МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Харківський національний університет імені В. Н. Каразіна

Факультет: **ННІ Каразінський банківський інститут**
Кафедра: **Інформаційних технологій та математичного моделювання**
Спеціальність: **122 Комп'ютерні науки**
Освітня програма: **Комп'ютерні науки**

Група: АК-21М **денна форма навчання**

КВАЛІФІКАЦІЙНА МАГІСТЕРСЬКА РОБОТА

на тему:

ВИКОРИСТАННЯ СУЧАСНИХ ЦИФРОВИХ ТЕХНОЛОГІЙ У СФЕРІ ОСВІТИ ТА НАУКИ
ЗА НАКАЗОМ № 4601-5_3262 ВІД 15 вересня 2025 РОКУ

здобувача вищої освіти **Пархоменка Олега Андрійовича**

Робота допущена до захисту в ЕК
протокол кафедри ІТММ № __ від _____2025 р.

Завідувач кафедри ІТММ
к.п.н., доцент
_____ **Н.І. Стяглик**

Науковий керівник
к.ф.-м.н., доцент
_____ **Г.В. Макарова**

м. Харків 2025 р.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Харківський національний університет імені В. Н. Каразіна

Факультет навчально-науковий інститут "Каразінський банківський інститут"

Кафедра Інформаційних технологій та математичного моделювання

Рівень вищої освіти другий (магістерський)

Спеціальність 122 Комп'ютерні науки

Освітня програма Комп'ютерні науки

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ **Н. І. Стяглик**

Підпис ініціали, прізвище

"15" вересня 2025 року

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ (ПРОЄКТ)

Пархоменко Олег Андрійович

(прізвище, ім'я, по батькові студента)

1. Тема роботи Використання сучасних цифрових технологій у сфері освіти та науки

керівник роботи Макарова Ганна Валеріївна, кандидат фіз.-мат. наук, доцент

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від "15" вересня 2025 року № 4601-5_3262

2. Строк подання студентом роботи 24 листопада 2025 року

3. Перелік питань, які потрібно розробити:

У розділі 1: Теоретичні основи та аналіз цифрових платформ в освітньо-науковій сфері. Дослідити сучасні тенденції цифровізації освіти та науки, зокрема роль онлайн-платформ, хмарних сервісів. Проаналізувати спефифіку, класифікацію та значення цифрових освітніх ресурсів та платформ для їх обміну. Визначити ключові виклики та вимоги при розробці веб-платформ для обміну навчальними матеріалами.

У розділі 2: Дослідження та порівняльний аналіз існуючих платформ для обміну освітніми матеріалами. Провести огляд та детальний аналіз функціональних можливостей існуючих освітніх платформ. Дослідити архітектурні рішення, моделі монетизації та технологічні стеки, що використовують у провідних освітніх веб-ресурсах. Виявити переваги та недоліки існуючих рішень для обгрунтування доцільності та актуальності розробки власного програмного застосунку (аналогу).

У розділі 3: Практична розробка та реалізація прототипу освітнього застосунку для обміну навчально-науковими матеріалами. Сформулювати функціональні та нефункціональні вимоги до авторського освітнього застосунку. Спроєктувати архітектуру програмного рішення, включаючи структуру бази даних, API та дизайн інтерфейсу користувача. Описати процес розробки та реалізації. Провести тестування розробленого прототипу, продемонструвати його роботу та сформулювати перспективи подальшого розвитку.

4. План роботи

№ з/п	Назви етапів роботи
1	Вибір здобувачем теми кваліфікаційної бакалаврської роботи
2	Затвердження плану і завдання кваліфікаційної бакалаврської роботи
3	Здача кваліфікаційної бакалаврської роботи керівнику
4	Підпис кваліфікаційної бакалаврської роботи керівника
5	Підпис кваліфікаційної бакалаврської роботи у нормоконтролера
6	Допуск завідувачем кафедри до захисту кваліфікаційної бакалаврської роботи
7	Захист кваліфікаційної бакалаврської роботи

5. Дата видачі завдання 15 вересня 2025 року

Студент _____ О. А. Пархоменко
підпис ініціали, прізвище

Керівник роботи _____ Г. В. Макарова
підпис ініціали, прізвище

РЕФЕРАТ

НА КВАЛІФІКАЦІЙНУ МАГІСТЕРСЬКУ РОБОТУ

«ВИКОРИСТАННЯ СУЧАСНИХ ЦИФРОВИХ ТЕХНОЛОГІЙ У СФЕРІ ОСВІТИ ТА НАУКИ»

ПАРХОМЕНКА ОЛЕГА АНДРІЙОВИЧА

Дипломна робота містить: 58 сторінок, 4 таблиці, 15 рисунків, список літератури з 33 найменувань.

Об'єктом дослідження є процес впровадження та специфіка використання сучасних цифрових технологій у сфері науки та освіти.

Предмет дослідження є практичні підходи, методи та інструменти застосування цифрових технологій у навчально-науковій діяльності.

Метою кваліфікаційної магістерської роботи полягає в аналізі ролі сучасних цифрових технологій у сфері освіти та науки, дослідженні їхнього впливу на ефективність освітнього процесу та наукових досліджень. Окрему увагу буде приділено авторській розробці освітнього застосунку, яка розглядатиметься як практичний приклад використання сучасних цифрових інструментів.

Завданнями роботи є:

– у першому розділі дослідити теоретичні основи цифровізації освіти, проаналізувати класифікацію цифрових освітніх ресурсів та визначити ключові виклики при розробці платформ для їх обміну;

– у другому розділі провести порівняльний аналіз існуючих аналогів (на прикладі платформ «Всеосвіта» та «На Урок»), дослідити архітектурні підходи до зберігання даних та обґрунтувати доцільність створення власного агрегатора контенту;

– у третьому розділі спроєктувати та програмно реалізувати веб-застосунок з використанням стеку технологій (MongoDB, Express, React, Node.js) та Next.js, а також провести тестування розробленого продукту.

Актуальність дослідження обумовлена глобальними тенденціями цифровізації суспільства, що безпосередньо впливають на сферу освіти та науки.

Використання цифрових технологій стає невід’ємною частиною навчального процесу, наукових досліджень та міжнародної співпраці. Зростання потреби у дистанційному навчанні, електронних освітніх ресурсах та цифрових наукових базах робить питання інтеграції сучасних технологій особливо важливим. Дослідження цього процесу дозволить виявити можливості та обмеження цифровізації освіти й науки та сприятиме підвищенню їхньої якості та доступності.

За результатами дослідження розроблено повнофункціональний прототип веб-агрегатора освітніх матеріалів. Зроблено висновок, що використання сучасних цифрових технологій у сфері освіти та науки є ключовим фактором підвищення ефективності освітньо-наукової діяльності. Цифрові інструменти сприяють інтерактивності навчального процесу, розвитку дистанційної та змішаної освіти, удосконаленню наукових методів і підвищенню рівня доступності знань.

Практична новизна полягає у виявленні тенденцій розвитку цифрових освітніх платформ, порівняльному аналізі їхніх можливостей та у представленні власного аналогу як практичного інструменту, що може бути застосований у навчально-науковій діяльності.

Одержані результати можуть бути використані освітніми установами, викладачами, студентами та науковцями для впровадження сучасних цифрових технологій у навчальний процес, удосконалення існуючих освітніх платформ, а також для розробки нових продуктів, орієнтованих на підвищення якості освіти та ефективності наукових досліджень.

КЛЮЧОВІ СЛОВА: ЦИФРОВІЗАЦІЯ, СУЧАСНІСТЬ, ЦИФРОВІ ОСВІТНІ РЕСУРСИ, ДОСТУПНІСТЬ, АДАПТИВНІСТЬ, API.

ABSTRACT

AT QUALIFICATION MASTER WORK

«USE OF MODERN DIGITAL TECHNOLOGIES IN THE SPHERE OF EDUCATION AND SCIENCE»

PARKHOMENKO OLEH ANDRIYOVYCH

The thesis contains: 58 pages, 4 tables, 15 figures, a list of references of 33 titles.

The object study is the process of implementation and specifics of the use of modern digital technologies in the field of science and education.

The subject research is practical approaches, methods and tools for applying digital technologies in educational and scientific activities.

The purpose of the work is to analyze the role of modern digital technologies in the field of education and science, to study their impact on the effectiveness of the educational process and scientific research. Special attention will be paid to the author's development of an educational application, which will be considered as a practical example of the use of modern digital tools.

The tasks of a bachelor's degree are:

– in the first section, to explore the theoretical foundations of the digitalization of education, analyze the classification of digital educational resources and identify key challenges in developing platforms for their exchange;

– in the second section, to conduct a comparative analysis of existing analogues (using the example of the platforms "Vseosvita" and "Na Urok"), to explore architectural approaches to data storage and to justify the feasibility of creating your own content aggregator;

– in the third section, to design and programmatically implement a web application using a technology stack (MongoDB, Express, React, Node.js) and Next.js, as well as to test the developed product.

The relevance of the study is due to global trends in the digitalization of society, which directly affect the sphere of education and science. The use of digital technologies is becoming an integral part of the educational process, scientific research and international cooperation. The growing need for distance learning, electronic

educational resources and digital scientific databases makes the issue of integrating modern technologies particularly important. The study of this process will identify the possibilities and limitations of the digitalization of education and science and will contribute to improving their quality and accessibility.

According to the results of the research, a fully functional prototype of a web aggregator of educational materials was developed. It was concluded that the use of modern digital technologies in the field of education and science is a key factor in increasing the efficiency of educational and scientific activities. Digital tools contribute to the interactivity of the educational process, the development of distance and blended education, the improvement of scientific methods and an increase in the level of accessibility of knowledge.

Practical novelty lies in identifying trends in the development of digital educational platforms, comparative analysis of their capabilities, and presentation of our own analogue as a practical tool that can be used in educational and scientific activities.

The results obtained can be used in educational institutions, teachers, students, and scientists to introduce modern digital technologies into the educational process, improve existing educational platforms, and develop new products aimed at improving the quality of education and the effectiveness of scientific research.

KEYWORDS: DIGITIZATION, MODERNITY, DIGITAL EDUCATIONAL RESOURCES, ACCESSIBILITY, ADAPTABILITY, API.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧОК, СИМВОЛІВ І ТЕРМІНІВ

API (Application Programming Interface) — інтерфейс прикладного програмування, набір готових класів, процедур, функцій, структур і констант, що надаються додатком для використання у зовнішніх програмних продуктах.

BSON (Binary JSON) — бінарно-кодований формат серіалізації JSON-подібних документів, що використовується в MongoDB.

CDN (Content Delivery Network) — географічно розподілена мережа серверів, що забезпечує швидку доставку контенту користувачам.

CRUD (Create, Read, Update, Delete) — чотири базові функції, що використовуються при роботі з базами даних: створення, читання, оновлення та видалення даних.

CSS (Cascading Style Sheets) — каскадні таблиці стилів, спеціальна мова, що використовується для опису зовнішнього вигляду сторінок, написаних мовами розмітки даних.

HTML (HyperText Markup Language) — стандартизована мова розмітки документів для перегляду веб-сторінок у браузері.

HTTP (HyperText Transfer Protocol) — протокол передачі гіпертексту, що використовується для обміну даними в мережі Інтернет.

JSON (JavaScript Object Notation) — текстовий формат обміну даними, заснований на JavaScript.

LCMS (Learning Content Management System) — система управління навчальним контентом, призначена для створення, зберігання та повторного використання навчальних матеріалів.

LMS (Learning Management System) — система управління навчанням, програмне забезпечення для адміністрування, документації, відстеження та звітності навчальних курсів.

MOOC (Massive Open Online Courses) — масові відкриті онлайн-курси, навчальні курси з масовою інтерактивною участю.

SEO (Search Engine Optimization) — комплекс заходів для підняття позицій сайту в результатах видачі пошукових систем.

SPA (Single Page Application) — односторінковий веб-застосунок, який завантажує єдину HTML-сторінку та динамічно оновлює її вміст.

UI (User Interface) — інтерфейс користувача, сукупність засобів, за допомогою яких користувач взаємодіє з програмою.

UX (User Experience) — досвід користувача, сукупність вражень, які отримує користувач від взаємодії з інтерфейсом.

ЗВО — заклад вищої освіти.

ЦОР — цифрові освітні ресурси.

ВСТУП

У сучасному світі цифрові технології стали невід'ємною частиною усіх сфер життя, включно з освітою та наукою. Швидкий розвиток інформаційних технологій та активний перехід до дистанційних і змішаних форм навчання створюють гостру потребу в ефективних інструментах для обміну навчально-науковими матеріалами. Викладачі, студенти та науковці дедалі частіше потребують централізованого, безпечного та зручного середовища для публікації, пошуку та розповсюдження авторських розробок: лекцій, методичних посібників, презентацій та результатів досліджень.

Актуальність дослідження полягає в тому, що існуючі рішення для обміну освітніми матеріалами часто є або фрагментованими (використання загальних хмарних сервісів, що не мають належної структуризації), або комерціалізованими, що обмежує вільний доступ до знань. Високий попит на спеціалізовані українські освітні платформи підтверджує необхідність створення та розвитку нових, конкурентних вітчизняних аналогів. Розробка власного програмного застосунку, що враховує специфічні потреби навчально-наукової спільноти , зокрема в аспекті захисту авторських прав та зручної категоризації і пошуку контенту , є важливим і своєчасним завданням для підвищення якості та доступності освіти.

Метою цієї магістерської роботи є дослідження ролі та функціональних можливостей сучасних цифрових платформ для обміну освітніми матеріалами, а також практична розробка прототипу веб-застосунку-аналогу, орієнтованого на потреби викладачів та студентів. Для досягнення поставленої мети в роботі спочатку досліджуються теоретичні основи та аналізуються сучасні тенденції використання цифрових платформ в освітньо-науковій сфері. На основі цього проводиться огляд та порівняльний аналіз існуючих платформ для обміну освітніми матеріалами», що дозволяє виявити їхні переваги, недоліки та обґрунтувати доцільність розробки власного програмного застосунку. Практична частина роботи охоплює формулювання функціональних та нефункціональних

вимог до авторського освітнього застосунку, проєктування його архітектури, включаючи структуру бази даних, API та дизайн інтерфейсу користувача, а також опис процесу розробки та реалізації ключових програмних модулів платформи. Завершальними етапами дослідження є тестування розробленого прототипу та формулювання перспектив його подальшого розвитку.

