13, 15].

Гібридна модель дозволяє логічно розділити функціональні зони відповідальності між двома типами збереження даних: MongoDB забезпечує гнучке середовище для розширення та зберігання варіативної інформації, тоді як Prisma забезпечує контроль, передбачуваність і структурованість критичних бізнес-операцій. Такий підхід сприяє підвищенню масштабованості, розділенню навантаження та спрощенню технічного супроводу системи.

У підсумку, розроблена модель даних орієнтована на забезпечення балансу між динамічністю та надійністю, що є ключовим для сучасних SaaS-рішень, які працюють у динамічному бізнес-середовищі.

2.4 Безпека, автентифікація, права доступу (NextAuth, JWT)

Питання безпеки є одним із ключових при проєктуванні вебплатформи для управління бізнес – додатками, оскільки система обробляє конфіденційну інформацію користувачів, комерційні дані та інтегрується з зовнішніми платіжними та аналітичними сервісами. Надійна автентифікація, контроль

доступу та захист від поширених вебзагроз є критично важливими для забезпечення довіри користувачів і дотримання сучасних вимог до захисту даних.

Для реалізації автентифікації та управління сесіями користувачів у платформі використано бібліотеку NextAuth, яка тісно інтегрується з фреймворком Next.js і підтримує різні методи входу, включаючи реєстрацію за електронною поштою, соціальні логіни (OAuth) та одноразові токени (див. рис. 2.4). Застосування NextAuth дозволяє ефективно організувати систему авторизації з урахуванням ролей користувачів і прав доступу, а також забезпечує зручну підтримку адаптерів для взаємодії з базою даних через Prisma [15].

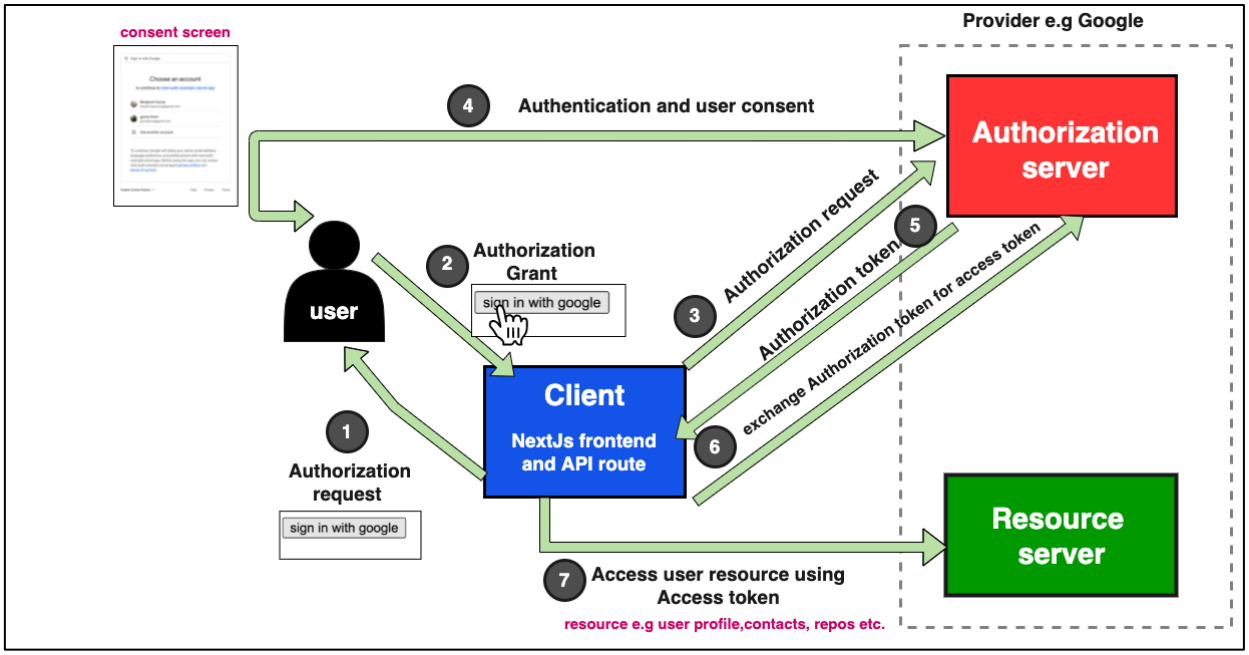


Рисунок 2.4 – Схема реалізації автентифікації користувачів через Google Authentication в додатку Next.js використовуючи NextAuth

Для забезпечення безпечної передачі даних між клієнтом і сервером використовуються JWT (JSON Web Tokens). Токени генеруються при вході в систему та зберігаються у клієнтській частині (через httpOnly cookies або LocalStorage), що дозволяє уникнути необхідності постійного зберігання сесій на сервері. При кожному запиті токен перевіряється на сервері для

підтвердження автентичності користувача. Такий підхід забезпечує масштабованість системи, зменшуючи навантаження на бекенд, а також дає змогу створювати захищені API, які доступні лише автентифікованим користувачам [16].

З метою обмеження доступу до окремих функціональних частин платформи реалізовано систему ролей, що дозволяє визначити рівень доступу до ресурсів залежно від призначеної ролі користувача (наприклад, «адміністратор», «менеджер», «користувач»). Розмежування прав доступу реалізовано як на рівні інтерфейсу користувача, так і на рівні API, що унеможливлює несанкціоноване використання адміністративних функцій через сторонні запити.

Крім того, забезпечено захист від типових вебзагроз, таких як XSS (міжсайтове скриптування), CSRF (міжсайтові запити), brute – force атаки та SQL/NoSQL ін’єкції. Для цього використано валідацію вхідних даних, обмеження кількості запитів до API, шифрування конфіденційної інформації та зберігання паролів у зашифрованому вигляді за допомогою бібліотеки bcryptjs.

Завдяки поєднанню перевірених технологій – таких як NextAuth та JWT – вдалося створити надійну систему безпеки, що відповідає сучасним вимогам до розробки хмарних сервісів і SaaS – рішень, і водночас забезпечити гнучкість та масштабованість для майбутнього розширення функціональності.

2.5 Проєктування модульної платформи для створення бізнес-додатків

Проєктування модульної універсальної платформи для створення бізнес – додатків є ключовим етапом розробки системи, здатної забезпечити швидке розгортання різноманітних SaaS – рішень. Модульність платформи дозволяє відокремити базову функціональність, спільну для всіх бізнес – додатків, від

специфічних компонентів, що реалізують логіку конкретного застосунку. Такий підхід знижує витрати на розробку, забезпечує масштабованість та створює умови для подальшого розвитку екосистеми додатків на основі єдиної інфраструктури.

Архітектурні принципи модульності полягають в ізоляції бізнес-логіки, завдяки якій кожен модуль реалізує чітко визначені функції і не залежить від внутрішньої реалізації інших компонентів. Повторне використання сервісів, таких як автентифікація, білінг, аналітика або email – повідомлення, дозволяє ефективно застосовувати ядро платформи для створення нових додатків, тоді як стандартизовані інтерфейси взаємодії забезпечують передбачувану поведінку системи. Принцип слабкої зв’язаності та високої когезії модулів сприяє легкій підтримці та масштабуванню платформи без необхідності модифікації базового ядра [26].

Структурно платформа складається з декількох рівнів, що взаємодіють між собою (див. рис. 2.5). На рівні базового ядра реалізовано фундаментальні функції, необхідні для роботи будь-якого бізнес – додатку. Це включає автентифікацію та управління сесіями користувачів, керування ролями та доступом, білінгові механізми, глобальні конфігурації застосунку та підтримку мультимовності. Також на цьому рівні формується єдиний інтерфейс користувача з використанням Radix UI та Tailwind CSS, що забезпечує уніфікований вигляд всіх додатків.

Рівень сервісів включає готові бізнес – сервіси, які можуть бути використані у різних типах SaaS – додатків. До них належать сервіси управління медіафайлами, автоматизовані email – розсилки, AI – адаптери для генерації та обробки контенту, а також сервіси для відкладених операцій і аналітики користувацької активності. Кожен сервіс реалізований як незалежний модуль, що дозволяє повторно використовувати його у різних додатках без модифікації ядра.

Рівень API та інтеграцій виконує роль «контракту» між клієнтською частиною, бізнес-додатками та зовнішніми сервісами. Стандартизовані API-

маршрути та middleware забезпечують коректність передачі даних, безпеку доступу та стабільність взаємодії між компонентами. На рівні модулів бізнес-додатків створюються конкретні SaaS-рішення, які використовують всі можливості базового ядра та сервісів платформи, але при цьому зберігають власну специфічну логіку, моделі даних та інтерфейси [26].

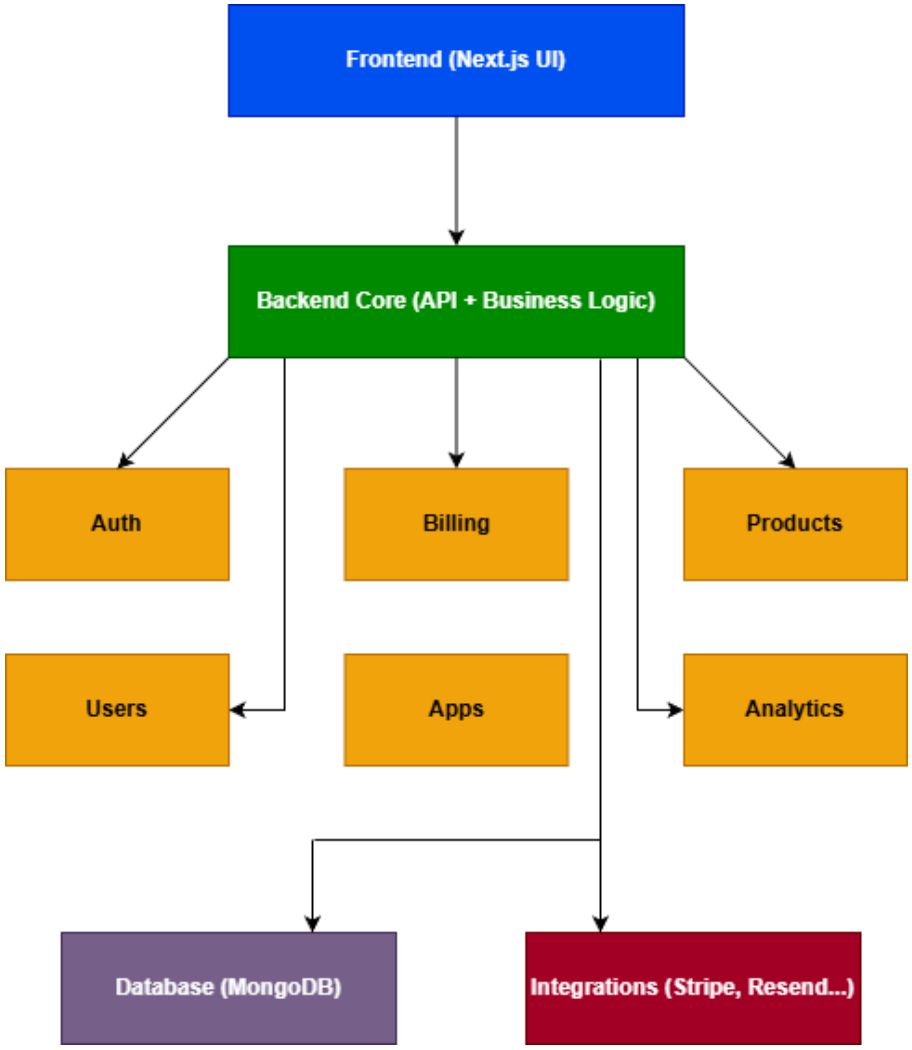


Рисунок 2.5 – Схема реалізації архітектури розроблюваної модульної платформи для створення бізнес-додатків

Використання Next.js як основи архітектури дозволяє поєднати серверні компоненти, маршрути API та edge-функції в єдину систему, що спрощує розробку та забезпечує високу продуктивність. Модульна структура директорій та можливість підключення розширень дозволяє створювати нові додатки без змін у базовому ядрі, а стандартизовані шаблони та конфігурації

забезпечують передбачуваність і безпеку системи. Централізована система управління доступом та логування подій гарантує цілісність даних і дозволяє підтримувати стабільну роботу платформи під час масштабування.

Отже, модульна архітектура забезпечує можливість створення масштабованих, повторно використовуваних і гнучких бізнес-додатків. На основі цієї архітектури у подальших розділах описується створення конкретного бізнес-додатку – гейміфікованого планувальника контенту, який демонструє практичну ефективність модульного підходу та універсальність платформи.

2.6 Проєктування бізнес-додатку «Планувальник контенту» на основі платформи

Проєктування бізнес-додатку «Планувальник контенту» ґрунтується на вже розробленій універсальній модульній платформі, яка забезпечує базову функціональність та стандартизовані сервіси, необхідні для швидкого створення і масштабування вебдодатків. Додаток орієнтований на користувачів, які ведуть активну присутність у соціальних мережах, та поєднує традиційні функції контент-планування з елементами для підвищення залученості та мотивації. Основна концепція передбачає інтеграцію функцій управління контентом, аналітики, автоматизованих рекомендацій і взаємодії користувачів через механізми, що стимулюють регулярну активність та послідовність публікацій [28].

Архітектура бізнес-додатку використовує усі сервіси, зокрема автентифікацію, управління ролями, API-маршрути, інтеграцію з AI-модулями та механізми білінгу для підключення платних тарифів. Логіка додатку побудована таким чином, що користувач може створювати контент-плани, призначати дати публікацій, завантажувати медіафайли та отримувати рекомендації від системи штучного інтелекту, що аналізує ефективність

минулих публікацій та оптимізує розклад для максимального охоплення аудиторії [29].

Схема реалізації архітектури розроблюваного бізнес-додатку відображає загальну структуру системи та взаємозв'язки між її основними компонентами (див. рис. 2.6). Архітектура побудована за принципами модульності та розподілу відповідальності, що забезпечує чітке відокремлення клієнтської частини, серверної логіки та зовнішніх сервісів. Бізнес-додаток функціонує на основі універсального ядра платформи, використовуючи спільні механізми автентифікації, доступу до даних, обробки запитів і інтеграції з модулями штучного інтелекту.

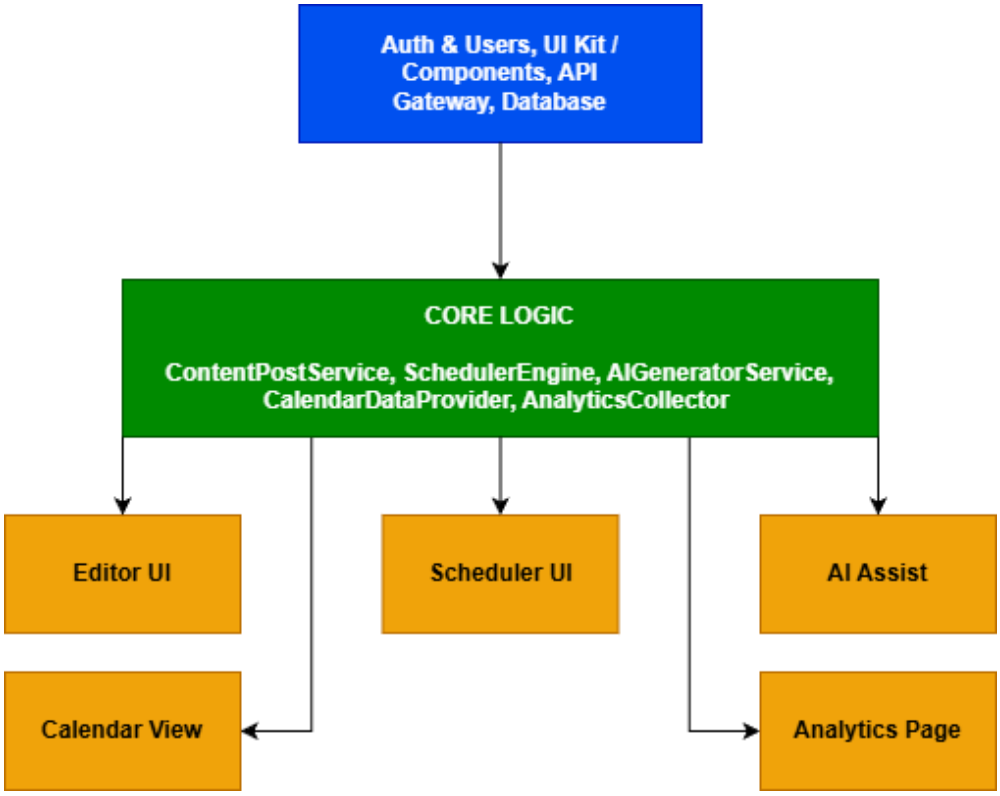


Рисунок 2.6 – Схема реалізації архітектури розроблюваного бізнес – додатку

Візуальна частина додатку реалізована на основі компонентів Radix UI та стилізована за допомогою Tailwind CSS, що забезпечує сучасний та адаптивний інтерфейс. Користувацький досвід передбачає інтуїтивну взаємодію з календарем публікацій, можливість перегляду аналітики в реальному часі та отримання нагород або балів за дотримання графіку

публікацій. Завдяки цьому елементи перетворюють рутинну задачу планування контенту на мотивуючу та залучаючу діяльність, що підвищує ефективність користувацьких кампаній у соціальних мережах.

