

КВАЛІФІКАЦІЙНА РОБОТА

Група ІПЗс 20-1
Король В.В

2024

ЗВО УНІВЕРСИТЕТ КОРОЛЯ ДАНИЛА

Факультет суспільних та прикладних наук

Кафедра інформаційних технологій

на правах рукопису

Король Володимир Васильович

УДК 004.4

Розробка веб-сайту для продажу та оренди нерухомості

Спеціальність 121 – «Інженерія програмного забезпечення»

Кваліфікаційна робота на здобуття кваліфікації бакалавр

Нормоконтроль

_____ Сτισло О.В.
(підпис, дата, розшифрування підпису)

Студент

_____ Король В.В.
(підпис, дата, розшифрування підпису)

Допускається до захисту

Завідувач кафедри

_____ к.т.н., доц. Ващишак С.П.
(підпис, дата, розшифрування підпису)

Керівник роботи

асистент каф. ІТ

_____ Шкатуляк В.В.
(підпис, дата, розшифрування підпису)

ЗВО УНІВЕРСИТЕТ КОРОЛЯ ДАНИЛА
Факультет суспільних та прикладних наук
Кафедра інформаційних технологій

Освітній ступінь: «бакалавр»

Спеціальність: 121 «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ

Завідувач кафедри

« ____ » _____ 2024 року

**ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ**

Король Володимир Васильович

(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи:

Розробка веб-сайту для продажу та оренди нерухомості

керівник роботи:

Шкатуляк Василь Васильович, асистент каф. ІТ

затверджена наказом вищого навчального закладу від « 12 » березня 2024 року

№ 19/1

2. Термін подання студентом роботи 05.06.2024

3. Вихідні дані роботи: мова програмування JavaScript, ReactJs, NextJs, HTML

4. Зміст кваліфікаційної роботи (перелік питань, які потрібно розробити)

1. Аналіз рішень в області розробки веб додатків

2. Обґрунтування вибору інструментів розробки

3. Реалізація веб додатку

5. Дата видачі завдання 14.03.2024

КОНСУЛЬТАНТИ РОЗДІЛІВ КВАЛІФІКАЦІЙНОЇ РОБОТИ

Розділ	Консультант (прізвище, ініціали та посада)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1.	Аналіз і вивчення аналогів	16.03.2024	Виконано
2.	Вибір мови програмування	20.03.2024	Виконано
3.	Вибір платформи для розробки	22.03.2024	Виконано
4.	Розроблення архітектури додатку	03.04.2024	Виконано
5.	Розробка веб додатку	28.04.2024	Виконано
6.	Перевірка та виправлення помилок	23.05.2024	Виконано
7.	Підготовка до захисту	24.05.2024	Виконано

Студент

(підпис)

Король В. В

(прізвище та ініціали)

Керівник роботи

(підпис)

Шкатуляк В. В

(прізвище та ініціали)

Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

Сторінка	Опис графічного матеріалу	Сторінка	Опис графічного матеріалу
21	Конфігурація TailwindCSS	23	Приклад готових діалогів та кнопки
25	Підключення до БД	26	Конфігурація NextAuth
28	Головна сторінка сервісу «DIMRIA»	29	Карта нерухомості України
29	Додавання оголошення про нерухомість	30	Головна сторінка веб-сайту «Flatfy»
31	Продаж квартири на веб-сайті «Flatfy»	31	Меню на сайті «Flatfy»
32	Веб-сайт для продажу нерухомості «Rent.ua»	33	Нерухомість на сайті «Rent.ua»
38	Інтеграція з мапами	38	Розділення фрагментів додатку на компоненти
40	Посилання на головній сторінці	41	Фільтри каталогу оголошень
42	Сторінка «Про нас»	44	Формування запити до БД з урахуванням сторінок
46	Передавання URL параметрів із клієнтської частини у серверну	47	Зберігання зображень користувачів у БД Uploadthing

49	Інтерфейс управління хостингом додатку	51	Форма входу з можливістю використання облікового запису гугл
53	Коміти та опис проекту на GitHub	54	Інструменти моніторингу
56	Головна сторінка додатку	57	Навігаційне меню
58	Блок авторизації	59	Сторінка оголошення для мобільних пристроїв
60	Пагінація оголошень	61	Функції додавання та видалення пошукових параметрів
62	Отримання ідентифікатора та запит до API	62	Експорт метаданих, які дозволять покращити SEO
64	Додавання нового оголошення	65	Зберігання Environment Variables всередині .env

АНОТАЦІЯ

У даній кваліфікаційній роботі розглядається розробка веб-сайту "Hive" для продажу та оренди нерухомості з використанням сучасних технологій. Мета роботи полягає у створенні ефективної та зручної платформи, що забезпечує швидкий доступ до інформації про нерухомість, полегшує процес купівлі, продажу та оренди, а також гарантує безпеку та надійність обробки даних.

Для досягнення мети було використано такі технології: react для створення інтерфейсу користувача, NextJS для серверного рендерингу та маршрутизації, TypeScript для статичної типізації, TailwindCSS для стилізації, Shadcn UI та Radix UI для готових компонентів, Framer Motion для анімацій, Lucide-react для іконок, NextAuth для автентифікації та авторизації користувачів, MongoDB для зберігання даних, Prisma ORM для роботи з базою даних, react-hook-form та zod для управління формами, GitHub для контролю версій, Uploadthing для завантаження зображень, Vercel для розгортання на сервері та VS Code для розробки коду.

У роботі було спроектовано архітектуру додатку, що включає клієнтську та серверну частини, роутинг та внутрішній API для взаємодії з базою даних. Реалізовано основні роути: головну сторінку, каталог оголошень, сторінку конкретного оголошення, сторінку про компанію, контакти, профіль користувача та авторизацію. Впроваджено систему авторизації користувачів з можливістю входу через Google або власні дані.

Проведено тестування додатку, виявлено та виправлено помилки, оптимізовано продуктивність. Веб-сайт був розгорнутий на платформі Vercel та наданий для пілотного використання, що підтвердило його функціональність та відповідність потребам користувачів.

Результати роботи мають значне практичне значення, сприяють розвитку ринку нерухомості, підвищують ефективність та зручність для користувачів та професіоналів, підтримують інновації та технологічний прогрес у галузі

веб-розробки. Апробація результатів дослідження була проведена через тестування прототипу, презентації на конференціях, публікації, пілотне впровадження та відгуки користувачів.

КЛЮЧОВІ СЛОВА: ВЕБ-САЙТ, ОРЕНДА НЕРУХОМОСТІ, ПРОДАЖ НЕРУХОМОСТІ, ПРОДАЖ ЖИТЛА, КУПІВЛЯ ЖИТЛА, ОРЕНДА ЖИТЛА, ПЛАТФОРМА НЕРУХОМОСТІ, РИНОК ЖИТЛА, ПРОФІЛІ КОРИСТУВАЧІВ, ВЕБ-РОЗРОБКА

SUMMARY

This qualification work explores the development of the "Hive" website for real estate sales and rentals using modern technologies. The aim of the work is to create an efficient and user-friendly platform that provides quick access to real estate information, facilitates the process of buying, selling, and renting, and ensures data security and reliability.

To achieve this goal, the following technologies were used: React for creating the user interface, NextJS for server-side rendering and routing, TypeScript for static typing, TailwindCSS for styling, Shadcn UI and Radix UI for ready-made components, Framer Motion for animations, Lucide-react for icons, NextAuth for user authentication and authorization, MongoDB for data storage, Prisma ORM for database interaction, react-hook-form and zod for form management, GitHub for version control, Uploadthing for image uploading, Vercel for server deployment, and VS Code for code development.

The work involved designing the architecture of the application, which includes the client and server parts, routing, and internal API for database interaction. The main routes were implemented: the home page, listings catalog, individual listing page, company information page, contacts, user profile, and authentication. A user authentication system was introduced, allowing users to log in via Google or their own credentials.

The application was tested, errors were identified and corrected, and performance was optimized. The website was deployed on the Vercel platform and provided for pilot use, confirming its functionality and user needs compliance.

The results of this work have significant practical value, contribute to the development of the real estate market, enhance efficiency and convenience for users and professionals, and support innovation and technological progress in web development. The results of the research were tested through prototype testing, presentations at conferences, publications, pilot implementation, and user feedback.

KEYWORDS: WEBSITE, REAL ESTATE RENTAL, SALE OF REAL ESTATE, SELL HOUSING, BUY HOUSING, RENT HOUSING, REAL ESTATE PLATFORM, HOUSING MARKETPLACE, USER PROFILES, WEB DEVELOPMENT

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ.....	11
ВСТУП.....	13
РОЗДІЛ 1. ВИЗНАЧЕННЯ ВИМОГ ДО ПРОЕКТУ ТА АНАЛІЗ НАЯВНИХ РІШЕНЬ.....	17
1.1 Вступ до технологій.....	17
1.2 Використання TailwindCSS.....	20
1.3 Готові компоненти UI.....	22
1.4 Робота з MongoDB та безпека.....	24
1.5 Авторизація з NextAuth.....	25
1.6 Сервіси-аналоги.....	26
1.7 Постановка задачі.....	33
Висновки до розділу 1.....	34
РОЗДІЛ 2. ОБҐРУНТУВАННЯ ВИБОРУ ТЕХНОЛОГІЙ РОЗРОБКИ.....	36
2.1 Архітектурний підхід.....	36
2.2 Основні функціональні можливості.....	39
2.3 Фільтрація та пагінація.....	43
2.4 Інтеграція зі сторонніми сервісами.....	47
2.5 Механізми авторизації.....	51
2.6 Керування версіями та розгортання.....	52
Висновки до розділу 2.....	55
РОЗДІЛ 3. РЕАЛІЗАЦІЯ ВЕБ-ДОДАТКА.....	56
3.1 Розробка головної сторінки.....	56
3.2 Реалізація каталогу оголошень.....	59

	10
3.3 Реалізація авторизації.....	63
3.4 Тестування додатку.....	64
Висновки до розділу 3.....	66
ВИСНОВКИ.....	67
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	68
ДОДАТКИ.....	70
Додаток А.....	70
Додаток Б.....	71
Додаток В.....	77

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

SPA – Single Page Application (Односторінковий додаток)

UI – User Interface (Користувацький інтерфейс)

API – Application Programming Interface (Програмний інтерфейс додатку)

ORM – Object-Relational Mapping (Об'єктно-реляційне відображення)

CRUD – Create, Read, Update, Delete (Створення, Читання, Оновлення, Видалення)

DB – Database (База даних)

Auth – Authentication (Автентифікація)

ID – Identifier (Ідентифікатор)

CSS – Cascading Style Sheets (Каскадні таблиці стилів)

HTML – HyperText Markup Language (Мова розмітки гіпертексту)

JS – JavaScript (Мова програмування JavaScript)

TS – TypeScript (Мова програмування TypeScript)

VS Code – Visual Studio Code (Редактор коду Visual Studio)

JSON – JavaScript Object Notation (Нотація об'єктів JavaScript)

HTTP – HyperText Transfer Protocol (Протокол передачі гіпертексту)

HTTPS – HyperText Transfer Protocol Secure (Захищений протокол передачі гіпертексту)

UI/UX – User Interface/User Experience (Користувацький інтерфейс/Користувацький досвід)

SEO – Search Engine Optimization (Оптимізація для пошукових систем)

JWT – JSON Web Token (Веб-токен на основі JSON)

DOM – Document Object Model (Об'єктна модель документа)

SDK – Software Development Kit (Набір засобів для розробки програмного забезпечення)

CDN – Content Delivery Network (Мережа доставки контенту)

CI/CD – Continuous Integration/Continuous Deployment (Безперервна інтеграція/Безперервне розгортання)

MVC – Model-View-Controller (Модель-Вид-Контролер)

REST – Representational State Transfer (Передача стану представлення)

CRUD – Create, Read, Update, Delete (Створення, Читання, Оновлення, Видалення)

SSR – Server-Side Rendering (Рендеринг на стороні сервера)

CSR – Client-Side Rendering (Рендеринг на стороні клієнта)

SaaS – Software as a Service (Програмне забезпечення як послуга)

PaaS – Platform as a Service (Платформа як послуга)

IaaS – Infrastructure as a Service (Інфраструктура як послуга)

ВСТУП

Актуальність теми. Актуальність теми розробки веб-сайту для продажу та оренди нерухомості зумовлена кількома важливими аспектами. По-перше, ринок нерухомості є одним із найбільших та найважливіших секторів економіки. Зі зростанням міського населення та підвищенням попиту на житлові та комерційні приміщення, потреба у зручних і ефективних платформах для купівлі, продажу та оренди нерухомості стає все більш актуальною. У сучасному світі цифрові технології відіграють ключову роль у розвитку бізнесу. Традиційні способи пошуку та купівлі нерухомості поступаються місцем онлайн-платформам, які пропонують більш зручні та ефективні рішення для користувачів. Веб-сайти для нерухомості дозволяють швидко знайти потрібну інформацію, порівняти різні варіанти та прийняти обґрунтоване рішення.

Розробка інтуїтивно зрозумілих і функціональних веб-сайтів полегшує процес взаємодії користувачів з ринком нерухомості. Використання сучасних технологій забезпечує високу продуктивність і гарний користувацький досвід, що є важливим для залучення і утримання клієнтів. Веб-сайт для продажу та оренди нерухомості, що містить функціонал для авторизації, управління обліковим записом, фільтрації та сортування оголошень, значно покращує взаємодію між продавцями та покупцями, орендодавцями та орендарями. Це сприяє більш ефективному обміну інформацією та швидшому здійсненню угод. Поточні умови показали важливість наявності онлайн-рішень для різних аспектів життя, включаючи ринок нерухомості.

Таким чином, розробка веб-сайту для продажу та оренди нерухомості є актуальним і важливим завданням, що відповідає сучасним тенденціям ринку та технологічному прогресу, задовольняє потреби користувачів і сприяє розвитку галузі нерухомості.

Мета роботи. Метою даної кваліфікаційної роботи є розробка ефективного та зручного у користуванні веб-сайту для продажу та оренди

нерухомості, що відповідає сучасним вимогам ринку та технологічним тенденціям. Основною метою є створення платформи, яка забезпечить швидкий та зручний доступ до інформації про нерухомість, полегшить процес купівлі, продажу та оренди для користувачів, а також забезпечить безпеку та надійність обробки даних.

Завдання роботи. Для досягнення цієї мети були поставлені певні завдання, першим з яких є аналіз ринку. Необхідно провести аналіз існуючих рішень на ринку веб-сайтів для продажу та оренди нерухомості, визначити їхні сильні та слабкі сторони, а також виявити потреби потенційних користувачів. Наступним кроком – обрати оптимальні технології для розробки веб-сайту, зокрема NextJS для серверного рендерингу та React для створення інтерфейсу, TypeScript для типізації коду, TailwindCSS для стилізації, Shadcn UI та Radix UI для використання готових компонентів, Framer Motion для анімацій, Lucide-react для іконок, NextAuth для авторизації та безпеки, MongoDB для зберігання даних, Prisma ORM для роботи з базою даних. Ще одним важливим завданням є проектування архітектури веб-сайту, що включає клієнтську та серверну частини, роутинг та внутрішній API для взаємодії з базою даних. Важливим завданням є інтеграція системи авторизації користувачів з можливістю входу через Google або власні дані за допомогою NextAuth, а також забезпечити захист персональних даних користувачів. В подальшому необхідно реалізувати внутрішній API для обробки запитів з клієнтської частини та взаємодії з базою даних через Prisma ORM. Забезпечити можливість фільтрації та сортування оголошень у каталозі. Дизайн та користувацький досвід також є важливим завданням: необхідно створити привабливий і зручний дизайн інтерфейсу з використанням TailwindCSS та готових компонентів Shadcn UI та Radix UI. Забезпечити інтерактивність та анімацію за допомогою Framer Motion. Що стосується фінальних кроків, то треба провести тестування веб-сайту для виявлення та виправлення помилок а також оптимізувати продуктивність та забезпечити швидке завантаження сторінок. Ну і нарешті, розгорнути веб-сайт на сервері за допомогою Vercel, забезпечивши його доступність для

користувачів. Виконання цих завдань дозволить створити сучасний, функціональний та безпечний веб-сайт для продажу та оренди нерухомості, що відповідає потребам ринку та очікуванням користувачів.

Об'єкт дослідження. Процеси розробки веб-додатків для продажу та оренди нерухомості, що включають використання сучасних технологій для створення інтерактивних та зручних у використанні платформ.

Предмет дослідження. Предметом дослідження є методи та засоби розробки веб додатку про оренду та продаж нерухомості, включаючи аналіз сучасних технологій веб-розробки, інтеграцію баз даних для зберігання інформації та розробку користувацького інтерфейсу.

Методи дослідження. Для досягнення мети та виконання завдань кваліфікаційної роботи були використані різноманітні методи дослідження, такі як аналіз літератури та джерел, порівняльний аналіз, проектування системи, та прототипування. Застосування таких методів дозволило забезпечити комплексний підхід до розробки веб-додатку, врахувати всі важливі аспекти створення сучасної, функціональної та надійної платформи для продажу та оренди нерухомості.

Практичне значення одержаних результатів. Одержані результати мають значне практичне значення для різних категорій користувачів та фахівців у галузі розробки веб-додатків і ринку нерухомості по низці причин. Розроблений веб-сайт надає зручний і інтуїтивно зрозумілий інтерфейс, що полегшує процес пошуку, купівлі, продажу та оренди нерухомості. Користувачі можуть швидко знаходити необхідну інформацію, використовувати фільтри для пошуку відповідних об'єктів та здійснювати операції з мінімальними зусиллями. Платформа пропонує інструменти для ефективного управління оголошеннями про продаж та оренду нерухомості, що дозволяє ріелторам і агентствам зменшити витрати часу на адміністрування та збільшити свою продуктивність. Інтеграція з базами даних, системами управління контентом та іншими сервісами спрощує роботу з великими обсягами даних. Використання сучасних технологій, таких як React, NextJS, TypeScript, TailwindCSS, Shadcn

UI, Radix UI, Framer Motion, Lucide-react, NextAuth, MongoDB, Prisma ORM, react-hook-form та zod, забезпечує високу продуктивність та надійність додатку. Це дозволяє зменшити витрати на розробку та обслуговування, а також забезпечити масштабованість платформи для підтримки великої кількості користувачів і оголошень. Інтеграція з NextAuth для автентифікації та авторизації користувачів забезпечує високий рівень безпеки персональних даних. Це особливо важливо у контексті операцій з нерухомістю, де конфіденційність та безпека інформації є критичними. Також розроблена архітектура додатку дозволяє легко додавати нові функції та адаптувати платформу під змінні потреби ринку. Це забезпечує гнучкість у розвитку та вдосконаленні сервісу, відповідаючи на нові виклики та потреби користувачів.