Робота спрямована на глибокий аналіз специфіки, функціональних можливостей та архітектурних рішень існуючих цифрових платформ, призначених для обміну освітніми та науковими матеріалами. Для досягнення поставленої мети в роботі послідовно вирішується низка завдань. Спочатку досліджуються теоретичні основи та сучасні тенденції цифровізації освітнього середовища. Далі проводиться детальний порівняльний аналіз наявних на ринку програмних рішень, зокрема таких аналогів як «Всеосвіта», з метою виявлення їхніх переваг, недоліків та обґрунтування доцільності створення власного інструменту.

Ключовим практичним результатом роботи є безпосередня авторська розробка освітнього застосунку, що слугує прикладом реалізації сучасного цифрового інструменту. Цей етап включає формулювання функціональних вимог, проєктування архітектури системи, структури бази даних та інтерфейсу користувача, а також опис реалізації основних програмних модулів. На основі проведеного аналізу та практичної розробки, у роботі формулюються висновки та рекомендації щодо оптимального використання і проєктування подібних цифрових платформ в освітньо-науковій діяльності.

РОЗДІЛ 1. ВСТУП ТА АНАЛІЗ СУЧАСНИХ ТЕНДЕНЦІЙ ВИКОРИСТАННЯ ЦИФРОВИХ ТЕХНОЛОГІЙ В ОСВІТІ ТА НАУЦІ

1.1 Сучасні тенденції цифровізації освіти та науки

Сучасна освітньо-наукова сфера перебуває в стані глибокої та системної трансформації, каталізатором якої виступає інтенсивна цифровізація суспільства. Традиційні підходи до організації навчального процесу та наукової діяльності, що базувалися на фізичній присутності та друкованих носіях інформації, активно доповнюються, а в багатьох випадках замінюються новими моделями, заснованими на використанні передових інформаційно-комунікаційних технологій. Ця еволюція зумовлена комплексом факторів, серед яких не лише стрімкий технологічний прогрес, але й глобальні соціально-економічні виклики, що вимагають від системи освіти більшої гнучкості, інклюзивності та персоналізації [1].

Однією з домінуючих тенденцій останнього десятиліття стало масове впровадження систем управління навчанням, які дозволяють автоматизувати адміністративні процеси, забезпечити контроль успішності та організувати дистанційну взаємодію між учасниками освітнього процесу. Паралельно з цим, розвиток хмарних обчислень та технологій обробки великих даних створив необхідну інфраструктуру для централізованого зберігання колосальних масивів освітньої інформації, що, у свою чергу, відкрило шлях для впровадження аналітичних інструментів. Важливим вектором розвитку є також інтеграція алгоритмів штучного інтелекту, які здатні аналізувати поведінку користувачів, формувати адаптивні навчальні траєкторії та автоматизувати рутинні процеси оцінювання, що значно підвищує ефективність навчання.

Однак, поряд із закритими інституційними системами, особливого значення набуває тенденція до відкритості та демократизації знань, що реалізується через розвиток концепції Відкритих освітніх ресурсів. Це стимулювало появу спеціалізованих платформ для обміну навчально-науковими матеріалами, які функціонують за принципом «економіки спільної участі». Ці

ресурси трансформуються з пасивних бібліотек у динамічні маркетплейси або соціальні мережі для освітян, де викладачі, студенти та науковці виступають не лише споживачами, а й активними творцями контенту (просьюмерами). Такі платформи, яскравим прикладом яких є вітчизняний ресурс «Всеосвіта», стають осередками професійної комунікації, сприяють налагодженню горизонтальних зв'язків, обміну передовим педагогічним досвідом та прискоренню трансферу знань між освітніми закладами.

1.2. Специфіка та класифікація цифрових освітніх ресурсів та платформ для їх обміну

Для глибокого розуміння предметної області дослідження необхідно чітко окреслити специфіку об'єктів обміну — цифрових освітніх ресурсів (ЦОР) — та провести класифікацію програмних засобів, що забезпечують їх життєвий цикл. ЦОР являють собою дидактичні матеріали, представлені у цифровому форматі, що можуть включати електронні підручники, мультимедійні презентації, набори тестових завдань, відеолекції, програмні симулятори, віртуальні лабораторії та бази наукових даних. Ключовою характеристикою сучасних ЦОР є їхня інтерактивність та можливість багаторазового використання у різних навчальних контекстах.

Аналіз ринку програмного забезпечення дозволяє класифікувати платформи для роботи з ЦОР за їхнім функціональним призначенням та архітектурними особливостями. Першу групу складають класичні Системи управління навчанням, такі як Moodle, Canvas чи Google Classroom. Вони є переважно закритими корпоративними системами, орієнтованими на вертикальну модель управління навчанням у межах конкретного закладу, що обмежує можливості вільного обміну контентом із зовнішнім світом. Другу групу формують платформи Масових відкритих онлайн-курсів, наприклад Coursera або Prometheus, які пропонують структуровані, завершені освітні продукти від провідних інституцій, орієнтовані на самоосвіту широкої аудиторії.

Третя група охоплює наукові репозиторії та соціальні мережі для вчених (ResearchGate, Academia.edu), які спеціалізуються на індексації та поширенні

результатів наукових досліджень, статей та монографій, забезпечуючи комунікацію в академічному середовищі. Четверту, найбільш релевантну для даного дослідження групу, становлять платформи управління навчальним контентом та освітні маркетплейси. Їхня унікальність полягає у фокусуванні на атомарних одиницях контенту (окремих файлах, розробках, тестах), а не на цілісних курсах. Вони створюють відкрите середовище, де якість матеріалів визначається спільнотою через механізми рейтингування та рецензування. Саме такий підхід забезпечує максимальну гнучкість, дозволяючи викладачам конструювати власні курси з готових модулів, знайдених на платформі, що значно економить час та підвищує якість підготовки до занять.

1.3. Ключові виклики та вимоги при розробці веб-платформ для обміну освітніми матеріалами

Першочерговим та найважливішим викликом є забезпечення інформаційної безпеки та захисту авторських прав. Враховуючи, що платформа оперує унікальним інтелектуальним продуктом (авторськими методиками, презентаціями, дослідженнями), вона повинна гарантувати надійний захист від несанкціонованого доступу, копіювання чи втрати даних. Це вимагає впровадження багаторівневої системи автентифікації користувачів, механізмів шифрування даних під час їх передачі та зберігання, а також чітких політик керування правами доступу.

Другим ключовим викликом є технічна доступність та адаптивність. Освітня спільнота є вкрай неоднорідною з точки зору технічного забезпечення. Користувачі можуть мати різну швидкість інтернет-з’єднання та використовувати широкий спектр пристроїв – від потужних стаціонарних комп’ютерів до застарілих смартфонів. Тому платформа повинна бути оптимізованою, мати адаптивний дизайн та мобільну версію , щоб забезпечити рівний та безперебійний доступ до матеріалів для всіх учасників процесу, підтримуючи його безперервність.

Третій, і, можливо, найбільш значущий виклик – це управління великими обсягами контенту та юзабіліті. Централізоване місце зберігання з мільйонами

документів швидко втрачає свою цінність, якщо користувач не може оперативно знайти необхідний матеріал. Це вимагає впровадження складних та ефективних алгоритмів індексації, глибокої системи категоризації та фільтрації контенту (за предметом, типом, рейтингом, автором), а також інтелектуального пошуку . Для покращення користувацького досвіду доцільно впроваджувати механізми персоналізації, які пропонують рекомендовані матеріали на основі попередніх дій, інтересів чи академічного профілю користувача, пришвидшуючи таким чином доступ до релевантних знань .

Окрім цього, залишаються актуальними психолого-педагогічні виклики. Платформа має бути інтуїтивно зрозумілою, а матеріали на ній – структурованими та інтерактивними, щоб не створювати додаткового когнітивного навантаження на викладачів та студентів. Також необхідно враховувати непередбачувані технічні ризики, що вимагає надійних механізмів резервного копіювання та моніторингу роботи системи. Успішне вирішення цих завдань дозволяє створити інтегровану цифрову екосистему, що реально сприяє підвищенню доступності знань та ефективності навчально-наукового процесу.

1.4 Теоретичні аспекти структуризації та матеданих цифрового освітнього контенту

Ефективне функціонування платформ для обміну навчально-науковими матеріалами неможливе без належної організації збережених даних. В умовах експоненційного зростання обсягів неструктурованої інформації, що генерується користувачами, критичного значення набуває поняття метаданих — структурованої інформації, що описує, пояснює та полегшує пошук, використання або управління інформаційними ресурсами. Для освітніх платформ метадані виступають ключовим інструментом, що перетворює хаотичний набір файлів на впорядковану бібліотеку знань, доступну для швидкого пошуку та фільтрації.

У контексті проєктування інформаційних систем для освіти необхідно враховувати існуючі стандарти опису освітніх ресурсів, такі як Dublin Core [2]. Хоча для користувацьких маркетплейсів суворе дотримання цих стандартів може

бути надлишковим, їхні базові принципи — обов'язкова атрибуція автора, чітке визначення типу ресурсу, тематична класифікація та прив'язка до освітніх рівнів — мають бути імплементовані в архітектуру бази даних. Відсутність або неповнота метаданих призводить до феномену «dark data» (темних даних), коли корисний контент існує на сервері, але залишається недоступним для користувачів через неможливість його знайти за релевантними запитами.

Особливу увагу слід приділити розробці гнучкої таксономії та системи тегування. На відміну від жорстких ієрархічних структур, характерних для традиційних бібліотечних каталогів, сучасні веб-платформи все частіше використовують фасетну класифікацію, що дозволяє користувачеві фільтрувати матеріали за кількома незалежними параметрами одночасно, наприклад, за предметом, класом, типом файлу та рейтингом автора (таб 1.1). Впровадження таких підходів на етапі проєктування системи дозволяє вирішити проблему інформаційного перевантаження та забезпечити високу релевантність пошукової видачі, що є одним із головних показників якості програмного продукту в освітній сфері.

Таблиця 1.1

Порівняльна характеристика стандартів інтероперабельності цифрових освітніх ресурсів

Стандарт/ Специфікація	Основне призначення	Переваги	Недоліки	Актуальність для платформ обміну (LCMS)
SCORM (Shareable Content Object Reference Model)	Стандартизація пакування навчальних об'єктів для сумісності з LMS.	Широка підтримка всіма класичними LMS (Moodle тощо).	Застаріла архітектура; складність створення контенту; погана підтримка	Низька. Використовується для складних курсів, а не для простих файлів.

			мобільних пристроїв.	
xAPI (Experience API / Tin Can)	Збір даних про навчальний досвід (хто, що зробив, де).	Гнучкість; відстеження навчання поза браузером (мобільні додатки, симулятори.	Складність реалізації; потребує окремого сховища LRS (Learning Record Store).	Середня. Корисно для аналітики поведінки користувачів.
LTI (Learning Tools Interoperability)	Інтеграція зовнішніх додатків та інструментів в LMS.	Дозволяє підключати зовнішні платформи до LMS без перенесення контенту.	Вимагає складної налаштування безпеки (OAuth) на стороні провайдера.	Висока. Дозволяє вашій платформі в майбутньому інтегруватися з Moodle університетів.
Dublin Core	Стандарт метаданих для опису цифрових ресурсів.	Простота; універсальність (автор, дата, формат, мова).	Не враховує специфіку освіти (наприклад, вік учня, тип уроку).	Висока. Ідеальна база для проєктування таблиць бази даних вашої системи.

РОЗДІЛ 2. ДОСЛІДЖЕННЯ ТА АНАЛІЗ ІСНУЮЧИХ ПЛАТФОРМ ДЛЯ ОБМІНУ ОСВІТНІМИ МАТЕРІАЛАМИ

У попередньому розділі було проаналізовано теоретичні основи та загальні тенденції цифровізації освітньо-наукової сфери. Було встановлено, що одним із ключових напрямків є розвиток спеціалізованих веб-платформ для обміну цифровими освітніми ресурсами між учасниками навчального процесу .

Метою даного розділу є проведення детального дослідження та порівняльного аналізу вже існуючих програмних рішень, що функціонують у цій ніші. Цей аналіз необхідний для виявлення сильних та слабких сторін конкурентних продуктів, ідентифікації незакритих потреб ринку та обґрунтування доцільності розробки власного програмного застосунку, визначивши його унікальну позицію та переваги.

2.1. Огляд та класифікація існуючих підходів до організації освітніх матеріалів

Сучасний ринок цифрових освітніх технологій пропонує широкий спектр інструментів для організації навчальних матеріалів. Як було зазначено в підрозділі 1.2, ці інструменти можна класифікувати за їхнім основним призначенням та архітектурою. Для чіткого визначення ніші майбутнього проєкту необхідно проаналізувати функціональні обмеження кожної категорії з точки зору обміну матеріалами.

Першою категорією є Системи управління навчанням, такі як Moodle, Google Classroom або Canvas. Їхня основна функція - адміністрування навчального процесу. Вони ефективно вирішують завдання організації матеріалів у межах конкретного курсу чи навчального закладу. Викладач завантажує лекції та завдання, студент має до них доступ, проте система за замовчуванням є закритою екосистемою. Обмін методичними матеріалами між викладачами різних закладів чи навіть різних факультетів у таких системах значно ускладнений або неможливий.

Друга категорія - Масові відкриті онлайн-курси, представлені платформами Coursera, Udemy або українською Prometheus. Ці платформи орієнтовані на надання користувачу структурованого, завершеного продукту — онлайн-курсу. Вони не передбачають функціоналу для обміну окремими цифровими ресурсами (ЦОР), такими як презентації, методичні посібники чи плани уроків. Контент на цих платформах є високо структурованим та створюється централізовано або через партнерство з університетами, що виключає можливість вільного обміну матеріалами між окремими освітянами.

Третя категорія - Наукові репозиторії (ResearchGate, Academia.edu, інституційні репозиторії). Ці платформи успішно вирішують завдання обміну науковими матеріалами (статтями, монографіями), але їхній функціонал та цільова аудиторія погано адаптовані для потреб освітнього процесу, особливо на рівні середньої школи чи прикладних дисциплін у вищій освіті.