Модель даних додатку адаптована під багатокористувацький режим і включає сутності для зберігання інформації про акаунти, пости, медіафайли, активності користувачів та їхні результати у системі гейміфікації. Для забезпечення масштабованості та продуктивності використовується MongoDB з ORM/ODM-шаром (Mongoose), що дозволяє динамічно розширювати схему даних без порушення роботи існуючих функцій.

Інтеграція штучного інтелекту виконує допоміжну функцію, надаючи рекомендації щодо оптимального часу публікацій, підбору хештегів та генерації текстового контенту. AI-модуль реалізований як сервіс-плагін, що взаємодіє з ядром платформи через стандартизований API, що дозволяє додатку масштабуватися та адаптуватися до потреб різних користувачів без втручання в базовий код [30].

Таким чином, проєктування бізнес-додатку «Планувальник контенту» демонструє ефективність використання модульної платформи, дозволяючи поєднати базову інфраструктуру з інноваційними механізмами та інтелектуальної підтримки контент-менеджменту. Цей підхід забезпечує не лише швидке створення додатку, але й високу залученість користувачів, масштабованість та готовність до подальшого розвитку нових функцій і інтеграцій.

2.7 Проєктування системи гейміфікації

Проєктування системи гейміфікації є невід’ємною частиною бізнес-додатку «Планувальник контенту» і спрямоване на підвищення мотивації користувачів, регулярності їхніх дій та ефективності управління контентом у соціальних мережах. Основна мета гейміфікації полягає в тому, щоб перетворити рутинні операції, такі як створення, планування та публікація

контенту, на інтерактивний та мотивуючий процес (див. рис. 2.7). Це досягається за рахунок впровадження елементів винагороди, досягнень, рівнів та системи балів, які стимулюють користувачів підтримувати регулярну активність і прагнути до покращення результатів.

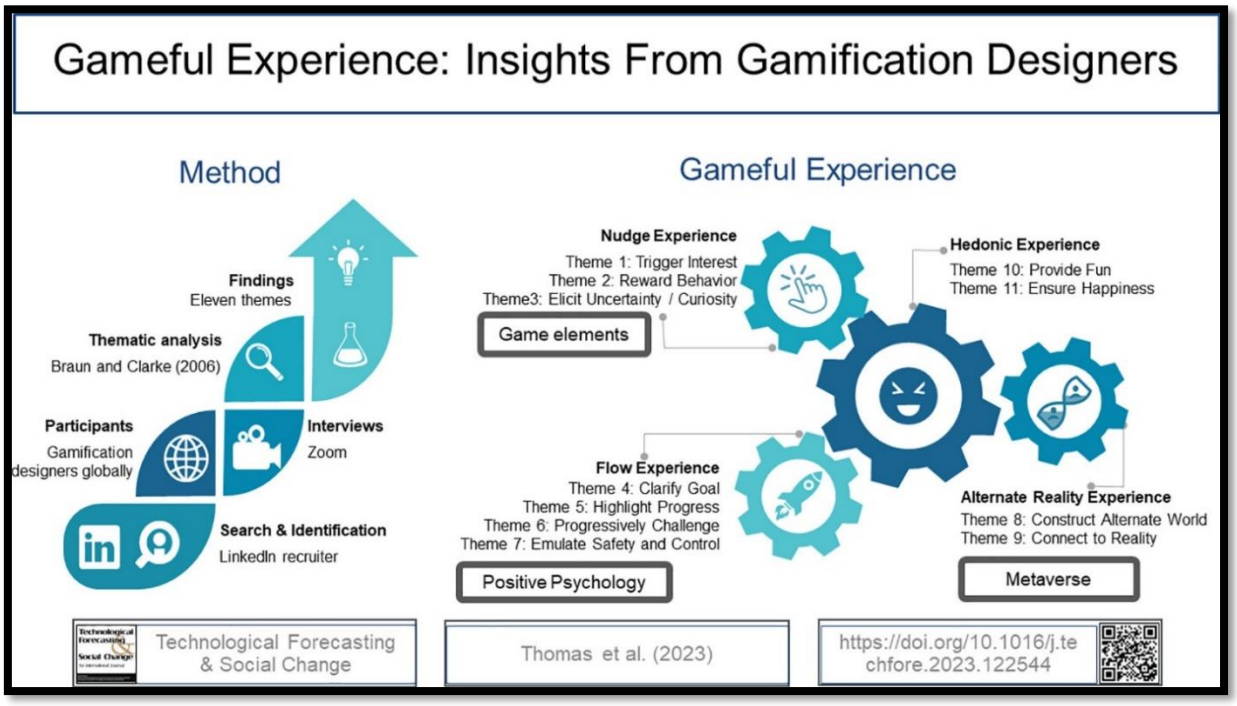


Рисунок 2.7 – Схема концепції гейміфікації і як вона впливає на сприйняття користувача

Система гейміфікації інтегрується безпосередньо у базову платформу, використовуючи її сервіси для автентифікації, керування користувачами, збору аналітичних даних та управління даними у базі. Вона передбачає обробку дій користувача, таких як створення публікацій, виконання завдань за розкладом або досягнення певних показників активності, і автоматичне нарахування балів, віртуальних нагород або підвищення рівня користувача в системі. Дані про активність зберігаються у структурованих моделях, що дозволяє проводити аналітику та генерувати рекомендації щодо підвищення ефективності контент-планування.

Візуальна інтеграція гейміфікаційних елементів забезпечує безшовний користувацький досвід. Інтерфейс відображає прогрес у вигляді шкал,

прогрес-барів, бейджів та інших графічних елементів, що мотивують користувачів виконувати завдання систематично. Паралельно система відстежує досягнення та результати кожного користувача, надаючи персоналізовані рекомендації та підказки для оптимізації процесів публікації.

Особливу увагу при проєктуванні приділено масштабованості та гнучкості системи. Архітектура гейміфікаційного модуля побудована таким чином, що нові елементи мотивації, правила нарахування балів або механіки нагородження можуть бути додані без зміни ядра платформи. Це забезпечує можливість адаптації системи до різних сценаріїв використання та потреб користувачів, а також підтримку багатокористувацького режиму з одночасним відстеженням активності великої кількості акаунтів.

Використання модульного підходу дозволяє системі гейміфікації взаємодіяти з іншими сервісами додатку, зокрема з AI-модулем, який може надавати персоналізовані поради щодо часу публікацій або тематики контенту на основі поведінки користувача та результатів попередніх публікацій. Це створює синергетичний ефект, коли гейміфікаційні механіки не лише стимулюють активність, але й підвищують ефективність використання самого планувальника контенту.

Таким чином, система гейміфікації виступає ключовим компонентом бізнес-додатку, що забезпечує залученість, регулярність дій та підвищує користувацьку цінність продукту. Вона демонструє, як інтеграція інноваційних механік мотивації у межах модульної платформи дозволяє створювати ефективні та сучасні вебдодатки для управління контентом у соціальних мережах.

3 РЕАЛІЗАЦІЯ ПРОЄКТУ

3.1. Вибір і налаштування стеку технологій (Next.js, React, MongoDB, Prisma)

У процесі створення вебплатформи для управління бізнес-додатками важливим етапом є вибір відповідного технологічного стеку, що забезпечить надійність, масштабованість та простоту в подальшій підтримці та розвитку системи. У цій кваліфікаційній роботі обґрунтовано вибір таких ключових інструментів, як Next.js, React, MongoDB та Prisma, які у сукупності створюють ефективне середовище для розробки повнофункціонального вебзастосунку.

Основою фронтенду платформи обрано Next.js – сучасний фреймворк, побудований на базі React, який поєднує можливості серверного та клієнтського рендерингу. Його застосування дозволяє реалізувати швидке завантаження сторінок, гнучку маршрутизацію та ефективну SEO-оптимізацію. Окрім того, Next.js підтримує генерацію статичних сторінок і серверну інтеграцію, що є доцільним для SaaS-платформи з потребою в динамічному контенті [7].

Бібліотека React використовується як базова технологія побудови інтерфейсу користувача. Її компонентний підхід дає змогу створювати повторно використовувані блоки інтерфейсу та забезпечує високу реактивність при взаємодії з даними. Застосування React у поєднанні з функціональністю Next.js дозволяє досягти ефективної організації коду та швидкого часу відгуку інтерфейсу.

Для зберігання даних обрано MongoDB – документоорієнтовану базу даних NoSQL, яка надає гнучку модель зберігання, що добре підходить для платформ із часто змінюваною структурою даних. Вона забезпечує високу продуктивність і горизонтальне масштабування, що особливо важливо в умовах можливого зростання кількості користувачів та даних у майбутньому.

Комунікація між базою даних та серверною частиною реалізується за допомогою Prisma – сучасного ORM (Object-Relational Mapping) інструменту, який спрощує роботу з MongoDB і дозволяє здійснювати типізований доступ до даних. Prisma забезпечує зручний підхід до міграцій, валідації, запитів і дає змогу уникнути помилок на етапі розробки завдяки статичній перевірці типів.

На етапі налаштування проєкту було створено базову структуру застосунку з інтеграцією Next.js і підключенням до MongoDB через Prisma. Також налаштовано систему маршрутизації, API-обробники, обробку помилок і з'єднання з базою даних, що забезпечує стабільну основу для подальшого розширення функціоналу платформи. Такий вибір технологій дозволяє створити масштабовану, продуктивну та сучасну систему, яка відповідає вимогам бізнес-користувачів.

3.2. Реалізація універсальної вебплатформи для бізнес-додатків

Реалізація універсальної вебплатформи для бізнес-додатків, яка отримала умовну назву SaaS Core, базується на архітектурних рішеннях, розроблених на етапі проєктування, і спрямована на забезпечення повторного використання функціональних компонентів та масштабованості додатків. Платформа реалізована з використанням сучасного стека технологій, до якого входять Next.js, React, MongoDB та Prisma, що дозволяє поєднати переваги серверного рендерингу, інтерактивного фронтенду та ефективного управління даними. Основною метою реалізації є створення ядра, яке забезпечує базову функціональність, стандартизовані сервіси та єдині інтерфейси взаємодії, що можуть бути використані у різноманітних бізнес-додатках без необхідності повторної розробки фундаментальних компонентів (див. рис. 3.1).

Реалізація ядра платформи передбачала створення механізмів автентифікації та управління сесіями користувачів на основі NextAuth і JWT, що забезпечує безпечний доступ до всіх компонентів системи та контроль ролей користувачів [31]. Крім того, були реалізовані сервіси для обробки

платежів і білінгу, які дозволяють інтегрувати різні моделі монетизації, включаючи платні тарифи та преміум-функції.

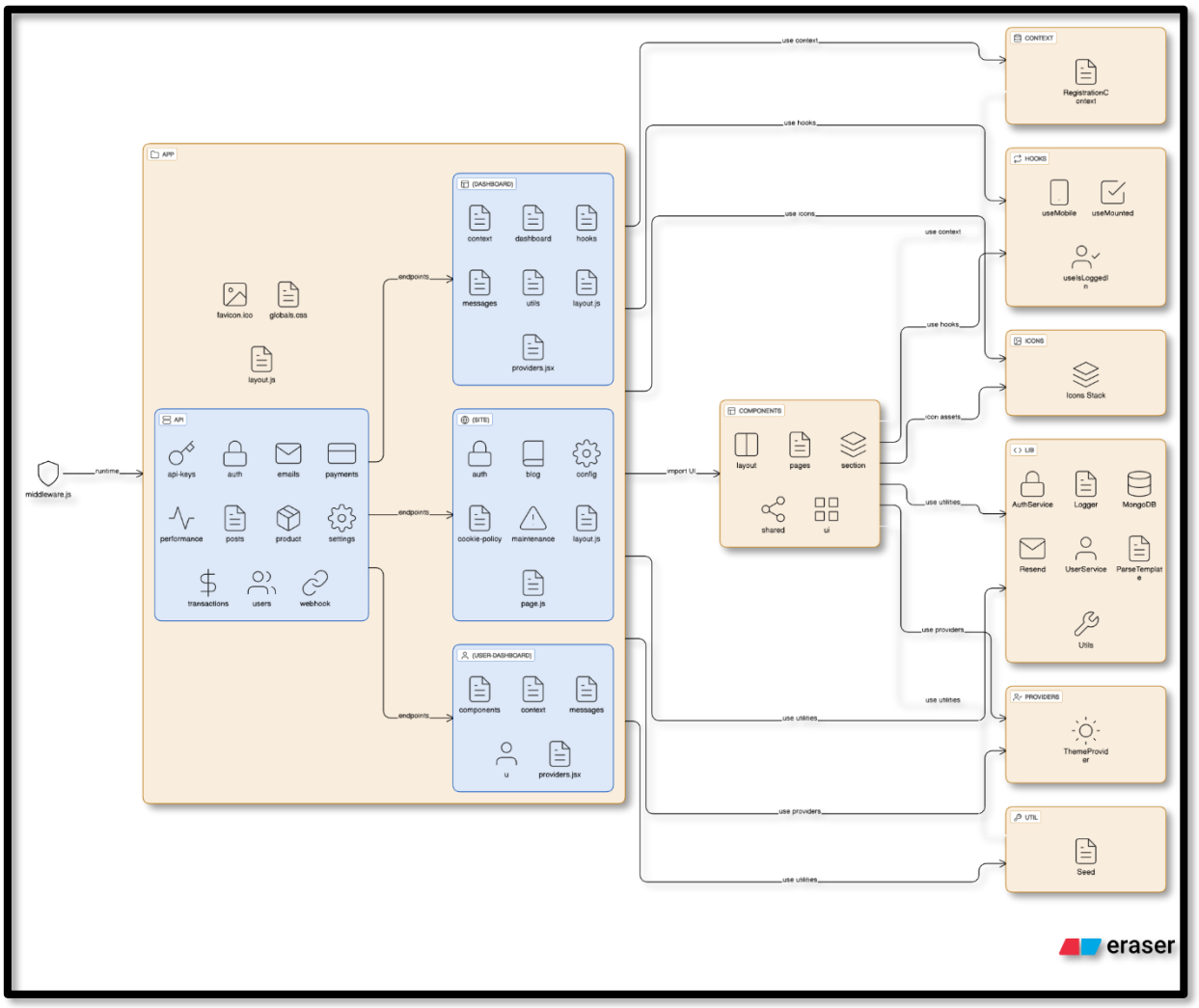


Рисунок 3.1 – Діаграма зв’язків файл/модуль

Для управління конфігураціями додатків та параметрами користувачів платформа використовує централізовану систему збереження даних у MongoDB, що забезпечує гнучке управління схемами даних і можливість масштабування при зростанні кількості користувачів (див. рис. 3.2).

Інтеграція бази даних MongoDB у проєкті реалізована через офіційний клієнт MongoClient з пакету mongodb, що забезпечує стабільний та масштабований доступ до даних. Підключення виконується на основі URI, який зберігається в середовищі .env.local, що підвищує безпеку та дозволяє легко змінювати конфігурацію без редагування коду.

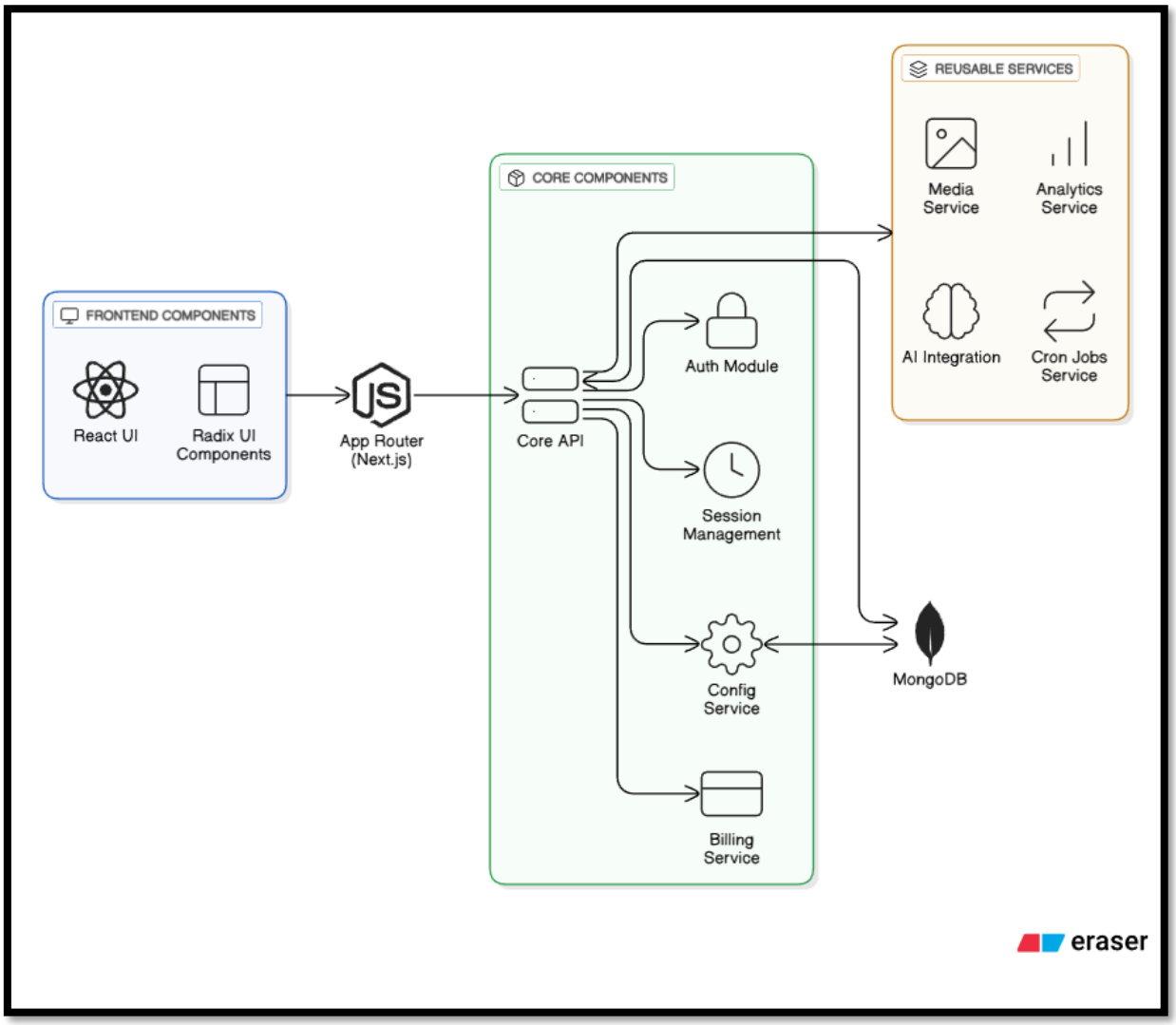


Рисунок 3.2 – Діаграма компонентів універсальної вебплатформи для бізнес-додатків

Механізм повторного використання існуючого клієнта через глобальну змінну _mongoClientPromise запобігає створенню надмірних підключень при гарячому перезавантаженні Next.js, завдяки чому робота з БД залишається ефективною навіть у режимі розробки. Функція connectToDatabase повертає доступ до клієнта та обраної бази, що робить її універсальним входом до всіх операцій CRUD у платформі – вставки, оновлення, читання та видалення даних (див. рис. 3.3).