Одержані результати можуть бути використані розробниками та ІТ-фахівцями як приклад застосування сучасних технологій і методологій у розробці веб-додатків. Вони також можуть бути корисними для студентів і дослідників у галузі ІІЗ як навчальний матеріал та базис для подальших досліджень.

Таким чином, результати даної роботи мають широкий спектр застосувань, сприяють розвитку ринку нерухомості, підвищують ефективність і зручність для користувачів та професіоналів, а також підтримують інновації та технологічний прогрес у галузі веб-розробки.

Апробація результатів дослідження. Матеріали кваліфікаційної роботи були представлені на XI Міжнародній науковій конференції «Студентські наукові дискусії поза форматом», у тезі на тему: «Технології розробки веб-сайту з продажу та оренди нерухомості», що проходила 11 квітня 2024 року в Університеті Короля Данила.

Структура. Розділи - 3. Обсяг основної частини - 59 сторінки. Список використаних джерел - 20

РОЗДІЛ 1. ВИЗНАЧЕННЯ ВИМОГ ДО ПРОЕКТУ ТА АНАЛІЗ НАЯВНИХ РІШЕНЬ

1.1 Вступ до технологій

У процесі розробки веб-додатку для продажу та оренди нерухомості використовувалися сучасні технології та інструменти, які забезпечують високу продуктивність, масштабованість і зручність у користуванні. Кожна з обраних технологій має свої особливості, що дозволяють ефективно вирішувати конкретні завдання. У цьому розділі ми розглянемо ключові технології, які стали основою для розробки додатку.

Основою технологій став React —бібліотека для розробки інтерфейсів користувача, створена компанією Facebook. Вона дозволяє будувати складні інтерфейси за допомогою компонентного підходу, що сприяє зручності розробки та підтримки коду. До основних переваг React-у можна віднести компонентний підхід, дозволяє розробникам створювати повторно використовувані компоненти, що спрощує розробку і тестування, віртуальний DOM, що підвищує продуктивність програми за рахунок ефективного оновлення інтерфейсу [6, 14].

Також перевагою є одностороння прив'язка даних. Вона забезпечує передбачувану поведінку додатку, що полегшує налагодження і тестування. Для серверної частини додатку використовувався фреймворк Next.js. Це — фреймворк для React, який забезпечує серверний рендеринг і генерацію статичних сайтів. Він надає низку переваг, які роблять його ідеальним вибором для розробки додатку [12].

Серверний рендеринг покращує продуктивність і SEO, забезпечуючи швидке завантаження сторінок. Система маршрутизації дозволяє створювати динамічні маршрути, що робить додаток більш гнучким і зручним у

використанні, а API-роути: дозволяють створювати бекенд-функціонал прямо в додатку Next.js, що спрощує архітектуру проекту [8].

Мовою програмування виступила TypeScript — надмножина JavaScript, яка додає типізацію до мови [1]. Вона допомагає виявляти помилки на етапі компіляції, що підвищує надійність коду. Типізація дозволяє уникнути багатьох помилок, пов'язаних з неправильним використанням змінних і функцій [4; 7].

Автокомпліт і інтелектуальні підказки полегшують процес розробки, надаючи розробникам підказки і автозавершення коду. TypeScript також можна поступово впроваджувати в існуючі проекти на JavaScript, що є його вагомою перевагою, хоч вона і не має значення в розроблюваному проекті [3].

Для стилізації використовувався Tailwind CSS — утилітарний CSS-фреймворк, який дозволяє швидко і ефективно створювати стильні інтерфейси. Він забезпечує утилітарний підхід, при якому кожен клас в Tailwind CSS відповідає за одне конкретне правило стилю, що робить стилізацію інтуїтивною і гнучкою. Висока кастомізація дозволяє легко налаштувати дизайн відповідно до потреб проекту. Як результат, бібліотека значно зменшує обсяг коду, оскільки не потрібно писати додаткові класи для елементів [9].

При розробці також використовувались Shadcn UI та Radix UI — це набори готових компонентів, які допомагають швидко створювати зручні та функціональні інтерфейси. Готові компоненти: прискорюють процес розробки, надаючи готові рішення для типових завдань. Вони забезпечують єдиний стиль компонентів, що підвищує естетичну привабливість додатку, а легка інтеграція компонентів, які легко інтегруються з іншими інструментами та фреймворками, значно підвищують швидкість розробки та зручність використання.

В додатку реалізовані анімації, у яких використовуються засоби технології Framer Motion — бібліотеки для створення анімацій у React-додатках. Вона надає простий і зручний API для додавання анімацій, що робить інтерфейс більш динамічним і привабливим. Простий синтаксис дозволяє легко створювати складні анімації, а гнучкість і налаштування забезпечує широкий спектр можливостей для налаштування анімацій. Підтримка ключових кадрів

дозволяє створювати анімації з використанням ключових кадрів, що дає більше контролю над рухом елементів.

Lucide-react — це бібліотека іконок для React, яка забезпечує широкий вибір іконок для використання в інтерфейсі. Широкий вибір іконок дозволяє швидко знайти необхідну іконку для будь-яких потреб, легкість у використанні та простий API роблять легкою інтеграцію іконок у компоненти. Також бібліотека надає можливість налаштовувати розмір, колір і інші властивості іконок, що значно збільшує кастомізаційний потенціал. Для аутентифікації використовується бібліотека NextAuth, яка підтримує різні способи авторизації, включаючи OAuth та JWT:

- підтримка різних провайдерів дозволяє легко інтегрувати авторизацію через Google, Facebook, GitHub тощо;
- безпека забезпечує безпечну аутентифікацію користувачів за допомогою сучасних методів шифрування і токенів;
- простота налаштування: легко інтегрується в Next.js-додаток з мінімальними налаштуваннями.

База даних реалізована засобами MongoDB — NoSQL бази даних, яка дозволяє зберігати великі обсяги даних у форматі документів. Для керування даними використовується Prisma ORM — інструмент для роботи з базами даних, який забезпечує зручний і безпечний доступ до даних. MongoDB дозволяє зберігати дані в динамічному форматі, що спрощує управління складними структурами даних. Висока продуктивність забезпечує швидкий доступ до даних і масштабування. Prisma ORM надає інтуїтивний API для роботи з даними, автоматично генерує SQL-запити і забезпечує типізацію на основі схеми бази даних [10].

Для обробки форм та надсилання даних використовуються інструменти React-hook-form та Zod. React-hook-form — це бібліотека для управління формами в React, а Zod — це бібліотека для валідації даних. Разом вони забезпечують ефективне управління і валідацію форм та забезпечує просте і інтуїтивне управління станом форм. Висока продуктивність мінімізує кількість

рендерів компонентів, що підвищує продуктивність додатку. Zod дозволяє легко валідувати дані форм, забезпечуючи високу якість і коректність введених даних.

В якості платформи для контролю версій використовується GitHub — платформа для контролю версій і співпраці над кодом, а також Vercel — платформа для розгортання веб-додатків. GitHub дозволяє зберігати історію змін коду, працювати в команді і вирішувати конфлікти. інтеграція з Vercel дозволяє автоматизувати процеси розгортання і тестування додатків. Vercel забезпечує швидке і просте розгортання додатків, інтеграцію з GitHub і автоматичні оновлення при кожному коміті [15].

Одна із технологій, які були використані - Uploadthing. Це сервіс для завантаження зображень, який дозволяє легко інтегрувати завантаження зображень в додаток, забезпечуючи високу продуктивність [19].

У якості IDE використовувався VS Code — редактор коду, який став стандартом де-факто серед розробників.

1.2 Використання TailwindCSS

TailwindCSS є CSS-фреймворком, засобами якого можна швидко і ефективно створювати сучасні та адаптивні інтерфейси користувача. Використання TailwindCSS у проекті забезпечило зручність стилізації та високу продуктивність розробки. У цьому розділі буде детально розглянуто основні особливості TailwindCSS, переваги його використання та приклади застосування в додатку [20].

TailwindCSS відрізняється від традиційних CSS-фреймворків своєю утилітарною природою. Замість попередньо створених компонентів і стилів, TailwindCSS пропонує набір класів, кожен з яких відповідає за окреме CSS-правило. Це дозволяє розробникам створювати власні компоненти швидко та ефективно, забезпечуючи високий рівень кастомізації.

Основні особливості TailwindCSS включають:

Утилітарні класи: кожен клас відповідає за одне правило стилю, наприклад, `text-center`, `mt-4`, `bg-blue-500`. Це дозволяє швидко застосовувати стилі без написання додаткового CSS.

Конфігурація: TailwindCSS має файл конфігурації `tailwind.config.js` (рис. 1.1), де можна налаштовувати кольори, відступи, шрифти та інші параметри відповідно до вимог проекту.



```
3
4 const config = withUt({
5   darkMode: ["class"],
6   content: [...],
7   prefix: "",
8   theme: {
9     container: {...},
10    extend: {
11      colors: {...},
12      borderRadius: {...},
13      keyframes: {...},
14      animation: {...},
15    },
16  },
17  plugins: [require("tailwindcss-animate")],
18 }) satisfies Config;
```

Рисунок 1.1 – Конфігурація TailwindCSS

Модульність: класи TailwindCSS легко комбінуються, що дозволяє створювати складні стилі без зайвого коду.

Адаптивність: вбудовані класи для адаптивного дизайну дозволяють створювати інтерфейси, які виглядають добре на будь-яких пристроях.

Переваги використання TailwindCSS. Використання TailwindCSS у проекті забезпечило низку переваг:

Швидка розробка: утилітарні класи дозволяють швидко створювати та змінювати дизайн інтерфейсу без необхідності писати додатковий CSS-код.

Легкість у підтримці: оскільки стилі застосовуються безпосередньо в класах HTML-елементів, зникає необхідність шукати відповідні стилі в зовнішніх файлах CSS.

Мінімізація конфліктів стилів: утилітарний підхід зменшує ймовірність конфліктів стилів, що часто виникають у великих проектах.

Адаптивний дизайн: вбудована підтримка медіа-запитів і класів для різних розмірів екрану дозволяє легко створювати адаптивні інтерфейси.

Висока продуктивність: автоматичне видалення невикористовуваних класів за допомогою інструменту PurgeCSS зменшує розмір кінцевого CSS-файлу, що покращує швидкість завантаження сторінок.

У додатку TailwindCSS використовується для стилізації всіх сторінок і компонентів, забезпечуючи єдність дизайну.

Для прикладу розглянемо розробку стилізації головної сторінки веб-застосунку (Додаток Б) – вона має кілька ключових секцій, таких як навігаційне меню, популярні питання, та блок для авторизації. Використання TailwindCSS дозволило легко створити сучасний і зручний інтерфейс. Сторінка каталогу відображає всі оголошення, дозволяючи користувачам фільтрувати та сортувати результати. TailwindCSS забезпечує зручну стилізацію форм і елементів інтерфейсу. На сторінці окремого оголошення у свою чергу TailwindCSS використовується для розміщення зображень, текстових блоків та контактної інформації.

1.3 Готові компоненти UI

Використання готових компонентів UI значно прискорило розробку додатку "react", забезпечивши високий рівень зручності та естетичної привабливості інтерфейсу. Для створення інтерфейсу користувача було застосовано бібліотеки Shadcn UI та Radix UI, які надали широкий спектр компонентів, що відповідають сучасним стандартам дизайну [16].

Shadcn UI надає комплект готових компонентів, що дозволяє швидко створювати стильні та функціональні інтерфейси. Основні переваги Shadcn UI включають високу якість компонентів, легкість налаштування та інтеграцію з

іншими інструментами, такими як TailwindCSS, оскільки готові компоненти повністю побудовані на його основі.

Основні компоненти Shadcn UI, використані в проєкті:

Кнопки: різноманітні стилі кнопок для різних дій, включаючи основні, вторинні та текстові кнопки.

Форми: компоненти для створення форм, такі як текстові поля, чекбокси, радіо-кнопки та селектори.

Модальні вікна: зручні модальні вікна для відображення додаткової інформації або підтвердження дій.

Карти: компоненти для відображення карток оголошень, включаючи зображення, заголовки та короткі описи.

На сторінках додатку використовуються кнопки та модальні вікна з Shadcn UI для створення інтуїтивно зрозумілого інтерфейсу (рис. 1.2).

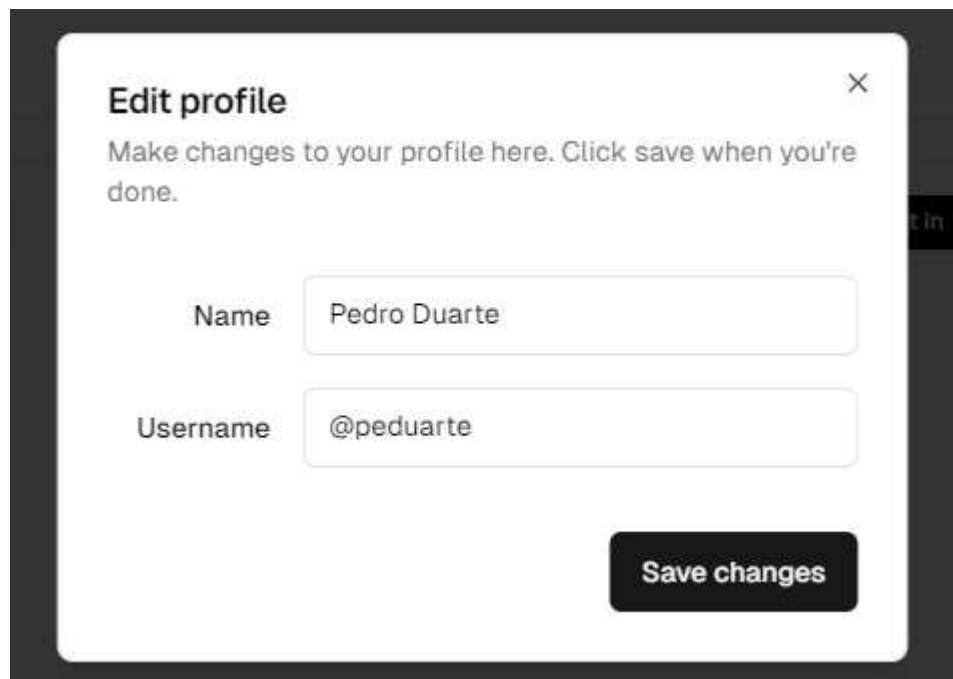


Рисунок 1.2 – Приклад готових діалогів та кнопок

Radix UI — це ще одна бібліотека компонентів, яка використовувалась для розробки додатку. Radix UI пропонує низку доступних і легко налаштовуваних компонентів, що допомагають створити сучасний і зручний інтерфейс.

Основні компоненти Radix UI, використані в проекті:

- акордеони для відображення часто задаваних питань (FAQ) на головній сторінці;
- випадаючі меню для навігації та управління аккаунтом користувача;
- діалогові вікна для створення нових оголошень.

Наприклад, на сторінці профілю користувача використовуються акордеони для відображення деталей щодо оголошення та частих запитань. Використання готових компонентів з Shadcn UI та Radix UI забезпечило ряд переваг для проекту:

- готові компоненти значно прискорили процес розробки, дозволяючи зосередитися на бізнес-логіці, а не на створенні базових елементів інтерфейсу;
- компоненти забезпечують єдиний стиль і користувацький досвід на всіх сторінках додатку, що покращує загальне сприйняття продукту;
- компоненти з цих бібліотек розроблені з урахуванням доступності, що гарантує зручність використання додатку для людей з різними потребами;
- компоненти легко налаштовуються та інтегруються з іншими інструментами і фреймворками, такими як TailwindCSS, що дозволяє створювати гнучкі та адаптивні інтерфейси.

Використання готових компонентів з Shadcn UI та Radix UI дозволило створити інтуїтивно зрозумілий, сучасний та зручний інтерфейс для додатку, що значно підвищило його якість і привабливість для користувачів.

1.4 Робота з MongoDB та безпека

Для зберігання даних у додатку використовується MongoDB, популярну NoSQL базу даних, яка забезпечує високу гнучкість і масштабованість. MongoDB ідеально підходить для роботи з великими обсягами даних і підтримує зберігання складних, багаторівневих документів у форматі BSON (Binary JSON). У цьому розділі розглянемо основні аспекти роботи з MongoDB у проекті та заходи для забезпечення безпеки даних.

MongoDB є потужним інструментом для роботи з даними, що має ряд переваг, таких як гнучкість моделі даних, масштабованість та висока продуктивність. У проєкті використовується MongoDB для зберігання інформації про оголошення, користувачів та інші важливі дані.

Для взаємодії з MongoDB використовується Prisma ORM, яка забезпечує простий і зручний API для роботи з базою даних. Prisma дозволяє визначати схеми даних і надає потужні інструменти для валідації та обробки даних [11].

Підключення до MongoDB здійснюється через Prisma ORM за допомогою коду, що на рисунку 1.3.

```
schema.prisma > Estate
1  datasource db {
2    provider = "mongodb"
3    url      = env("DATABASE_URL")
4  }
5
6  Generate
7  generator client {
8    provider = "prisma-client-js"
9  }
```

Рисунок 1.3 – Підключення до БД

Робота з MongoDB та Prisma ORM у проєкті забезпечує надійне та ефективне управління даними, забезпечуючи високу продуктивність і безпеку додатку. Для зберігання даних у MongoDB визначаються схеми за допомогою Prisma ORM. Використання сучасних підходів до зберігання та обробки даних дозволяє створювати масштабовані та безпечні веб-додатки.

1.5 Авторизація з NextAuth

Авторизація є важливим аспектом будь-якого веб-додатку, оскільки вона забезпечує контроль доступу до ресурсів і захищає персональні дані

користувачів. У проекті використовується NextAuth для реалізації системи авторизації та автентифікації. NextAuth — це гнучка і проста у використанні бібліотека для автентифікації в додатках, розроблених з використанням Next.js. Вона підтримує різноманітні методи автентифікації, такі як OAuth, одноразові паролі (OTP), електронну пошту і паролі, та інші.

Основні можливості NextAuth:

- підтримка багатьох провайдерів: Google, Facebook, GitHub та інші;
- легка інтеграція: простота інтеграції з Next.js, мінімальні налаштування, швидкий початок роботи;
- збереження сесій: підтримка збереження сесій на сервері або за допомогою токенів JWT;
- безпека: підтримка шифрування і захисту даних користувачів.