Четверта категорія, яка є найбільш релевантною для даного дослідження - це платформи-маркетплейси освітніх матеріалів. Яскравими українськими прикладами є «Всеосвіта» (рис 2.1) та «На Урок» (рис. 2.2) [3] [4]. На відміну від LMS та MOOC, ці платформи побудовані за принципом user-generated content (контент, що генерується користувачами). Вони створюють відкрите середовище, де будь-який викладач може зареєструватися, завантажити власні авторські розробки (презентації, тести, плани-конспекти) та зробити їх доступними (платно або безкоштовно) для всієї спільноти. Саме ця модель забезпечує найвищий рівень обміну матеріалами і є цільовою моделлю для розробки в даній роботі.

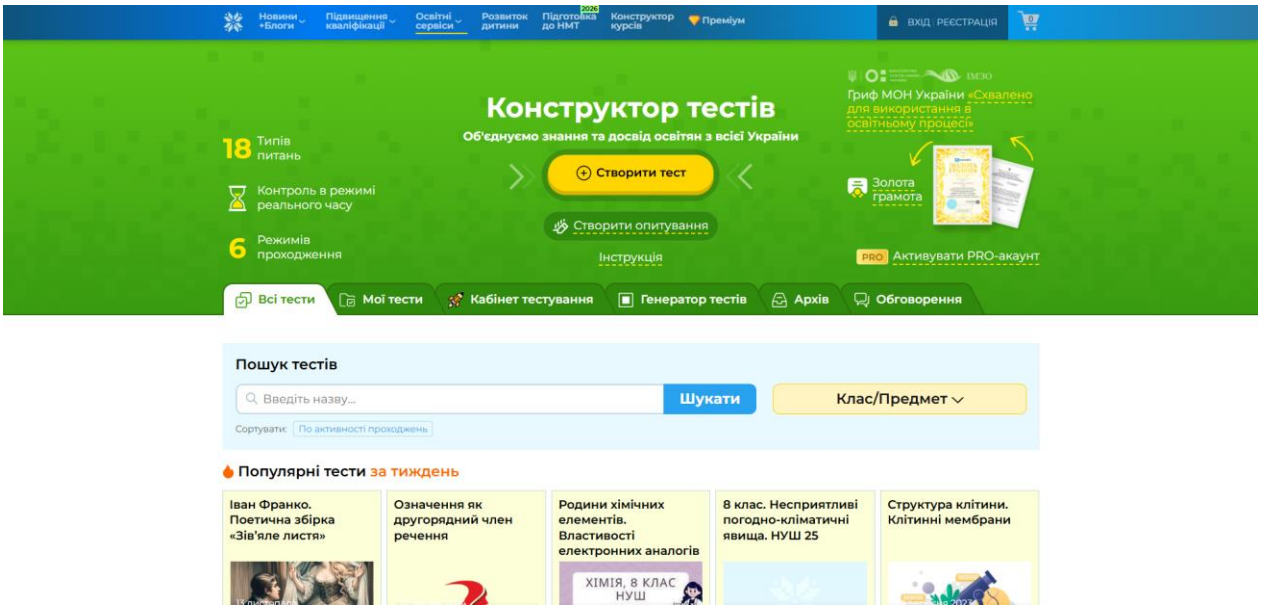

Рис. 2.1. Платформа «Всеосвіта» та підрозділ тести

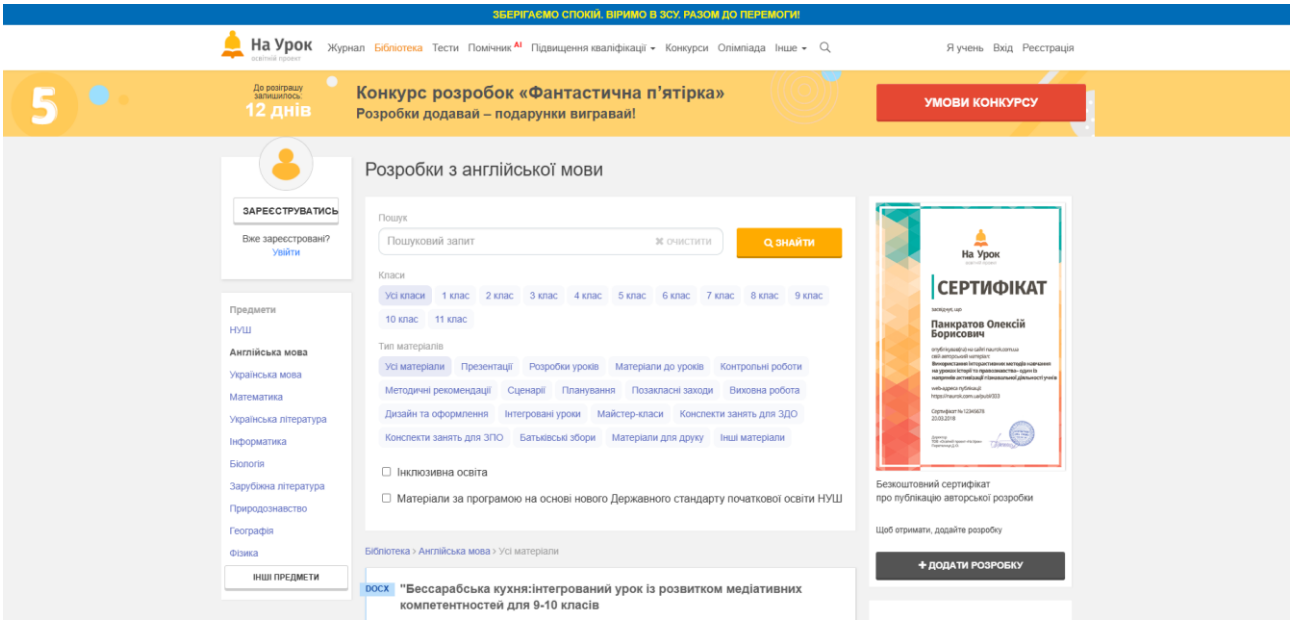

Рис. 2.2. Платформа «На Урок»

2.2. Порівняльний аналіз функціональних можливостей платформ-аналогів

Для обґрунтування розробки власного продукту необхідно провести детальний аналіз ключових гравців українського ринку освітніх маркетплейсів, зокрема «Всеосвіта» та «На Урок» (таб. 2.1 - 2.2). Аналіз проводитиметься за критеріями, визначеними у викликах (підрозділ 1.3): модель функціонування, управління контентом, контроль якості та юзабіліті.

«Всеосвіта» Платформа позиціонує себе як «національна освітня платформа» і є однією з найбільших в Україні.

Модель функціонування: Це класичний маркетплейс user-generated content. Викладачі (автори) завантажують матеріали та можуть встановлювати на них ціну або поширювати безкоштовно. Платформа виступає посередником, надаючи інфраструктуру та інструменти для продажу.

Управління контентом: Платформа має потужну, хоча й складну систему каталогізації. Матеріали можна фільтрувати за типом (презентації, тести, конспекти), предметом, класом. Пошук є ключовою функцією, проте через величезну кількість матеріалів (понад 1 мільйон) він часто видає занадто багато нерелевантних результатів.

Контроль якості та авторське право: Якість контенту контролюється переважно спільнотою через систему рейтингів та відгуків. Існує премодерація, але її ефективність при таких обсягах є сумнівною. Питання захисту авторського права вирішується шляхом додавання автоматичних водяних знаків на роботи, проте це не захищає від прямого копіювання змісту.

Юзабіліті (UI/UX): Це найслабше місце платформи. Інтерфейс є перевантаженим. Окрім обміну матеріалами, «Всеосвіта» пропонує десятки супутніх сервісів: олімпіади, курси підвищення кваліфікації, вебінари, конструктори тестів, новини. Це створює високе когнітивне навантаження на користувача, який шукає лише матеріали.

«На Урок» Платформа є прямим конкурентом «Всеосвіти» і має схожу функціональну модель.

Модель функціонування: Аналогічна маркетплейсу «Всеосвіта». Надає можливість публікувати авторські матеріали, брати участь у вебінарах та олімпіадах.

Управління контентом: Також використовує детальну систему рубрикації за предметами, класами та типами матеріалів. Пошукова система є важливою складовою, але, як і в конкурента, страждає від великої кількості контенту різної якості.

Контроль якості та авторське право: Використовується схожий підхід — комбінація модерації та зворотного зв'язку від спільноти (відгуки, рейтинги).

Юзабіліті (UI/UX): Інтерфейс також є багатофункціональним і дещо перевантаженим. Фокус зміщений з простого обміну файлами на комплексне обслуговування потреб вчителя (конкурси, підвищення кваліфікації).

Таблиця 2.1.

Порівняльний аналіз платформ-аналогів

Критерії	**«Всеосвіта»**	**«На Урок»**
Основна модель	Маркетплейс ЦОР (User-Generated)	Маркетплейс ЦОР (User-Generated)
Цільова аудиторія	Переважно вчителі середньої школи	Переважно вчителі середньої школи
Управління контентом	Розгалужена система фільтрів, пошук	Розгалужена система фільтрів, пошук
Контроль якості	Модерація, рейтинги, відгуки	Модерація, рейтинги, відгуки
Захист авт.права	Автоматичні водяні знаки	Автоматичні водяні знаки
Інтерфейс	Перевантажений	Перевантажений
Додаткові сервіси	Дуже багато (олімпіади, курси, новини)	Дуже багато (олімпіади, курси, тести)

Таблиця 2.2.

SWOT-аналіз існуючих платформ-аналогів («Всеосвіта», «На Урок»)

Категорія	Фактори аналізу
Сильні сторони	1. Велика база користувачів та матеріалів (понад 1 млн). 2. Висока впізнаваність бренду та довіра.

	3. Потужна SEO-оптимізація (матеріали легко гугляться). 4. Розвинена система мотивації (сертифікати, конкурси).
Слабкі сторони	1. Перевантажений інтерфейс: складність навігації через велику кількість функцій. 2. Нерелевантний пошук: важко знайти якісний матеріал серед застарілих. 3. Слабка адаптація для вищої школи: відсутність структури для ЗВО. 4. Застаріла модель хостингу: дорогі сервери, повільне завантаження.
Можливості	1. Впровадження штучного інтелекту для рекомендацій. 2. Інтеграція з державними цифровими сервісами (Дія.Освіта). 3. Вихід на міжнародні ринки.
Загрози	1. Поява нішевих конкурентів (наша розробка) зі зручнішим інтерфейсом. 2. Зміни в законодавстві щодо авторського права. 3. Відтік аудиторії на глобальні платформи (Google Classroom, Canva).

2.3 Аналіз архітектурних моделей зберігання та агрегації освітнього контенту

При проєктуванні системи обміну навчальними матеріалами критично важливим є вибір стратегії роботи з даними. Аналіз існуючих рішень дозволяє

виділити два діаметрально протилежних підходи: модель безпосереднього хостингу та модель агрегації посилань (таб 2.3).

Більшість класичних платформ-аналогів («Всеосвіта», «На Урок») використовують модель безпосереднього хостингу. Вона передбачає фізичне завантаження файлів користувачів на сервери платформи (рис. 2.3). Хоча це дає повний контроль над контентом, такий підхід має суттєві технічні та економічні недоліки: висока вартість хмарного сховища, необхідність у складних механізмах транскодування відео та зображень, а також пряма юридична відповідальність платформи за порушення авторських прав у завантажених файлах.

Альтернативним, більш сучасним та гнучким підходом, є модель агрегації ресурсів. У цій архітектурі платформа виступає не як сховище файлів, а як каталогізатор метаданих (рис 2.4). Автор розміщує навчальні матеріали на спеціалізованих зовнішніх сервісах (Google Drive – для документів, YouTube – для відео, GitHub – для коду, Figma – для дизайну), а на платформі зберігається лише пряме посилання та згенероване прев'ю (зображення попереднього перегляду).

Таблиця 2.3.

Вибір бази даних

Критерії порівняння	Реляційні (SQL)	Документоорієнтовані (NoSQL) (MongoDB)	Висновок для проекту
Гнучкість схеми даних	Низька. Вимагає чіткої структури таблиць. Додавання нового поля (наприклад, "тривалість відео") потребує міграції всієї БД.	Висока (Schema-less). Кожен запис може мати унікальний набір полів. Ідеально для різнорідних матеріалів.	MongoDB краще, оскільки дозволяє легко додавати нові типи ресурсів без зміни коду.
Формат даних	Таблиці та рядки. Потребує	JSON-подібні документи (BSON).	MongoDB краще,

	перетворення (ORM) для роботи з JSON на фронтенді.	Нативний формат для JavaScript/Node.js.	оскільки спрощує розробку (немає зайвих перетворень даних).
Швидкодія при читанні	Висока для складних зв'язків (JOIN), але може падати при великій кількості запитів на читання.	Дуже висока. Дані зберігаються разом (демократизовано), тому читання одного об'єкта відбувається миттєво без "склеювання" таблиць.	MongoDB краще для каталогу, де основне навантаження — це перегляд карток матеріалів.
Масштабованість	Вертикальна (потрібен потужніший сервер). Складно налаштувати кластеризацію.	Горизонтальна. Легко масштабується шляхом додавання нових серверів (Sharding).	MongoDB краще для потенційного зростання бази матеріалів.

Технічні переваги моделі агрегації для проєктованої системи:

Оптимізація ресурсів: База даних зберігає лише легковагові текстові метадані (назва, опис, теги, URL) та оптимізовані зображення-прев'ю. Це дозволяє системі працювати швидко навіть на бюджетному хостингу, оскільки «важкий» контент віддається через потужні CDN зовнішніх провайдерів (Google, Microsoft тощо) .

Прев'ю: Ручне завантаження обкладинки автором гарантує якісний візуальний ряд. Це дозволяє автору створити привабливу «вітрину» для свого матеріалу, а платформі — підтримувати єдиний стиль каталогу. Технічна задача

зводиться до реалізації механізмів завантаження та автоматичної оптимізації (стиснення, зміни розміру) зображень на стороні клієнта або сервера.

Стійкість та безпека: Оскільки файли не зберігаються на сервері платформи, зникає вектор атак через завантаження шкідливого ПЗ (вірусів у файлах .doc, .exe). Захист авторських прав також спрощується: власник матеріалу зберігає повний контроль над оригіналом на своєму хмарному диску і може в будь-який момент обмежити до нього доступ.