Такий підхід закладає основу для побудови надійних модулів управління бізнес-додатками та забезпечує гнучку взаємодію сервера з даними у реальному часі.

```
import { MongoClient } from "mongodb";

const uri = process.env.MONGODB_URI;

if (!uri) {
  throw new Error("MONGODB_URI is not defined in .env.local");
}
const options = {};
let client;
let clientPromise;

if (global._mongoClientPromise) {
  clientPromise = global._mongoClientPromise;
} else {
  client = new MongoClient(uri, options);
  global._mongoClientPromise = client.connect().catch((err) => {
    console.error("MongoDB connection error:", err);
    throw err;
  });
  clientPromise = global._mongoClientPromise;
}
export async function connectToDatabase() {
  const client = await clientPromise;
  const db = client.db();
  return { db };
}
```

Рисунок 3.3 – Інтеграція бази даних MongoDB що реалізовано у проєкті

Фронтенд платформи реалізований на основі React і компонентів Radix UI, що дозволяє створювати адаптивний і сучасний інтерфейс, який можна легко налаштовувати під різні типи бізнес-додатків. Для організації структури компонентів та маршрутизації застосовується App Router Next.js, що дозволяє інтегрувати серверні та клієнтські компоненти в єдину систему і забезпечує оптимальний час завантаження сторінок (див. рис. 3.4). Взаємодія між клієнтською частиною та сервером здійснюється через стандартизовані API-маршрути, що забезпечує передбачуваність та стабільність обміну даними.

Особливу увагу приділено реалізації сервісів, які можуть повторно використовуватися у різних додатках. Серед них виділяються сервіси для обробки медіафайлів, інтеграції з AI-модулями, аналітики користувацької активності та відкладених операцій за допомогою cron-завдань. Кожен сервіс реалізований як незалежний модуль, що дозволяє масштабувати платформу та

інтегрувати нові бізнес-додатки без зміни базового ядра. Завдяки цьому реалізація SaaS Core забезпечує гнучкість та швидкість розвитку нових продуктів, зменшуючи витрати часу на повторну розробку базової функціональності.

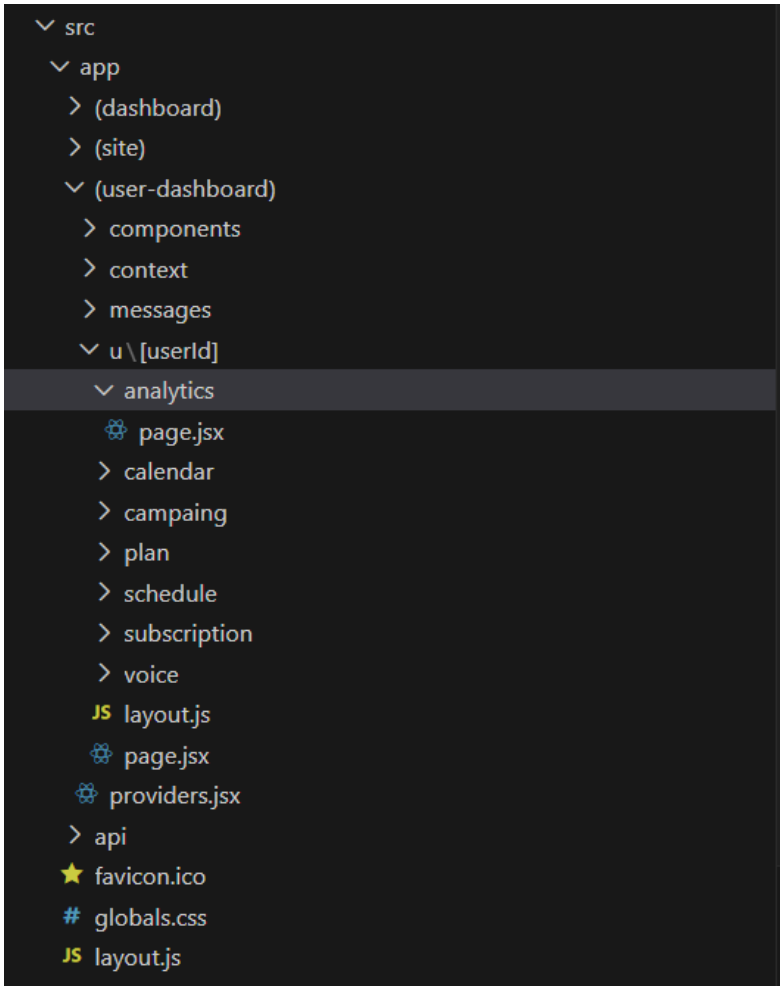


Рисунок 3.4 – Структура проєкту реалізованого за допомогою React та App Router

Ключовим аспектом реалізації є забезпечення безпеки та надійності системи. Центральна система прав доступу та логування дій користувачів дозволяє відстежувати активність, запобігати несанкціонованому доступу та підтримувати цілісність даних (див. рис. 3.5). Інтеграція з AI-модулями здійснюється через стандартизовані адаптери, що дозволяє забезпечити розширюваність та адаптивність платформи до нових алгоритмів без втручання у базовий код.

```
import { getToken } from "next-auth/jwt";
import { NextResponse } from "next/server";

const SECURITY_CONFIG = {
  // Routes that don't need any authentication
  PUBLIC_ROUTES: [
    "/",                    // Home page - EXACT match only
    "/auth/signin",         // Sign in page
    "/auth/signup",         // Sign up page
    "/blog",               // Public blog listing
    "/api/product",        // Public product catalog
  ],
  // Routes that require admin or owner role
  ADMIN_ONLY_ROUTES: [
    "/dashboard",      // Admin dashboard
    "/api/users/admins",   // Admin management
    "/api/users/",         // User analytics
    "/api/emails",         // Email management (if not user-specific)
  ],
  USER_SPECIFIC_ROUTES: {
    "/u": 2,          // /u/[userId] - ID is at position 3
    "/api/users": 3,       // /api/users/[userId] - ID is at position 3
    "/api/emails": 3,      // /api/emails/[userId] - ID is at position 3 (if
user-specific)
  },
```

Рисунок 3.5 – Комплексна система прав користовачів застосунку

Для наочного представлення архітектури універсальної вебплатформи та взаємозв’язків між її основними функціональними частинами доцільно використати діаграми компонентів UML. Дані діаграми відображають логічну структуру платформи, основні програмні модулі, їх відповідальність та способи взаємодії між собою (див. рис. 3.6 – 3.8).

Використання діаграм компонентів дозволяє узагальнити архітектурні рішення, продемонструвати модульність і масштабованість системи, а також спростити аналіз повторного використання сервісів при створенні бізнес-додатків на базі платформи.

Таким чином, реалізація універсальної вебплатформи SaaS Core створила основу для швидкого розвитку бізнес-додатків, забезпечуючи гнучкість, масштабованість і повторне використання сервісів та компонентів.

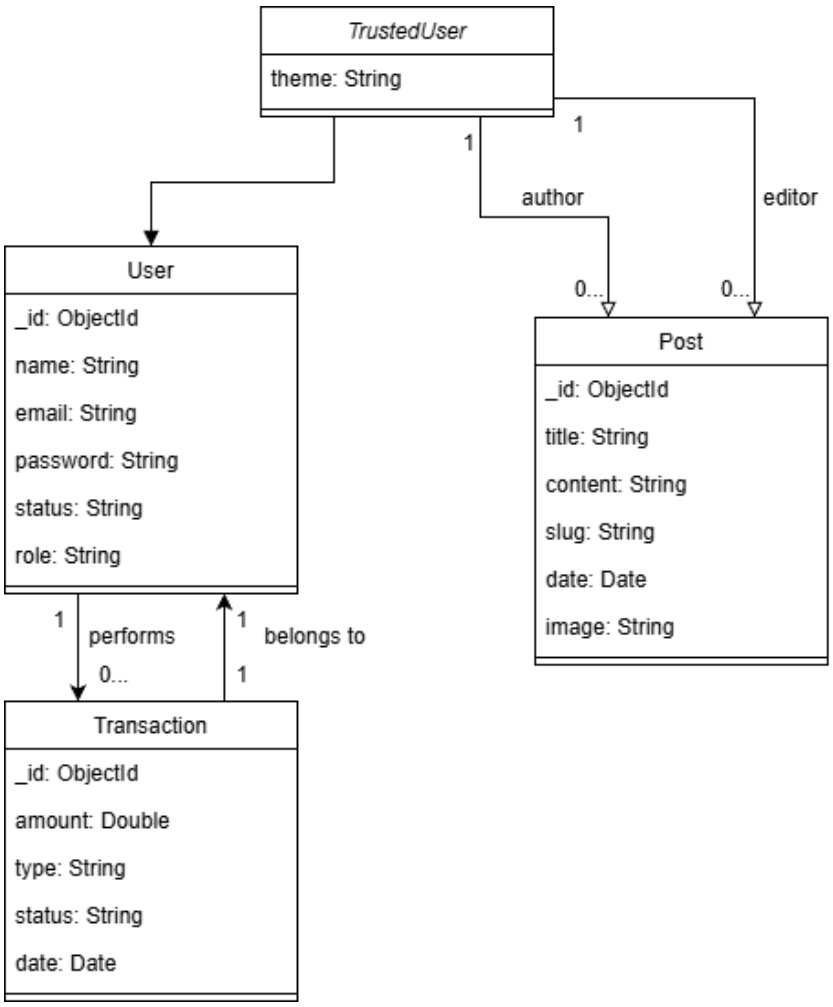


Рисунок 3.6 – Класи User, TrustedUser, Post та Transaction

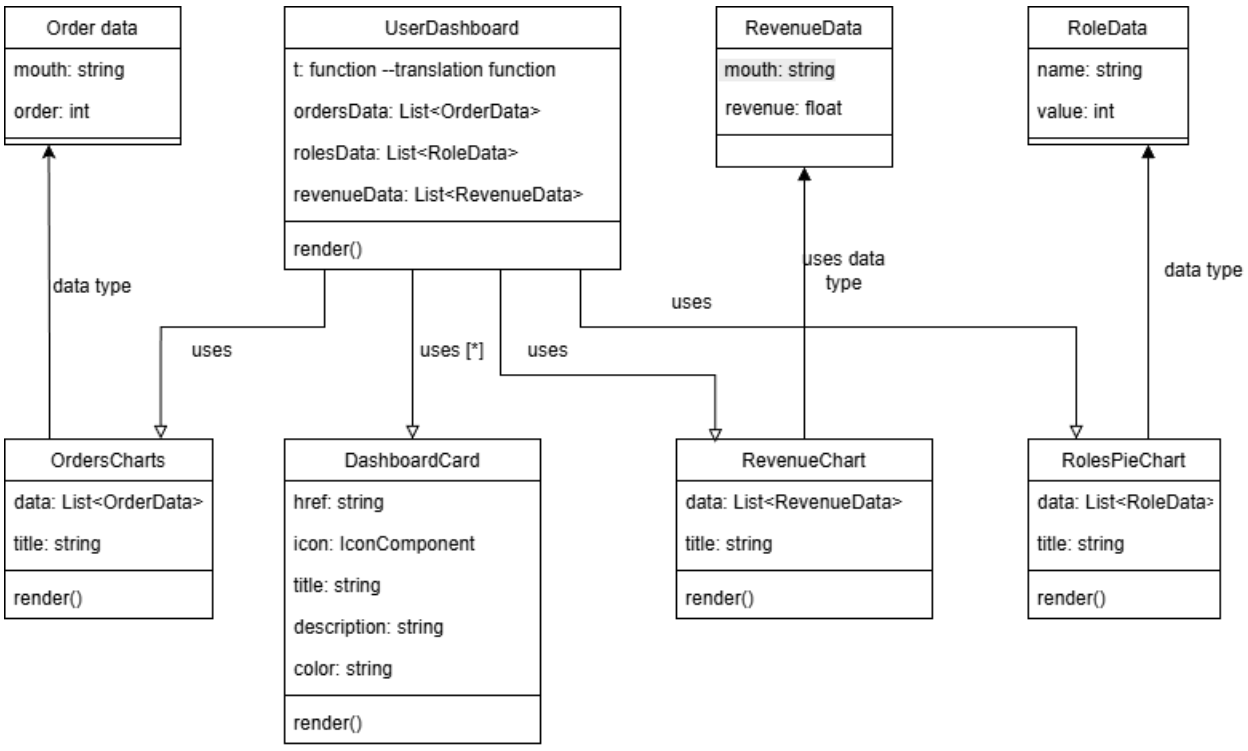


Рисунок 3.7 – Діаграма класів сторінки адміністратора

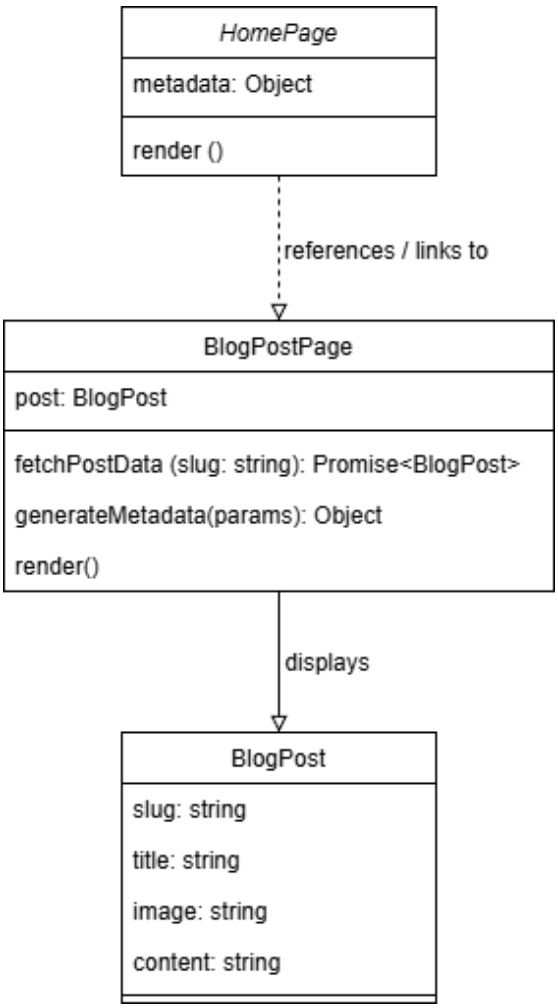


Рисунок 3.8 – Діаграма класів сторінки блогу

3.3. Реалізація бізнес-додатку «Планувальник контенту»

Реалізація бізнес-додатку «Планувальник контенту» була виконана на базі універсальної модульної платформи SaaS Core, що забезпечує повторне використання базових сервісів, стандартизованих API та компонентів інтерфейсу. Додаток призначений для користувачів, які ведуть активну діяльність у соціальних мережах, та поєднує класичні функції планування контенту з елементами гейміфікації для підвищення залученості і регулярності публікацій. Основною метою реалізації було створення інтерактивного, адаптивного та масштабованого інструменту, здатного забезпечити ефективне управління контентом та мотивувати користувачів до регулярної роботи з платформою.

Фронтенд додатку реалізований з використанням React та компонентів Radix UI, що дозволяє створювати сучасний і зручний інтерфейс, який адаптується до різних пристроїв і розмірів екранів. Для стилізації та анімації використовувалися Tailwind CSS та Framer Motion, що забезпечує плавний користувацький досвід та інтуїтивно зрозумілу взаємодію з функціоналом додатку. Основна увага приділялася організації інтерфейсу календаря публікацій, візуалізації прогресу завдань та відображенню системи нагород і балів, що складають ядро гейміфікаційного механізму.