Для налаштування NextAuth у проекті було створено файл [...nextauth].js у директорії app/api/auth. Цей файл містить конфігурацію NextAuth (рис. 1.4).

```

6  const handler = NextAuth({
7    providers: [
8      GoogleProvider({
9        clientId: process.env.GOOGLE_CLIENT_ID!,
10       clientSecret: process.env.GOOGLE_CLIENT_SECRET!,
11     }),
12
13     Credentials({
14       credentials: {
15         email: { label: "email", type: "email", required: true },
16         password: { label: "password", type: "password", required: true },
17       },
18       async authorize(credentials) {

```

Рисунок 1.4 – Конфігурація NextAuth

1.6 Сервіси-аналоги

Аналіз веб-сайтів, що продають нерухомість, включає огляд різних аспектів, таких як дизайн сайту, його функціональність, зручність навігації, якість контенту та ефективність маркетингових стратегій.

Ось деякі ключові аспекти, на які варто звернути увагу при аналізі веб-сайтів, що продають нерухомість:

1. Чітка та проста навігація. Навігація відіграє важливу роль, особливо для сайтів, що продають нерухомість.
2. Продаючий контент. Контент має бути не лише продаючим, але й корисним для відвідувачів.
3. Зручний пошук. Ефективний і зручний пошук нерухомості - ключовий аспект для забезпечення задоволення користувачів.
4. Адаптивний дизайн. Зважаючи на те, що більшість користувачів відвідують сайти через мобільні пристрої, важливо мати адаптивний дизайн.

Крім того, важливо оцінити ефективність різних маркетингових каналів для продажу нерухомості, таких як "сарафанне радіо" або WOM-маркетинг, лідогенерація через Facebook / Instagram та інші.

Крім основних аспектів, важливо також звернути увагу на інші елементи, що можуть впливати на ефективність веб-сайту з продажу нерухомості. Наприклад, швидкість завантаження сторінок грає важливу роль у враженні від сайту. Повільне завантаження може знизити зацікавленість користувачів та збільшити ймовірність того, що вони покинуть сайт. Також важливо враховувати аспекти безпеки, забезпечуючи захист персональних даних користувачів та інформації про нерухомості.

Одним із ключових факторів успіху веб-сайту є його можливість привертати та утримувати трафік. Це може бути досягнуто за допомогою різних методів, таких як оптимізація контенту для пошукових систем (SEO), використання соціальних медіа для просування, а також платний рекламний трафік. Аналіз ефективності цих стратегій та їхнє поєднання може допомогти вдосконалити маркетинговий підхід та збільшити конверсію на сайті.

Необхідно також враховувати функціональні особливості, які можуть зробити веб-сайт більш привабливим для відвідувачів. Наприклад, впровадження інтерактивних інструментів, таких як віртуальні тури по об'єктах

нерухомості або можливість зв'язку з агентом нерухомості, може покращити користувацький досвід та збільшити ймовірність успішної угоди.

Сайт DIM RIA спеціалізується на продажі та оренді житла, головна сторінка якого представлена на рисунку 1.5, пропонуючи широкий вибір нерухомості для різних потреб

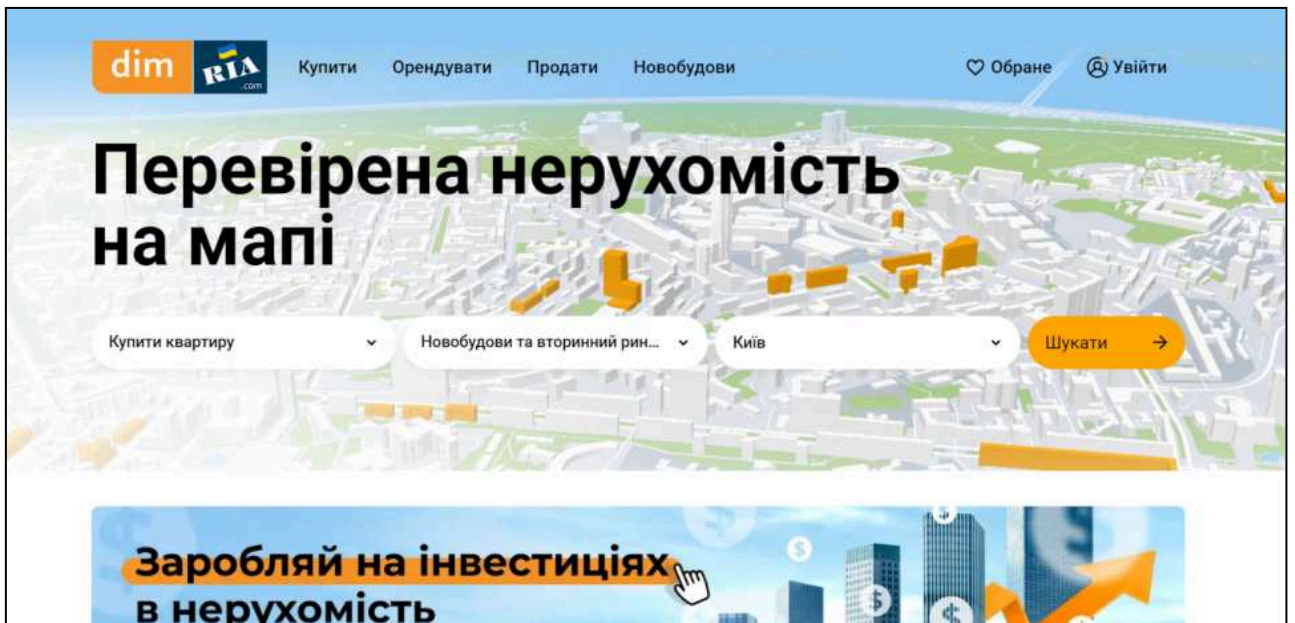


Рисунок 1.5 – Головна сторінка сервісу «DIM RIA»

У веб-сайті представлено кілька варіантів авторизації. Існує авторизація через google, телеграм чи дію. Також можна зареєструватись через електронну пошту та пароль. Порівняно з іншими сервісами «DIM RIA» має більш розширені можливості.

Основною функцією сайту є купівля та продаж нерухомості. Інформація про нерухомість містить дані про його ціну, площу, район, опис. Також є можливість додати нерухомість в обране для подальшої покупки.

На сайті існує великий вибір фільтрів для пошуку потрібної квартири, що включає фільтрування за ціною, площею, за районом.

Ще однією функцією є можливість переглянути доступну нерухомість та її ціну на інтерактивній карті України (рис.1.6).

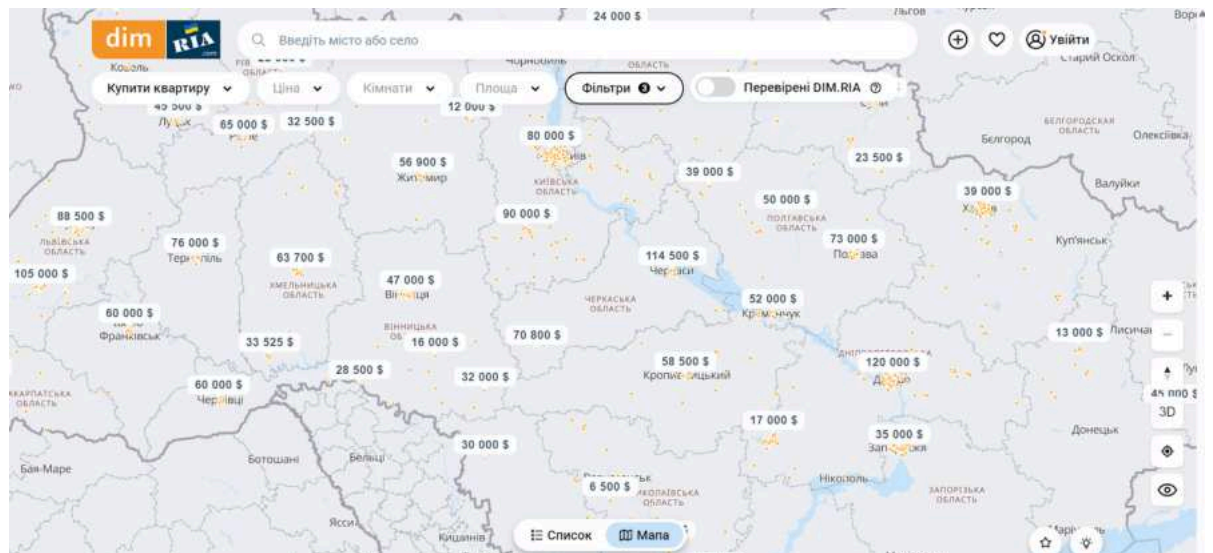


Рисунок 1.6 – Карта нерухомості України

Ця карта допоможе оцінити вартість нерухомості в окремих районах та зробити висновок про те, де вигідніше можна взяти нерухомість.

При натисканні на нерухомість користувачу відкривається її опис. Також користувач тут має можливість оглянути рейтинг району. Також є можливість подивитись відстань до лікарень, дитячих, садків та шкіл.

Крім цього, на сайті є форма для зв'язку з продавцем для покупки нерухомості. Також у сервісі є можливість додати оголошення для продажу нерухомості (рис.1.7), що полегшує взаємодію між користувачами.

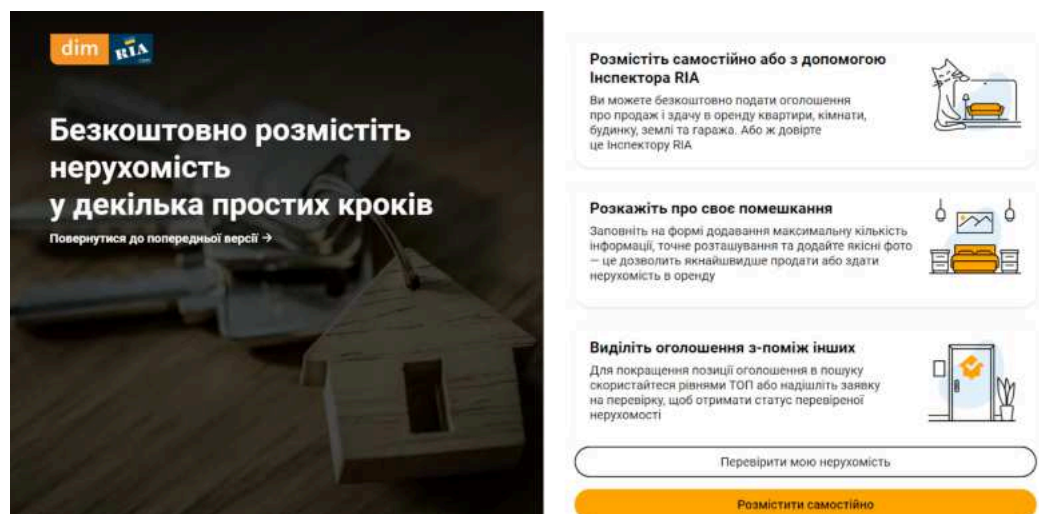


Рисунок 1.7 – Додавання оголошення про нерухомість

Додавання оголошення доступне тільки для авторизованих користувачів. Для його додавання потрібно придумати опис, розмістити фотографії та написати адресу нерухомості.

У підсумку, сервіс «DIM RIA» є потужним сервісом для продажу та оренди нерухомості. Тут є можливість детально розглянути пропозицію щодо продажу нерухомості: опис, фото, рейтинг району, карту, відстань від об'єктів критичної інфраструктури. Також на сайті є можливість зв'язатись з продавцем щодо купівлі та додавання нових оголошень. Сайт має велику кількість варіантів для авторизації, що є перевагою. Крім того веб-сервіс надає простий та інтуїтивно зрозумілий інтерфейс.

Головна сторінка веб-сайту Flatfy представлена на рисунку 1.8.

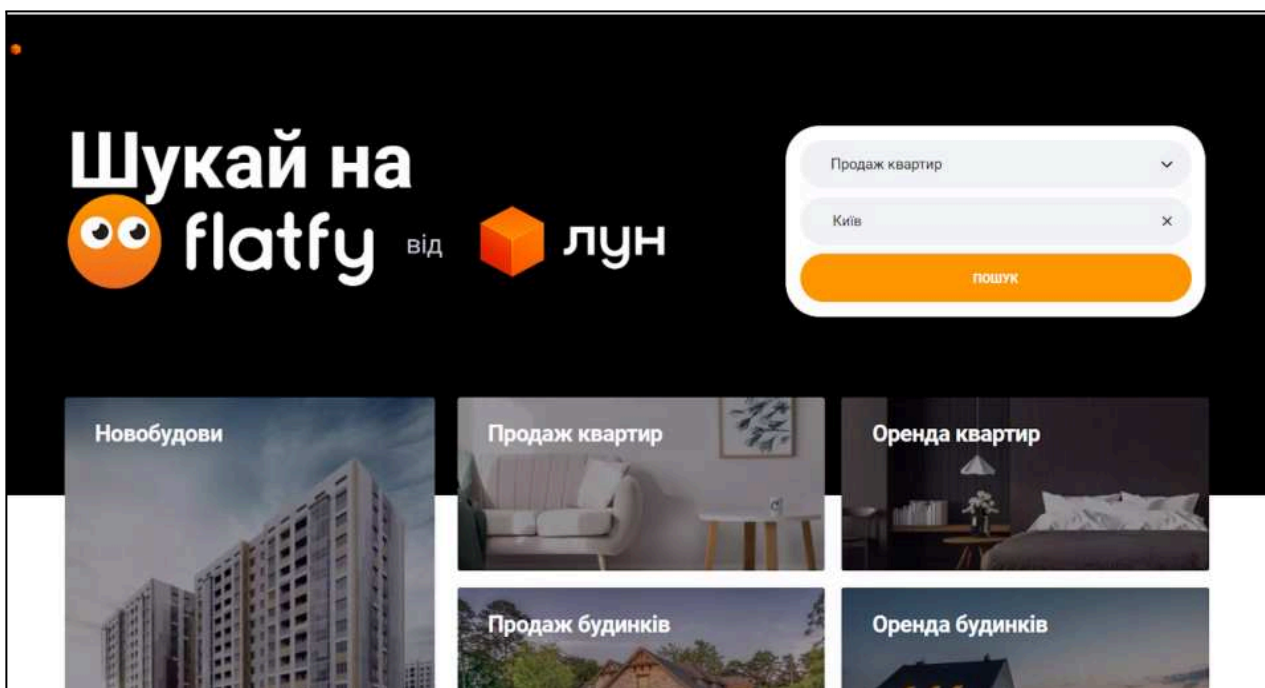


Рисунок 1.8 – Головна сторінка веб-сайту «Flatfy»

На головній сторінці присутня інформація про те чим займається даних сайт. Є можливість фільтрувати нерухомість по районах, після чого потрібно натиснути пошук. Після чого користувачу надається перелік доступних варіантів, які теж можна відфільтрувати по кількості кімнат, площі (рис.1.9).

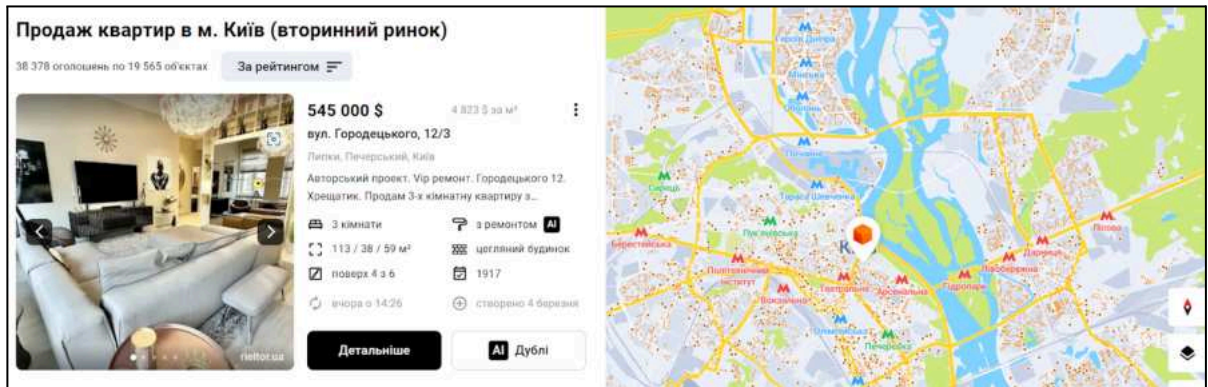


Рисунок 1.9 – Продаж квартири на веб-сайті «Flatfy»

В списку квартир є можливість переглянути кількість кімнат, площа, ремонт, поверх. Також при наведенні на пропозицію зліва одразу показується точка на карті де знаходиться нерухомість. Також існує велика кількість фото для ознайомлення з квартирою.

При натисканні на пропозицію користувач може переглянути її опис. Також тут є можливість переглянути інформацію про район та кількість ріелторів. Крім того на сайті є інформація для зв'язку з продавцем.

Також на сайті є викидне гамбургер-меню (рис.1.10).

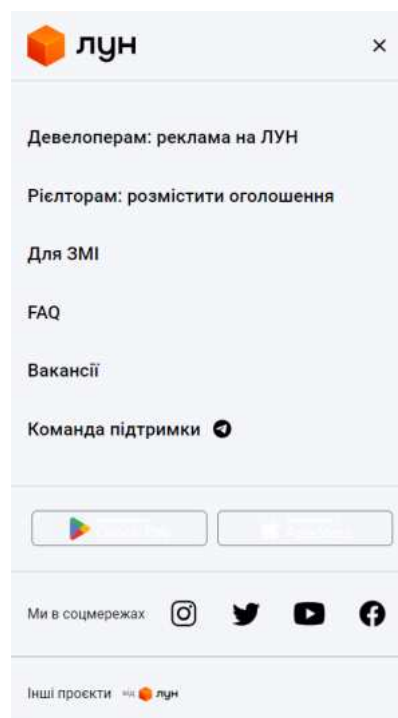


Рисунок 1.10 – Меню на сайті «Flatfy»

Тут є можливість зв'язатись з службою підтримки, скачати мобільний застосунок, подивитись рекламу. Крім того можна розмістити оголошення для ріелторів або перейти в часто задавані питання.

У підсумку веб-сервіс «Flatfy» надає велику кількість різноманітних квартир та будинків для оренди та продажі квартири. Також є можливість відфільтрувати дані, та подивитись інформацію про нерухомість на карті. Користувач має можливість переглянути фотогалерею нерухомості для ознайомлення з пропозицією. Недоліком веб-сайту є відсутність авторизації. Сайт має простий та зручний інтерфейс та просту навігацію між сторінками.