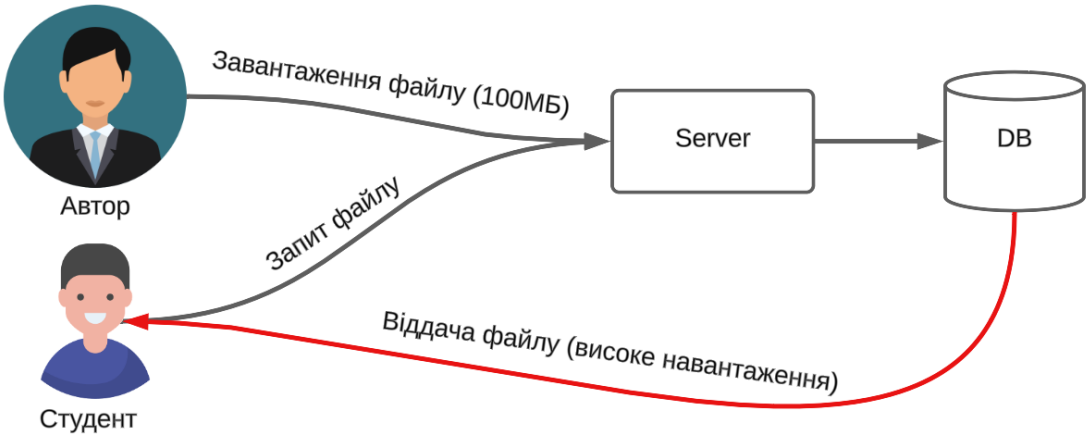

Рис. 2.3. Схематичне відображення організації роботи платформи «Всеосвіта»

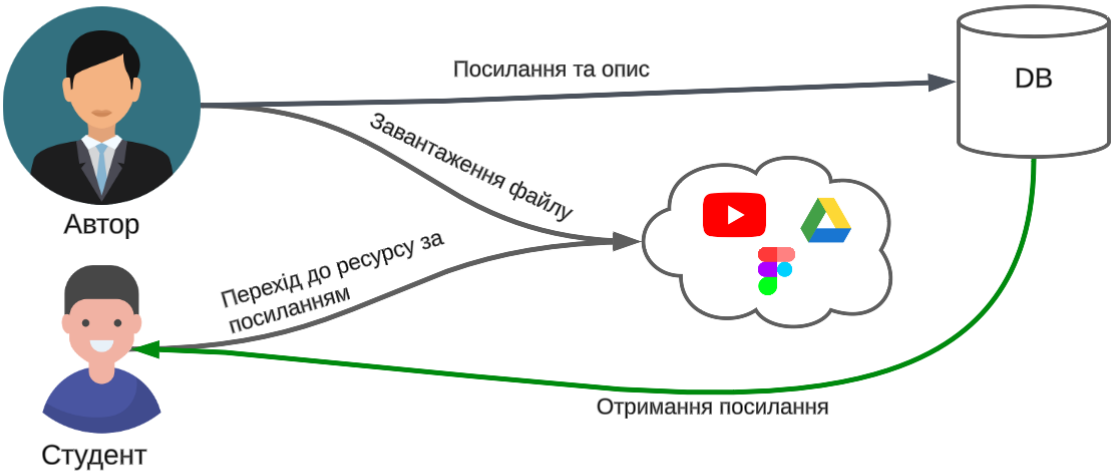

Рис. 2.4. Запропонована модель організації роботи розроблюваної платформи

Отже, аналіз показує, що для створення сучасного, легкого та безпечного освітнього застосунку доцільно відмовитись від застарілої моделі файлообмінника на користь архітектури агрегатора посилань із розширеною генерацією прев'ю. Це дозволить сфокусуватися на зручності пошуку та каталогізації, переклавши завдання зберігання «важких» даних на спеціалізовані хмарні сервіси.

2.4. Обґрунтування доцільності розробки власного освітнього застосунку

Проведений аналіз провідних українських платформ для обміну освітніми матеріалами демонструє, що ринок, хоч і зайнятий великими гравцями, має низку суттєвих недоліків та незакритих потреб, що й обґрунтовує доцільність розробки нового програмного продукту.

Виявлені недоліки існуючих рішень:

1. Надмірна функціональність та перевантаженість інтерфейсу: І «Всеосвіта», і «На Урок» еволюціонували від простих репозиторіїв до багатофункціональних «комбайнів». Вони намагаються охопити всі можливі потреби освітян (новини, вебінари, курси, конкурси), що робить їхні інтерфейси складними, повільними та відволікаючими. Користувач, який має просту задачу (наприклад, знайти презентацію з конкретної теми), змушений пробиватися крізь безліч банерів, пропозицій та другорядних функцій.
2. Відсутність вузької спеціалізації: Обидві платформи орієнтовані переважно на середню освіту (вчителів та учнів 1-11 класів). Ніша для обміну матеріалами у сфері вищої освіти (між викладачами та студентами ЗВО) або у сфері наукової діяльності (обмін специфічними методиками, програмним кодом, даними експериментів, що не є повноцінними статтями) практично не зайнята.

3. Недостатній захист авторського права: Механізми водяних знаків є лише базовим захистом і не вирішують проблему копіювання та недобросовісного використання авторських матеріалів.

Обґрунтування розробки:

Враховуючи вищезазначені недоліки, розробка власного програмного застосунку є доцільною та актуальною. Новий продукт може зайняти свою нішу, якщо він буде спроєктований з урахуванням наступних принципів:

1. Мінімалізм та фокус на основній функції: На відміну від конкурентів, пропонований застосунок буде зосереджений виключно на одній задачі — швидкому, зручному та структурованому обміні навчально-науковими матеріалами. Це дозволить створити чистий, інтуїтивно зрозумілий та швидкий інтерфейс (UI/UX).
2. Нішева орієнтація: Проєкт може бути вузько спрямований на аудиторію, потреби якої існуючі платформи ігнорують, наприклад, на спільноту викладачів та студентів ЗВО. Це дозволить реалізувати більш релевантну систему категоризації (за спеціальностями, кафедрами, дисциплінами), ніж просто «за класами».
3. Покращені механізми безпеки: У новому продукті можна закласти більш сучасні підходи до захисту контенту (наприклад, інтегрований переглядач документів без можливості прямого завантаження, більш гнучкі налаштування приватності).

У другому розділі проведено комплексне дослідження та аналіз існуючих цифрових платформ для обміну освітніми матеріалами. Встановлено, що український ринок представлений переважно універсальними рішеннями, такими як «Всеосвіта» та «На Урок», які функціонують за моделлю безпосереднього хостингу контенту.

Порівняльний аналіз виявив, що попри популярність, ці платформи мають низку суттєвих недоліків: перевантаженість інтерфейсу, складність навігації, відсутність спеціалізації для вищої освіти та застарілі підходи до організації зберігання даних. Ключовою проблемою є спроба поєднати в одній системі

функції соціальної мережі, магазину, новинного порталу та LMS, що негативно впливає на користувацький досвід (UX).

Таким чином, розробка власного веб-застосунку не буде простим копіюванням існуючих рішень, а запропонує ринку ціннісну пропозицію у вигляді простоти, сфокусованості та кращого користувацького досвіду, що повністю відповідає меті даної дипломної роботи. Наступний розділ буде присвячений практичній реалізації прототипу такої системи.

РОЗДІЛ 3 ПРАКТИЧНА РОЗРОБКА ТА РЕАЛІЗАЦІЯ ОСВІТНЬОГО ЗАСТОСУНКУ

3.1. Формування вимог та моделювання бізнес-процесів системи

Початковим та визначальним етапом життєвого циклу розробки будь-якого програмного забезпечення є етап системного аналізу та формування вимог. Базуючись на висновках другого розділу, де було обґрунтовано доцільність створення спеціалізованого веб-агрегатора освітніх матеріалів для закладів вищої освіти, необхідно формалізувати завдання, які має вирішувати проєктована система. Цей етап включає деталізацію функціональних та нефункціональних вимог, визначення ролей користувачів та моделювання ключових бізнес-процесів за допомогою уніфікованої мови моделювання (UML).

3.1.1. Ідентифікація стейкхолдерів та ролей користувачів

Для забезпечення коректної роботи системи та розмежування прав доступу необхідно чітко визначити акторів, які взаємодіятимуть із платформою. Враховуючи концепцію відкритого обміну знаннями, у системі виділено три ключові ролі, кожна з яких має специфічний набір повноважень.

Першою роллю є Гість (Неавторизований користувач). Це відвідувач, який взаємодіє з платформою з метою ознайомлення. Гість має доступ до базового функціоналу читання: перегляду головної сторінки, використання публічного каталогу матеріалів та здійснення пошуку за ключовими словами. Він може переглядати картки ресурсів, бачити прев’ю та опис, доступ до прямих посилань на зовнішні джерела, а також можливість взаємодії з контентом (коментування, оцінювання), проте мне має можливості додавати матеріали до збережених, створювати авторські матеріали.

Другою та центральною роллю в системі є Авторизований користувач. Ця роль є універсальною і поєднує в собі функції як шукача, так і творця контенту (автора). Після проходження процедури автентифікації користувач отримує повний доступ до інформаційних ресурсів платформи, включаючи перехід за зовнішніми посиланнями. Одночасно з цим він отримує право публікувати власні

матеріали через механізм створення карток ресурсів. Авторизований користувач має особистий кабінет, де він може керувати своїми публікаціями (редагувати або видаляти їх), а також формувати власну колекцію обраних матеріалів, збережених від інших авторів. Крім того, ця роль передбачає активну соціальну взаємодію: можливість залишати відгуки, ставити оцінки та формувати рейтинг матеріалів спільноти.

Третьою роллю є Адміністратор. Це привілейований користувач, відповідальний за технічну підтримку та модерацію платформи. До його повноважень належить керування довідниками системи (наприклад, додавання нових категорій дисциплін), керування обліковими записами користувачів (включно з блокуванням за порушення правил) та модерація контенту. Адміністратор має право видаляти або приховувати матеріали, що містять недостовірну інформацію, порушують авторські права або не відповідають тематиці ресурсу.

3.1.2. Специфікація функціональних вимог

Функціональні вимоги визначають поведінку системи та набір сервісів, які вона надає користувачам. Враховуючи обрану архітектуру агрегатора та уніфікацію ролей, вимоги згруповано за логічними модулями.

Модуль автентифікації та профілю. Система повинна забезпечувати реєстрацію та авторизацію користувачів. Важливою вимогою є наявність особистого профілю, де користувач може вказати або змінити своє ім’я, додати опис про себе та контакти-ресурси через які інші користувачі зможуть отримати зв’язок з автором . Це дозволить шукачам матерілів більше розуміти з ким вони мають справу та мати більшу обізнаність про автора.

Модуль керування контентом. Цей модуль є доступним для будь-якого Авторизованого користувача. Він має реалізовувати логіку повного життєвого циклу картки матеріалу (CRUD). Система повинна надавати зручну форму для додавання ресурсу, яка включає поля для назви, опису, вибору категорії (тип ресурсу, вікова категорія, предмет, та вставки зовнішнього URL-посилання, також у автора буде можливість обрати який початковий доступ буде у метеріалу

(публічний або приватний). Критичною вимогою є функція завантаження зображеннь-прев'ю, яка повинна підтримувати валідацію формату файлу та його автоматичну оптимізацію для веб-відображення. Користувач повинен мати можливість редагувати або видаляти лише ті матеріали, автором яких він є. Окрім цього у автора має бути змога навіть після публікації матеріалу, відредагувати його вміст за потреби, або видалити.

Модуль пошуку та каталогізації. Система повинна забезпечувати швидкий та релевантний пошук по базі метаданих. Вимагається реалізація механізмів фільтрації за кількома критеріями одночасно: типом ресурсу (лекція, презентація, тест і тд.), предмет(географія, природознавство, метематика і тд.) та рівень навчання (фахова, вища, шкільна освіта і тд.). Пошукова видача повинна відображатися у вигляді сітки карток із прев'ю, заголовком, типом доступу до матеріалу (безкоштовно, за плату), автором та тегами з типами матеріалу, що дозволяє користувачеві швидко оцінити візуальну якість матеріалу.

Модуль взаємодії та бібліотеки. Цей функціонал дозволяє Авторизованому користувачу формувати персональний простір знань. Система повинна підтримувати функцію «Додати у вибране», що зберігає посилання на корисні матеріали в окремому розділі кабінету. Також модуль відповідає за збір зворотного зв'язку: розрахунок середнього рейтингу матеріалу на основі оцінок користувачів.

3.1.3. Специфікація нефункціональних вимог

Нефункціональні вимоги визначають атрибути якості програмного забезпечення та накладають архітектурні обмеження, дотримання яких є необхідною умовою для забезпечення життєздатності системи в реальних умовах експлуатації. Для проєктування спеціалізованого веб-агрегатора освітніх матеріалів було виділено низку критичних параметрів, що охоплюють продуктивність, масштабованість, надійність, зручність використання та інформаційну безпеку.

Першочерговим пріоритетом є забезпечення високої продуктивності та швидкодії системи, оскільки основна цінність для користувача полягає у

миттєвому доступі до каталогу матеріалів. Згідно з сучасними стандартами веб-розробки, час завантаження сторінки, зокрема показник First Contentful Paint, який фіксує момент появи першого значущого контенту, не повинен перевищувати 1.5 секунди для десктопної версії та 2.0 секунд для мобільних пристроїв навіть за умов нестабільного з’єднання. Для досягнення таких показників у системі необхідно реалізувати механізми автоматичної оптимізації медіа-контенту на стороні сервера. Це передбачає відтворення та збереження значної частини контенту на серверній стороні що не навантажувати пристрій користувача для завантаження контенту. Додатково для зменшення навантаження на базу даних та пришвидшення відгуку інтерфейсу доцільно впровадити багаторівневе кешування: на стороні клієнта для статичних ресурсів та на стороні сервера для результатів частих пошукових запитів.

Окрему увагу при проєктуванні архітектури приділено вимогам масштабованості, що обумовлено потенційним експоненційним зростанням обсягу бази даних та кількості одночасних користувачів. Вибір документоорієнтованої системи керування базами даних MongoDB дозволяє задовольнити вимогу щодо масштабування даних завдяки вбудованій підтримці механізму шардінгу. Це технічне рішення дозволить у перспективі розподілити колекції матеріалів між декількома фізичними серверами, забезпечуючи стабільну швидкість читання даних навіть при мільйонах записів. На рівні бекенду архітектура повинна будуватися за принципом Stateless (без збереження стану), що уможливлює горизонтальне масштабування серверної частини шляхом запуску додаткових екземплярів застосунку за балансувальником навантаження у періоди пікової активності.