На серверній стороні додатку використовувалися сервіси SaaS Core для автентифікації користувачів, управління ролями, обробки даних та інтеграції з AI-модулем. Це дозволяє користувачам отримувати персоналізовані рекомендації щодо часу публікацій, тематики контенту та хештегів, базуючись на аналітиці минулих публікацій та поведінки користувачів. Для зберігання даних застосовувалися MongoDB.

Для відображення логіки взаємодії між користувачем, інтерфейсом бізнес-додатку та сервісами універсальної платформи доцільно використати UML-діаграми послідовності. Дані діаграми ілюструють порядок виконання основних сценаріїв роботи бізнес-додатку, зокрема процеси створення, планування та оптимізації контенту, а також взаємодію з сервісами автентифікації, зберігання даних і модулями штучного інтелекту. Використання діаграм послідовності дозволяє наочно продемонструвати динамічну поведінку системи, узгодженість бізнес-логіки та коректність інтеграції бізнес-додатку з ядром універсальної платформи платформи (див. рис. 3.9).

Особливу увагу було приділено інтеграції гейміфікаційних механік у функціонал планувальника. Система нарахування балів, досягнень і рівнів мотивує користувачів дотримуватися регулярного графіку публікацій, а візуальна індикація прогресу дозволяє контролювати ефективність та залученість. Інтерактивні елементи були реалізовані таким чином, щоб вони не лише стимулювали активність, а й покращували загальний користувацький

досвід, роблячи процес управління контентом більш захопливим та ефективним.

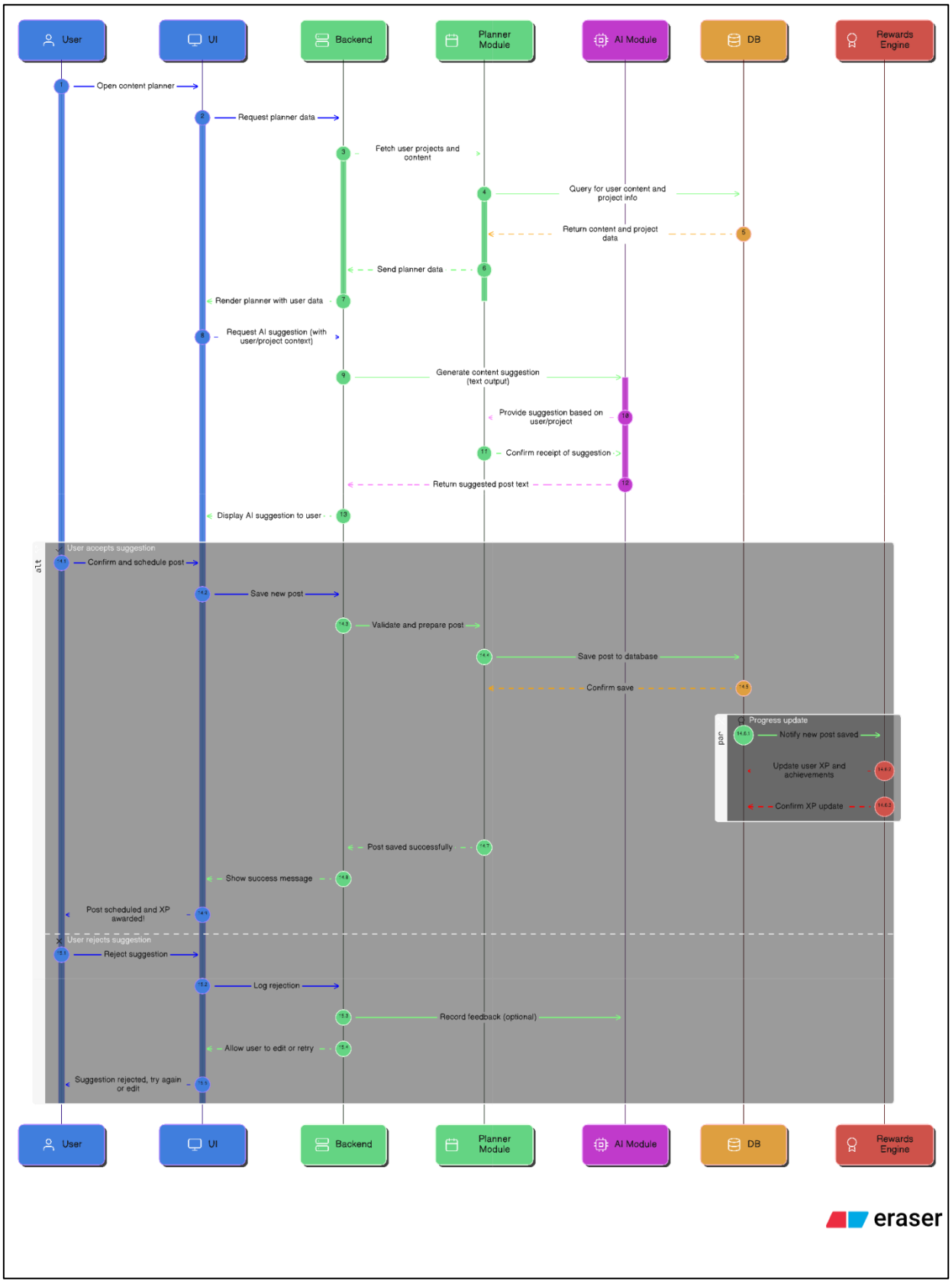


Рисунок 3.9 – Діаграма послідовності розроблюваного бізнес-додатку

Інтеграція AI здійснюється через стандартизовані адаптери, що взаємодіють із ядром платформи, забезпечуючи генерацію та оптимізацію контенту без необхідності зміни базових сервісів. Це дозволяє масштабувати функціонал додатку та додавати нові алгоритми без порушення архітектури. Розробка бізнес-логіки була виконана з урахуванням можливості подальшого розширення функцій, таких як аналітика ефективності контенту, автоматичне планування серій публікацій та інтеграція з іншими сервісами соціальних мереж.

На представлених скріншотах продемонстровано процес авторизації у бізнес-додатку «Планувальник контенту». Перший скріншот ілюструє основний екран логіну, де користувач може обрати метод входу: за допомогою облікового запису Google, GitHub або традиційним способом через електронну пошту (див. рис. 3.10). Другий скріншот відображає інтегрований процес авторизації через Google, включаючи перелік доступних акаунтів для швидкого входу (див. рис. 3.11). Дані зображення підкреслюють реалізацію багатомодального механізму автентифікації, що забезпечує зручність користувача та безпечний доступ до функціоналу додатку.

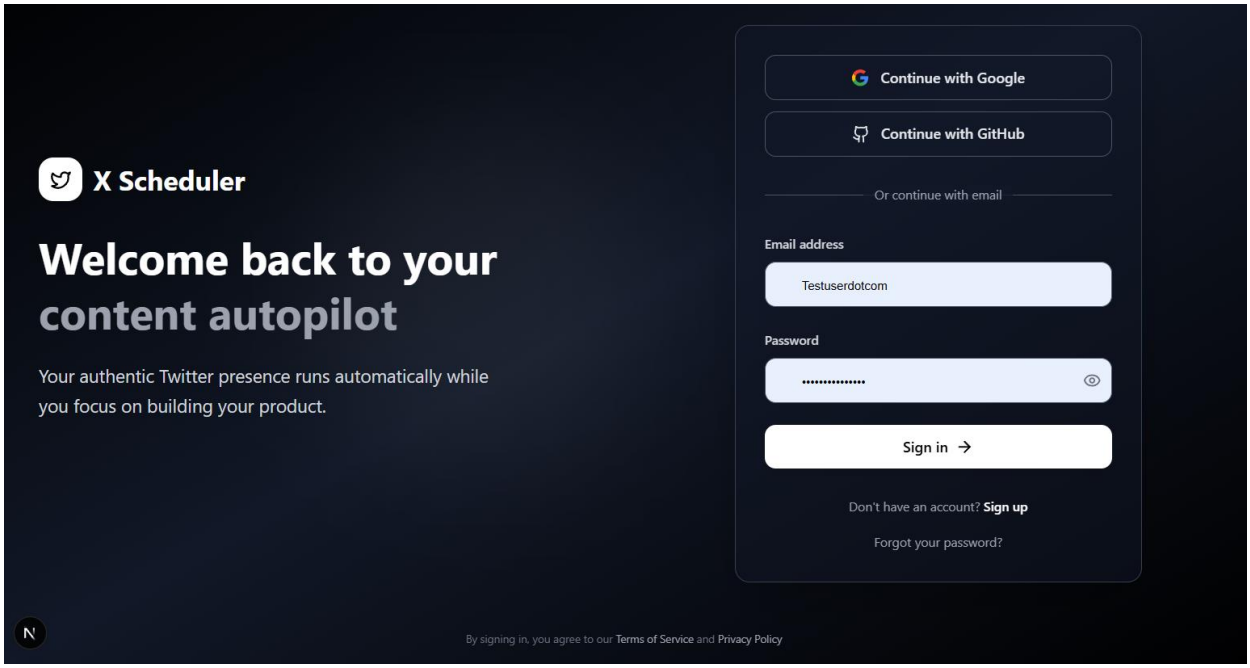


Рисунок 3.10 – Авторизація користувача у бізнес-додатку «Планувальник контенту»

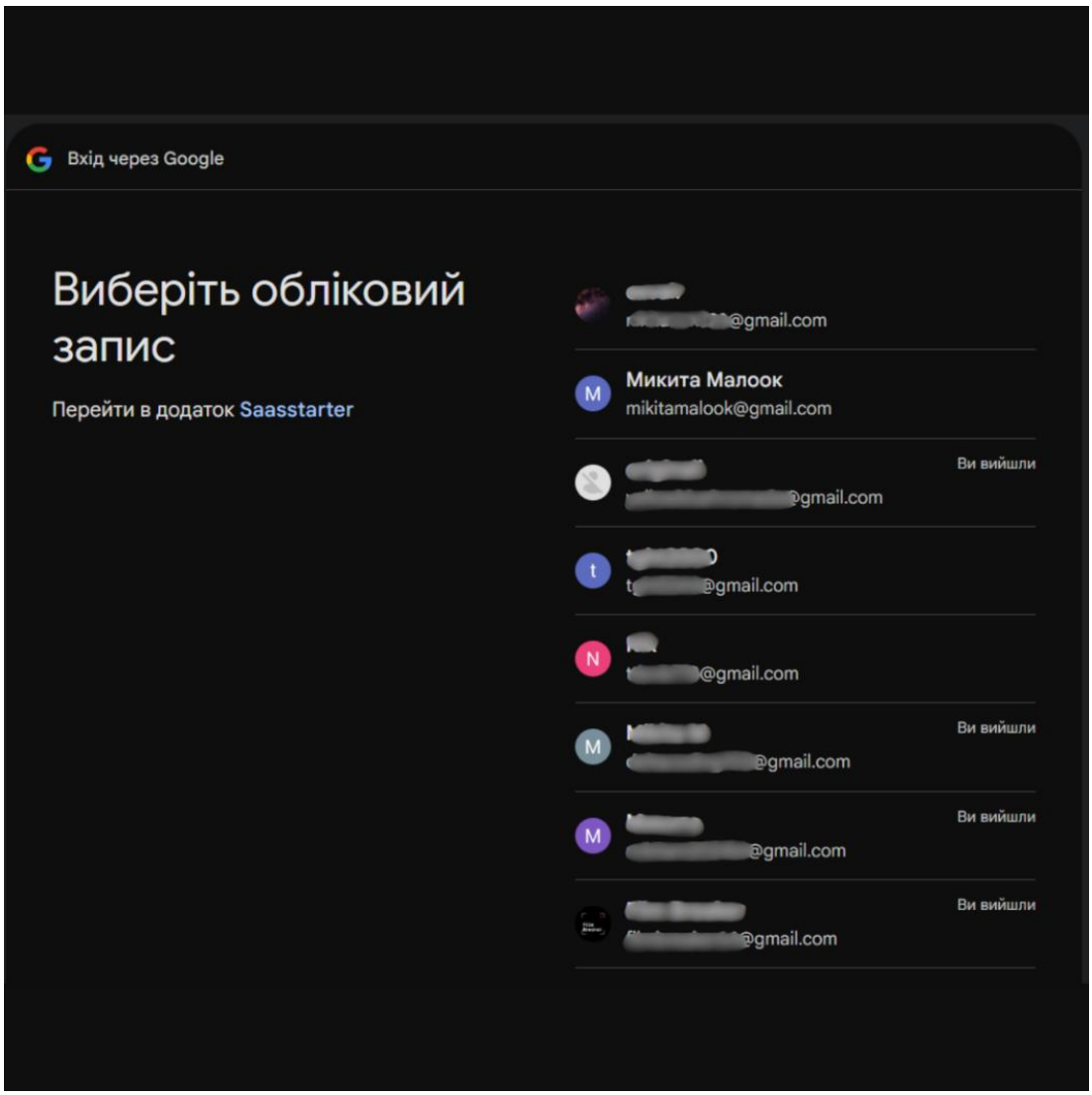


Рисунок 3.11 – Процес авторизації через Google Oauth

При вході в бізнес-додаток, користувача зустрічає головна панель керування додатку. Інтерфейс демонструє персоналізовану інформаційну панель користувача, де відображаються ключові показники продуктивності: кількість згенерованих публікацій, середній рівень залученості аудиторії, час, заощаджений завдяки автоматизації, та оцінка точності голосового помічника. Також представлений статус навчання унікальному стилю користувача та можливості швидкого генерування контенту, включно з створенням окремих публікацій, а також коментарів щодо галузевих тем. У верхній частині екрану розташована кнопка генерації контенту, а праворуч відображаються майбутні публікації, що дозволяє планувати роботу та моніторити активність (див. рис. 3.12).

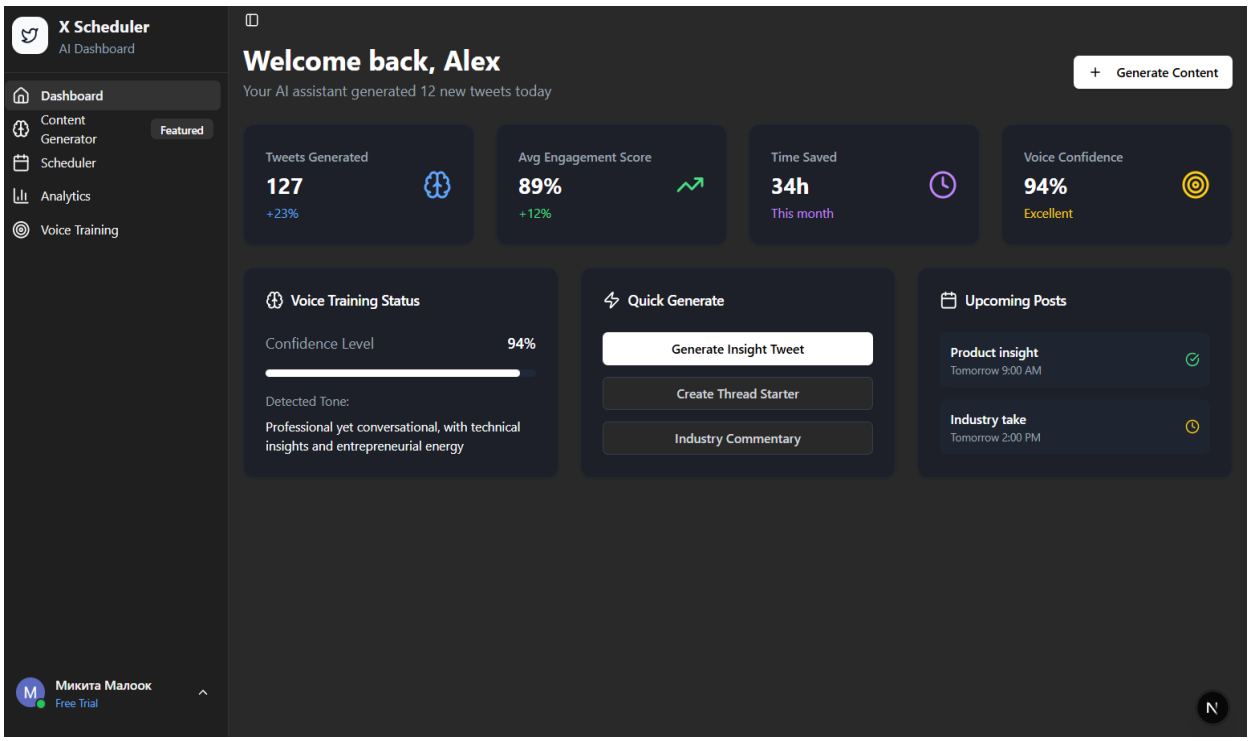


Рисунок 3.12 – Вид головної панелі керування бізнес-додатком

Наступний скріншот демонструє вкладку “Content Generator”, який використовується для планування та створення контенту (див. рис. 3.13).