Головна сторінка веб-сайту Rent.ua представлена на рисунку 1.11.

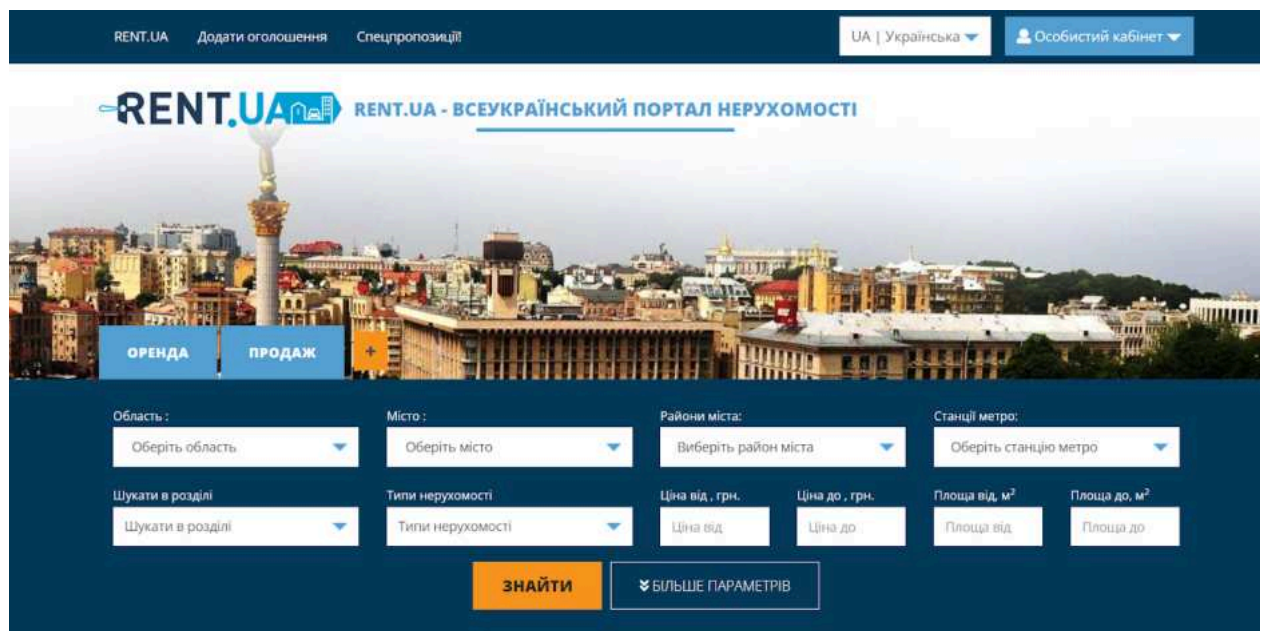


Рисунок 1.11 – Веб-сайт для продажу нерухомості «Rent.ua»

На сайті є можливість відфільтрувати пропозиції, обравши область, місто, ціну, площу. Після натискання на кнопку "знайти" користувачу нададуть релевантні результати пошуку. Після натискання на пропозицію користувачу відкриється наступне вікно з детальною інформацією про обраний об'єкт нерухомості, включаючи фотографії, опис, контактну інформацію продавця та інші важливі деталі (рис.1.12).



Рисунок 1.12 – Недухомість на сайті «Rent.ua»

На сторінці вибору пропозиції користувач має можливість переглянути фото нерухомості, її опис. Крім цього користувач має можливість зв'язатись з продавцем щодо покупки нерухомості.

На веб-сайті також є можливість зареєструватись за допомогою електронної пошти та пароллю. Користувач після авторизації має можливість потрапити в особистий кабінет.

У підсумку веб-сервіс має функціонал для продажу та оренди квартир. Основною перевагою є можливість легко та без проблем додати оголошення про нерухомість. Дизайн веб-сайту простий, навігація зрозуміла користувачу.

1.7 Постановка задачі

В результаті процесів планування та проектування були визначені основні завдання, які необхідно реалізувати в рамках розробки застосунку:

1. Розробити зручний та інтуїтивно зрозумілий користувацький інтерфейс, який має забезпечувати легкий доступ до всіх функціональних можливостей сайту, включаючи перегляд оголошень, фільтрацію за різними параметрами, додавання нових оголошень та управління профілем користувача.

2. Забезпечити ефективну та безпечну автентифікацію та авторизацію, включаючи можливість входу на сайт через Google або за допомогою власних облікових даних.

3. Реалізувати надійну систему управління базою даних з використанням MongoDB у якості бази даних та Prisma ORM для зручного доступу до неї. Система має забезпечувати швидке та надійне збереження, оновлення та видалення даних.

4. Забезпечити високий рівень продуктивності та масштабованості сайту. Це означає, що веб-сайт повинен швидко завантажуватися та реагувати на дії користувачів, навіть при великій кількості одночасних відвідувачів (що також позитивно вплине на SEO). Для цього слід використовувати серверний рендеринг для оптимізації продуктивності.

5. Забезпечити належний рівень безпеки веб-додатку. Потрібно приділити час для впровадження певних механізмів, які передбачатимуть потрапляння до бази даних загрозливих або неправильних даних. MongoDB досить добре захищає базу даних від поширених веб-загроз, таких як SQL-ін'єкції, проте бажано всеодно передбачати такі помилки завчасно.

6. Веб-сайт повинен бути розгорнутий на платформі Vercel для забезпечення його доступності та надійності.

Виконання цих задач є необхідним для створення ефективного, зручного та безпечного веб-сайту, що задовольнятиме потреби користувачів та відповідатиме сучасним стандартам веб-розробки.

Висновки до розділу 1

У розділі було розглянуто ключові аспекти та технології, що використовуються при розробці додатку. Початковий огляд включав вступ до технологій, що забезпечують сучасну, продуктивну та зручну розробку веб-додатків. Також було більш детально розглянуто використання TailwindCSS для стилізації додатку. Було показано, як утиліта дозволяє ефективно та швидко

створювати інтерфейси. Було зроблено акцент на використанні готових компонентів та продемонстровано, як бібліотеки допомагають швидко створювати інтерактивні елементи інтерфейсу користувача. Було присвячено час роботі з MongoDB та питанням безпеки, а також описано, як MongoDB використовується для зберігання даних, які підходи застосовуються для забезпечення надійності та швидкості доступу до даних.

У підрозділі 1.5 розглянуто механізми авторизації з використанням NextAuth та висвітлено, як NextAuth інтегрується в додаток.

Нарешті, у підрозділі 1.6 проведено аналіз сервісів-аналоги, що існують на ринку. Було проведено порівняння функціональності та технологічних рішень, що використовуються в аналогічних платформах, виявлено їх сильні та слабкі сторони, що дозволило визначити унікальні переваги, недоліки та задачі розроблюваного додатку.

РОЗДІЛ 2. ОБҐРУНТУВАННЯ ВИБОРУ ТЕХНОЛОГІЙ РОЗРОБКИ

2.1 Архітектурний підхід

Архітектура додатку "Hive" для продажу та оренди нерухомості спроектована з урахуванням вимог масштабованості, модульності, продуктивності та зручності підтримки. Додаток реалізований як SPA (Single Page Application) з використанням сучасних технологій, що дозволяють забезпечити високу швидкість роботи та зручність для користувачів. Основними технологіями є Next.js, React, TypeScript та MongoDB.

Архітектура додатку складається з декількох основних компонентів:

1. Клієнтська частина (Frontend).
2. Серверна частина (Backend).
3. База даних.
4. Інтеграція зі сторонніми сервісами.

Клієнтська частина додатку побудована з використанням React та Next.js, що дозволяє реалізувати SPA з підтримкою SSR (Server-Side Rendering) для покращення SEO та продуктивності [13].

- React: бібліотека для побудови інтерфейсів користувача, яка забезпечує високу продуктивність та гнучкість;
- Next.js: фреймворк для React, який додає можливості серверного рендерингу, статичного експорту та маршрутизації [12];
- TypeScript: мова програмування, що є надбудовою над JavaScript і забезпечує статичну типізацію, що покращує якість коду, зменшуючи кількість помилок під час розробки програмного забезпечення. [2];
- TailwindCSS: утилітарний CSS фреймворк, що дозволяє швидко та ефективно стилізувати компоненти інтерфейсу;
- Shadcn UI та Radix UI: набори готових компонентів, які забезпечують швидку розробку та уніфікований стиль інтерфейсу;

- `framer-motion`: бібліотека для створення анімацій, що додає інтерфейсу динамічність та покращує взаємодію з користувачем;
- `lucide-react`: іконки для інтерфейсу, що додають візуальної привабливості та покращують користувацький досвід;
- серверна частина додатку реалізована з використанням `Next.js`, який дозволяє створювати API маршрути та обробляти запити на сервері;
- `Next.js API Routes`: механізм створення API маршрутів, що дозволяє реалізувати серверну логіку без необхідності налаштування окремого сервера;
- `Prisma ORM`: інструмент для взаємодії з базою даних, що спрощує роботу з `MongoDB` та забезпечує високу продуктивність запитів;
- `NextAuth`: бібліотека для реалізації автентифікації та авторизації користувачів, яка підтримує різні методи автентифікації.

Для зберігання даних у додатку використовується `MongoDB`, яка є документно-орієнтованою базою даних, що забезпечує гнучкість, масштабованість та високу продуктивність.

- `MongoDB`: NoSQL база даних, що підтримує зберігання даних у форматі `BSON (Binary JSON)`, дозволяючи зберігати складні структури даних;
- `Mongoose`: бібліотека для роботи з `MongoDB` у `Node.js`, що забезпечує зручний API для створення схем та моделей даних;
- додаток інтегрується з декількома сторонніми сервісами для покращення функціональності та забезпечення високої якості роботи;
- `Uploadthing`: сервіс для завантаження та зберігання зображень, що дозволяє користувачам додавати фотографії до оголошень;
- `Vercel`: платформа для розгортання додатків, що забезпечує швидке і просте розгортання проекту;
- `Google Maps`: інтеграція з картою `Google` для відображення розташування офісу компанії на сторінці контактів, що дозволяє користувачам з легкістю знаходити місцезнаходження та маршрути до офісу (рис. 2.1).

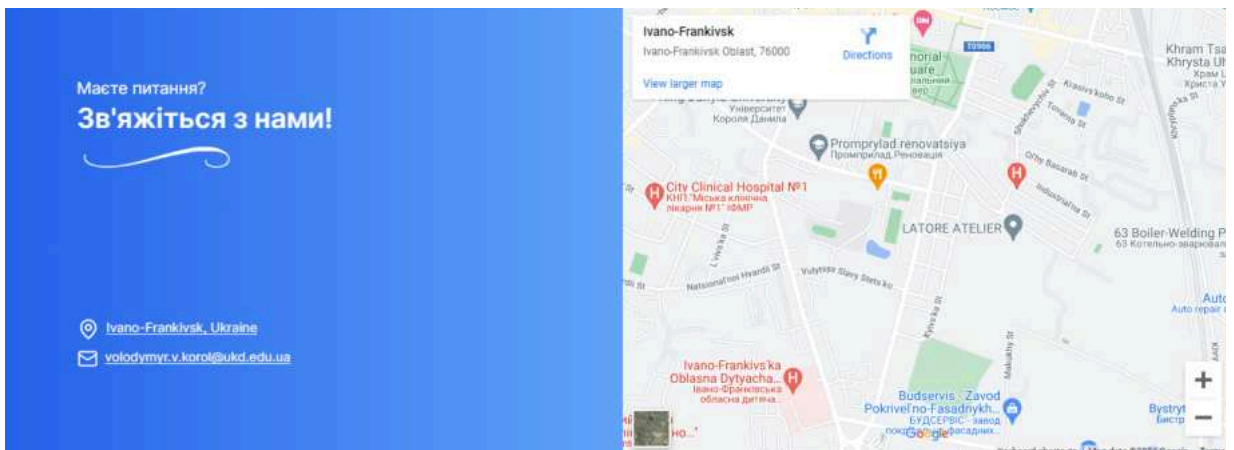


Рисунок 2.1 – Інтеграція з мапами

Основні принципи архітектури:

1. Модульність: кожен компонент додатку реалізується окремо, що дозволяє легко додавати нові функціональні можливості та підтримувати існуючі, спрощуючи розробку програмного забезпечення (рис. 2.2).

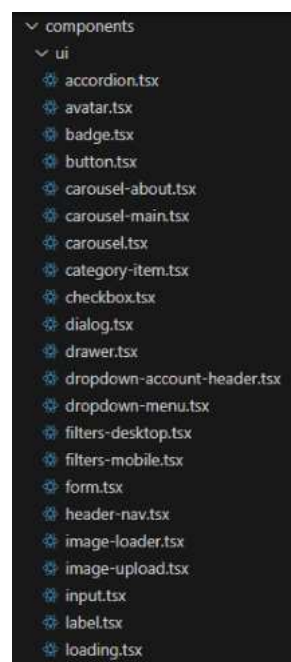


Рисунок 2.2 – Розділення фрагментів додатку на компоненти

2. Реактивність: використання React для створення інтерфейсів забезпечує високу продуктивність і швидкість реагування на дії користувачів.

3. Безпека: використання NextAuth для аутентифікації користувачів та Prisma ORM для роботи з базою даних забезпечує високий рівень безпеки.

4. Масштабованість: використання MongoDB та сучасних фреймворків надає можливість легко масштабувати додаток.

5. Зручність розробки та підтримки: використання TypeScript та сучасних інструментів розробки забезпечує високу якість коду [5].

Архітектура додатку дозволяє створити надійний, масштабований та зручний у використанні веб-додаток для продажу та оренди нерухомості, який відповідає сучасним вимогам до якості та безпеки.

2.2 Основні функціональні можливості

Додаток "Nive" призначений для забезпечення зручного і ефективного інтерфейсу для продажу та оренди нерухомості. Він надає користувачам різноманітні функціональні можливості, що охоплюють всі аспекти роботи з оголошеннями, від авторизації до управління оголошеннями. Основні функціональні можливості додатку можна розділити на кілька категорій:

1. Авторизація та управління обліковим записом.
2. Перегляд та фільтрація оголошень.
3. Додавання та редагування оголошень.
4. Взаємодія з оголошеннями.
5. Інформаційні сторінки.

Авторизація є першочерговою функцією, що забезпечує безпеку і персоналізацію досвіду користувача.

— реєстрація та вхід: користувачі можуть зареєструватися або увійти за допомогою електронної пошти та пароля або через обліковий запис Google;

— управління профілем: користувачі можуть переглядати інформацію свого профілю, включаючи електронну пошту, ім'я та аватар, а також додавати свої оголошення, забезпечуючи особисту керуваність та можливість розміщення власного контенту на платформі;

– перегляд особистих оголошень: у профілі користувачів є можливість переглядати всі оголошення, які вони розмістили.

Основна функціональність додатку полягає в наданні користувачам можливості переглядати оголошення про нерухомість і фільтрувати їх за різними параметрами, такими як область, місто, ціна та площа, що дозволяє швидко знаходити релевантні пропозиції відповідно до індивідуальних потреб користувачів..

– головна сторінка: на головній сторінці представлено корисні посилання на категорії оголошень, що дозволяє користувачам швидко орієнтуватися та переходити до потрібного розділу для перегляду відповідних оголошень (рис. 2.3).

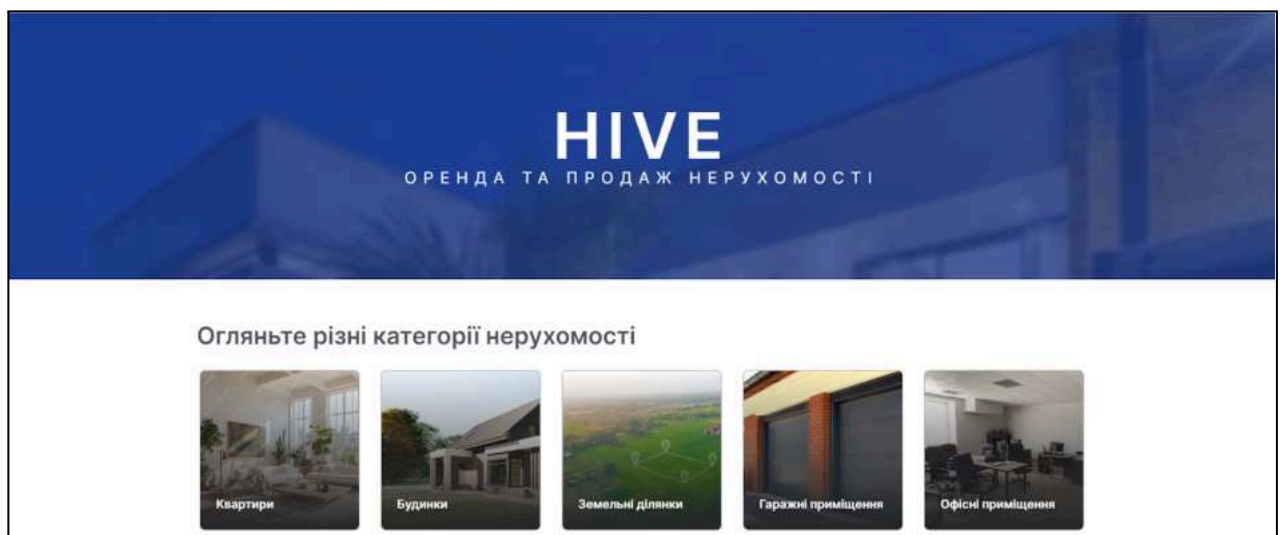


Рисунок 2.3 – Посилання на головній сторінці

– каталог оголошень: всі оголошення зібрані в каталозі, де вони представлені у вигляді списку з можливістю пагінації, що дозволяє користувачам легко переглядати великий обсяг пропозицій, зручно перемикаючись між сторінками.;

– фільтрація: користувачі можуть фільтрувати оголошення за різними параметрами (рис. 2.4). Фільтри представлені у вигляді чекбоксів і автоматично оновлюються при виборі.

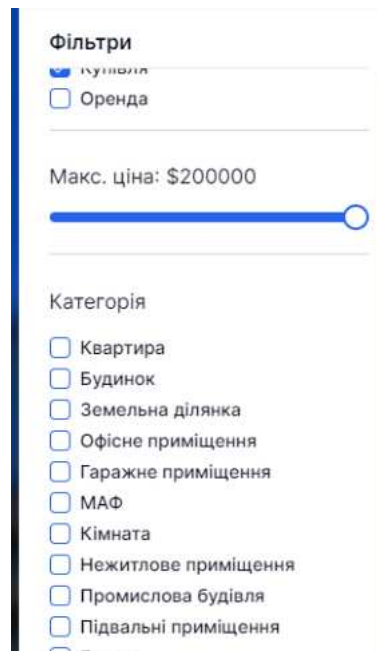


Рисунок 2.4 – Фільтри каталогу оголошень

Зареєстровані користувачі можуть створювати нові оголошення, а також редагувати або видаляти свої існуючі оголошення.