Вимоги до надійності та доступності системи спрямовані на забезпечення стабільного доступу до контенту та коректну обробку позаштатних ситуацій. Специфічним викликом для моделі агрегатора є ризик виникнення неактивних посилань, коли автор видаляє файл на зовнішньому хмарному диску. Для вирішення цієї проблеми система повинна містити механізми «м'якої» обробки помилок та автоматичного моніторингу доступності зовнішніх ресурсів. У разі

виявлення неактивного посилання користувач має отримувати зрозуміле повідомлення, а автору матеріалу повинно надсилатися автоматичне сповіщення з пропозицією оновити URL-адресу. Загальний показник безперебійної роботи системи має складати не менше 99.5%, що забезпечується, зокрема, регулярним автоматичним резервним копіюванням бази даних для запобігання втраті метаданих.

Критично важливими є вимоги до інтерфейсу та юзабіліті, оскільки платформа орієнтована на широку аудиторію з різним рівнем технічних навичок. Дизайн веб-застосунку повинен бути повністю адаптивним, гарантуючи коректне відображення та функціональність на всьому спектрі пристроїв — від смартфонів до широкоформатних моніторів. Візуальне оформлення має відповідати принципам мінімалізму, уникаючи інформаційного шуму, що дозволяє користувачеві сфокусуватися на освітньому контенті. Окрім того, для забезпечення рівного доступу система повинна відповідати базовим вимогам стандарту доступності веб-контенту, зокрема забезпечувати достатній контраст елементів інтерфейсу та підтримку навігації за допомогою клавіатури [5] .

3.1.4. Моделювання бізнес-процесів (UML-діаграми)

Для візуалізації та формалізації вимог було розроблено моделі бізнес-процесів. Центральним процесом системи є додавання нового матеріалу автором. Цей процес є нетривіальним через особливості моделі агрегації.

Розглянемо сценарій «Публікація освітнього ресурсу», який можна описати за допомогою діаграми діяльності.Процес розпочинається з ініціалізації Автором форми додавання. Система пропонує ввести назву, опис та обрати типізацію. Далі відбувається розгалуження логіки: Автор вводить URL-адресу зовнішнього ресурсу (наприклад, посилання на Google Drive). Система виконує валідацію формату посилання. Якщо формат невірний, виводиться помилка. Якщо посилання валідне, Автор переходить до етапу завантаження візуального прев'ю (до 8 зображень).

На цьому кроці відбувається завантаження зображення з локального пристрою користувача. Система на стороні сервера перевіряє тип файлу

(дозволені лише зображення), його розмір та виконує компресію. У разі успіху файл зберігається у файловій системі сервера або хмарному сховищі, а посилання на нього повертається клієнту. Після вибору типу доступу до матеріалу (публічний або приватний) та підтвердження публікації створюється запис у базі даних, і матеріал стає доступним відповідно до обраного типу доступу.

Іншим важливим процесом є «Доступ до матеріалу» з боку інших користувачів. Оскільки матеріал знаходиться на зовнішньому ресурсі, система виступає проміжною ланкою. Студент натискає кнопку «Перейти до матеріалу». Система фіксує факт переходу (інкрементує лічильник переглядів) для статистики та перенаправляє користувача на зовнішній URL у новій вкладці браузера. Такий підхід дозволяє збирати аналітику популярності ресурсів, навіть не хостячи їх фізично.

Розроблені моделі та сформульовані вимоги створюють надійний фундамент для наступного етапу — безпосереднього проєктування архітектури та вибору технологічного стеку, що буде детально розглянуто у наступному підрозділі.

3.1.5. Вимоги до пошукової оптимізації та технічної інтеграції з пошуковими системами

Враховуючи специфіку роботи платформи-агрегатора, ключовим джерелом трафіку та нових користувачів є органічний пошук через глобальні пошукові системи, зокрема Google. Тому на етапі формування вимог критично важливо закласти архітектурні рішення, що забезпечують ефективну індексацію динамічного контенту. Оскільки сучасні веб-застосунки, побудовані на базі бібліотеки React, функціонують як односторінкові додатки, їхній вміст формується динамічно на стороні клієнта за допомогою JavaScript. Це створює суттєвий технічний бар'єр для пошукових роботів (crawlers), які можуть некоректно зчитувати контент сторінки, що призводить до низьких позицій у пошуковій видачі.

Для вирішення цієї проблеми система повинна підтримувати технологію серверного рендерингу або статичної генерації для публічних сторінок

матеріалів. Це дозволить серверу віддавати пошуковому роботу повністю сформований HTML-код сторінки з контентом, що гарантує миттєву індексацію. Окрім цього, кожен окремий матеріал повинен мати унікальну, людино-зрозумілу URL-адресу, яка містить транслітеровану назву ресурсу, що позитивно впливає на ранжування та сприйняття посилання користувачами.

Важливою вимогою є впровадження семантичної розмітки структурованих даних згідно зі стандартом. Це дозволить пошуковій системі Google не просто індексувати текст, а й розуміти контекст сторінки: визначати автора, дату публікації, рейтинг, тип файлу та цільову аудиторію. Завдяки цьому у результатах пошуку формуватимуться розширені сніпети із зірочками рейтингу та додатковою інформацією, що значно підвищує клікабельність посилань на платформу.

Також система повинна автоматично керувати мета-тегами (Meta Title, Meta Description) для кожної сторінки, динамічно генеруючи їх на основі назви та опису матеріалу, введених автором. Для забезпечення коректного відображення посилань при поширенні у соціальних мережах та месенджерах необхідна інтеграція протоколу Open Graph [6]. Це дозволить автоматично підтягувати завантажене автором зображення-прев'ю як обкладинку посилання. Технічна реалізація SEO-стратегії також передбачає автоматичну генерацію та регулярне оновлення карти сайту (sitemap.xml), яка повідомляє пошуковим системам про появу нових матеріалів.

3.2. Обґрунтування вибору стеку технологій та проєктування архітектури програмного забезпечення

Ефективність реалізації проєктованої системи, її надійність та можливість подальшого масштабування безпосередньо залежать від обґрунтованого вибору інструментальних засобів розробки. Для створення веб-агрегатора освітніх матеріалів було обрано сучасний технологічний стек, який базується на екосистемі JavaScript, що дозволяє використовувати єдину мову програмування як на стороні клієнта, так і на стороні сервера. Такий підхід забезпечує

уніфікацію кодової бази, спрощує процес розробки та дозволяє ефективно працювати з даними у форматі JSON, що є природним для обраної архітектури.

Початковий етап проєктування інтерфейсу та користувацького досвіду реалізується за допомогою графічного редактора Figma. Цей хмарний інструмент дозволяє створити деталізовані прототипи сторінок, спроєктувати адаптивну сітку для мобільних пристроїв та розробити дизайн-систему компонентів ще до написання програмного коду. Використання Figma забезпечує узгодженість візуального стилю та дозволяє протестувати логіку навігації на етапі макетування, що значно знижує ризик виникнення помилок у юзабіліті під час програмування.

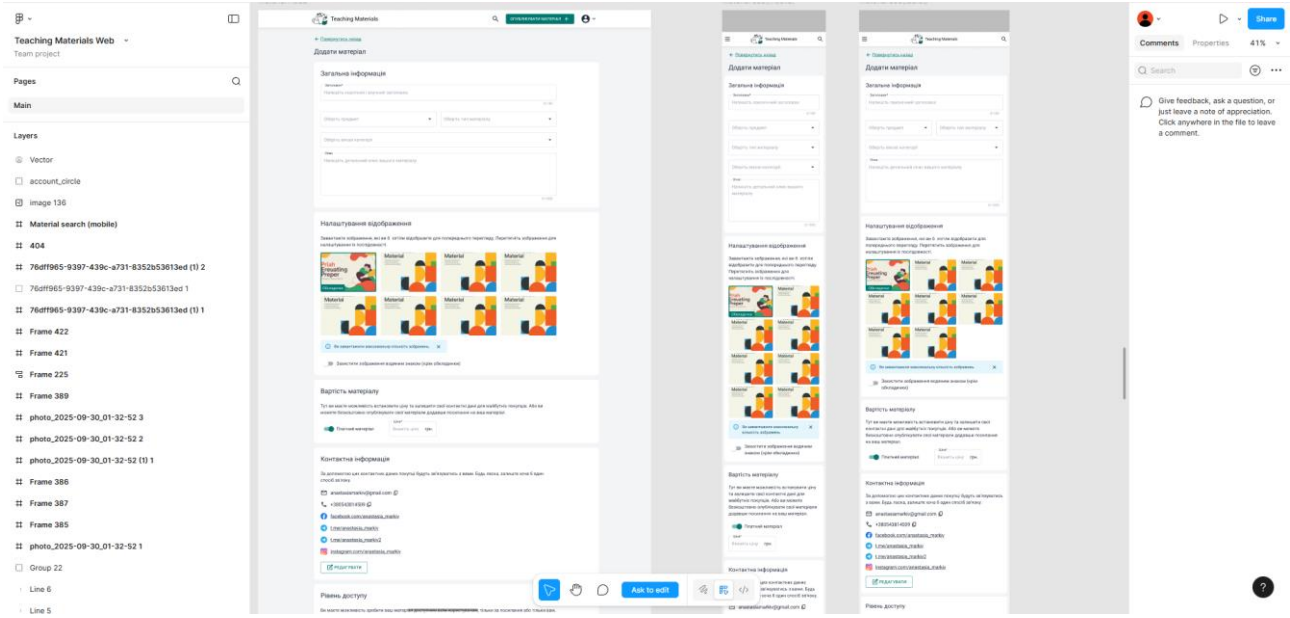

Рис. 3.1. Платформа Figma з макетами сторінки створення матеріалу та врахуванням адаптації до мобільних пристроїв

Ключовим елементом технологічного ядра системи обрано фреймворк Next.js, який базується на бібліотеці React [7] [8]. Вибір на користь Next.js зумовлений специфічними вимогами до платформи-агрегатора, зокрема необхідністю забезпечення якісної пошукової оптимізації (SEO). На відміну від класичних односторінкових застосунків (SPA), які рендеряться виключно в браузері, Next.js підтримує технологію серверного рендерингу (та статичної генерації сторінок. Це дозволяє серверу віддавати пошуковим роботам та

користувачам повністю сформований HTML-код, що гарантує миттєве відображення контенту та високі позиції ресурсу в пошуковій видачі Google.

Для стилізації інтерфейсу використовується CSS-фреймворк Tailwind CSS. Цей інструмент реалізує підхід utility-first, дозволяючи конструювати унікальний дизайн безпосередньо в розмітці компонентів без написання громіздких каскадних таблиць стилів. Важливою перевагою Tailwind CSS у поєднанні з Next.js є автоматичне видалення невикористаних стилів під час збірки проєкту, що забезпечує мінімальний розмір файлів та критично впливає на швидкість завантаження сторінок на мобільних пристроях [9].

Серверна частина будується на базі платформи Node.js. Вибір цього середовища виконання зумовлений його неблокуючою моделлю введення-виведення, яка дозволяє ефективно обробляти велику кількість одночасних легких запитів, характерних для платформ-агрегаторів. Node.js забезпечує високу продуктивність при роботі з API та інтеграції із зовнішніми сервісами [10].

У якості системи керування базами даних використовується MongoDB [11]. Ця документоорієнтована NoSQL-база даних зберігає інформацію у форматі JSON, що дозволяє уникнути складних перетворень даних при передачі їх на клієнтську частину. Гнучка схема даних MongoDB дає можливість легко змінювати структуру опису навчальних матеріалів без необхідності виконання складних міграцій бази даних, що є важливою перевагою на етапі активної розробки та розвитку продукту.

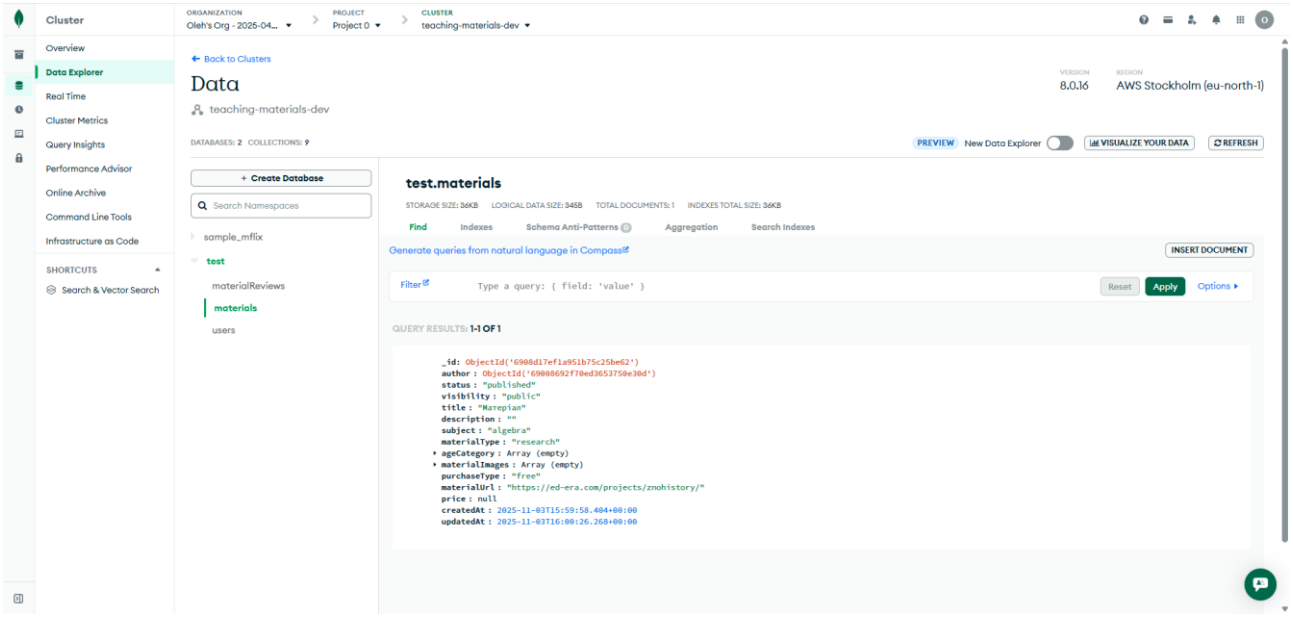

Рис. 3.2. Приклад документу зі збереженим матеріалом в базі даних MobgoDB

Для забезпечення безпечної автентифікації та керування сесіями користувачів інтегровано платформу Firebase від Google. Використання готового рішення Firebase Authentication дозволяє делегувати складні завдання криптографічного захисту паролів та відновлення доступу надійному зовнішньому провайдеру, що підвищує загальний рівень безпеки системи. Крім того, Firebase надає зручні інструменти для реалізації входу через соціальні мережі, що спрощує процес реєстрації для нових користувачів [12].