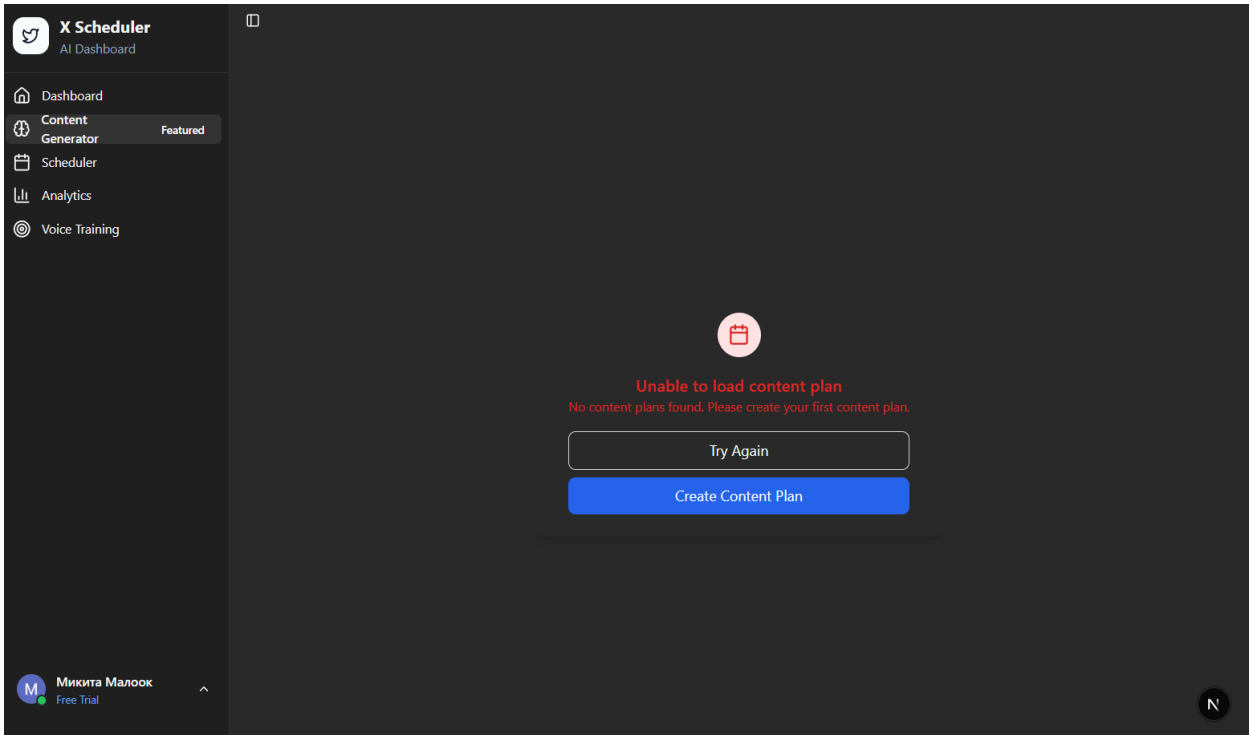


Рисунок 3.13 – Вкладка “Content Generator” з повідомлення про відсутність контент-планів

У центрі екрану відображається повідомлення про відсутність контент-планів та пропозиція створити перший план публікацій. Цей екран ілюструє функціональність, що спрямована на організацію та структурування контенту, а також забезпечує користувача інтуїтивними підказками для початку роботи з платформою.

Екран налаштування плану контенту є першим етапом у процесі формування контент-стратегії в додатку, що відображає підхід до керованого онбордингу користувача. Дизайн екрана виконано у темній палітрі з акцентом на мінімалізмі та функціональності. Перший крок ідентифікується як вибір тем, вимагає від користувача визначення принаймні п’яти ключових тем контенту для забезпечення різноманітності та послідовності публікацій на цільовій платформі. У поточній версії додатку, на вибір пропонується дев’ять категорій, орієнтованих переважно на ІТ-сферу та професійний розвиток, включаючи такі опції, як поради для розробників, новини галузі, навчальний шлях та поради щодо кар’єри. Кожна тема представлена карткою з іконкою, заголовком та коротким описом, що сприяє швидкому та інтуїтивно зрозумілому вибору (див. рис. 3.14). Цей функціонал підкреслює зосередженість системи на структурованому плануванні контенту та персоналізації рекомендацій для досягнення цілей, які будуть встановлені на наступному кроці “Set Goals” (Встановлення цілей).

Наступний скріншот фіксує стан успішного завершення першого етапу налаштування контент-стратегії. Користувач вибрав шість із дев’яти запропонованих тем, що візуально підтверджується яскравим виділенням карток і позначками (див. рис. 3.15). Критично важливо, що індикатор мінімального вибору в нижньому лівому куті тепер відображає текст “6/5 minimum” та супроводжується зеленим статусом “Ready to proceed” (Готово до продовження). Ця зміна не тільки підтверджує, що користувач виконав необхідну умову, але й активує кнопку для наступного кроку, відкриваючи шлях до наступного етапу. Таким чином, скріншот демонструє ефективну систему валідації введених даних та прогресивний перехід у процесі

онбордингу, забезпечуючи, щоб користувач розпочав формування стратегії, маючи достатньо диверсифіковану базу контентних напрямків.

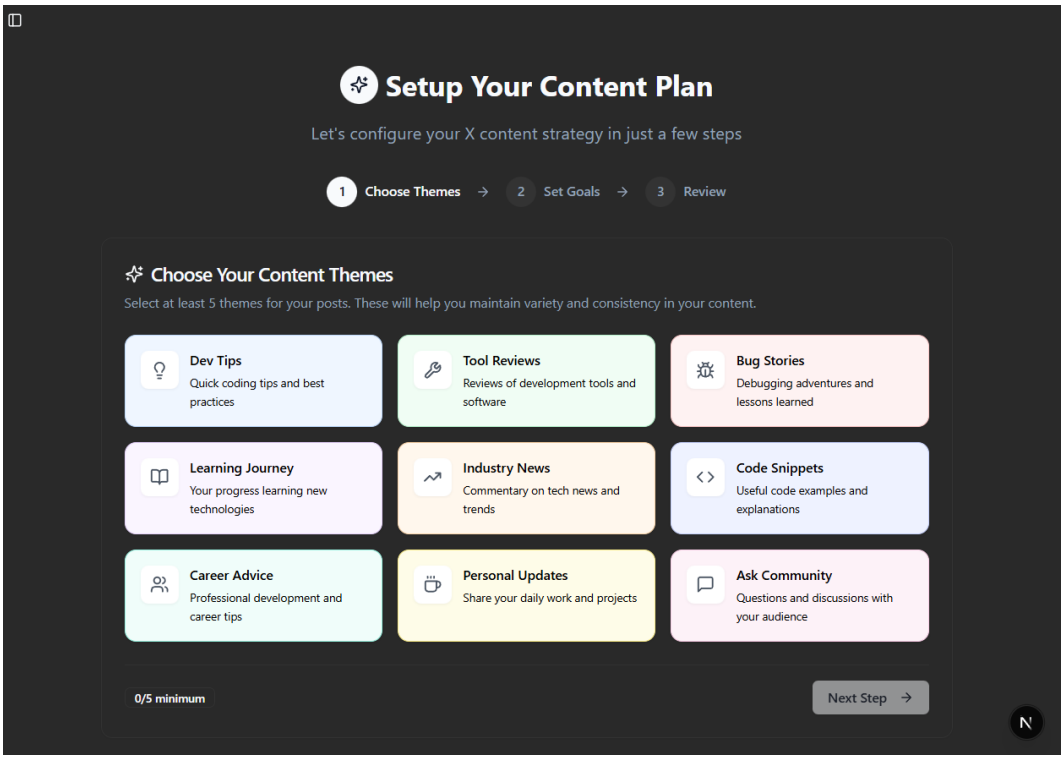


Рисунок 3.14 – Вкладка першого кроку налаштування плану контенту

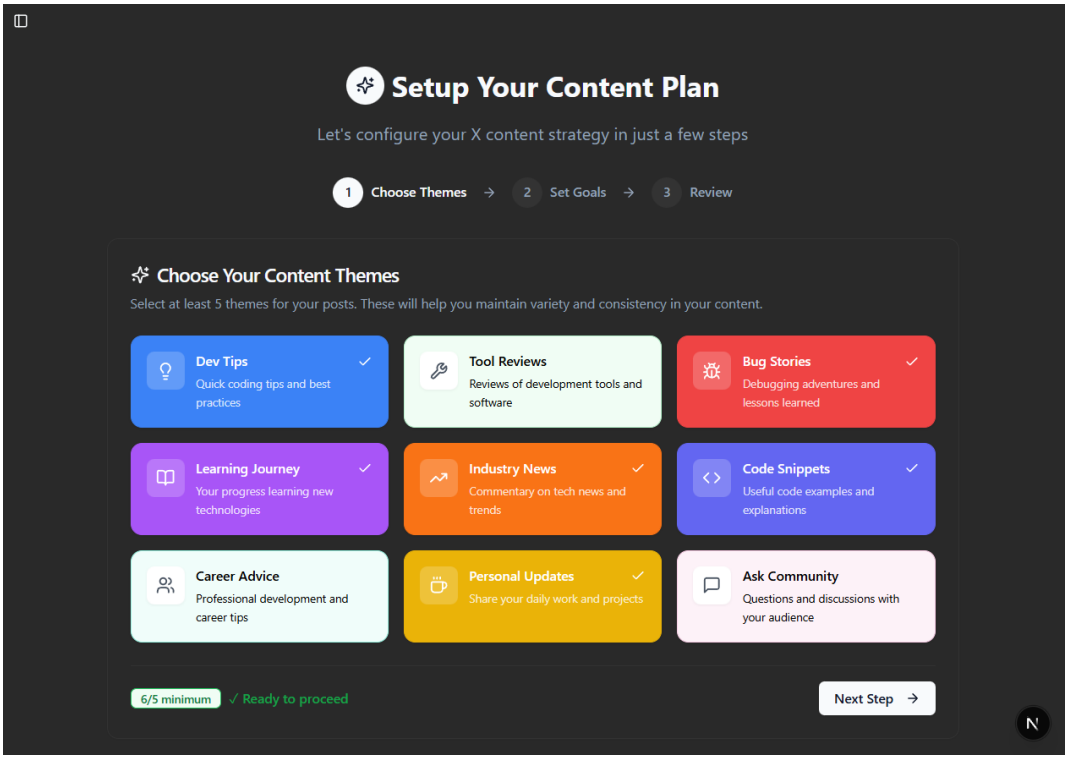


Рисунок 3.15 – Вкладка першого кроку налаштування плану контенту з вибраними темами

Наступний скріншот ілюструє перехід до другого етапу налаштування контент-стратегії, “Set Goals” (Встановлення цілей), який фокусується на кількісних метриках планування (див. рис. 3.16). У секції “Set Your Posting Goals” користувачеві надається можливість визначити частоту публікацій (Posts per day) та тривалість планування (Planning duration). Користувач обрав 2 пости на день (із рекомендованим діапазоном 2-3 для оптимальної роботи) та 1 місяць як часовий горизонт стратегії. Найважливішим елементом є автоматично згенероване підсумкове резюме “Your Content Plan Summary”. Воно інтегрує вибір із цього кроку (2 публікації на день, 1 місяць) із даними попереднього етапу (використання 6 різних тем). Результуючий розрахунок демонструє, що план передбачає створення 60 публікацій протягом наступного 1 місяця. Такий підхід забезпечує прозорість та конкретизацію цілей (SMART-критеріїв), що є необхідним кроком перед переходом до фінального етапу і завершального етапу “Review” (Перегляд), який активується кнопкою “Review Setup”.

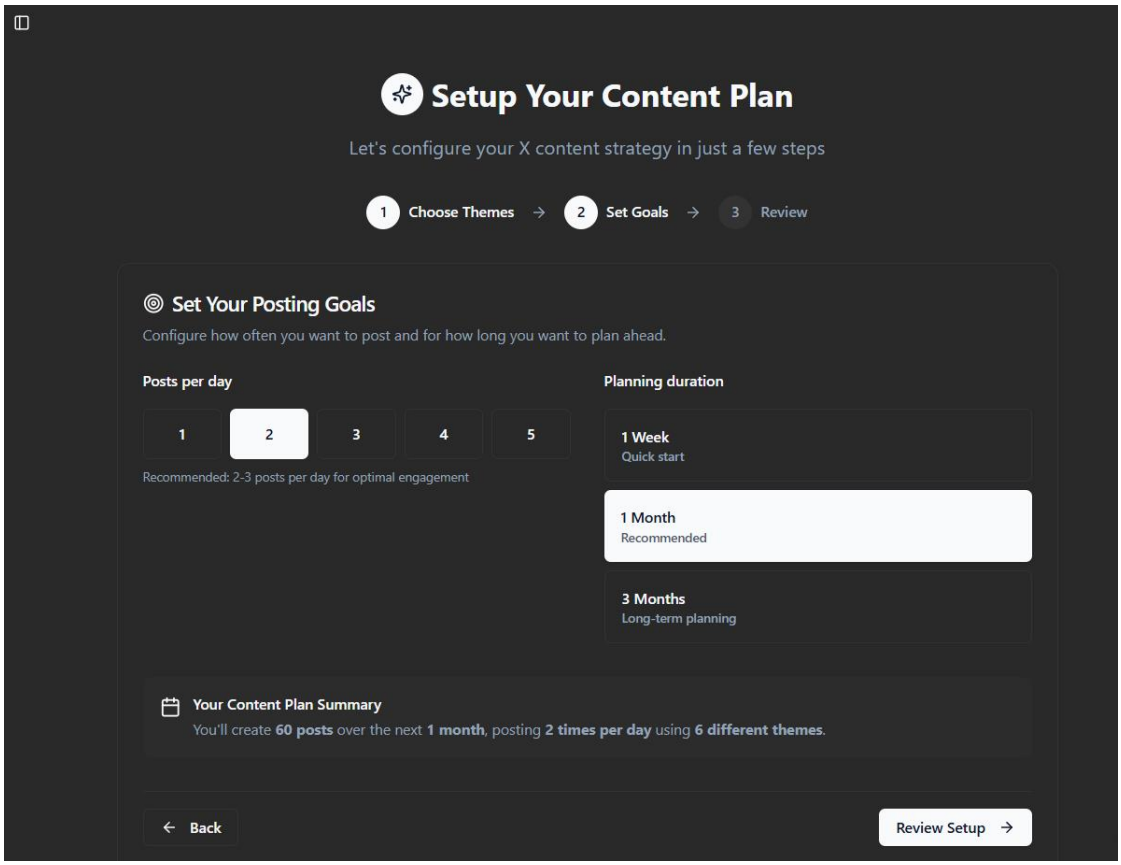


Рисунок 3.16 – Вкладка другого кроку налаштування плану контенту

Завершальним етапом налаштування є “Review” (Перегляд), який слугує для підтвердження та фіналізації контент-стратегії перед генерацією календаря (див. рис. 3.17). Екран “Review Your Setup” підсумовує всі попередні рішення, забезпечуючи користувачеві останню можливість перевірки. Дані чітко розділені на три блоки: вибрані теми, де перелічено шість обраних напрямків; графік публікацій із підтвердженою частотою 2 публікації на день; та тривалість планування, встановлена на 1 місяць (30 днів). Ключовим елементом є зелене повідомлення про успішну валідацію: “Ready to go! We’ll generate your content calendar starting from today, with 60 post slots distributed across your selected themes”. Це повідомлення остаточно підтверджує логічну послідовність та внутрішню узгодженість усіх налаштувань, інтегруючи частоту, тривалість та теми.

Фінальна кнопка “Create My Content Plan” активує генерацію кінцевого продукту системи – деталізованого календаря, завершуючи керований процес онбордингу та планування.

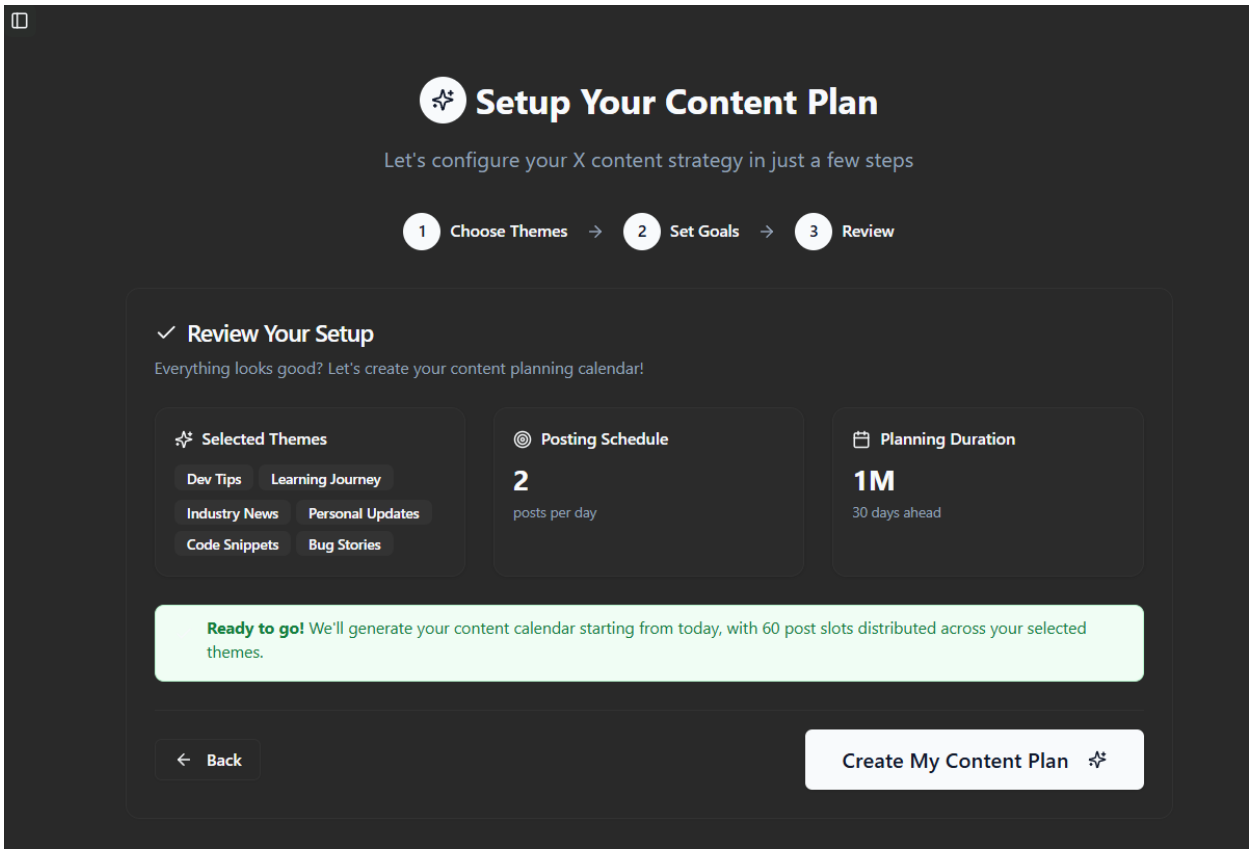


Рисунок 3.17 – Вкладка третього кроку налаштування плану контенту

Фінальним підтвердженням процесу налаштування контент-плану, що з’являється після натискання кнопки “Create My Content Plan” з попереднього етапу (див. рис. 3.18). Цей екран демонструє успішне завершення трикрокового процесу налаштування плану контенту. Фінальна кнопка “Go to Calendar” (Перейти до Календаря) є CTA (закликом до дії), який переводить користувача з процесу налаштування безпосередньо в робочий інтерфейс, де він зможе використовувати згенеровані 60 слотів для публікацій, що відповідають вибраним цілям та темам.