- створення оголошень: користувачі можуть заповнювати форму для створення нового оголошення, вказуючи необхідну інформацію, таку як назва, опис, ціна, розташування, контактні дані та додавати зображення;

- завантаження зображень: використовується сервіс Uploadthing для завантаження і зберігання зображень, що додаються до оголошень.

Користувачі можуть переглядати деталі оголошень, зв'язуватися з авторами і залишати відгуки.

- перегляд деталей оголошення: кожне оголошення має сторінку з детальною інформацією, де представлено зображення, опис, ціна, контакти автора, характеристики і т.д;

- зв'язок з автором: на сторінці оголошення є можливість зв'язатися з автором через контактні дані, зазначені в оголошенні.

Додаток також містить декілька інформаційних сторінок, що надають користувачам додаткову інформацію про компанію та способи зв'язку.

- про нас: сторінка з інформацією про компанію, її місію, досягнення та команду. Можна ознайомитись на рисунку 2.5;

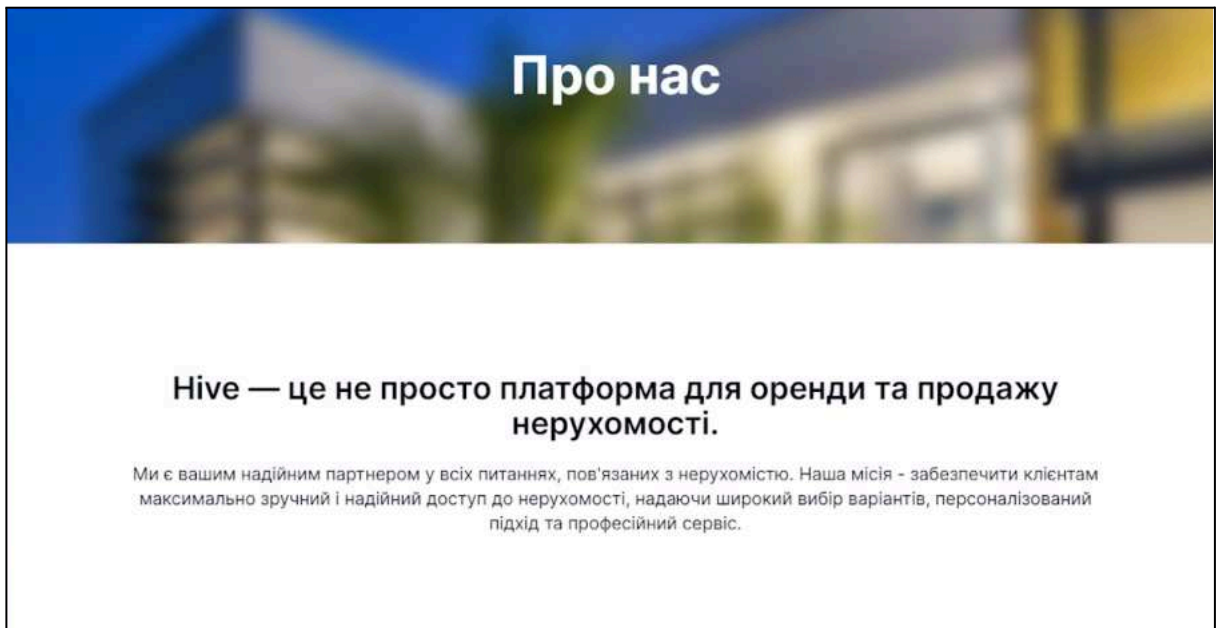


Рисунок 2.5 – Сторінка «Про нас»

- контакти: сторінка з контактною інформацією компанії, включаючи електронну пошту, фізичну адресу та інтеграцію з Google Maps для відображення місця розташування офісу.

Кожна з цих функціональних можливостей реалізована за допомогою різних технологій та інструментів, що забезпечують ефективність та надійність роботи додатку.

- Next.js API Routes: для обробки запитів на сервері та взаємодії з базою даних;
- Prisma ORM: для взаємодії з MongoDB і забезпечення зручного доступу до даних;
- React та Next.js: для побудови інтерфейсу користувача і забезпечення швидкої та зручної навігації;

- TailwindCSS: для швидкої і ефективною стилізації компонентів інтерфейсу, надаючи гнучкість та можливість швидкої кастомізації за допомогою набору утилітарних класів;

- NextAuth: для реалізації системи автентифікації та авторизації користувачів, забезпечуючи просту інтеграцію з різноманітними постачальниками ідентифікації і підтримуючи різні стратегії входу;

- Uploadthing: для завантаження і зберігання зображень оголошень.

Завдяки цим технологіям та функціональним можливостям додаток забезпечує зручний інтерфейс для користувачів, дозволяючи легко переглядати, фільтрувати, створювати і взаємодіяти з оголошеннями про нерухомість.

2.3 Фільтрація та пагінація

Фільтрація та пагінація є ключовими функціональними можливостями додатку, оскільки вони забезпечують користувачам можливість легко і швидко знаходити потрібні оголошення серед великої кількості доступних. Ці функції реалізовані з використанням сучасних технологій і підходів, що забезпечують високу продуктивність і зручність для користувачів.

Фільтрація оголошень дозволяє користувачам звужити пошук до конкретних критеріїв, що відповідають їхнім потребам і бажанням. У додатку реалізована багатокритеріальна фільтрація, яка включає такі параметри, як:

- тип нерухомості (квартира, будинок, офіс тощо);
- ціна (максимальна);
- розміри;
- велика кількість категорій;
- фільтрація реалізована за допомогою наступних технологій;
- React: використовується для створення інтерфейсу фільтрів, таких як чекбокси, селект-бокси та повзунки;

- Next.js: забезпечує серверну частину, де обробляються запити на фільтрацію, що дозволяє створювати швидкодіючі додатки, особливо у контексті динамічного рендерингу та побудови сторінок на стороні сервера.;
- MongoDB: база даних, де зберігаються оголошення та їх атрибути, що використовуються для фільтрації;
- Prisma ORM: інструмент для побудови та виконання запитів до бази даних, який надає зручний інтерфейс для взаємодії з базою даних за допомогою типобезпечних запитів і авто генерації коду..

Процес фільтрації:

1. Вибір параметрів фільтрації: користувач обирає параметри фільтрації у формі на сторінці каталогу.
2. Формування запиту: обрані параметри фільтрації формуються в запит, який відправляється на сервер.
3. Обробка запиту на сервері: на сервері Next.js приймає запит і використовує Prisma для виконання відповідного запиту до MongoDB (рис. 2.6).
4. Повернення результатів: відфільтровані оголошення повертаються на клієнтську частину і відображаються користувачеві.

```
const offset = (page - 1) * perPage;

const estates = await prisma.estate.findMany({
  where,
  take: perPage,
  skip: offset,
});
```

Рисунок 2.6 – Формування запиту до БД з урахуванням сторінок

Пагінація дозволяє розбити велику кількість оголошень на менші сторінки, що робить процес перегляду оголошень зручнішим і швидшим,

сприяючи кращій навігації та полегшуючи сприйняття контенту. Вона дозволяє користувачам швидко переміщатися між різними частинами списку без необхідності завантажувати всі дані одночасно.

Технологічна реалізація пагінації

- React: використовується для побудови інтерфейсу пагінації, включаючи кнопки "попередня", "наступна" і номери сторінок;
- Next.js: виконує серверну частину, обробляючи запити на отримання конкретних сторінок оголошень;
- MongoDB: зберігає оголошення, які повертаються по частинах відповідно до запитів пагінації;
- Prisma ORM: інструмент для побудови та виконання запитів до бази даних з урахуванням параметрів пагінації.

Процес пагінації відбувається наступним чином:

1. запит на певну сторінку: користувач натискає на кнопку пагінації або обирає номер сторінки, формуючи запит на конкретну сторінку оголошень.
2. формування запиту з параметрами пагінації: запит включає номер сторінки і кількість оголошень на сторінці.
3. обробка запиту на сервері: на сервері Next.js приймає запит і використовує Prisma для виконання запиту до MongoDB з урахуванням параметрів пагінації.
4. повернення результатів: відповідна сторінка оголошень повертається на клієнтську частину і відображається користувачеві.

Фільтрація та пагінація працюють разом для забезпечення максимальної зручності користувачів. Коли користувач застосовує фільтри, результати автоматично розбиваються на сторінки, і навпаки, при переході між сторінками фільтри залишаються застосованими, що дозволяє користувачам переглядати саме ті оголошення, які відповідають їхнім критеріям.

Технологічні аспекти такого комбінування заключаються насамперед у використанні URL параметрів для зберігання стану фільтрів і номера сторінки, що дозволяє легко передавати ці дані між клієнтською і серверною частинами.

Це реалізується шляхом маніпулювання параметрами запиту, що дозволяє відтворювати стан сторінки або фільтрувати дані, необхідні для пагінації, зберігаючи при цьому коректну адресу в браузері користувача (рис. 2.7).

```
const searchParams = useSearchParams();
const params = new URLSearchParams(searchParams.toString());

const getAllEstates = async () => {
  setLoading(true);
  const res = await fetch(
    "/api/estate?page=" + currentPage + "&" + params.toString()
  );
  const data = await res.json();
  setItems(data);
  setLoading(false);
};

useEffect(() => {
  getAllEstates();
}, [currentPage]);
```

Рисунок 2.7 – Передавання URL параметрів із клієнтської частини у серверну

Зберігання стану фільтрів і пагінації в React state забезпечує динамічне оновлення інтерфейсу без необхідності перезавантаження сторінки, а використання API маршрутів для обробки складних запитів на сервері, які включають як параметри фільтрації, так і пагінації, дозволяє зробити пагінацію ще більш зручною. Це поєднання сприяє плавному та інтерактивному досвіду користувача, дозволяючи швидко застосовувати фільтри та переміщатися між сторінками результатів без затримок..

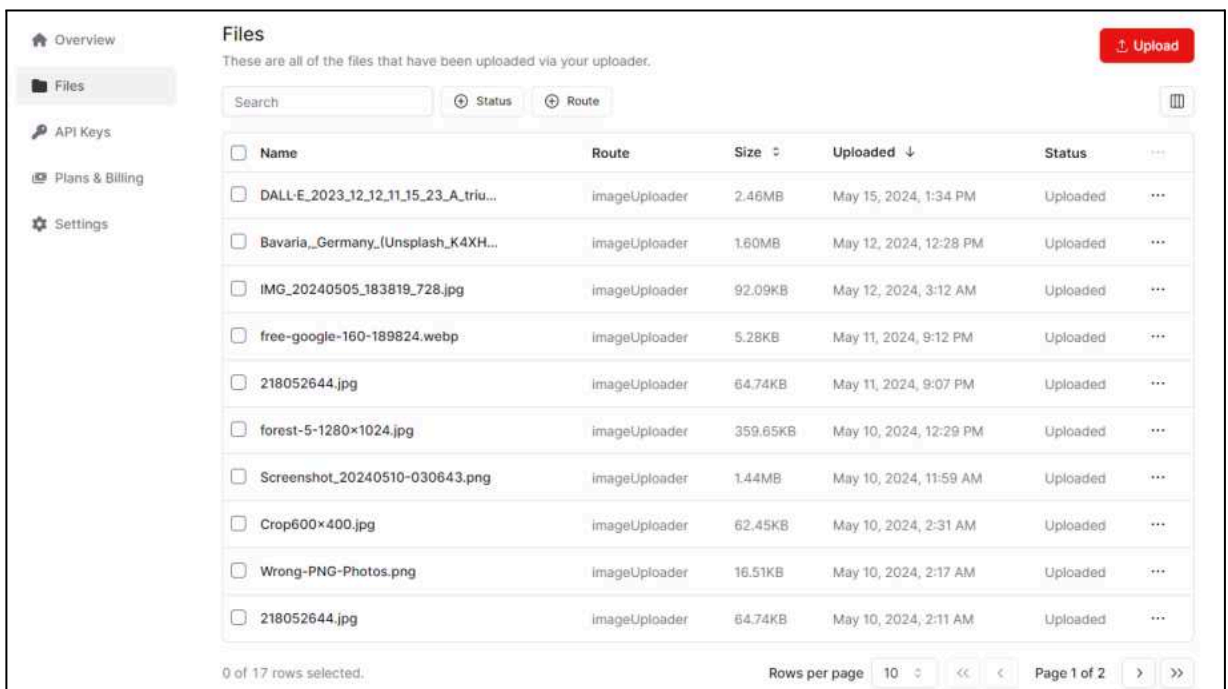
Завдяки ефективній реалізації фільтрації та пагінації додаток забезпечує користувачам можливість швидко знаходити релевантні оголошення про нерухомість, що значно підвищує зручність використання додатку, роблячи процес пошуку швидким і інтуїтивно зрозумілим.

2.4 Інтеграція зі сторонніми сервісами

Інтеграція зі сторонніми сервісами є важливою складовою функціональності додатку, оскільки вона дозволяє розширити можливості додатку і забезпечити високий рівень користувацького досвіду. У проекті використовуються декілька сторонніх сервісів для реалізації різних аспектів функціональності, включаючи завантаження зображень, автентифікацію користувачів, розгортання додатку та інтеграцію з картами Google.

Uploadthing використовується для завантаження і зберігання зображень, які додаються до оголошень користувачів. Цей сервіс забезпечує надійне і швидке зберігання файлів, що є критичним для додатків, які працюють з великою кількістю зображень.

— завантаження зображень: користувачі можуть завантажувати зображення своїх оголошень безпосередньо через інтерфейс додатку. Uploadthing забезпечує зберігання зображень у хмарі (рис. 2.8) і повертає URL для доступу до них;



The screenshot shows the 'Files' section of the Uploadthing dashboard. It features a sidebar with navigation options: Overview, Files, API Keys, Plans & Billing, and Settings. The main area displays a table of uploaded files with the following columns: Name, Route, Size, Uploaded, and Status. A search bar and buttons for 'Status' and 'Route' are located above the table. A red 'Upload' button is in the top right corner. The table lists 10 files, all with a status of 'Uploaded'.

Name	Route	Size	Uploaded	Status
DALL-E_2023_12_12_11_15_23_A_triu...	imageUploader	2.46MB	May 15, 2024, 1:34 PM	Uploaded
Bavaria,,Germany_(Unsplash_K4XH...	imageUploader	1.60MB	May 12, 2024, 12:28 PM	Uploaded
IMG_20240505_183819_728.jpg	imageUploader	92.09KB	May 12, 2024, 3:12 AM	Uploaded
free-google-160-189824.webp	imageUploader	5.28KB	May 11, 2024, 9:12 PM	Uploaded
218052644.jpg	imageUploader	64.74KB	May 11, 2024, 9:07 PM	Uploaded
forest-5-1280x1024.jpg	imageUploader	359.65KB	May 10, 2024, 12:29 PM	Uploaded
Screenshot_20240510-030643.png	imageUploader	1.44MB	May 10, 2024, 11:59 AM	Uploaded
Crop600x400.jpg	imageUploader	62.45KB	May 10, 2024, 2:31 AM	Uploaded
Wrong-PNG-Photos.png	imageUploader	16.51KB	May 10, 2024, 2:17 AM	Uploaded
218052644.jpg	imageUploader	64.74KB	May 10, 2024, 2:11 AM	Uploaded

0 of 17 rows selected. Rows per page: 10. Page 1 of 2.

Рисунок 2.8 – Зберігання зображень користувачів у БД Uploadthing

– інтеграція з формами: Uploadthing інтегрований з формами створення і редагування оголошень, що дозволяє користувачам легко додавати зображення при створенні оголошення. Це полегшує процес створення оголошень, забезпечуючи зручність та ефективність для користувачів.

Технологічна реалізація:

1. Форма завантаження: реалізована за допомогою React і компонентів для завантаження файлів.

2. API запити: використання Fetch API для відправки запитів на завантаження зображень до Uploadthing.

3. Зберігання URL: збереження отриманих URL зображень у базі даних MongoDB через Prisma ORM.

NextAuth використовується для автентифікації та авторизації користувачів. Він забезпечує підтримку різних методів автентифікації, включаючи автентифікацію через облікові записи Google.

– Google Auth: користувачі можуть увійти в додаток за допомогою своїх облікових записів Google, що забезпечує швидку і безпечну автентифікацію, додатково підвищуючи зручність і безпеку вхідного процесу;

– електронна пошта та пароль: також підтримується традиційний метод автентифікації через електронну пошту та пароль.

Технологічна реалізація

1. Конфігурація NextAuth: налаштування NextAuth у проекті Next.js для підтримки різних провайдерів автентифікації.

2. Захищені маршрути: реалізація захищених маршрутів і компонентів, які доступні лише для автентифікованих користувачів, що забезпечує безпеку додатку та обмежує доступ до конфіденційної інформації лише для уповноважених користувачів..

3. Сеанси користувачів: управління сеансами користувачів за допомогою NextAuth для забезпечення безперервної авторизації.

Vercel використовується для розгортання додатку на сервері. Це платформа, яка забезпечує швидке і просте розгортання додатків Next.js з високою продуктивністю і масштабованістю.

– автоматичне розгортання: Vercel інтегрується з GitHub, що дозволяє автоматично розгортати нові версії додатку при кожному пуші в основну гілку репозиторію, забезпечуючи швидкий і безпроблемний процес розгортання;

– хостинг: забезпечує надійний і масштабований хостинг для додатку.

Технологічна реалізація:

1. Інтеграція з GitHub: підключення репозиторію GitHub до Vercel для автоматичного розгортання (рис. 2.9).

2. Налаштування розгортання: конфігурація параметрів розгортання і середовищ виконання (наприклад, налаштування змінних середовища).