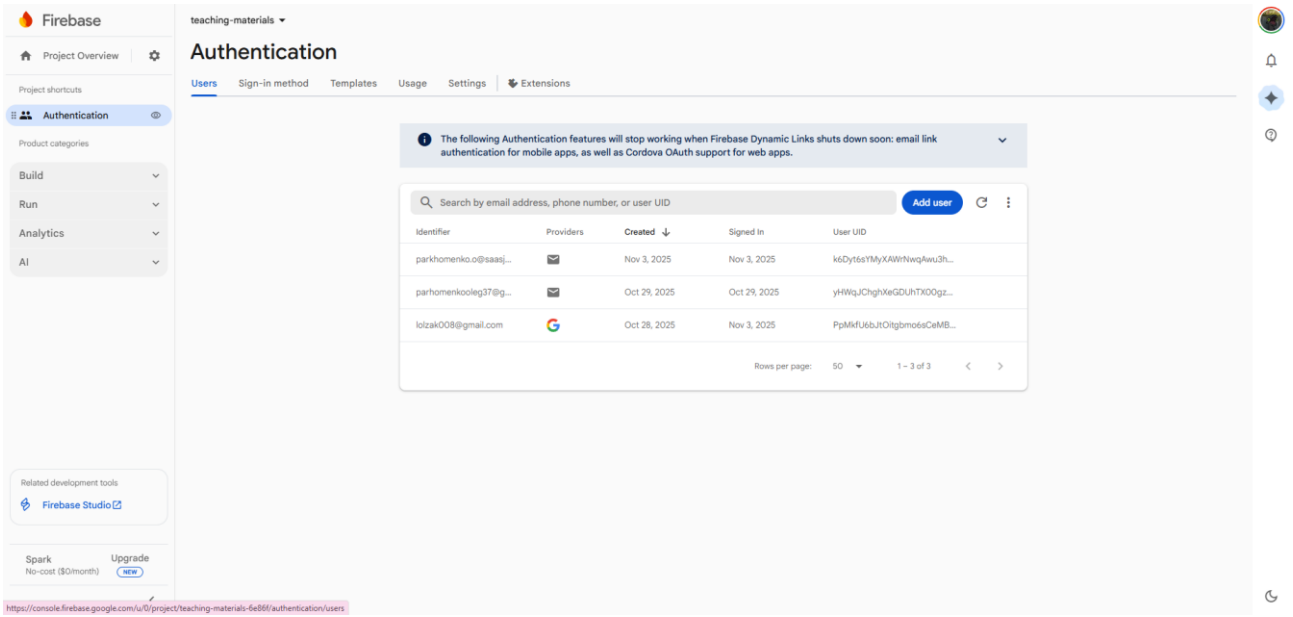

Рис. 3.3. Збережені зареєстровані користувачі у ресурсі Firebase

Забезпечення продуктивності, безпеки та захисту від DDoS-атак покладено на сервіс Cloudflare. Він виступає як мережа доставки контенту (CDN), кешуючи статичні ресурси (зображення прев'ю, скрипти, стилі) на серверах, розташованих географічно близько до користувача. Це дозволяє суттєво зменшити час завантаження сайту та знизити навантаження на основний сервер застосунку. Також Cloudflare забезпечує автоматичне надання SSL-сертифікатів для шифрування трафіку за протоколом HTTPS [13].

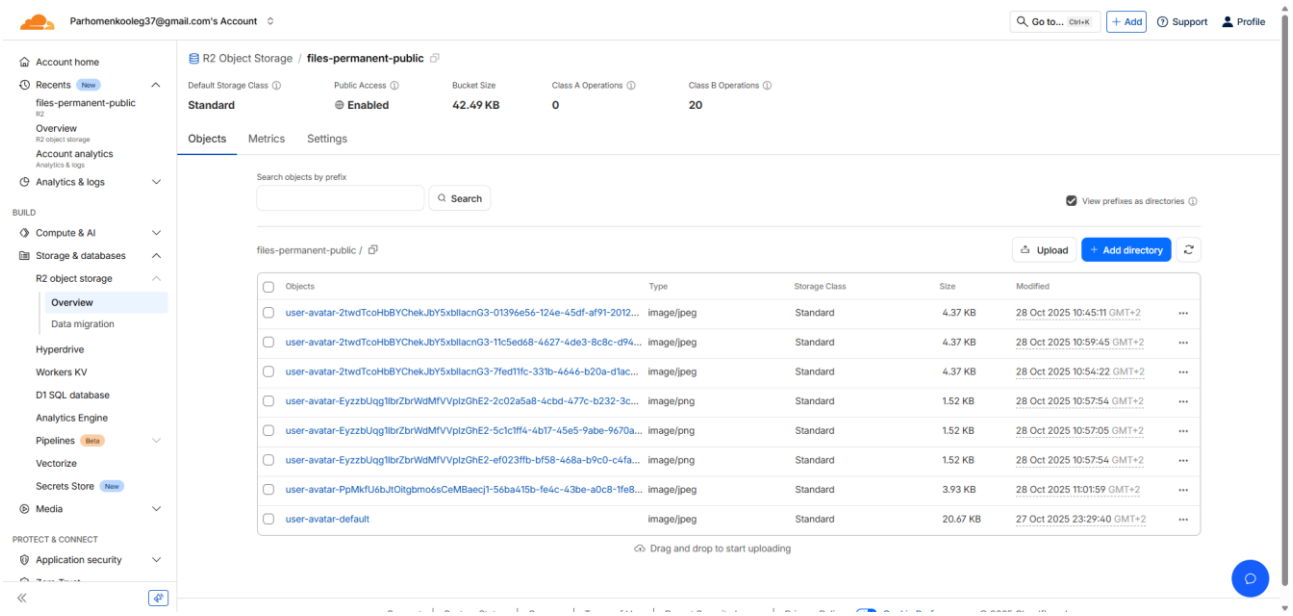

Рис. 3.4. Збережені зображення аватару користувачів на платформі Cloudflare

Для аналізу поведінки користувачів та покращення користувацького досвіду (UX) у систему інтегровано інструмент Hotjar. Він дозволяє створювати теплові карти кліків та записувати сеанси взаємодії відвідувачів із інтерфейсом. Аналіз цих даних дає можливість виявляти проблемні зони у навігації, розуміти, на яких етапах користувачі стикаються з труднощами, та приймати обґрунтовані рішення щодо оптимізації інтерфейсу в наступних ітераціях розробки.

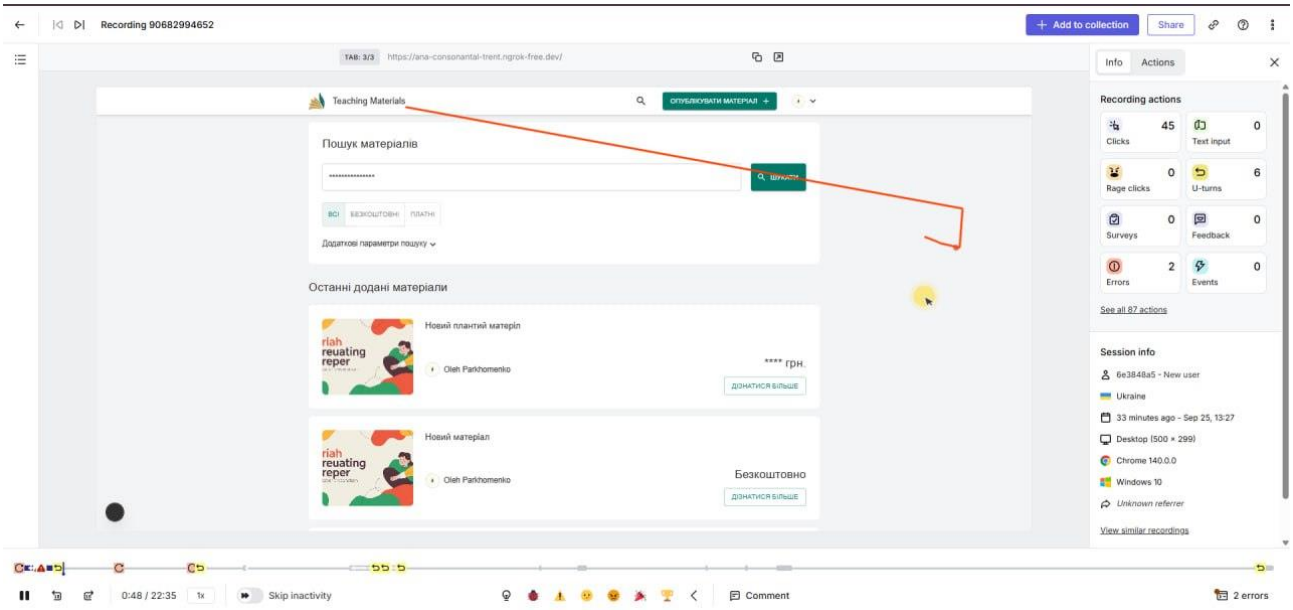

Рис. 3.5. Приклад принципу роботи Hotjar та слідкування за курсором

Загальна архітектура системи будується за принципом односторінкового застосунку, де клієнтська частина на React завантажується один раз, а подальший обмін даними з сервером на Node.js відбувається через REST API у фоновому режимі. Такий підхід, у поєднанні з обраним стеком технологій, дозволяє створити швидкий, масштабований та зручний у використанні освітній ресурс, що повністю відповідає сформульованим функціональним та нефункціональним вимогам.

3.3. Проєктування структури бази даних та інформаційної моделі

Програмна реалізація серверної частини веб-агрегатора виконана на базі платформи Node.js із використанням бібліотеки Mongoose, яка забезпечує об’єктно-документне відображення для взаємодії з базою даних. Такий підхід дозволив описати структуру даних на рівні коду застосунку, забезпечивши сувору типізацію та валідацію даних перед їх збереженням у базу, що є критичним для підтримки цілісності інформаційної системи.

Реалізація моделі користувача спроєктована з урахуванням використання зовнішнього провайдера автентифікації. Ключовим ідентифікатором виступає поле uid типу String, яке зберігає унікальний токен, отриманий від сервісу Firebase Authentication. Таке архітектурне рішення дозволяє розділити логіку безпеки та зберігання профілю: паролі та сесії обробляються на стороні

провайдера, тоді як локальна база даних зберігає лише публічні метадані, такі як відображуване ім'я, опис та посилання на аватар. Для зберігання контактної інформації використано гнучку структуру масиву об'єктів, що дозволяє динамічно додавати різні типи соціальних мереж або месенджерів без зміни схеми бази даних.

Рис. 3.6. Приклад запису користувача в базі даних

Центральним компонентом системи є реалізація моделі освітнього матеріалу, яка технічно втілює концепцію агрегатора, обґрунтовану у другому розділі. Схема документа містить поле materialUrl, призначене для зберігання прямого посилання на зовнішнє джерело контенту, що звільняє сервер від необхідності зберігати «важкі» файли. Візуальна складова реалізована через масив materialImages, який містить шляхи до попередньо завантажених та оптимізованих зображень-прев’ю. Для забезпечення зв’язності даних використано посилання на документ автора через ObjectId, що дозволяє ефективно виконувати операції об’єднання колекцій (population) при формуванні видачі каталогу.

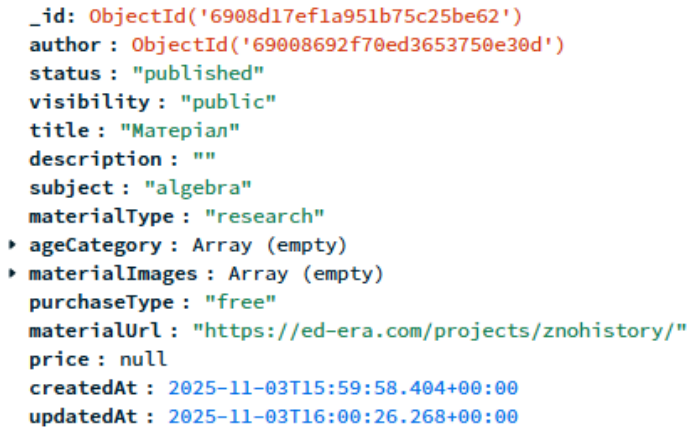

Рис. 3.6. Приклад запису матеріалу в базі даних

Для гарантування якості даних у схемі матеріалів використано механізм перерахувань (enums) для полів статусу, видимості та типу покупки. Це обмежує можливі значення набору даних заздалегідь визначеними константами, унеможливлюючи появу некоректних станів системи. Класифікація ресурсів здійснюється через набір текстових полів для предмету, типу матеріалу та вікової категорії, що надалі використовується для фасетної фільтрації.

Окремої уваги заслуговує реалізація пошукового механізму. Замість використання стандартних, але повільних операторів пошуку за регулярними виразами, у системі імплементовано функціонал повнотекстового пошуку на базі MongoDB Atlas Search. Для цього на рівні коду описано конфігурацію пошукового індексу, який автоматично створюється або оновлюється при запуску сервера. Конфігурація використовує аналізатор lucene.keyword для точного співпадіння полів фільтрації (категорія, статус) та механізм autocomplete для полів заголовка та опису. Такий підхід забезпечує миттєву видачу релевантних результатів навіть при неповному введенні пошукового запиту користувачем, значно підвищуючи юзабіліті системи та швидкість доступу до навчальних матеріалів.

3.4. Реалізація серверної частини та логіки агрегації контенту

Програмна реалізація серверної частини веб-агрегатора виконана на базі платформи Node.js із використанням фреймворку Express.js, який забезпечує маршрутизацію HTTP-запитів та обробку бізнес-логіки. Взаємодія з базою даних реалізована за допомогою бібліотеки Mongoose, що надає інструменти об’єктно-документного відображення. Використання Mongoose дозволило формалізувати структуру даних на рівні програмного коду, забезпечивши сувору типізацію та валідацію інформації перед її збереженням, що є критичним аспектом для підтримання цілісності інформаційної системи.