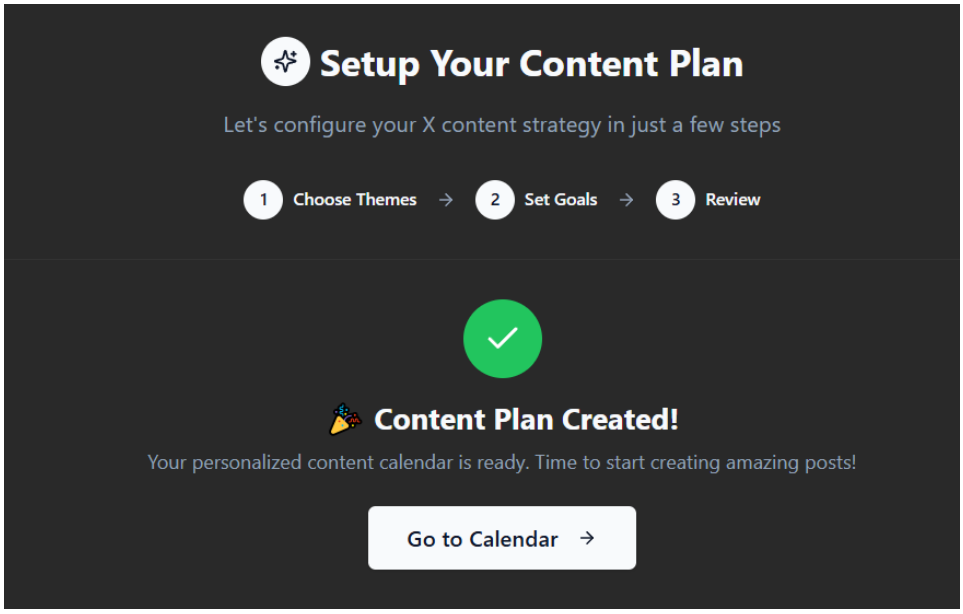


Рисунок 3.18 – Повідомлення про успішно створений контент-план

Далі представлено основний робочий інтерфейс додатку після успішного створення контент-плану (див. рис. 3.19). Екран відображає деталі плану: 2 публікації на день, на протязі 30 днів із загальним числом публікацій 60. Центральне місце займає календар. Поточний день, 6 грудня, виділено синім кольором, є початком кампанії. Всі дні в рамках плану відмічені двома крапками, які символізують необхідність двох публікацій на день. Праворуч від календаря реалізовано елементи гейміфікації та трекінгу: секція “Level Progress” (Прогрес користувача) показує, що користувач має досвід і знаходиться на певному рівні, мотивуючи до послідовного виконання плану. При виборі конкретного дню відображаються конкретні завдання на поточний

день. Нижче розташований індикатор “Today’s Goal” (Сьогоднішня ціль), який показує публікації у статусі “In Progress”. Цей інтерфейс ефективно інтегрує персоналізований контент-план у щоденний робочий процес, використовуючи календар і гейміфікацію для підтримки високої залученості та послідовності користувача.

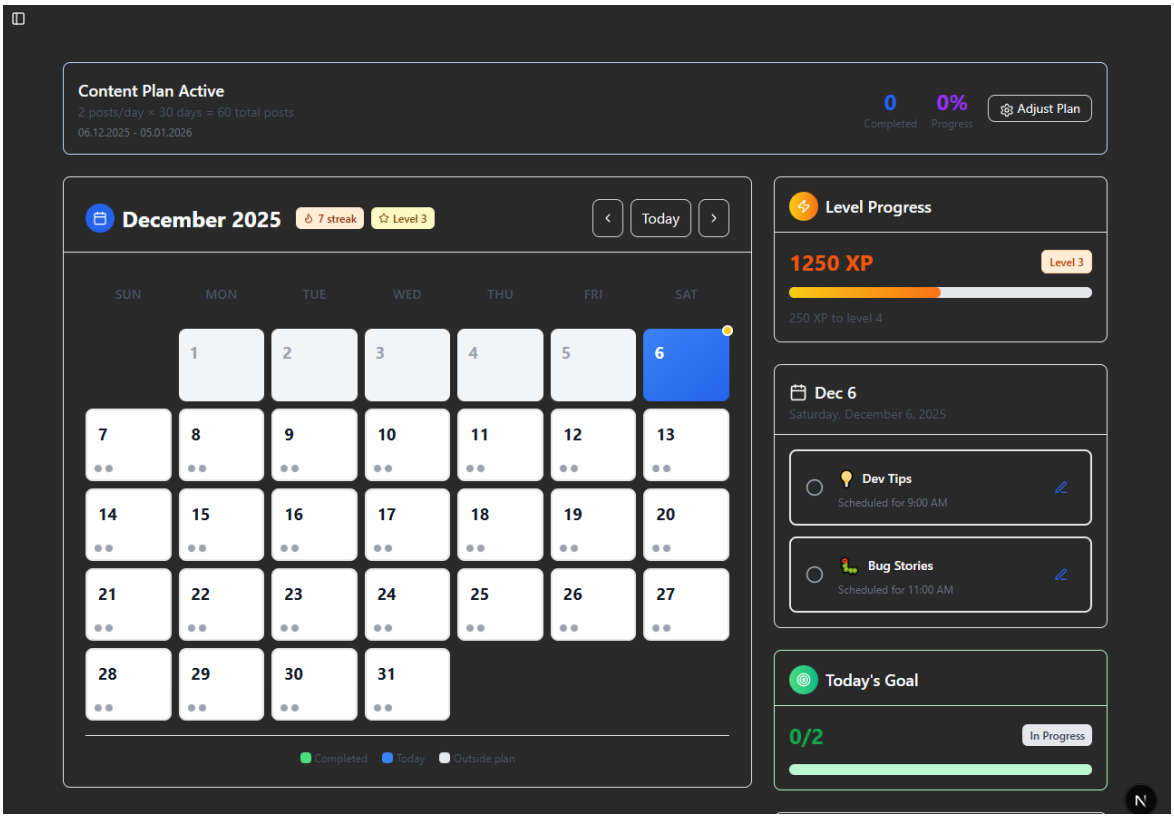


Рисунок 3.19 – Основний робочий інтерфейс додатку після успішного створення контент-плану

Наступний скріншот зображує інтерфейс календаря контент-плану (подібний до попереднього), але у стані активного виконання (див. рис. 3.20). У верхньому правому куті секція “Level Progress” показує, що користувач досяг 1950 досвіду за виконані публікації. Прогресія виконання відображена в календарі: дні з 7 по 11 грудня, а також день 18, тепер виділені зеленим кольором, що свідчить про успішне виконання щоденної норми. У нижній частині екрана, секція “Today’s Goal” (Сьогоднішня ціль) відображає статус “2/2 Complete!”, і супроводжується позитивним мотиваційним повідомленням. Користувач може виконувати завдання одразу або дотримуватися

згенерованого графіку. Це підтверджує ефективність інтеграції гейміфікації та візуального зворотного зв'язку, які підтримують постійну активність, підкреслюючи, що система автоматично планує контент, розподіляючи теми та підтримуючи послідовність відповідно до заданих користувачем цілей.

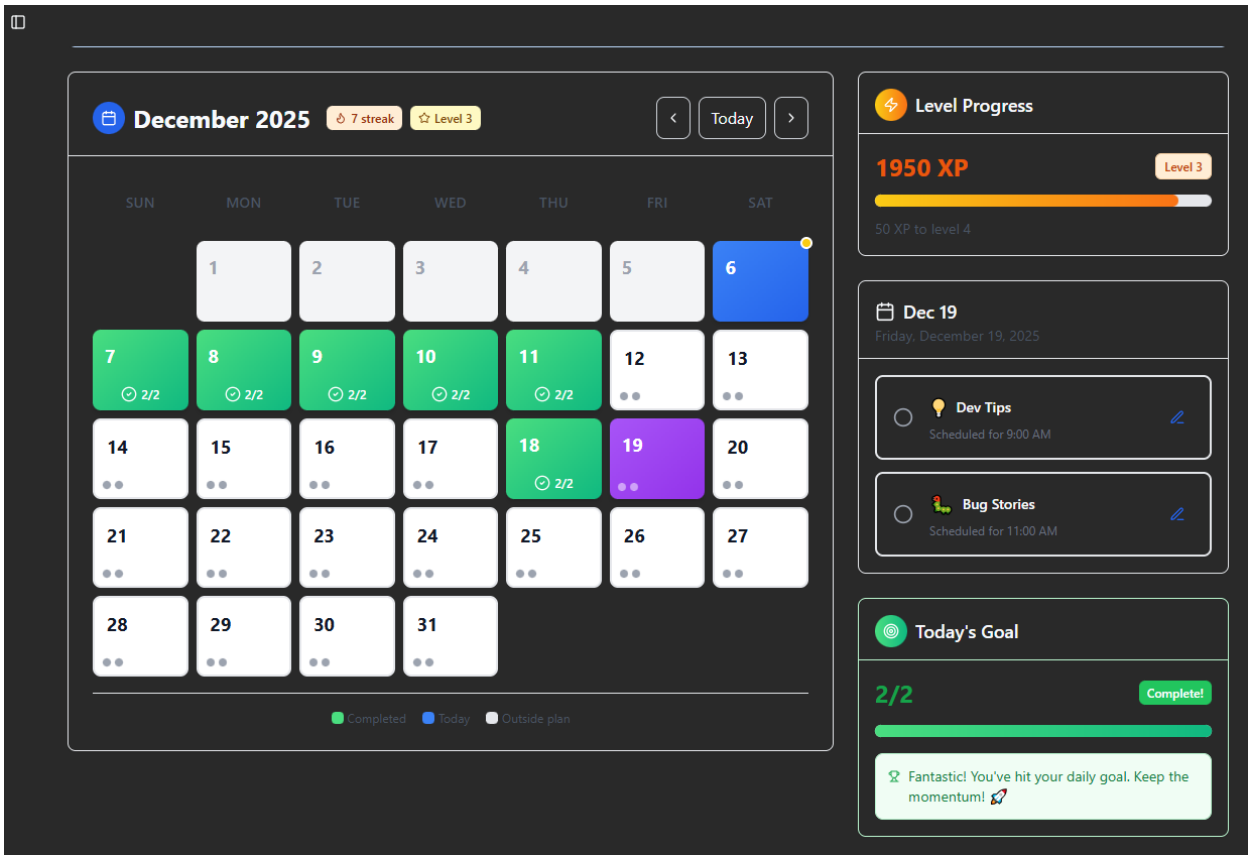


Рисунок 3.20 – Інтерфейс календаря контент-плану у стані активного виконання

Наступний скріншот зображує інтерфейс активного контент-плану на етапі виконання, ближче до кінця місячного циклу (див. рис. 3.21). У верхній частині екрана відображається загальний прогрес: 43 публікації завершено, що становить 72% від загальної кількості (60 постів), і цільовий період планування (06.12.2025 – 05.01.2026) залишається незмінним. У календарі візуалізація прогресу збережена: більшість днів, починаючи з 6 грудня. Картка завдань на 30 грудня показує, що одне завдання ("Code Snippets") вже завершене і позначене зеленим, а друге ("Personal Updates") заплановане на 11:00 AM і знаходиться в очікуванні. Індикатор "Today's Goal" підтверджує

поточний статус «2/2» і позначку “Complete!”, що вказує на те, що ціль на 30 грудня вже виконана, демонструючи систему відстеження ефективності в реальному часі та підтримуючи високий рівень залученості за допомогою елементів гейміфікації та чітких візуальних підтверджень успіху.

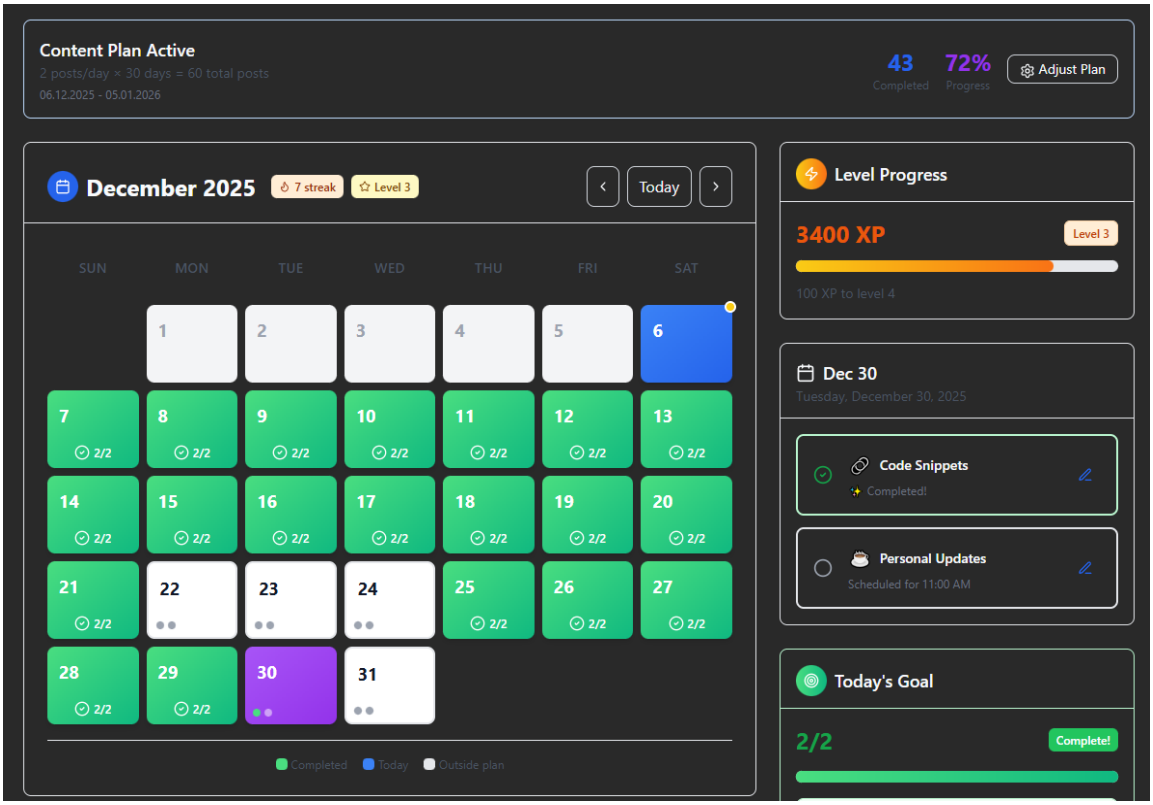


Рисунок 3.21 – Інтерфейс активного контент-плану на етапі виконання, ближче до кінця місячного циклу

Далі наведено інтерфейс редактора контенту та інтеграцію генеративного ШІ, який запускається після вибору слоту публікації в календарі (див. рис. 3.22). Екран пропонує користувачеві структуровані формати контенту, такі як “Quick Tip” (Коротка порада) та “Code Snippet” (Фрагмент коду). Це підтверджує, що обрані раніше теми використовуються для шаблонування процесу створення. Основна секція, “Your Post” (Ваша публікація), містить текст, який може бути введений вручну або згенерований за допомогою ШІ (кнопка “Generate with AI”). У цьому прикладі згенерована порада “Developer Tip – Write Cleaner Code in 3 Steps” містить чіткі інструкції, а також підтримує форматування Markdown.

Нижче відображається “Preview” (Попередній перегляд), який точно імітує, як публікація виглядатиме на цільовій платформі (у даному прикладі X), включаючи автоматично додані релевантні хештеги. У нижній частині розташовані кнопки “Save Draft” (Зберегти чернетку) та “Schedule Post” (Запланувати публікацію), що підкреслює гнучкість управління контентом.

Цей інтерфейс демонструє ключовий аспект автоматизації та підтримки контент-стратегії, поєднуючи планування, шаблонізацію та ШІ-генерацію для підвищення ефективності роботи користувача.