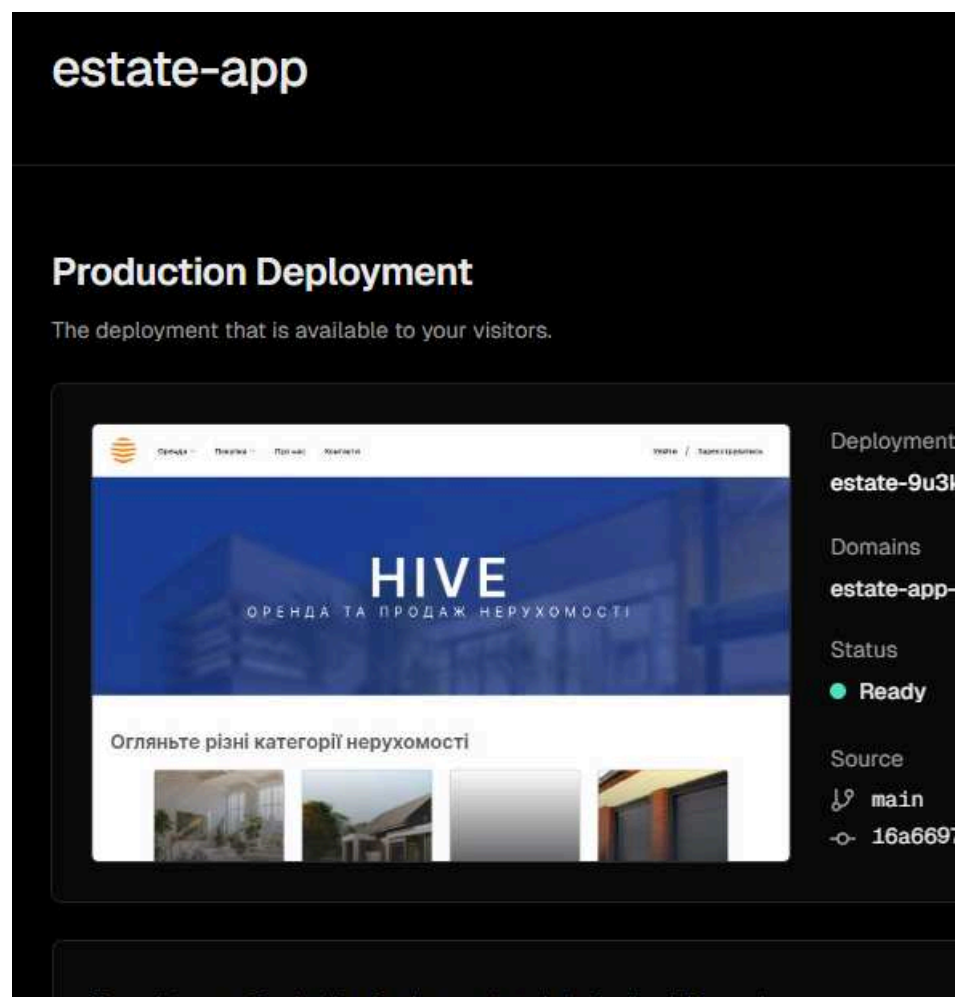


Рисунок 2.9 – Інтерфейс управління хостингом додатку

Google Maps інтегрується на сторінці контактів для відображення розташування офісу компанії, що допомагає користувачам легко знайти місцезнаходження компанії.

– карта з маркером: на сторінці контактів відображається інтерактивна карта з маркером, який вказує на розташування офісу;

– навігація: користувачі можуть використовувати карту для прокладання маршруту до офісу.

Технологічна реалізація:

1. Google Maps API: використання Google Maps JavaScript API для вбудовування карти на сторінку.

2. Компоненти React: створення компонентів для відображення карти і маркера на сторінці контактів.

Інтеграція зі сторонніми сервісами надає низку переваг для додатку:

1. Швидкий запуск і розширення функціональності: використання готових рішень дозволяє швидко додати нові можливості до додатку без необхідності розробляти їх з нуля.

2. Надійність і масштабованість: сторонні сервіси, такі як Vercel і Uploadthing, забезпечують високу надійність і масштабованість, що дозволяє обробляти великий обсяг даних і запитів.

3. Безпека: використання перевірених сервісів для автентифікації і зберігання даних забезпечує високий рівень безпеки.

4. Покращення користувацького досвіду: інтерактивні карти, швидка автентифікація і надійне завантаження зображень покращують загальний користувацький досвід і зручність використання додатку.

Завдяки інтеграції зі сторонніми сервісами, додаток надає користувачам потужні функціональні можливості і високу якість обслуговування, що робить процес пошуку і розміщення оголошень про нерухомість швидким, зручним і ефективним.

2.5 Механізми авторизації

Авторизація є важливою складовою безпеки та персоналізації в додатку "Hive". Використовуючи NextAuth, додаток забезпечує різноманітні методи автентифікації, включаючи соціальні логіни та автентифікацію за допомогою електронної пошти і пароля. Це дозволяє користувачам легко входити в систему і отримувати доступ до своїх профілів та персоналізованих функцій.

NextAuth є потужною бібліотекою для автентифікації в додатках Next.js. Вона підтримує різні методи автентифікації та забезпечує високий рівень безпеки і простоту інтеграції [18].

Бібліотека передбачає декілька методів аутентифікації, про які представлена інформація нижче.

Соціальні логіни: Google: користувачі можуть увійти в додаток, використовуючи свої облікові записи Google. Це забезпечує швидкий і безпечний вхід без необхідності запам'ятовувати ще один пароль (рис. 2.10).

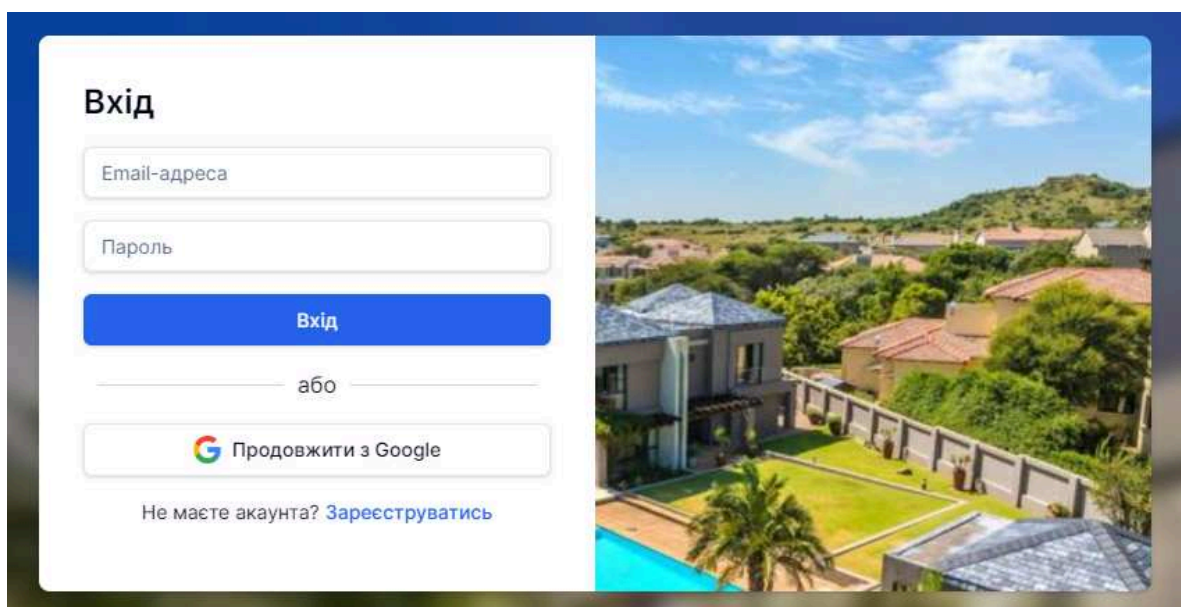


Рисунок 2.10 – Форма входу з можливістю використання облікового запису гугл

Інші провайдери: NextAuth підтримує багато інших соціальних логінів, таких як Facebook, GitHub, але в даному додатку інтегровано лише Google.

Електронна пошта і пароль: користувачі можуть зареєструватися і увійти в систему, використовуючи свою електронну пошту і пароль. Це стандартний метод автентифікації, який є зрозумілим для більшості користувачів.

Процес авторизації: користувачі можуть зареєструвати новий обліковий запис, заповнивши форму реєстрації, де вони вказують свою електронну пошту і пароль. При реєстрації з соціальним логіном, користувачі використовують свій існуючий обліковий запис Google.

Вхід в систему: для входу в систему користувачі можуть використовувати або свою електронну пошту і пароль, або обрати соціальний логін Google.

Верифікація: NextAuth забезпечує верифікацію користувачів, перевіряючи їхні облікові дані перед наданням доступу до захищених ресурсів.

Керування сеансами: NextAuth управляє сеансами користувачів, зберігаючи інформацію про вхід і забезпечуючи постійний доступ до захищених сторінок додатку.

2.6 Керування версіями та розгортання

Управління версіями та розгортання додатку є важливою складовою процесу розробки, оскільки воно забезпечує ефективну роботу команди, забезпечує стабільність і безпеку додатку, а також швидкість впровадження нових функцій і виправлень помилок. Для додатку використовуються сучасні інструменти управління версіями та автоматизованого розгортання, що дозволяє забезпечити швидкий і надійний процес розробки та впровадження змін.

Управління версіями коду дозволяє зберігати історію змін, відстежувати розвиток проекту та ефективно співпрацювати з іншими членами команди. Для цього використовується система контролю версій Git, яка дозволяє створювати різні гілки розробки, об'єднувати їх та вирішувати конфлікти. Код проекту "Hive" зберігається на платформі GitHub, що забезпечує зручний і безпечний спосіб спільної роботи над проектом (рис. 2.11) [17].

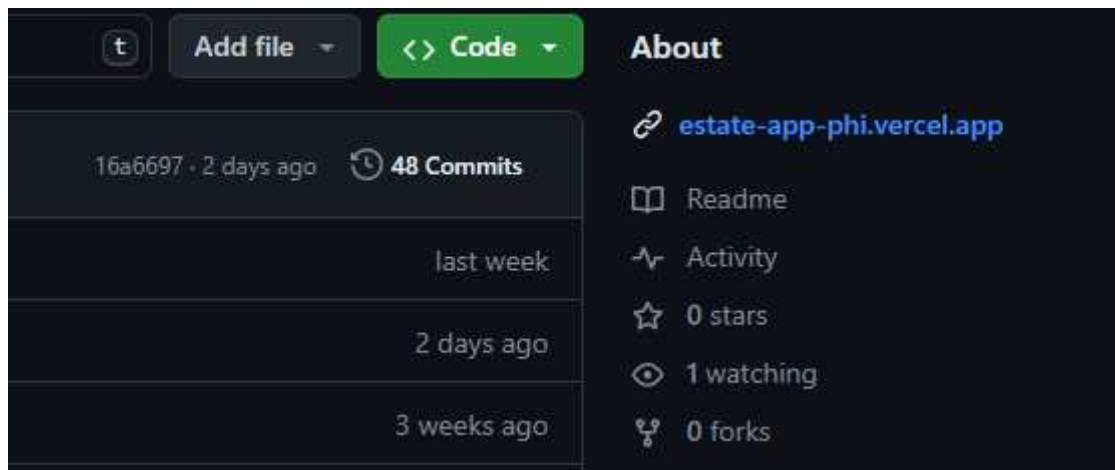


Рисунок 2.11 – Коміти та опис проекту на GitHub

Основні концепції Git: гілки (Branches): використовуються для розділення різних розвитку функцій або виправлень помилок, що дозволяє розробникам працювати паралельно без впливу на основну гілку (зазвичай main або master).

Коміти (Commits): представляють собою зміни, які здійснені в коді. Кожен коміт має повідомлення, що пояснює зроблені зміни.

Об'єднання (Merging): поєднання різних гілок, яке дозволяє внести зміни з однієї гілки в іншу.

Конфлікти (Conflicts): ситуації, коли Git не може автоматично об'єднати зміни з різних гілок, що потребує ручного втручання розробника для вирішення.

Розгортання додатку здійснюється за допомогою платформи Vercel, яка забезпечує простий і швидкий процес розгортання. Vercel інтегрується з GitHub, що дозволяє автоматично розгортати нові версії додатку при кожному коміті в основну гілку репозиторію. Це забезпечує швидке впровадження нових функцій та виправлень помилок без необхідності ручного розгортання.

Основні переваги розгортання з Vercel: автоматичне розгортання: нові версії додатку автоматично будуть застосовуватись при завантаженні даних на систему контролю версій.

Масштабування: платформа Vercel автоматично масштабує додаток відповідно до потреб та навантаження, що дозволяє забезпечити стабільну роботу додатку навіть під час періодів великого навантаження.

Швидкість розгортання: Vercel надає швидкий час розгортання нових версій додатку, що дозволяє швидко впроваджувати зміни та виправлення помилок, сприяючи ефективній розробці програмного забезпечення.

Підтримка прев'ю (Preview): Vercel дозволяє створювати прев'ю відгалужень (branches) коду, що дає змогу перевіряти нові функції або виправлення помилок перед об'єднанням коду з основною гілкою.

Інтеграція зі сторонніми сервісами: платформа Vercel підтримує інтеграцію з різними сервісами, такими як GitHub, GitLab, Bitbucket, що спрощує процес розгортання та спільної роботи з кодом.

Аналітика і моніторинг: Vercel надає інструменти для моніторингу (рис. 2.12) та аналізу продуктивності додатку, що дозволяє виявляти проблеми та оптимізувати роботу додатку.

Time	Status	Host	Request	Message
MAY 15 14:03:51.42	GET 200	estate-app-pnl.vercel...	/api/auth/session	
MAY 15 14:03:34.22	GET 200	estate-app-pnl.vercel...	/api/auth/session	
MAY 15 14:03:29.17	GET 200	estate-app-pnl.vercel...	/api/auth/session	
MAY 15 14:02:32.87	GET 200	estate-app-pnl.vercel...	/api/auth/session	
MAY 15 13:57:58.13	GET 200	estate-9u3k0d2iy-calm...	/api/auth/session	
MAY 15 13:57:58.06	GET 200	estate-9u3k0d2iy-calm...	/api/auth/session	
MAY 15 13:57:47.78	GET 200	estate-9u3k0d2iy-calm...	/api/auth/session	
MAY 15 13:57:36.75	GET 200	estate-app-pnl.vercel...	/api/auth/session	
MAY 15 13:57:36.36	GET 200	estate-app-pnl.vercel...	/api/auth/session	
MAY 15 13:57:34.38	GET 200	estate-9u3k0d2iy-calm...	/api/auth/session	
MAY 15 13:54:51.83	GET 200	estate-app-pnl.vercel...	/api/auth/session	
MAY 15 13:54:04.55	GET 200	estate-app-pnl.vercel...	/api/auth/session	
MAY 15 13:53:37.29	GET 200	estate-app-pnl.vercel...	/api/auth/session	
MAY 15 13:52:58.28	GET 200	estate-app-pnl.vercel...	/api/auth/session	
MAY 15 13:52:57.72	GET 200	estate-app-pnl.vercel...	/api/auth/session	
MAY 15 13:52:49.54	GET 200	estate-app-pnl.vercel...	/api/auth/session	
MAY 15 13:52:16.16	GET 200	estate-app-pnl.vercel...	/api/auth/session	
MAY 15 13:51:19.22	GET 200	estate-app-pnl.vercel...	/api/auth/session	
MAY 15 13:49:28.46	GET 200	estate-app-pnl.vercel...	/api/uploadthing	
MAY 15 13:49:18.38	GET 200	estate-app-pnl.vercel...	/api/auth/session	
MAY 15 13:49:18.92	GET 200	estate-app-pnl.vercel...	/api/auth/session	
MAY 15 13:48:56.06	GET 404	estate-app-pnl.vercel...	/_next/image	
MAY 15 13:48:05.06	GET 200	estate-app-pnl.vercel...	/estate/6830f519e6ff2e61353e684	
MAY 15 13:48:05.03	GET 200	estate-app-pnl.vercel...	/estate/6830f519e6ff2e61353e684	

Рисунок 2.12 – Інструменти моніторингу

Безкоштовний план для невеликих проектів: Vercel пропонує безкоштовний план для невеликих проектів, що дозволяє розгортати та використовувати додаток без додаткових витрат. Це дозволяє почати розробку та тестування проекту без необхідності великих витрат на інфраструктуру або хостинг, забезпечуючи швидке та доступне розгортання для розпочинання роботи над проектом.

Загалом, використання платформи Vercel для розгортання додатку "Hive" забезпечує швидкий, зручний та надійний процес впровадження змін та підтримки додатку.

Висновки до розділу 2

У другому розділі було розглянуто ключові аспекти архітектури та функціональності проекту. Розділ розпочинається з опису архітектурного підходу, що використовується для побудови додатку. Було показано, як застосування сучасних архітектурних шаблонів дозволяє забезпечити модульність, масштабованість та зручність у підтримці коду. Були описані основні функціональні можливості додатку, такі як створення та управління оголошеннями, перегляд каталогу нерухомості, взаємодія користувачів через профільні сторінки та інші важливі функції, що роблять платформу корисною та зручною для користувачів. Підрозділ 2.3 був присвячений механізмам фільтрації та пагінації, що забезпечують користувачам можливість легко знаходити необхідну інформацію. Було розглянуто, як реалізовані функції фільтрації за різними параметрами та як забезпечується ефективна робота пагінації для обробки великих обсягів даних. Також було розглянуто інтеграцію зі сторонніми сервісами і було показано, як ці інтеграції покращують функціональність додатку та надають користувачам додаткові можливості. Була продемонстрована реалізація різних методів аутентифікації, (Google та власні облікові дані). Була висвітлена робота з керуванням версіями та розгортанням проекту. (використання GitHub для контролю версій коду та платформи Vercel для розгортання додатку)

Таким чином, у другому розділі було детально розглянуто всі ключові аспекти архітектури, функціональності та технологій, використаних у розробці.

РОЗДІЛ 3. РЕАЛІЗАЦІЯ ВЕБ-ДОДАТКА

3.1 Розробка головної сторінки

Головна сторінка є ключовою точкою входу для користувачів у додаток. Вона повинна надати користувачам зручний доступ до основного функціоналу, відображати актуальну інформацію та стимулювати їх до подальшої взаємодії з додатком. Розробка головної сторінки включає в себе наступні аспекти:

Дизайн і компоненти:

1. Вигляд і структура: головна сторінка має привабливий та інтуїтивно зрозумілий дизайн (рис. 3.1). Це включає в себе зручне розташування основних елементів, таких як меню навігації, блоки з категоріями оголошень, секції з популярними питаннями тощо.

2. Компоненти і анімації: використання готових компонентів UI і анімацій покращує користувацький досвід та робить сторінку більш привабливою. Наприклад, використання акордеонів для показу популярних питань чи анімаційних ефектів при прокручуванні сторінки.