В основі підсистеми даних лежать дві ключові моделі: модель користувача та модель матеріалу. Реалізація схеми користувача спроєктована з урахуванням інтеграції із зовнішнім провайдером автентифікації Firebase. Ключовим ідентифікатором у системі виступає поле uid типу String, яке зберігає унікальний

токен користувача. Таке архітектурне рішення дозволяє чітко розділити зони відповідальності: питання безпеки паролів та сесій делегуються спеціалізованому сервісу, тоді як локальна база даних зберігає публічні метадані профілю, такі як відображуване ім'я, опис та посилання на аватар. Гнучкість NoSQL-підходу використана для зберігання контактної інформації у вигляді масиву об'єктів, що дозволяє користувачеві динамічно додавати різні канали зв'язку без необхідності зміни структури бази даних.

Центральним компонентом серверної логіки є реалізація моделі освітнього матеріалу, яка технічно втілює концепцію агрегації контенту. Схема документа містить поле materialUrl, призначене для зберігання прямого посилання на зовнішнє джерело, що звільняє серверну інфраструктуру від навантаження, пов'язаного зі зберіганням та роздачею «важких» файлів. Візуальна репрезентація ресурсу реалізована через масив materialImages, який містить шляхи до попередньо оброблених зображень-прев’ю. Для забезпечення реляційної зв’язності даних використано посилання на документ автора через тип ObjectId, що дозволяє ефективно виконувати операції об’єднання колекцій при формуванні відповіді API. Для гарантування консистентності даних у схемі використано механізм перерахувань для полів статусу, видимості та типу покупки, що унеможливлює перехід системи у невизначений стан.

Особливої уваги заслуговує реалізація механізму пошуку. Замість використання стандартних операторів запитів, які є неефективними на великих обсягах текстових даних, у системі імплементовано функціонал повнотекстового пошуку на базі технології MongoDB Atlas Search. На рівні програмного коду описано процедуру ініціалізації пошукового індексу, яка виконується при запуску сервера. Конфігурація індексу використовує різні типи аналізаторів для різних полів: lucene.keyword для точного співпадіння фільтрів (категорія, статус) та autocomplete для полів заголовка та опису. Це забезпечує реалізацію функціоналу «живого пошуку», коли релевантні результати пропонуються користувачеві вже в процесі введення запиту, значно підвищуючи ергономіку системи.

Бізнес-логіка агрегації реалізована через набір контролерів API, які обробляють запити від клієнтської частини. Процес додавання нового матеріалу відбувається у кілька етапів. Спочатку сервер приймає метадані та файл зображення, завантажений автором. Після успішного збереження зображення у хмарному сховищі формується документ матеріалу, до якого записуються посилання на зображення та зовнішній ресурс. Перед записом у базу даних виконується валідація URL-адреси, щоб уникнути збереження некоректних або шкідливих посилань.

Захист API-ендпоінтів реалізовано за допомогою проміжного програмного забезпечення, яке перевіряє валідність токена авторизації в заголовках запиту. Сервер верифікує токен через Admin SDK Firebase, отримуючи унікальний ідентифікатор користувача, та надає доступ до захищених ресурсів лише у разі успішної перевірки. Така модель забезпечує надійний захист від несанкціонованого доступу та підміни даних. Для забезпечення стабільності роботи сервера впроваджено глобальний обробник помилок, який перехоплює виключення на всіх рівнях обробки запиту та повертає клієнту стандартизовану відповідь з описом проблеми, приховуючи при цьому деталі реалізації з міркувань безпеки.

3.5. Розробка клієнтської частини та інтерфейсу користувача

Практична реалізація клієнтської частини веб-агрегатора освітніх матеріалів базується на використанні сучасного фреймворку Next.js, який є надбудовою над бібліотекою React. Вибір саме цього технологічного рішення зумовлений необхідністю забезпечення високої продуктивності, SEO-оптимізації та підтримки гібридного рендерингу сторінок. Архітектура фронтенд-застосунку спроєктована за модульним принципом, що передбачає поділ інтерфейсу на незалежні, повторно використовувані компоненти, логіка яких інкапсульована всередині окремих модулів. Такий підхід дозволив значно підвищити швидкість розробки, спростити процес тестування та забезпечити легку масштабованість проєкту в майбутньому.

Структурна організація проєкту відповідає стандартам Next.js і базується на файловій системі маршрутизації. Логіка застосунку розділена на декілька ключових директорій: сторінки, компоненти, сервіси для взаємодії з API та контексти для керування глобальним станом. Для забезпечення цілісності користувацького досвіду реалізовано компонент-обгортку макету, який містить наскрізні елементи інтерфейсу, такі як «хедер» та «футер». Завдяки механізму збереження стану в React, при переході між сторінками ці елементи не перезавантажуються, що створює відчуття роботи з нативним застосунком і зменшує навантаження на мережу.

Візуальна складова системи розроблена з використанням підходу Mobile First, що передбачає першочергове проєктування інтерфейсу для мобільних пристроїв із подальшою адаптацією для планшетів та десктопів. Для стилізації використано CSS-фреймворк Tailwind CSS, який реалізує методологію utility-first. Це дозволило відмовитися від традиційних каскадних таблиць стилів на користь декларування дизайну безпосередньо в розмітці JSX. Такий підхід не лише пришвидшує верстку, але й дозволяє автоматично видаляти невикористані стилі під час збірки проєкту, що критично важливо для мінімізації розміру бандлу та пришвидшення завантаження сторінок на мобільних мережах.

Центральним елементом інтерфейсу каталогу є компонент картки матеріалу, який відповідає за візуалізацію метаданих ресурсу. Реалізація цього компонента включає відображення назви, категорії, найголовніше, зображення-прев’ю, завантаженого автором (рис 5.1). Для оптимізації роботи з графікою використано вбудований компонент Next.js Image (ліст. 5.1), який автоматично конвертує зображення у сучасний формат WebP та генерує кілька варіантів роздільної здатності для різних пристроїв. Додатково застосовано техніку «лінивого завантаження», завдяки якій зображення завантажуються лише тоді, коли потрапляють у зону видимості екрану користувача, що суттєво покращує показники Web Vitals.

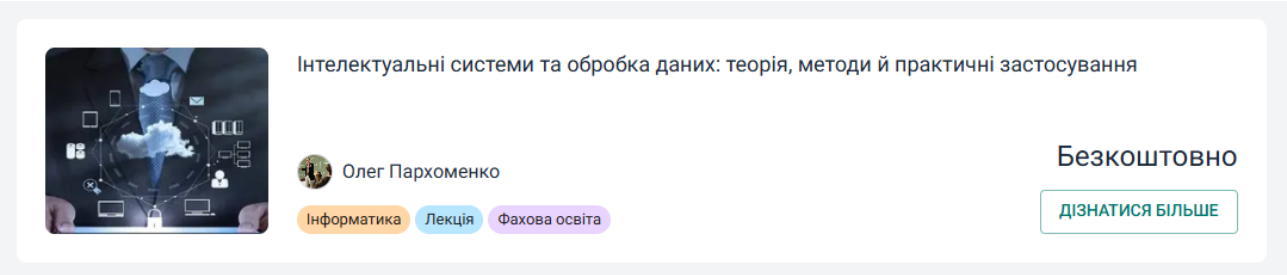

Рис 5.1. Візуалізація інформаційної картки матеріалу

```
<Image
    fill={true}
    sizes="360px"
    alt={`${material.title} Preview`}
    className="object-cover rounded-lg"
    src={material.materialImages[0].url}
/>
```

Лістинг 5.1. Приклад використання Next.js Image для завантаження прев'ю матеріалу

Взаємодія клієнтської частини з серверним API реалізована через шар сервісів, який використовує бібліотеку Axios для виконання асинхронних HTTP-запитів. Управління фільтрацією та пошуком у каталозі базується на синхронізації стану інтерфейсу з параметрами URL-адреси. Це дозволяє користувачам ділитися посиланнями на конкретні результати пошуку та коректно використовувати історію браузера для навігації.

Окремої уваги заслуговує реалізація механізму публікації нових матеріалів. Розроблена форма додавання ресурсу містить інтерактивні поля для введення метаданих та спеціалізований віджет для завантаження обкладинки (рис 5.2). Для забезпечення валідності даних на стороні клієнта використано бібліотеку Formik у поєднанні зі схемою валідації Yup. Це дозволяє миттєво перевіряти коректність введених даних, наприклад, формат URL-посилання, ще до відправки запиту на сервер. Для покращення UX реалізовано функцію попереднього перегляду зображення: обраний файл зчитується браузером і відображається у формі миттєво, що дає автору впевненість у коректності вибору файлу.

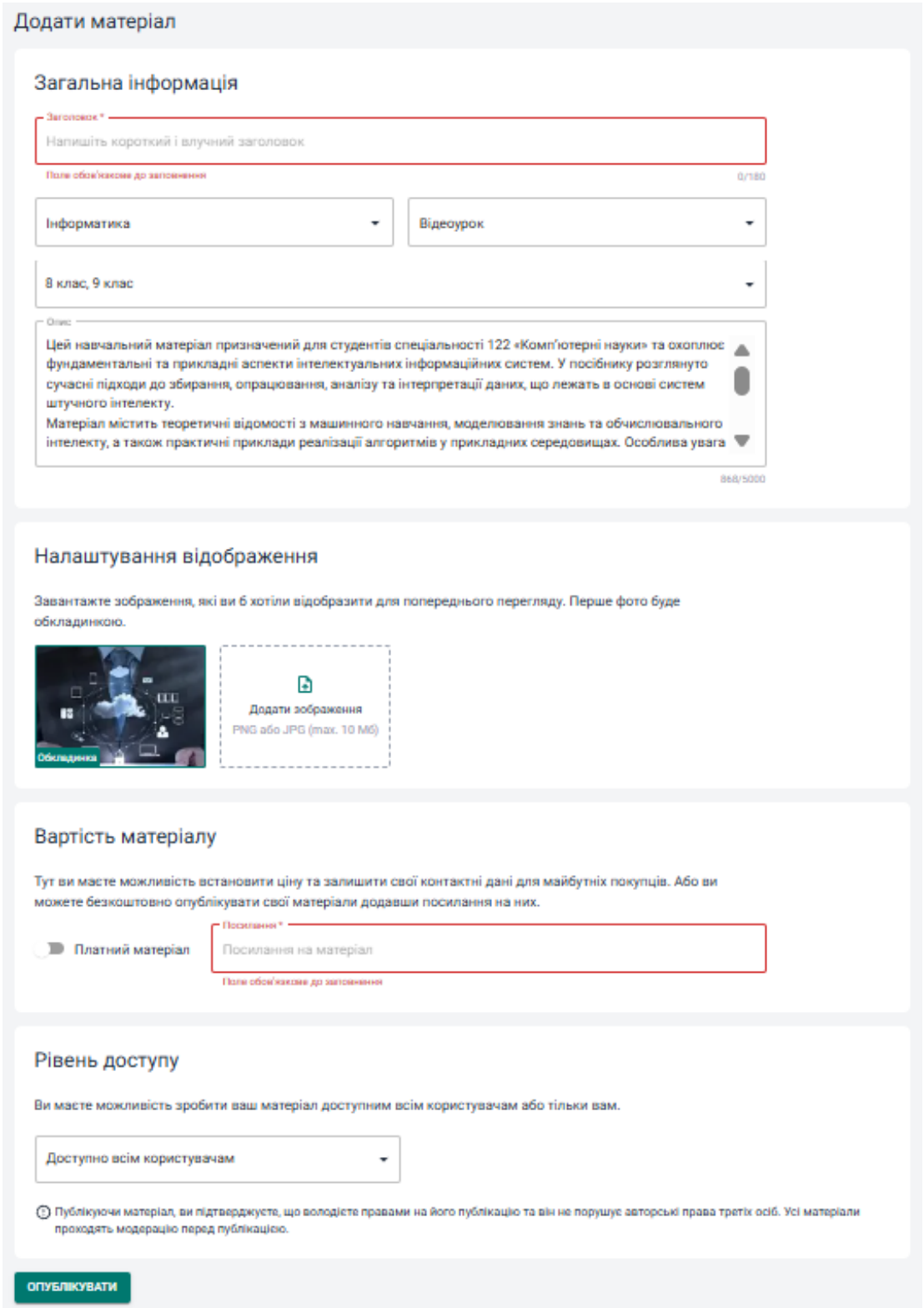

Рис. 5.2. Заповнення форми для публікації матеріалу

Система автентифікації та авторизації інтегрована у застосунок через React Context API. Цей глобальний контекст взаємодіє з сервісом Firebase Authentication, відслідковуючи статус сесії користувача в реальному часі (рис 5.3).

```
const loginWithEmailAndPassword = async (email, password) : Promise<...> => {
    return await withFirebaseHandler(
        fn: async () : Promise<{...}> => {
            const userCredentials : UserCredential = await signInWithEmailAndPassword(firebaseClient.auth, email, password);
            if (!userCredentials.user.emailVerified) {
                await sendVerificationEmail(userCredentials.user);
                await logout();
                return { shouldVerify: true };
            }

            await login();
            return { shouldVerify: false };
        },
        {t}: { t },
    );
};
```

Рис. 5.3. Функція авторизації користувача за email та паролем у Firebase

Залежно від стану авторизації інтерфейс динамічно змінюється: для гостей відображаються кнопки входу та реєстрації, а доступ до захищених маршрутів, таких як сторінка додавання матеріалу, блокується з автоматичним перенаправленням на сторінку логіну. Для авторизованих користувачів стають доступними персоналізовані елементи меню та функції управління контентом. Така глибока інтеграція фронтенду з логікою безпеки та даними забезпечує надійну, швидку та зручну роботу веб-агрегатора, повністю відповідаючи сучасним вимогам до веб-розробки.

3.6. Тестування програмного продукту та оцінка якості реалізації

Завершальним етапом життєвого циклу розробки програмного забезпечення є комплексне тестування, метою якого є верифікація відповідності реалізованого продукту сформульованим функціональним та нефункціональним вимогам. Для перевірки веб-агрегатора освітніх матеріалів було обрано стратегію комбінованого тестування, що охоплює модульну перевірку окремих компонентів, інтеграційне тестування взаємодії клієнта з сервером та системне тестування продуктивності. Такий підхід дозволяє виявити дефекти на різних рівнях абстракції та гарантувати стабільність роботи системи перед її розгортанням у продуктивному середовищі.