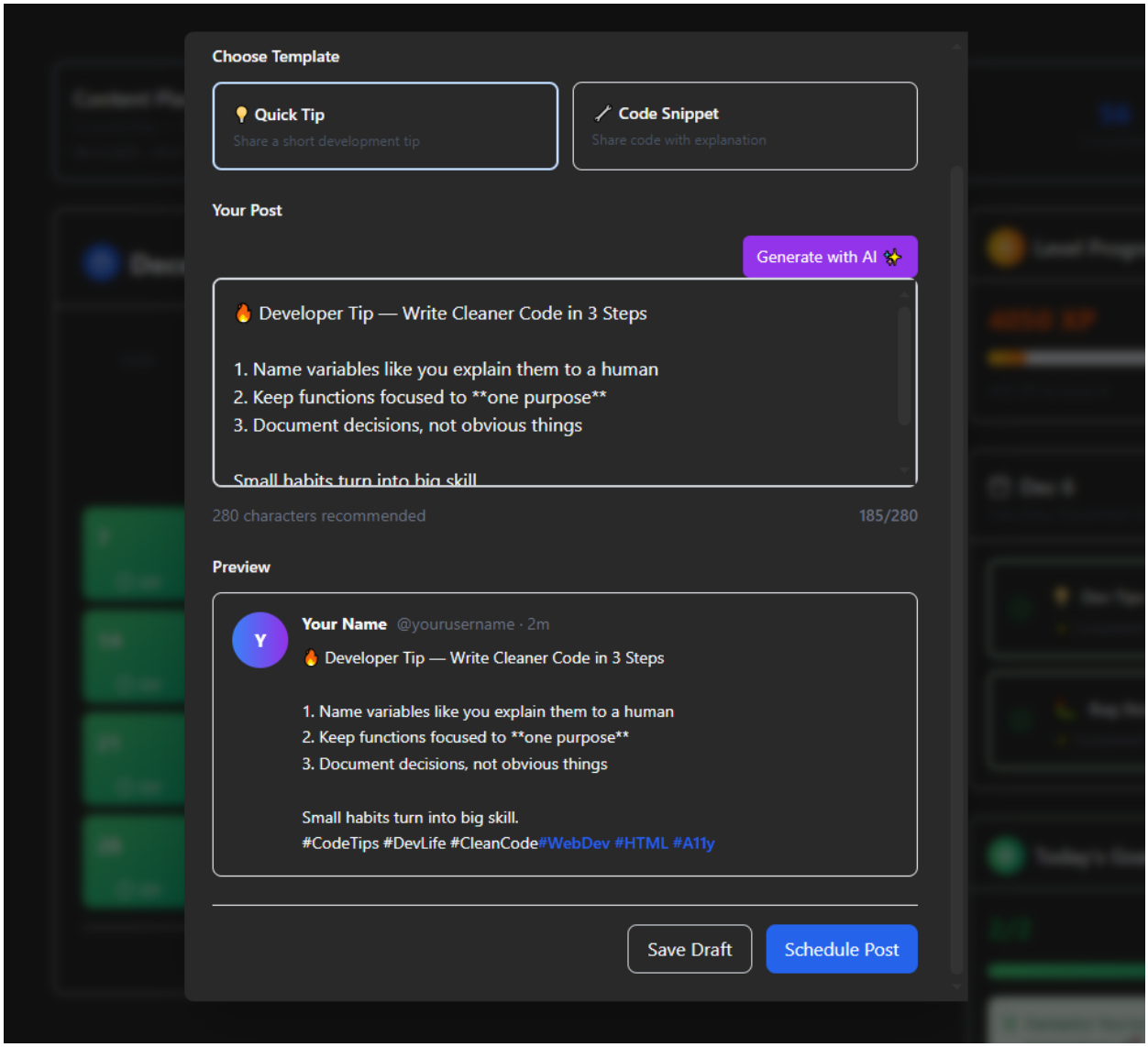


Рисунок 3.22 – Вкладка редагування публікації

Таким чином, реалізація бізнес-додатку «Планувальник контенту» демонструє практичну ефективність модульної платформи SaaS Core,

дозволяючи поєднувати базову інфраструктуру з інноваційними механізмами гейміфікації та AI-підтримки. Додаток забезпечує високу залученість користувачів, регулярність їхньої діяльності та готовність до подальшого розвитку та інтеграцій, що підтверджує універсальність і масштабованість розробленої платформи.

3.4. Інтеграція AI для генерації та оптимізації контенту

Інтеграція штучного інтелекту у бізнес-додаток «Планувальник контенту» є ключовим компонентом, який підвищує ефективність роботи користувачів та спрощує процес створення публікацій. Основна мета впровадження AI полягає в автоматизації рутинних завдань – генерації текстів, підборі хештегів, аналізі аудиторії та рекомендаціях щодо часу публікацій. Це дозволяє користувачам скоротити час на створення контенту, підвищити його якість і отримати доступ до інструментів, які раніше були доступні лише у професійних маркетингових командах.

Функціонально інтеграція AI включає генерацію текстового контенту, рекомендації щодо часу публікацій, підбір оптимальних хештегів та оцінку потенційного охоплення аудиторії. Модуль AI аналізує попередні дії користувачів, результати їхніх публікацій та взаємодії аудиторії, формуючи персоналізовані поради, які допомагають підвищити ефективність контент-плану. Такий підхід дозволяє не лише автоматизувати частину робочих процесів, але й створює основу для адаптивного та динамічного управління контентом, що відповідає сучасним потребам соціальних мереж та маркетингових стратегій.

На наведеному скріншоті показано результат роботи AI-модуля для генерації контенту на один і той самий запит. Користувач може спостерігати дві різні версії тексту, але зберігає загальну тематику та сенс запиту (див. рис. 3.23). Такий підхід дозволяє оцінити різноманіття варіантів публікацій, обрати найбільш підходящий або надихнутися для подальших редагувань, що

демонструє гнучкість і адаптивність інтегрованого AI-модуля в процесі створення контенту для соціальних мереж.

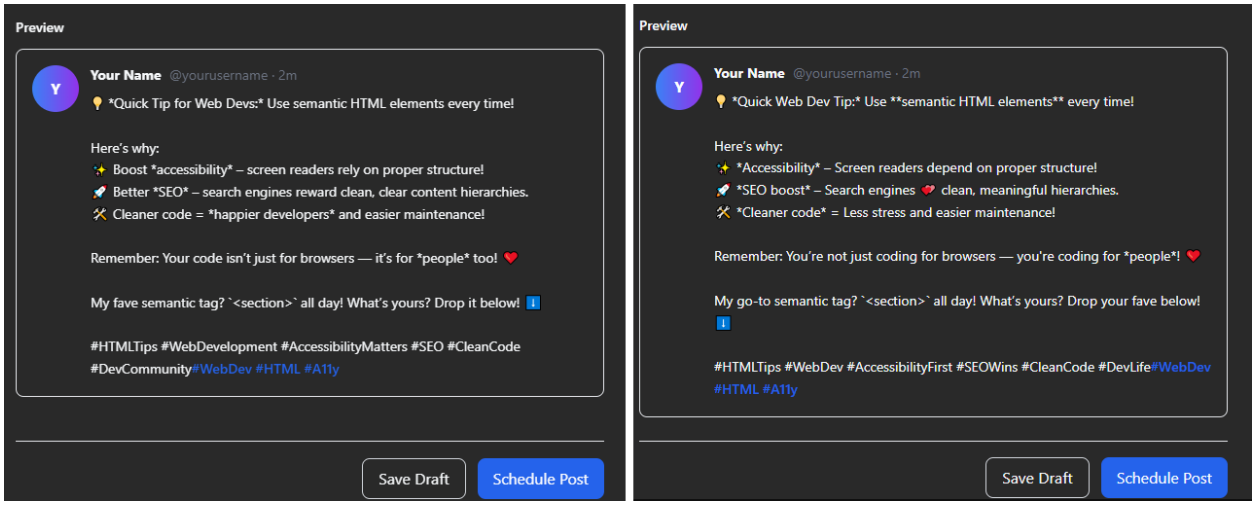


Рисунок 3.23 – Результат роботи AI-модуля для генерації контенту на один і той самий запит

AI-функціонал реалізований у вигляді окремого модулю платформи SaaS Core. Модуль працює як мікросервіс і взаємодіє з основним додатком через стандартизований REST API. Такий підхід забезпечує:

– можливість масштабування незалежно від інших модулів;
– швидке підключення нових моделей без змін у ядрі платформи;
– повторне використання модуля в інших бізнес-додатках.

Технічно інтеграція здійснюється через API-підключення AI-сервісів до бекенду додатку (див. рис. 3.24), що забезпечує обробку даних у реальному часі та взаємодію з базою даних MongoDB. Це дозволяє зберігати результати аналізу, формувати рекомендації та синхронізувати їх із інтерфейсом користувача. Завдяки використанню модульної архітектури платформи, AI-сервіси можуть підключатися до будь-якого бізнес-додатку, що створюється на базі SaaS Core, з мінімальними змінами у кодовій базі, що значно підвищує гнучкість та адаптивність системи.

Архітектурно система складається з таких компонентів:

– AI Service – окремий API, який приймає текстові запити та повертає результати генерації чи аналізу;

– SaaS Core Backend – посередник між інтерфейсом та AI-модулем, який керує запитами, зберігає результати в MongoDB, кешує відповіді;
– Frontend UI – інтерактивні інтерфейси користувача з візуальними підказками, формами та інструментами роботи з контентом.

Візуальна частина інтеграції AI реалізована через інтуїтивні інтерфейси у фронтенді додатку. Користувачі отримують рекомендації у вигляді підказок, графіків, прогрес-барів або динамічних підсказок щодо оптимізації контенту. Такий підхід забезпечує безшовну взаємодію з іншими функціями гейміфікованого планувальника та підвищує мотивацію користувачів дотримуватися регулярного графіку публікацій.

```
const client = new OpenAI({
  baseURL: "https://models.github.ai/inference",
  apiKey: token,
});
export async function POST(req) {
  try {
    const { task, payload } = await req.json();

    let userPrompt = "";

    switch (task) {
      case "generate":
        userPrompt = `Write a social media post about: ${payload.topic}. Make
it engaging and concise.`;
        break;
      case "optimize":
        userPrompt = `Optimize the following text for Instagram engagement:
${payload.text}`;
        break;
      case "hashtags":
        userPrompt = `Suggest 15 relevant hashtags for the topic:
${payload.topic}`;
        break;
      case "time":
        userPrompt = `Analyze the following user activity data and suggest the
best time to post: ${payload.data}`;
        break;
```

Рисунок 3.24 – API-підключення модулю ШІ

Модуль штучного інтелекту в додатку охоплює кілька ключових функціональних можливостей, спрямованих на автоматизацію та покращення процесу створення контенту. Передусім він забезпечує генерацію текстових публікацій: користувач вводить тему або короткий опис, після чого система формує оптимізовані варіанти постів для соціальних мереж. Додатково модуль виконує аналіз стилю, довжини та ключових слів, пропонуючи рекомендації щодо покращення тексту для підвищення залученості аудиторії. На основі тематики та актуальних трендів AI автоматично підбирає релевантні хештеги, а також пропонує оптимальний час публікації, враховуючи історичні дані користувача та загальні патерни активності. Окремо система може оцінювати потенційне охоплення, прогнозуючи рівень взаємодії залежно від структури та змісту посту. Такий комплекс інструментів дозволяє значно спростити роботу з контентом та підвищити його ефективність.

Отже, інтеграція AI у бізнес-додаток дозволяє поєднати автоматизацію, інтелектуальний аналіз та гейміфікаційні механіки, створюючи високоефективний інструмент для управління контентом у соціальних мережах. Використання штучного інтелекту підвищує практичну цінність додатку, сприяє залученості користувачів та демонструє ефективність застосування модульної платформи SaaS Core для створення сучасних вебдодатків.

3.5. Реалізація системи гейміфікації

Реалізація системи гейміфікації у бізнес-додатку «Планувальник контенту» є логічним продовженням процесу проєктування та спрямована на підвищення залученості користувачів, стимулювання регулярності дій та формування позитивного користувацького досвіду.

Гейміфікація у додатку реалізована для стимулювання послідовності та довгострокової залученості користувача до виконання встановленого контент-

плану. Цей механізм ґрунтується на двох ключових метриках: досвіді (XP) та рівнях (Levels) (див. рис. 3.25).

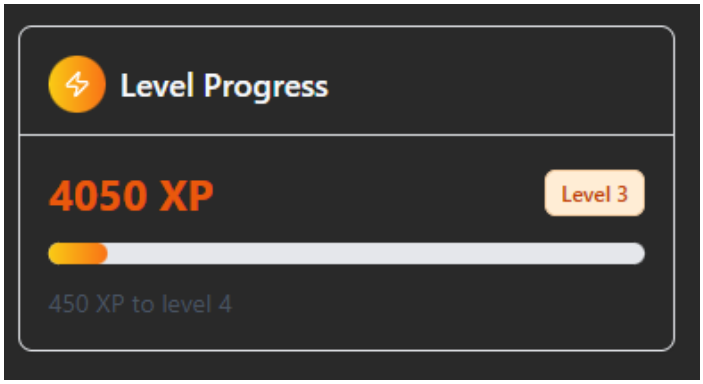


Рисунок 3.25 – Система гейміфікації реалізована через систему досвіду, що накопичується користувачем

Користувач отримує XP за успішне виконання щоденних цілей публікацій. Накопичення XP веде до підвищення Рівня (наприклад, з Рівня 3 до Рівня 4), що візуально відображається у секції “Level Progress”. Елемент гейміфікації підтримується через систему залученості, яка відстежує послідовну щоденну активність і заохочує користувача не переривати виконання плану. Таким чином, гейміфікація перетворює рутинне планування контенту на досяжну і вимірювану ціль, використовуючи внутрішню мотивацію та системи винагород для забезпечення сталості контент-стратегії протягом усього запланованого періоду (див. рис. 3.26).

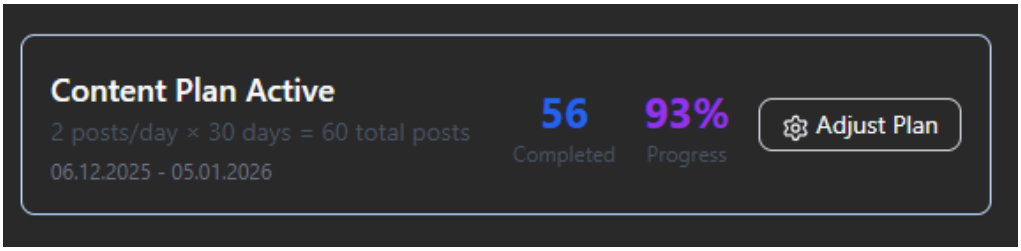


Рисунок 3.26 – Відстеження прогресу користувача протягом періоду контент-плану

Функціональна реалізація передбачає автоматичне відстеження дій користувача, таких як створення та публікація контенту, виконання завдань за

розкладом або досягнення певних результатів у системі. На основі цих дій система нараховує бали, віртуальні нагороди та підвищує рівень користувача, що стимулює регулярність використання додатку. Дані про активність зберігаються у структурованих моделях у MongoDB, що дозволяє проводити аналіз поведінки користувачів і формувати персоналізовані рекомендації щодо підвищення ефективності їхніх дій.

Інтерфейс користувача інтегрує гейміфікаційні елементи у вигляді прогрес-барів, бейджів, шкал досягнень та інших візуальних індикаторів, які демонструють прогрес виконання завдань та мотиваційні стимули. Анімаційні та інтерактивні компоненти реалізовані з використанням Framer Motion та Tailwind CSS, що дозволяє забезпечити плавність анімацій, адаптивність та інтуїтивність інтерфейсу. Користувачі отримують зворотний зв’язок у режимі реального часу, що сприяє формуванню звички регулярного використання планувальника та підвищенню ефективності контент-менеджменту.

Архітектура системи гейміфікації передбачає її масштабованість і гнучкість. Механізми нарахування балів, правил підвищення рівня та надання нагород реалізовані як окремі модулі, що дозволяє додавати нові елементи мотивації або змінювати існуючі правила без необхідності модифікації базового ядра платформи. Крім того, система гейміфікації взаємодіє з AI-модулем, що дозволяє формувати персоналізовані підказки, оцінювати ефективність дій та рекомендувати оптимальні стратегії публікацій на основі аналітики поведінки користувачів.

Таким чином, реалізація системи гейміфікації демонструє ефективність поєднання інноваційних мотиваційних механізмів із модульною архітектурою SaaS Core. Вона забезпечує підвищену залученість користувачів, регулярність виконання завдань та високий рівень інтерактивності додатку, що є ключовими чинниками успіху бізнес-додатку «Планувальник контенту».

4 ТЕСТУВАННЯ ТА ОЦІНКА ЯКОСТІ

4.1 Методологія тестування

Методологія тестування розробленої системи визначає підхід до перевірки функціональної та нефункціональної якості як універсальної вебплатформи SaaS Core, так і бізнес-додатку «Планувальник контенту». Вона ґрунтується на поєднанні кількох методів тестування, що дозволяють оцінити стабільність, продуктивність, безпеку та зручність користування системою в умовах реального використання. Основним завданням методології є забезпечення комплексної перевірки, яка дозволяє виявити помилки на ранніх етапах, підтвердити відповідність розроблених функцій вимогам та забезпечити високу якість кінцевого продукту.