Рисунок 3.1 – Головна сторінка додатку

Функціональність:

1. Навігація: меню навігації містить посилання на основні розділи додатку, такі як каталог оголошень, сторінка про компанію, контактні дані тощо (рис. 3.2). Це дозволяє користувачам швидко знаходити необхідну інформацію, полегшуючи навігацію та забезпечуючи доступ до ключових функцій та розділів додатку.

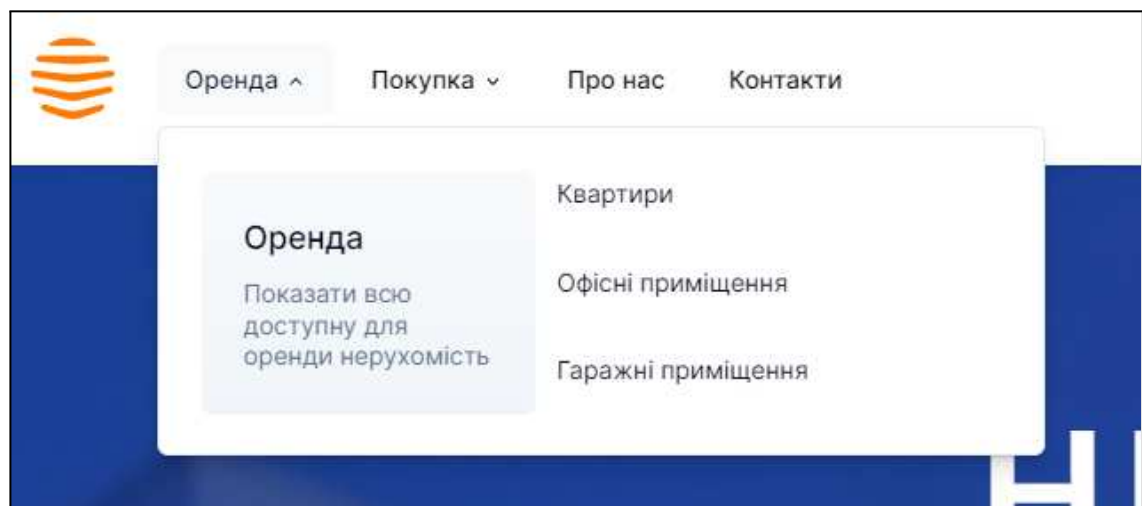


Рисунок 3.2 – Навігаційне меню

2. Відображення категорій оголошень: блок з категоріями оголошень дозволяє користувачам швидко знайти потрібні оголошення за їхнім типом або категорією, полегшуючи процес пошуку необхідної інформації. Це забезпечує зручність і ефективність користування додатком, дозволяючи швидко орієнтуватися в доступних категоріях та знаходити потрібні оголошення.

3. Популярні питання: секція з популярними питаннями може допомогти новим користувачам зорієнтуватися в додатку та знайти відповіді на найбільш поширені запитання.

4. Форма для авторизації: блок "Увійдіть, щоб розмістити оголошення" (рис. 3.3) надає можливість входу користувачам, які вже мають обліковий запис, а також посилання на реєстрацію для нових користувачів.

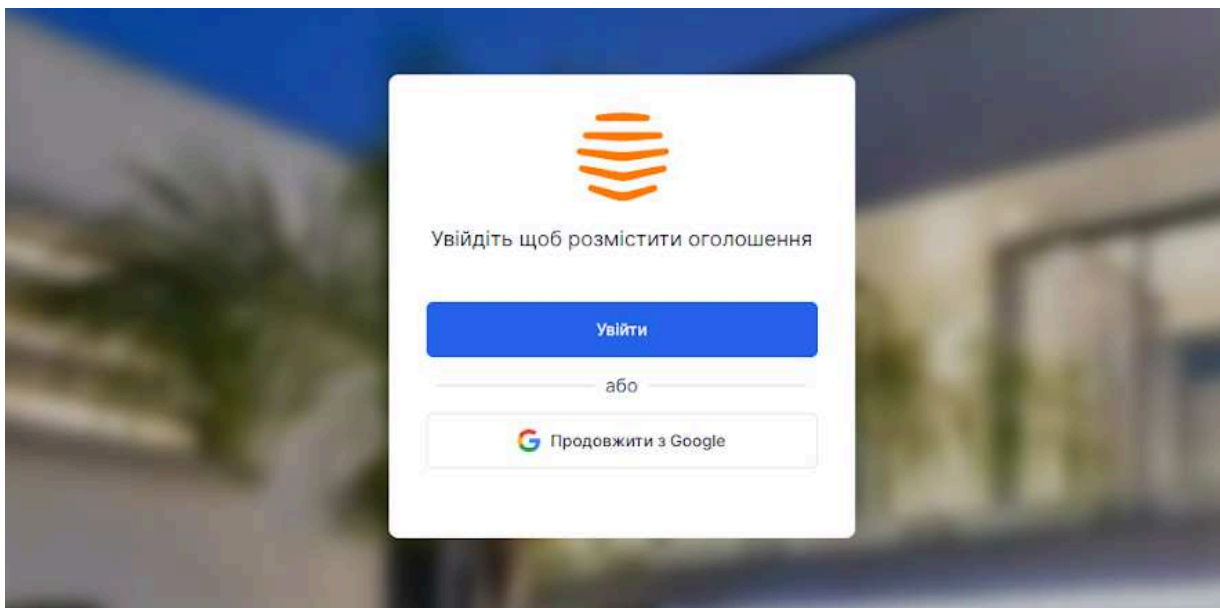


Рисунок 3.3 – Блок авторизації

Оптимізація та доступність:

1. Швидкість завантаження: забезпечено швидке завантаження головної сторінки, оскільки це перший контакт користувача з додатком. Мінімізація ресурсів та оптимізація коду покращують час завантаження, забезпечуючи приємний і швидкий досвід взаємодії з додатком, що важливо для залучення та утримання користувачів.

2. Доступність: враховуючи різні потреби користувачів, була забезпечена доступність головної сторінки для людей з обмеженими можливостями, включаючи користувачів з вадами зору або моторикою за допомогою tabindex. Це дозволяє забезпечити рівний доступ до інформації та функціоналу додатку для всіх користувачів, незалежно від їхніх фізичних можливостей чи обмежень..

3. Респонсивний дизайн: додаток адаптивний до різних пристроїв та розмірів екранів, щоб забезпечити зручний перегляд на будь-яких пристроях, від комп'ютерів до смартфонів. Це дозволяє користувачам отримувати консистентний та оптимізований досвід взаємодії з додатком незалежно від пристрою, який вони використовують (рис 3.4).

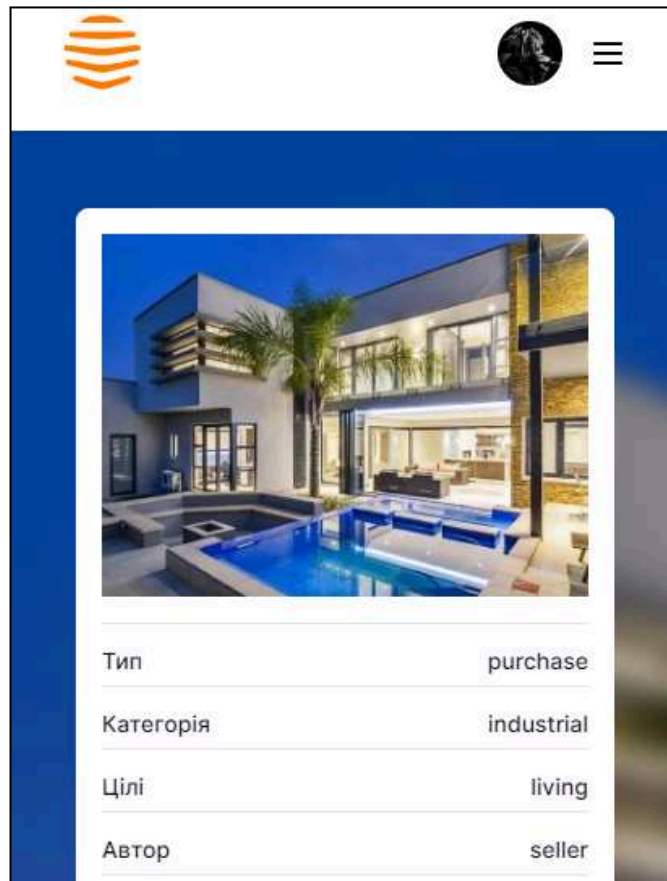


Рисунок 3.4 – Сторінка оголошення для мобільних пристроїв

При розробці головної сторінки додатку враховувались усі вимоги до дизайну, функціональності та оптимізації, щоб забезпечити зручний та привабливий користувацький досвід.

3.2 Реалізація каталогу оголошень

Каталог оголошень є ключовим функціональним елементом додатку, який дозволяє користувачам здійснювати пошук та перегляд доступних оголошень за різними критеріями. Реалізація каталогу оголошень включає в себе:

Дизайн і компоненти:

1. Відображення оголошень: кожне оголошення відображено у вигляді картки з необхідною інформацією, такою як зображення, назва та ціна. Це може зроблено за допомогою сітки карточок з можливістю фільтрації та сортування.

2. Фільтрація і сортування: користувачам надана можливість фільтрувати оголошення за різними параметрами, такими як категорія, ціновий діапазон, тощо. Користувачам дозволено також сортувати результати за різними критеріями, та великою кількістю категорій, що стосуються не лише нерухомості, а й автора, який створив це оголошення.

3. Пагінація: було створено пагінацію для розділення результатів на окремі сторінки, що полегшує навігацію користувача та зменшує навантаження на сервер (рис. 3.5).

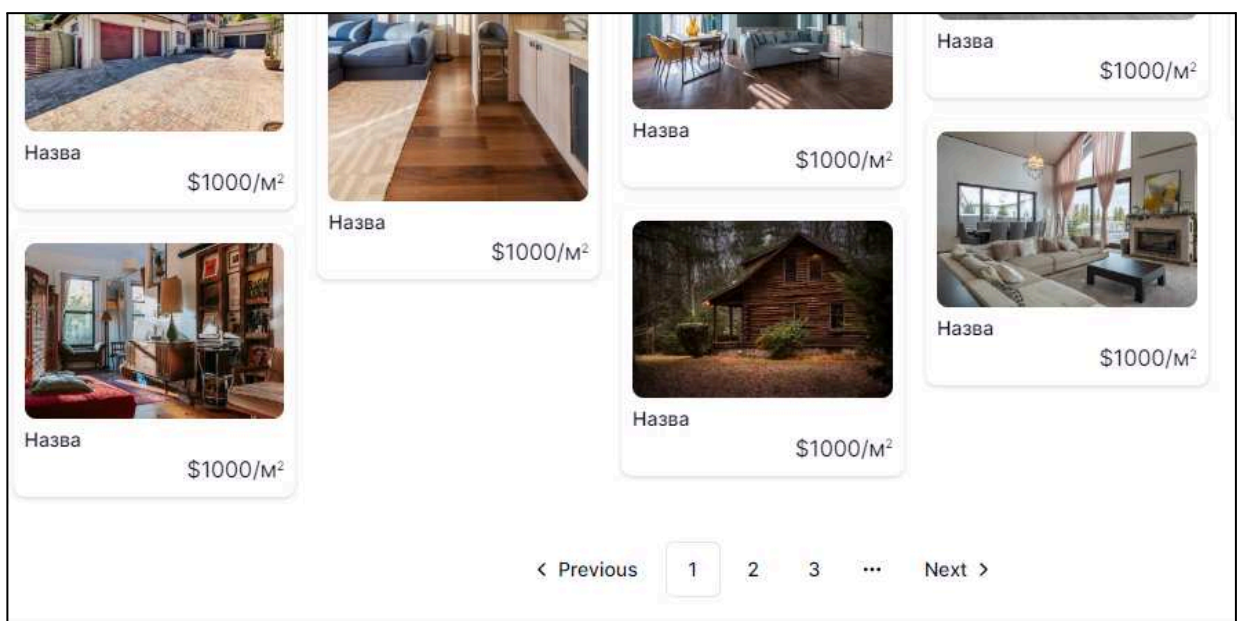


Рисунок 3.5 – Пагінація оголошень

Функціональність:

1. Завантаження оголошень з сервера: каталог оголошень отримує дані про оголошення з сервера, використовуючи апі-запити. Це відбувається асинхронно з використанням fetch-запитів JS та внутрішнього API.

2. Фільтрація та сортування на клієнті: для полегшення роботи користувача фільтрація та сортування реалізовані на клієнтській стороні за допомогою JavaScript, що дозволяє швидше оновлювати результати без перезавантаження сторінки при виборі користувачем певної категорії, за допомогою хука useRouter NextJS/navigation у адресну строку користувача

пушиться нове посилання, але сторінка не оновлюється, при цьому оновлюючи необхідні дані на сторінці. З функціоналом можна ознайомитись на рисунку 3.6.

```
30   const addSearchParameter = (parameter: string, value: string) => {
31     if (!params.has(parameter) || parameter === "maxPrice") {
32       params.set(parameter, value);
33     } else {
34       params.set(parameter, params.get(parameter) + " " + value);
35     }
36     router.push(currentLink + params.toString());
37     refresh();
38   };
39
40   const removeSearchParameter = (parameter: string, value: string) => {
41     if (!params.has(parameter)) {
42       return;
43     } else {
44       const existingValue = params.get(parameter);
45       const valuesArray = existingValue!.split(" ");
46       const newValueArray = valuesArray.filter((val) => val !== value);
47
48       if (newValueArray.length > 0) {
49         params.set(parameter, newValueArray.join(" "));
50       } else {
51         params.delete(parameter);
52       }
53     }
54     router.push(currentLink + params.toString());
55     refresh();
56   };
```

Рисунок 3.6 – Функції додавання та видалення пошукових параметрів

3. Детальний перегляд оголошень: користувачам надана можливість перегляду детальної інформації про кожне оголошення, включаючи зображення, характеристики, опис та всі інші дані про оголошення із бази даних. Це реалізовано за допомогою окремих сторінок для кожного оголошення. При переході на сторінку з оголошенням в параметри посилання користувача вкладається ідентифікатор конкретного оголошення, яке уже на сторінці оголошення звідти «витягується» та передається у запиті на внутрішній API (рис. 3.7), який в свою чергу приймає його та передає уже до бази даних за допомогою Prisma ORM з ціллю отримати інформацію про конкретне оголошення. Потім передача даних відбувається в зворотньому напрямку, тим самим відображаючи на сторінці користувача необхідні дані.

```

16  const EstatePage = ({ params }: { params: { id: string } }) => {
17    const [estate, setEstate] = useState<any>({});
18
19    const getEstate = async () => {
20      const res = await fetch("/api/singleEstate?id=" + params.id);
21      const data = await res.json();
22      setEstate(data);
23    };
24
25    useEffect(() => {
26      getEstate();
27    }, []);

```

Рисунок 3.7 – Отримання ідентифікатора та запит до API

Оптимізація та доступність:

1. Кешування запитів: для покращення швидкодії каталогу оголошень застосовується кешування результатів запитів на клієнті, що дозволяє зменшити кількість запитів до сервера, полегшуючи навантаження на сервер та підвищуючи продуктивність додатку для кінцевого користувача. Це сприяє швидшому завантаженню сторінок і покращує загальний досвід користувача, забезпечуючи швидкий доступ до інформації...

2. SEO оптимізація: важливо оптимізувати каталог оголошень для пошукових систем, щоб забезпечити його високий рейтинг у пошукових результатах. Це включає в себе використання правильних метатегів (рис. 3.8), оптимізованих URL-адрес та текстового контенту.

```

8  const inter = Inter({ subsets: ["latin"] });
9
10 export const metadata: Metadata = {
11   title: "Hive",
12   description:
13     "Hive - Веб-сайт для придбання та оренди нерухомості по всій Україні",
14 };
15

```

Рисунок 3.8 – Експорт метаданих, які дозволять покращити SEO

3. Доступність: каталог оголошень доступний для всіх користувачів, включаючи тих, що використовують програми для читання екрану або мають обмежені можливості. Для цього дотримуються стандартів доступності, такі як використання правильних HTML-тегів, атрибутів та розмітки.

3.3 Реалізація авторизації

Авторизація є важливою функціональністю для будь-якого веб-додатка. Реалізація авторизації включає в себе низку етапів та можливостей для користувачів, таких як створення облікового запису, вхід в систему, відновлення пароля, керування правами доступу та інше. Це дозволяє забезпечити безпеку додатка та конфіденційність даних користувачів, а також надає можливість персоналізованого досвіду використання для кожного користувача:

Механізм авторизації: використання NextAuth: для реалізації авторизації в використовується бібліотека NextAuth, яка забезпечує швидку та зручну інтеграцію з різними методами авторизації, включаючи авторизацію з гуглом та власними даними.

Синхронізація даних: при використанні як авторизації з гуглом, так і авторизації з власними даними, користувач отримує однаковий доступ до особистого профілю та функціональності додатку. Це означає, що користувач може здійснювати ті ж самі дії, незалежно від того, яким методом він увійшов у систему.

Функціональність авторизованих користувачів:


Перегляд власних оголошень: авторизований користувач може переглядати список своїх оголошень на сторінці профілю. Це дозволяє користувачеві швидко переглядати, редагувати або видаляти власні оголошення.

Додавання нових оголошень: користувач має можливість додавати нові оголошення через відповідну форму, доступну на сторінці профілю (рис. 3.9). Це дозволяє користувачам швидко розміщувати нові оголошення про нерухомість, яку вони пропонують.

Нове оголошення ⓘ

Назва

Зображення



Ціна

Створити Відміна

Рисунок 3.9 – Додавання нового оголошення

3.4 Тестування додатку

Тестування є невід'ємною частиною розробки будь-якого програмного забезпечення, включаючи веб-додатки. Для забезпечення якості та надійності додатку "Hive", було проведено комплексне тестування на різних етапах розробки. Нижче наведено розділений підхід до тестування додатку:

Інтеграційне тестування:

1. Тестування інтеграції компонентів: були проведені тести для перевірки коректної взаємодії між різними компонентами та модулями додатку.
2. Тестування API взаємодії: була здійснена перевірка роботи API та його взаємодії з клієнтською частиною додатку шляхом відправлення запитів та перевірки відповідей.

Енд-тестування:

1. Тестування відповідності вимогам функціональності: були проведені тести, які перевіряють, чи відповідає функціональність додатку вимогам та очікуванням користувачів.

2. Тестування сумісності та відповідності стандартам: була проведена перевірка сумісності додатку з різними веб-браузерами, пристроями та відповідність стандартам, таким як HTML, CSS та іншим.