Першим етапом стало тестування серверного API (Back-end Testing). Оскільки серверна частина реалізує критичну бізнес-логіку агрегації та автентифікації, для її перевірки було використано інструментарій Postman. Було

створено колекцію тестових запитів, що емулюють усі можливі сценарії взаємодії клієнта із сервером: реєстрацію нового користувача, вхід у систему, додавання нового матеріалу, пошук за фільтрами та отримання даних для конкретної сторінки. Особлива увага приділялася тестуванню механізмів захисту маршрутів. Перевірялася реакція сервера на запити без токена авторизації, з простроченим токеном або з токеном користувача, який не має прав на редагування чужого контенту. Результати тестування підтвердили коректність роботи middleware-функцій: сервер стабільно повертає відповідні HTTP-статуси помилок (401 Unauthorized, 403 Forbidden) при спробах несанкціонованого доступу, що свідчить про надійність реалізованої системи безпеки.

Наступним кроком стало функціональне тестування клієнтської частини (Front-end Testing). Основний акцент було зроблено на перевірці коректності роботи форм введення даних, зокрема форми додавання нового матеріалу. Тестування підтвердило ефективність валідаційних схем Yup: система коректно блокує відправку форми при введенні невалідних URL-адрес або спробі завантаження файлів недопустимих форматів, миттєво відображаючи зрозумілі повідомлення про помилки. Також було перевірено логіку роботи динамічних фільтрів каталогу та збереження стану інтерфейсу при перезавантаженні сторінки, що забезпечується завдяки синхронізації з параметрами URL.

Критично важливим для веб-агрегатора є тестування адаптивності та кросбраузерності. Використовуючи інструменти розробника Google Chrome DevTools (рис 5.4), було проведено емуляцію роботи застосунку на широкому спектрі пристроїв: від мобільних телефонів з невеликими екранами (iPhone SE) до планшетів та широкоформатних моніторів. Перевірка показала, що завдяки використанню фреймворку Tailwind CSS інтерфейс коректно адаптується під будь-яку ширину екрану: сітка карток матеріалів автоматично перебудовується, навігаційне меню трансформується у мобільний формат, а зображення оптимізуються під роздільну здатність пристрою.

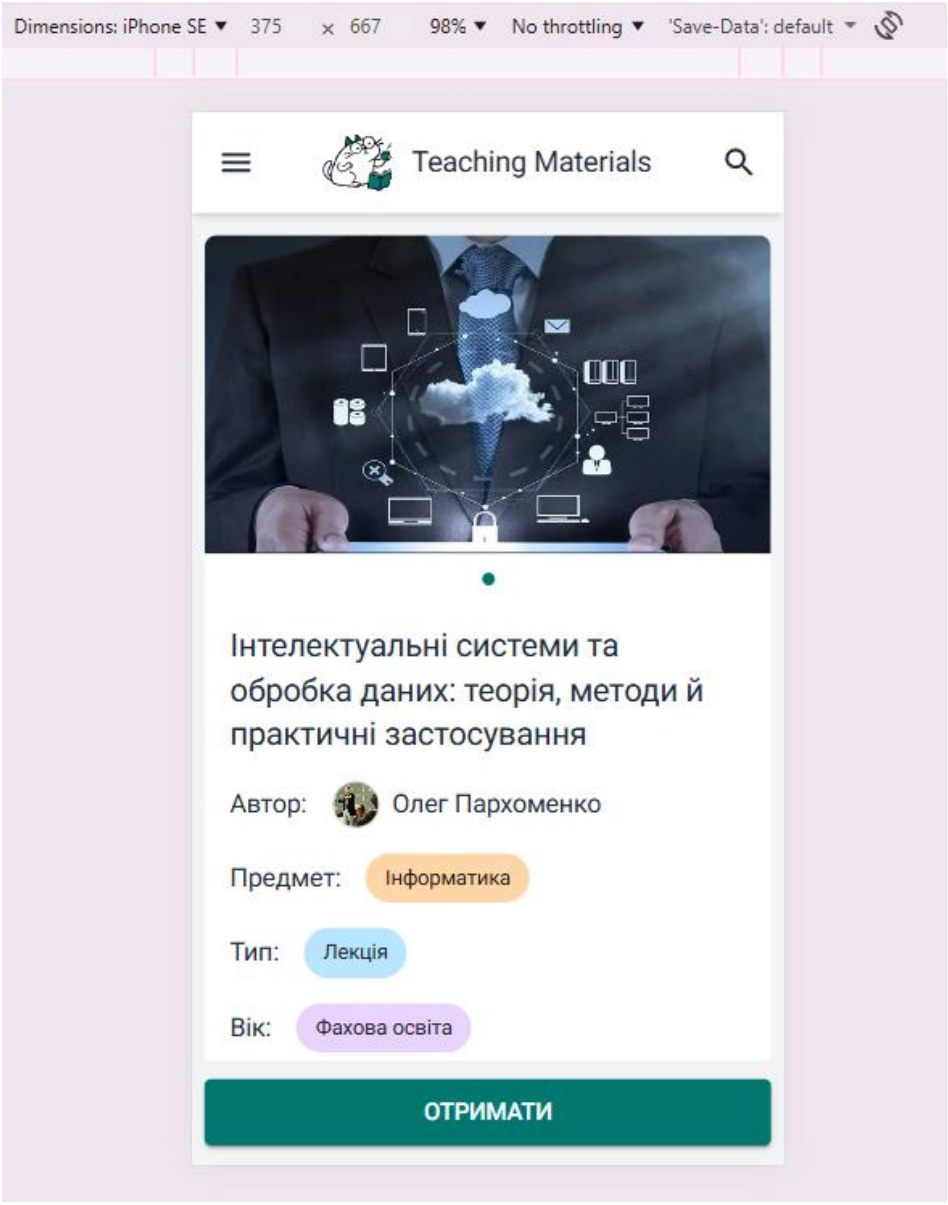

Рис. 5.4. Тестування відображення на телефоні типу Iphone SE з використанням Google Chrome DevTools

Таким чином, проведене комплексне тестування підтвердило працездатність розробленого програмного забезпечення, його стабільність та відповідність сучасним стандартам веб-розробки. Система готова до етапу розгортання (деплойменту) та подальшої експлуатації користувачами.

ВИСНОВКИ

У ході виконання кваліфікаційної магістерської роботи було проведено комплексне дослідження та практичну реалізацію веб-платформи для обміну освітніми матеріалами, що базується на сучасних цифрових технологіях та архітектурних підходах. Основною метою роботи було створення ефективного, масштабованого та зручного інструменту, який вирішує проблеми існуючих рішень на ринку, зокрема у сегменті вищої освіти. У результаті проведеного теоретичного аналізу сучасних тенденцій цифровізації освіти було встановлено, що поряд із класичними системами управління навчанням та платформами масових відкритих онлайн-курсів, дедалі більшого значення набувають відкриті репозиторії та маркетплейси освітнього контенту. Визначено, що ключовими викликами при розробці таких систем є забезпечення захисту авторських прав, технічна доступність для користувачів з різними пристроями, ефективність пошуку великих обсягів даних та зручність інтерфейсу, який не повинен створювати надмірного когнітивного навантаження.

Важливим етапом дослідження став порівняльний аналіз провідних українських платформ, таких як «Всеосвіта» та «На Урок», який виявив низку суттєвих недоліків у їхній організації. Зокрема, було ідентифіковано проблему перевантаженості інтерфейсу зайвими функціями, відсутність належної спеціалізації для закладів вищої освіти та використання застарілої моделі безпосереднього хостингу файлів, що призводить до високих витрат на інфраструктуру. На основі цих висновків було обґрунтовано доцільність розробки власного застосунку, що функціонує за моделлю агрегатора контенту. Такий архітектурний підхід передбачає зберігання посилань на зовнішні хмарні ресурси замість фізичних файлів, що дозволяє суттєво знизити навантаження на сервер, зменшити витрати на зберігання даних та спростити юридичні аспекти захисту авторських прав, оскільки контроль над оригіналом файлу залишається у власника.

Практична реалізація проєкту полягала у проєктуванні та розробці повнофункціонального прототипу веб-застосунку з використанням сучасного стеку технологій. В якості технологічної основи було обрано екосистему JavaScript, що дозволило уніфікувати розробку клієнтської та серверної частин. Серверна логіка реалізована на базі платформи Node.js та фреймворку Express.js, що забезпечує високу продуктивність обробки запитів. Для зберігання даних обрано документоорієнтовану базу даних MongoDB, гнучка схема якої ідеально підходить для роботи з різнорідними метаданими освітніх ресурсів. Клієнтська частина розроблена на основі фреймворку Next.js, використання якого дозволило реалізувати технологію серверного рендерингу, що є критично важливим фактором для забезпечення пошукової оптимізації та швидкого першого відображення контенту.

Для підвищення рівня безпеки та зручності користувачів у систему було інтегровано низку хмарних сервісів. Зокрема, автентифікація користувачів реалізована за допомогою Firebase Authentication, що гарантує надійний захист облікових записів, а захист від веб-атак та кешування контенту забезпечується сервісом Cloudflare. Особливу увагу було приділено реалізації ефективної системи пошуку, яка базується на технології MongoDB Atlas Search

Результати комплексного тестування, що включало модульну перевірку компонентів, інтеграційне тестування API та перевірку користувацького інтерфейсу, підтвердили стабільність та надійність розробленої системи. Практична новизна отриманих результатів полягає у створенні архітектурно легкого веб-агрегатора, який, на відміну від громіздких аналогів, фокусується на швидкості роботи, мінімалізмі інтерфейсу та безшовній інтеграції з існуючими хмарними сховищами користувачів. Розроблений програмний продукт повністю готовий до розгортання та подальшої експлуатації. Перспективи подальшого розвитку системи вбачаються у розширенні соціальних функцій взаємодії, впровадженні алгоритмів машинного навчання для персоналізації рекомендацій та використання штучного інтелекту для аналізу та обробки завантажений матеріалів для подальшої їх модерації.

ПЕРЕЛІК ПОСИЛАНЬ

1. Биков В. Ю. Цифрова трансформація суспільства і розвиток комп’ютерно-технологічної платформи освіти і науки України / В. Ю. Биков // Інформаційно-цифровий освітній простір України: трансформаційні процеси і перспективи розвитку. — 2019. https://lib.iitta.gov.ua/id/eprint/718692/1/Microsoft%20Word%20-%20%D0%91%D0%B8%D0%BA%D0%BE%D0%B2%20%D0%92_2019_2.pdf
2. Dublin Core Metadata Element Set, Version 1.1. https://www.dublincore.org/specifications/dublin-core/dces/
3. Всеосвіта: Національна освітня платформа. https://vseosvita.ua/
4. На Урок: освітній проект для вчителів. https://naurok.com.ua/
5. Web Content Accessibility Guidelines. https://www.w3.org/TR/WCAG21/
6. The Open Graph protocol. https://ogp.me/
7. Next.js Documentation. https://nextjs.org/docs
8. A JavaScript library for building user interfaces. https://react.dev/
9. Tailwind CSS. Rapidly build modern websites without ever leaving your HTML. https://tailwindcss.com/
10. Node.js. JavaScript runtime built on Chrome's V8 JavaScript engine https://nodejs.org/uk
11. MongoDB Documentation. https://www.mongodb.com/docs/
12. Firebase Authentication Documentation https://firebase.google.com/docs/auth
13. Cloudflare CDN. Content Delivery Network https://www.cloudflare.com/learning/cdn/what-is-a-cdn/
14. Закон України «Про освіту» від 05.09.2017 № 2145-VIII. https://zakon.rada.gov.ua/laws/show/2145-19#Text

15. Стратегія розвитку вищої освіти в Україні на 2022–2032 роки, схвалена розпорядженням Кабінету Міністрів України від 23 лютого 2022 р. № 286-р. https://zakon.rada.gov.ua/laws/show/286-2022-%D1%80#Text
16. Закон України «Про захист персональних даних» від 01.06.2010 № 2297-VI. https://zakon.rada.gov.ua/laws/show/2297-17#Text
17. Fielding R. T. Architectural Styles and the Design of Network-based Software Architectures : Dissertation / Roy Thomas Fielding. — Irvine : University of California, 2000.
18. Google Developers. Web Vitals: Essential metrics for a healthy site. https://web.dev/articles/vitals
19. MDN Web Docs. HTTP response status codes. https://developer.mozilla.org/en-US/docs/Web/HTTP/Reference/Status
20. Chacon S. Pro Git / Scott Chacon, Ben Straub. — Apress, 2014.
21. Martin R. C. Clean Code: A Handbook of Agile Software Craftsmanship / Robert C. Martin. — Prentice Hall, 2008.
22. Спірін О. М. Інформаційно-комунікаційні та цифрові технології в науковій діяльності : монографія / О. М. Спірін. — Київ : ІІТЗН НАПН України, 2019.
23. Кухаренко В. М. Дистанційне навчання : теорія і практика / В. М. Кухаренко, О. В. Рибалко, Н. Г. Сиротенко. — Харків : НТУ «ХПІ», 2018.
24. Affouneh S. The inevitable digitalization of education / S. Affouneh, S. Salha, Z. Khlaif // Jurnal Pendidikan. — 2020.
25. Kleppmann M. Designing Data-Intensive Applications: The Big Ideas Behind Reliable, Scalable, and Maintainable Systems / M. Kleppmann. — O'Reilly Media, 2017.
26. NoSQL vs SQL Databases: A Comparative Study / S. G. K. Patro et al. // International Journal of Engineering Research & Technology. — 2020.
27. Richardson C. Microservices Patterns: With examples in Java / Chris Richardson. — Manning Publications, 2018.

28. Banks A. Learning React: Modern Patterns for Developing React Apps / A. Banks, E. Porcello. — O'Reilly Media, 2020.
29. Freeman A. Pro React 16 / Adam Freeman. — Apress, 2019.
30. Warchol J. Next.js: The React Framework for Production, 2023.
31. Chinnathambi K. Learning Node.js: A Hands-On Guide to Building Web Applications in JavaScript / K. Chinnathambi. — Addison-Wesley Professional, 2016.
32. Krug S. Don't Make Me Think, Revisited: A Common Sense Approach to Web Usability / Steve Krug. — New Riders, 2014.
33. OWASP Top 10:2021. The Ten Most Critical Web Application Security Risks. https://owasp.org/Top10/