Тестування здійснювалося за принципами модульного, інтеграційного та системного підходів. Модульне тестування передбачало перевірку окремих компонентів платформи та додатку на коректність виконання функцій та взаємодії з іншими елементами системи. Інтеграційне тестування забезпечувало оцінку роботи сервісів у комбінації, зокрема взаємодію фронтенд-компонентів з API, базою даних, системами автентифікації та AI-модулями. Системне тестування включало перевірку роботи всієї платформи у реальних умовах, що дозволяло оцінити ефективність взаємодії користувачів із функціоналом та продуктивність системи при збільшенні навантаження.

Особлива увага приділялася перевірці ключових функцій бізнес-додатку, включаючи планування контенту, взаємодію з календарем публікацій, генерацію рекомендацій AI, облік досягнень та нарахування балів. Тестування проводилося з урахуванням різних сценаріїв використання, що дозволяло виявити потенційні проблеми у взаємодії користувача з інтерфейсом, затримки у відображенні даних та невідповідності логіки системи очікуваним результатам.

Методологія також включала перевірку нефункціональних характеристик, таких як, масштабованість та безпека. Безпека оцінювалася шляхом перевірки механізмів автентифікації, контролю доступу та збереження конфіденційних даних. Такий комплексний підхід дозволяє забезпечити високий рівень надійності та стійкості системи, а також підтвердити відповідність платформи та додатку вимогам користувачів і сучасним стандартам розробки вебдодатків.

4.2 Тестування базової SaaS-платформи

Тестування базової SaaS-платформи було спрямоване на оцінку коректності роботи фундаментальних сервісів, що складають ядро платформи SaaS Core, і визначення рівня їх стабільності та надійності при виконанні ключових операцій. Основна мета цього етапу полягає в тому, щоб забезпечити повторне використання компонентів при створенні бізнес-додатків без виникнення критичних помилок, а також підтвердити відповідність базового функціоналу встановленим вимогам.

Процес тестування включав перевірку роботи механізмів автентифікації та управління користувачами, що реалізовані на основі NextAuth та JWT. Було проведено перевірку реєстрації нових користувачів, логіну та виходу із системи, контролю доступу до різних рівнів привілеїв і ролей, а також обробки сесій. Це дозволило оцінити стабільність роботи платформи в умовах одночасного доступу декількох користувачів та ефективність захисту даних.

Додатково тестування охоплювало перевірку функцій обробки даних і взаємодії з базою даних MongoDB. Були перевірені операції створення, читання, оновлення та видалення даних, а також коректність збереження і відновлення інформації в умовах підвищеного навантаження. Цей підхід забезпечив підтвердження надійності основних сервісів платформи та готовності їх до інтеграції з бізнес-додатками.

Інтерфейсні компоненти також проходили тестування для оцінки їх адаптивності та стабільності при взаємодії з різними частинами платформи. Було перевірено відображення елементів користувацького інтерфейсу, коректність навігації між модулями та взаємодії з API. Особлива увага приділялася перевірці компонентів, що повторно використовуються у бізнес-додатках, включаючи елементи управління контентом, візуальні індикатори та сервіси інтеграції з AI.

Таким чином, тестування базової SaaS-платформи дозволило підтвердити працездатність та стабільність фундаментальних компонентів, що забезпечують масштабованість, повторне використання сервісів і готовність платформи до подальшої інтеграції бізнес-додатків. Результати цього етапу створюють надійну основу для перевірки специфічних функцій додатків, які використовують ядро платформи, і служать основою для подальшого тестування функціоналу гейміфікованого планувальника контенту.

4.3 Тестування функцій бізнес-додатку (планувальника)

Тестування функцій бізнес-додатку «Планувальник контенту» було спрямоване на оцінку коректності реалізації ключових можливостей додатку, зокрема планування публікацій, інтеграції механік та роботи AI-сервісів для оптимізації контенту. Основна мета цього етапу полягала у підтвердженні відповідності додатку встановленим функціональним вимогам, забезпеченні надійності роботи системи під навантаженням і перевірці взаємодії користувачів з інтерфейсом у реальних сценаріях використання.

Під час тестування було перевірено функціонал календаря публікацій, який дозволяє створювати, редагувати та видаляти записи, а також автоматично відображати їх у відповідних часових слотах. Особлива увага приділялася синхронізації даних між клієнтською частиною та сервером, що забезпечує актуальність інформації та запобігає виникненню конфліктів при

одночасній роботі декількох користувачів. Взаємодія користувача з календарем оцінювалася на предмет зручності, швидкості реакції системи та коректності відображення даних у різних режимах перегляду.

Окрему увагу було приділено тестуванню гейміфікаційних механік, включаючи нарахування балів, підвищення рівнів користувача, отримання віртуальних нагород та відображення прогресу. Тестування здійснювалося за допомогою моделювання типових сценаріїв користування, що дозволило перевірити правильність логіки нарахування винагород та ефективність мотиваційного впливу на користувачів. Результати тестування підтвердили, що інтегровані механізми гейміфікації коректно взаємодіють із ядром платформи та AI-модулем, забезпечуючи персоналізований та адаптивний користувацький досвід.

Також було проведено перевірку інтеграції AI для генерації та оптимізації контенту, включаючи формування рекомендацій щодо часу публікацій, тематики контенту та підбору хештегів. Тестування показало коректність роботи алгоритмів у реальному часі та їхню здатність адаптуватися до історичних даних користувача. Взаємодія AI з іншими сервісами додатку була оцінена на предмет стабільності, швидкості обробки запитів та точності результатів.

Результати тестування функцій бізнес-додатку демонструють високу надійність та стабільність роботи ключових компонентів, підтверджують відповідність розробленого функціоналу встановленим вимогам та забезпечують основу для ефективного використання додатку у реальних умовах. Успішне проходження тестів свідчить про практичну готовність гейміфікованого планувальника контенту до впровадження та його здатність забезпечувати інтерактивність, мотивацію користувачів і оптимізацію контенту.

4.4 Тестування продуктивності

Тестування продуктивності розробленої системи було спрямоване на оцінку здатності універсальної платформи SaaS Core та бізнес-додатку «Гейміфікований планувальник контенту» ефективно працювати під різними рівнями навантаження. Основна мета цього етапу полягала у визначенні стабільності роботи системи при одночасному використанні великої кількості користувачів, швидкості обробки запитів, затримок у взаємодії з базою даних та ефективності функцій, що взаємодіють із AI-модулем.

Процес тестування продуктивності включав проведення навантажувальних експериментів із імітацією одночасної роботи сотень користувачів, що виконують основні дії, такі як планування публікацій, редагування контенту, перегляд аналітики та взаємодія з гейміфікаційними механізмами. Було оцінено час відповіді системи на стандартні запити, швидкість відображення даних у інтерфейсі, а також затримки при взаємодії з базою даних MongoDB через Prisma та Mongoose. Особлива увага приділялася обробці одночасних запитів до AI-сервісів, які генерують рекомендації та оптимізують контент.

Додатково тестування включало оцінку ефективності кешування, балансування навантаження та оптимізації запитів до бази даних. Ці перевірки дозволили визначити вузькі місця у продуктивності системи та підтвердити, що архітектура SaaS Core здатна забезпечити масштабованість і стабільну роботу як базових сервісів, так і бізнес-додатків, створених на її основі. У процесі тестування також було оцінено вплив інтеграції AI та гейміфікаційних механізмів на загальну швидкодію системи, що дозволило підтвердити ефективність взаємодії між компонентами платформи.

Результати тестування продуктивності свідчать про те, що розроблена платформа здатна обробляти високі навантаження без критичних затримок та втрати функціональності. Система демонструє стабільність та надійність роботи, що є важливим чинником для впровадження бізнес-додатку у

реальних умовах та забезпечення позитивного користувацького досвіду при одночасній роботі великої кількості користувачів.

4.5 Оцінка якості розробленої системи

Оцінка якості розробленої системи включає комплексну характеристику функціональних, нефункціональних та користувацьких параметрів як універсальної платформи SaaS Core, так і бізнес-додатку «Гейміфікований планувальник контенту». Основна мета оцінки полягала у визначенні ступеня відповідності системи встановленим вимогам, надійності та ефективності її роботи, а також у підтвердженні готовності до практичного використання.

Функціональна якість системи оцінювалася на основі результатів тестування окремих модулів, інтеграції сервісів та взаємодії користувачів із функціоналом. Результати показали коректну роботу ключових компонентів, включаючи планувальник публікацій, модулі гейміфікації, AI-сервіси для генерації та оптимізації контенту, а також механізми автентифікації та управління ролями. Кожен модуль забезпечує стабільну та передбачувану поведінку системи, що підтверджує ефективність модульної архітектури SaaS Core та її здатність підтримувати різноманітні бізнес-додатки.

Нефункціональні характеристики оцінювалися з урахуванням продуктивності, масштабованості, безпеки та зручності користування. Навантажувальні тести підтвердили стабільну роботу платформи при одночасній взаємодії великої кількості користувачів, швидку обробку запитів та мінімальні затримки у відображенні даних. Перевірка безпеки засвідчила ефективність механізмів автентифікації, контролю доступу та збереження конфіденційної інформації, що є критично важливим для комерційного використання системи. Зручність користування була підтверджена інтуїтивністю інтерфейсу, адаптивністю компонентів та логічністю організації взаємодії з платформою та додатком.

Оцінка ефективності системи також включала аналіз інтеграції гейміфікаційних механізмів і AI-сервісів. Виявлено, що наявність елементів мотивації та персоналізованих рекомендацій сприяє підвищенню залученості користувачів та регулярності виконання завдань. Це підтверджує практичну користь інтеграції інтелектуальних та ігрових механізмів у бізнес-додатки, що створюються на базі SaaS Core.

Таким чином, проведена оцінка якості розробленої системи дозволяє зробити висновок про її високу функціональну і нефункціональну надійність, готовність до практичного застосування та ефективність впроваджених інноваційних рішень. Система відповідає сучасним вимогам веброзробки, забезпечує стабільну роботу під навантаженням та створює позитивний користувацький досвід, що підтверджує успішність реалізації проєкту і досягнення поставлених цілей.

ВИСНОВКИ

У процесі виконання кваліфікаційної роботи було розроблено універсальну вебплатформу для управління бізнес-додатками з використанням сучасного стеку технологій, включаючи Next.js, React, MongoDB та Prisma. Розроблена система дозволяє адміністратору керувати контентом, користувачами, SaaS-додатками, налаштуваннями вебсайту та інтегрованими сервісами через зручну адміністративну панель, що забезпечує масштабованість та повторне використання функціональних модулів.

На основі розробленої платформи було реалізовано бізнес-додаток «Гейміфікований планувальник контенту», призначений для організації публікацій у соціальних мережах із додатковими елементами мотивації та інтерактивності. Додаток включає календар публікацій, систему гейміфікації, механізми нарахування балів та нагород, а також інтеграцію AI для генерації рекомендацій та оптимізації контенту. Такий підхід забезпечує підвищену залученість користувачів, регулярність виконання завдань та ефективну організацію роботи з контентом.

У ході роботи було здійснено детальний аналіз вимог до платформи та бізнес-додатку, визначено їх функціональні та нефункціональні характеристики, розроблено архітектуру системи, структуру інтерфейсу та модель бази даних. Для зберігання динамічного контенту використано документоорієнтовану базу MongoDB, а для типізованої взаємодії з нею – ORM Prisma. Next.js дозволив реалізувати як серверний, так і клієнтський рендеринг, що забезпечує високу швидкість завантаження сторінок та ефективну SEO-оптимізацію.

Платформа включає реалізовані модулі автентифікації та управління ролями користувачів, керування контентом та продуктами, конфігурацію налаштувань вебсайту, підтримку багатомовності та інтеграцію з платіжними системами. Інтерфейс адміністративної панелі побудовано за компонентним

підходом React, що забезпечує гнучкість, повторне використання коду та можливість динамічного розширення функціоналу. Реалізація гейміфікаційних механізмів і AI-сервісів демонструє ефективне поєднання інноваційних технологій із модульною архітектурою SaaS Core, що дозволяє створювати адаптивні та персоналізовані бізнес-додатки.

У процесі розробки було вирішено низку технічних завдань: створення конфігураційного механізму на базі JSON, розділення привілеїв користувачів, зберігання і завантаження зображень, а також підготовка інтерфейсу до динамічного розширення модулів. Забезпечено адаптивність інтерфейсу для різних пристроїв та оптимізовано структуру сторінок для кращої продуктивності.

Фізичні характеристики вебплатформи:

– адміністративна панель містить понад 10 окремих сторінок для управління системою;
– кодова база налічує понад 8000 рядків коду;
– впроваджено понад 25 унікальних компонентів React.

Результати виконаної роботи свідчать, що створена платформа і бізнес-додаток повністю відповідають поставленим цілям та вимогам технічного завдання. Платформа SaaS Core забезпечує ефективну основу для подальшого створення інших бізнес-додатків, а реалізований гейміфікований планувальник контенту демонструє практичну користь інтеграції мотиваційних механізмів і AI у вебсервіси. Отримані знання, технічні рішення та досвід можуть бути використані для розробки масштабованих SaaS-продуктів, орієнтованих на управління цифровими ресурсами та підвищення залученості користувачів.

ПЕРЕЛІК ПОСИЛАНЬ

1. What is SaaS Enterprise Resource Planning (ERP)? URL: https://www.oracle.com/ua/erp/what-is-erp/saas-erp/ (дата звернення: 18.03.2025).

2. What Is Customer Relationship Management (CRM)? URL: https://www.ibm.com/think/topics/crm (дата звернення: 18.03.2025).

3. What Is ERP? URL: https://www.oracle.com/erp/what-is-erp/ (дата звернення: 30.03.2025).

4. React Official Website. URL: https://reactjs.org/ (дата звернення: 30.03.2025).

5. Angular Documentation. URL: https://angular.dev/overview (дата звернення: 30.03.2025).

6. Vue.js – The Progressive JavaScript Framework | Vue.js Introduction. URL: https://vuejs.org/guide/introduction (дата звернення: 30.03.2025).

7. Next.js Docs | Next.js. URL: https://nextjs.org/docs (дата звернення: 02.11.2025).

8. AI in Web Development: Breaking Down the Future. URL: https://nextjs.org/docs (дата звернення: 15.03.2025).

9. OpenAI Official API Documentation. URL: https://platform.openai.com/docs (дата звернення: 15.03.2025).

10. Representational State Transfer (REST). URL: https://roy.gbiv.com/pubs/dissertation/rest_arch_style.htm (дата звернення: 25.03.2025).

11. Microservices Design Patterns. URL: https://www.geeksforgeeks.org/system-design/microservices-design-patterns/ (дата звернення: 29.03.2025).

12. Tailwind CSS Documentation. URL: https://tailwindcss.com/docs (дата звернення: 14.03.2025).

13. Prisma ORM Documentation. URL: https://www.prisma.io/docs (дата звернення: 14.03.2025).

14. MongoDB Documentation. URL: https://www.mongodb.com/docs/ (дата звернення: 14.03.2025).

15. NextAuth.js OAuth. URL: https://next-auth.js.org/configuration/providers/oauth (дата звернення: 14.03.2025).

16. JavaScript MDN Web Docs. URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript (дата звернення: 31.05.2025).

17. Dashboard UI Design Guide | User Insights. URL: https://www.anoda.mobi/ux-blog/dashboard-ui-design-guide-best-practices (дата звернення: 31.05.2025).

18. Resend – Email API Documentation. URL: https://resend.com/docs (дата звернення: 31.05.2025).

19. Buffer: Social media management for everyone About Us. URL: https://buffer.com/about (дата звернення: 31.05.2025).

20. Hootsuite: Social Media Marketing and Management Tool. URL: https://www.hootsuite.com/platform/integrations (дата звернення: 31.05.2025).

21. Later: Influencer Marketing and Social Media Management About. URL: https://later.com/about/ (дата звернення: 31.05.2025).

22. METRICOOL : Social Media Management Tool. URL: https://metricool.com/ (дата звернення: 31.05.2025).

23. What Is No-Code Development? Building Apps Without Coding. URL: https://www.oracle.com/ua/application-development/no-code/ (дата звернення: 31.05.2025).

24. Building with Patterns: A Summary – MongoDB. URL: https://www.mongodb.com/company/blog/building-with-patterns-a-summary (дата звернення: 31.05.2025).

25. Multi-Tenancy Architecture – System Design. URL: https://www.geeksforgeeks.org/system-design/multi-tenancy-architecture-system-design/ (дата звернення: 31.05.2025).

26. Pattern: Microservice Architecture. URL: https://microservices.io/patterns/microservices.html (дата звернення: 31.05.2025).

27. Vercel Deployment Documentation. URL: https://vercel.com/docs (дата звернення: 12.11.2025).

28. Complete Guide to Clean Architecture. URL: https://www.geeksforgeeks.org/system-design/complete-guide-to-clean-architecture/ (дата звернення: 12.11.2025).

29. W3C. Web Content Accessibility Guidelines (WCAG). URL: https://www.w3.org/WAI/standards-guidelines/wcag/ (дата звернення: 05.11.2025).

30. IBM. AI in Software Development. URL: https://www.ibm.com/think/topics/ai-in-software-development (дата звернення: 31.05.2025).

31. RFC 7519 – JSON Web Token (JWT). URL: https://datatracker.ietf.org/doc/html/rfc7519 (дата звернення: 06.11.2025).