Тестування безпеки:

1. Тестування на вразливості: було здійснене проведення аудиту та тестування на вразливості для забезпечення безпеки додатку.

2. Перевірка обробки та збереження конфіденційної інформації: були виконані тести для перевірки правильної обробки конфіденційних даних користувачів. Також інструментами GitHub та Vercel була перевірена правильність зберігання Environment Variables, доступ до якої не повинен отримувати користувач та будь-яка інша стороння особа (рис. 3.10).



```
.env
1 GOOGLE_CLIENT_ID= [REDACTED]
2 GOOGLE_CLIENT_SECRET= [REDACTED]
3 NEXTAUTH_SECRET= [REDACTED]
4 NEXTAUTH_URL= [REDACTED]
5 DATABASE_URL= [REDACTED]
6 APP_NAME= Hive
7 API_BASE= [REDACTED]
8 UPLOADTHING_SECRET= [REDACTED]
9 UPLOADTHING_APP_ID= [REDACTED]
```

Рисунок 3.10 – Зберігання Environment Variables всередині .env

Ручне тестування:

1. Тестування вручну: були виконані ручні тести для перевірки зручності використання додатку користувачами на різних етапах взаємодії.

2. Тестування у реальних умовах: було проведено тестування додатку у реальних умовах, зокрема на різних пристроях, з різними швидкостями Інтернет-з'єднання та на різних мережах.

Висновки до розділу 3

У третьому розділі було розглянуто практичні аспекти розробки та впровадження ключових компонентів веб-сайту. Був описаний процес розробки головної сторінки та показано створення інтерфейсу, який надає користувачам доступ до основних розділів сайту та містить інформацію про ресурс. Також було описано реалізацію каталогу оголошень, і розглянуто, як здійснено структурування та відображення великої кількості оголошень, включаючи механізми фільтрації та пагінації, що дозволяють користувачам знаходити пропозиції з нерухомості. Було з практичної точки зору описано, як інтегровано систему NextAuth для входу користувачів. Також було розглянуто процес тестування додатку - було проведено різні види тестування, включаючи Unit-тестування, функціональне тестування та тестування безпеки. Це дозволило виявити та виправити помилки, тим самим забезпечивши якість та стабільність додатку.

Тож, у третьому розділі було детально розглянуто практичні аспекти створення ключових компонентів проекту. Це забезпечило реалізацію функціональних можливостей проекту та його готовність до використання кінцевими користувачами.

ВИСНОВКИ

У рамках кваліфікаційної роботи було розроблено проект "Hive" - веб-додаток для продажу та оренди нерухомості, розроблений з використанням технологій React, Next.js, та ряду інших інструментів. Були використані сучасні технології та інструменти розробки, такі як React, Next.js, MongoDB, Tailwind CSS, для створення функціонального та зручного у використанні веб-додатку. Додаток надає ряд корисних функціональностей, включаючи реєстрацію та авторизацію користувачів, перегляд оголошень про нерухомість, додавання нових оголошень, редагування профілю користувача та інші. Була розроблена чітка архітектура додатку, розділення на клієнтську та серверну частини, використання компонентного підходу та зручної навігації. Після всіх процесів розробки було проведено комплексне тестування додатку на різних етапах розробки, включаючи unit-тестування, інтеграційне тестування, тестування безпеки та ручне тестування для забезпечення якості та надійності продукту.

Загальною метою проекту було створення функціональності, проектування архітектури використання сучасних та інструментів, які забезпечили створення якісного продукту, що в свою чергу дозволяє користувачам легко і зручно шукати нерухомість для покупки чи оренди. Пояснювальна записка до проекту слугує як важливий ресурс для розробників, тестувальників та користувачів, щоб краще розуміти та використовувати додаток.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Крокфорд, Дуглас. "JavaScript: The Good Parts." O'Reilly Media, 2008. 176 с.
2. Дакетт, Джон. "JavaScript і JQuery: Interactive Front-End Web Development." Wiley, 2014. 640 с.
3. Ресіг, Джон, і Беар Бібелт. "Secrets of the JavaScript Ninja." Manning Publications, 2016.
4. Хавербеке, Марейн. "Eloquent JavaScript: A Modern Introduction to Programming." 3-е видання, No Starch Press, 2018.
5. Зак, Ніколас С. "Understanding ECMAScript 6: The Definitive Guide for JavaScript Developers." No Starch Press, 2016.
6. Бенкс, В. Джейсон, і Ітан Браун. "Learning React: Modern Patterns for Developing React Apps." O'Reilly Media, 2020. 484 с.
7. Хейлсберг, Андерс, і Даніель Розенвассер. "Programming TypeScript: Making Your JavaScript Applications Scale." O'Reilly Media, 2019. 324 с.
8. Тафт, Метт. "Mastering Next.js: Building Scalable Server-Side Web Applications." Leanpub, 2020.
9. Сонг, Адам. "Tailwind CSS: From Zero to Production." Leanpub, 2021.
10. Реймерс, Свен. "The Definitive Guide to MongoDB: A Complete Guide to Dealing with Big Data using MongoDB." Apress, 2019. 572 с.
11. Копеланд, Райан, і Лука Маттеїс. "Prisma: The Complete Guide to Build a Scalable Server-Side Application." Leanpub, 2020.
12. Папазян, Володимир. "Building Modern Web Applications with Next.js: Your Comprehensive Guide." Leanpub, 2021.
13. Server-side Rendering (SSR). Веб-сайт.URL: <https://nextjs.org/docs/pages/building-your-application/rendering/server-side-rendering> (Дата звернення: 25.03.2024)

14. React – JavaScript-бібліотека для створення користувацьких інтерфейсів. (n.d.). React. Веб-сайт. URL: <https://uk.legacy.reactjs.org/> (Дата звернення: 23.03.2024)
15. How to Build a Multi-Tenant App with Custom Domains Using Next.js Веб-сайт. URL: <https://nextjs.org/docs/pages/building-your-application/rendering/server-side-rendering> (Дата звернення: 20.04.2024)
16. Introduction. Веб-сайт. URL: <https://ui.shadcn.com/docs> (Дата звернення: 23.03.2024)
17. Git Guide. Веб-сайт. URL: <https://github.com/git-guides> (Дата звернення: 29.03.2024)
18. Introduction About NextAuth.js. Веб-сайт. URL: <https://next-auth.js.org/getting-started/introduction> (Дата звернення: 28.03.2024)
19. What Is UploadThing? Веб-сайт. URL: <https://docs.uploadthing.com/> (Дата звернення: 05.04.2024)
20. Get started with Tailwind CSS? Веб-сайт. URL: <https://tailwindcss.com/docs/installation/> (Дата звернення: 22.03.2024)

ДОДАТКИ

Додаток А

Лейаут усіх сторінок додатку

```
import type { Metadata } from "next";
import { Inter } from "next/font/google";
import "./globals.css";
import Providers from "@components/Providers";
import Header from "@components/Header";
import Footer from "@components/Footer";

const inter = Inter({ subsets: ["latin"] });
export const metadata: Metadata = {
  title: "Hive",
  description:
    "Hive - Веб-сайт для придбання та оренди нерухомості по всій Україні",
};

export default function RootLayout({
  children,
}: Readonly<{
  children: React.ReactNode;
}>) {
  return (
    <html lang="en">
      <Providers>
        <body className={inter.className}>
          <Header />
          {children}
          <Footer />
        </body>
      </Providers>
    </html>
  );
}
```

Додаток Б

Код головної сторінки додатку

```

export default function Home() {
  return (
    <main className="flex min-h-screen flex-col overflow-x-hidden">
      <div className="h-[240px] md:h-[400px]">
        <div className="absolute bg-[url('/218052644.jpg')] h-[240px] md:h-[400px] bg-cover
w-full -z-20"></div>
        <div className="absolute bg-blue-900 opacity-80 h-[240px] md:h-[400px] w-full
-z-20"></div>
        <div className="absolute w-full h-[240px] md:h-[400px] flex flex-col items-center
justify-center -z-10 px-2 backdrop-blur-sm">
          <h1 className="text-4xl md:text-8xl font-semibold text-white text-center
tracking-[6px] md:tracking-[10px]">
            HIVE
          </h1>
          <p className="text-white text-md md:text-2xl tracking-[6px] md:tracking-[10px]
text-center">
            ОРЕНДА ТА ПРОДАЖ НЕРУХОМОСТІ
          </p>
        </div>
      </div>
      <motion.section
        className="container"
        initial={{ opacity: 0, translateX: -200 }}
        animate={{ opacity: 1, translateX: 0 }}
        transition={{ duration: 0.4 }}
      >
        <h2 className="font-semibold text-2xl md:text-4xl text-zinc-600 mt-16 text-center
md:text-left">
          Огляньте різні категорії нерухомості
        </h2>
        <ul className="flex flex-wrap items-center justify-center gap-8 my-8">
          {categories.map((el, i) => {
            return (
              <motion.li
                key={el.title}

```



```

        className="size-60"
        transition={{ duration: 0.15 }}
        whileHover={{ scale: 1.05 }}
        whileTap={{ scale: 0.95 }}
      >
        <CategoryItem item={el} />
      </motion.li>
    );
  })}
</ul>
<Separator className="bg-zinc-400" />
{ /* <h2 className="font-semibold text-2xl md:text-4xl text-zinc-600 mt-16 text-center
md:text-left">
  Бажаєте придбати житло чи орендувати?
</h2> */}
<Tabs defaultValue="buy" className="w-full my-6">
  <TabsList className="mb-4 bg-blue-600 text-white w-full md:w-auto">
    <TabsTrigger value="buy" className="w-full">
      Купівля
    </TabsTrigger>
    <TabsTrigger value="rent" className="w-full">
      Оренда
    </TabsTrigger>
  </TabsList>
  <TabsContent value="buy">
    <ul className="w-full grid grid-cols-1 md:grid-cols-2 lg:grid-cols-3 items-center
justify-center gap-6">
      {toBuy.map((el, i) => {
        return (
          <motion.li
            key={el.title}
            className="w-full h-48"
            transition={{ duration: 0.15 }}
            whileHover={{ scale: 1.05 }}
            whileTap={{ scale: 0.95 }}
          >
            <CategoryItem item={el} />
          </motion.li>
        );
      })}
    </ul>
  </TabsContent>
</Tabs>

```

```

    }}
  </ul>
</TabsContent>
<TabsContent value="rent">
  <ul className="w-full grid grid-cols-1 md:grid-cols-2 lg:grid-cols-3 items-center
justify-center gap-6">
    {toRent.map((el, i) => {
      return (
        <motion.li
          key={el.title}
          className="w-full h-48"
          transition={{ duration: 0.15 }}
          whileHover={{ scale: 1.05 }}
          whileTap={{ scale: 0.95 }}
        >
          <CategoryItem item={el} />
        </motion.li>
      );
    })}
  </ul>
</TabsContent>
</Tabs>
<Separator className="bg-zinc-400" />
  {/* <h2 className="font-semibold text-3xl md:text-4xl text-zinc-600 mt-16 text-center
md:text-left">
    Популярні пропозиції
  </h2>

  <CarouselMain /> */}
  <h2 className="font-semibold text-2xl md:text-4xl text-zinc-600 mt-24 text-center
md:text-left">
    Популярні питання
  </h2>
  <div className="mt-8 mx-auto mb-20">
    <Separator />

    <Accordion type="single" collapsible className="w-full">
      <AccordionItem value="item-1">
        <AccordionTrigger>

```

```

    Як я можу оголосити свою нерухомість на вашому сайті?
</AccordionTrigger>
<AccordionContent>
    Ви можете розмістити своє оголошення про оренду або продаж
    нерухомості, зареєструвавшись на сайті та перейшовши у Ваш
    профіль
</AccordionContent>
</AccordionItem>
<AccordionItem value="item-2">
    <AccordionTrigger>
        Як я можу знайти нерухомість для оренди або покупки на вашому
        сайті?
    </AccordionTrigger>
    <AccordionContent>
        Використовуйте наш розширений пошук у розділі "Оренда" або
        "Покупка", де ви можете вказати потрібні параметри, такі як
        місце розташування, тип нерухомості, цінний діапазон тощо, щоб
        знайти нерухомість, що відповідає вашим потребам.
    </AccordionContent>
</AccordionItem>
<AccordionItem value="item-3">
    <AccordionTrigger>
        Чи є у вас можливість перевірити нерухомість перед покупкою або
        орендою?
    </AccordionTrigger>
    <AccordionContent>
        На жаль, ми не здійснюємо безпосередньо перевірку нерухомості.
        Однак, ми рекомендуємо звертатися до фахівців з нерухомості або
        використовувати послуги незалежного експерта з оцінки
        нерухомості для отримання додаткової інформації.
    </AccordionContent>
</AccordionItem>
<AccordionItem value="item-4">
    <AccordionTrigger>
        Чи можу я змінити своє оголошення після його публікації?
    </AccordionTrigger>
    <AccordionContent>
        Ні, на жаль, після публікації оголошення зміни в ньому
        неможливі. Однак, якщо у вас з{``}явилися нові дані або

```

потреби, ви можете зняти це оголошення та опублікувати нове, враховуючи оновлену інформацію.

```

</AccordionContent>
</AccordionItem>
</Accordion>
</div>
</motion.section>
  <div className="bg-[url('/218052644.jpg')] h-[550px] bg-fixed bg-cover
backdrop-blur-md w-full">
  <div className="backdrop-blur w-full h-[550px] bg-black/20 flex flex-col items-center
justify-center">
    <div className="h-[400px] w-[400px] bg-white rounded-lg p-8 flex flex-col
items-center">
      <Image src="/logo.svg" width={80} height={80} alt="logo" />
      <h2 className="mb-8 mt-4 text-lg text-center">
        Увійдіть щоб розмістити оголошення
      </h2>
      <Link href="/auth/signin" className="w-full my-2 h-12">
        <Button type="button" className="w-full h-12">
          Увійти
        </Button>
      </Link>
      <div className="flex w-full justify-evenly items-center my-1">
        <Separator className="w-2/5 bg-zinc-300" />
        <span className="">або</span>
        <Separator className="w-2/5 bg-zinc-300" />
      </div>
      <Button
        type="button"
        className="w-full my-2 h-12"
        variant="outline"
      >
      <Image
        src="/free-google-160-189824.webp"
        width={20}
        height={20}
        alt="google"
        className="mr-2"
      />

```

```
        <span>Продовжити з Google</span>
    </Button>
</div>
</div>
</div>
</main>
);
}
```

Додаток В

Глобальні стилі додатку

```
@tailwind base;
@tailwind components;
@tailwind utilities;
@layer base {
  :root {
    --background: 0 0% 100%;
    --foreground: 222.2 84% 4.9%;
    --card: 0 0% 100%;
    --card-foreground: 222.2 84% 4.9%;
    --popover: 0 0% 100%;
    --popover-foreground: 222.2 84% 4.9%;
    --primary: 221.2 83.2% 53.3%;
    --primary-foreground: 210 40% 98%;
    --secondary: 210 40% 96.1%;
    --secondary-foreground: 222.2 47.4% 11.2%;
    --muted: 210 40% 96.1%;
    --muted-foreground: 215.4 16.3% 46.9%;
    --accent: 210 40% 96.1%;
    --accent-foreground: 222.2 47.4% 11.2%;
    --destructive: 0 84.2% 60.2%;
    --destructive-foreground: 210 40% 98%;
    --border: 214.3 31.8% 91.4%;
    --input: 214.3 31.8% 91.4%;
    --ring: 221.2 83.2% 53.3%;
    --radius: 0.5rem;
  }

  .dark {
    --background: 222.2 84% 4.9%;
    --foreground: 210 40% 98%;
    --card: 222.2 84% 4.9%;
    --card-foreground: 210 40% 98%;
    --popover: 222.2 84% 4.9%;
    --popover-foreground: 210 40% 98%;
    --primary: 217.2 91.2% 59.8%;
```

```
--primary-foreground: 222.2 47.4% 11.2%;  
--secondary: 217.2 32.6% 17.5%;  
--secondary-foreground: 210 40% 98%;  
--muted: 217.2 32.6% 17.5%;  
--muted-foreground: 215 20.2% 65.1%;  
--accent: 217.2 32.6% 17.5%;  
--accent-foreground: 210 40% 98%;  
--destructive: 0 62.8% 30.6%;  
--destructive-foreground: 210 40% 98%;  
--border: 217.2 32.6% 17.5%;  
--input: 217.2 32.6% 17.5%;  
--ring: 224.3 76.3% 48%;  
}  
}
```

```
@layer base {  
  * {  
    @apply border-border;  
  }  
  body {  
    @apply bg-background text-foreground;  
  }  
}
```



метадані

Заголовок

Розробка веб-сайту для продажу та оренди нерухомості

Автор

Король В. В. Науковий керівник / Експерт

підрозділ

King Danylo University

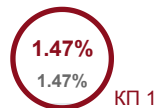
Тривога

У цьому розділі ви знайдете інформацію щодо текстових спотворень. Ці спотворення в тексті можуть говорити про **МОЖЛИВІ** маніпуляції в тексті. Спотворення в тексті можуть мати навмисний характер, але частіше характер технічних помилок при конвертації документа та його збереженні, тому ми рекомендуємо вам підходити до аналізу цього модуля відповідально. У разі виникнення запитань, просимо звертатися до нашої служби підтримки.

Заміна букв		0
Інтервали		0
Мікропробіли		0
Білі знаки		0
Парафрази (SmartMarks)		5

Обсяг знайдених подібностей

Коефіцієнт подібності визначає, який відсоток тексту по відношенню до загального обсягу тексту було знайдено в різних джерелах. Зверніть увагу, що високі значення коефіцієнта не автоматично означають плагіат. Звіт має аналізувати компетентна / уповноважена особа.



25
Довжина фрази для коефіцієнта подібності 2



9724
Кількість слів

76062
Кількість символів

Подібності за списком джерел

Нижче наведений список джерел. В цьому списку є джерела із різних баз даних. Колір тексту означає в якому джерелі він був знайдений. Ці джерела і значення Коефіцієнту Подібності не відображають прямого плагіату. Необхідно відкрити кожне джерело і проаналізувати зміст і правильність оформлення джерела.

10 найдовших фраз

Колір тексту

ПОРЯДКОВИЙ НОМЕР	НАЗВА ТА АДРЕСА ДЖЕРЕЛА URL (НАЗВА БАЗИ)	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)	
1	https://cad.kpi.ua/attachments/093_2017d_Krasiuk.pdf	23	0.24 %
2	https://openarchive.nure.ua/bitstream/document/10952/1/2019_M_IUS_Volovik_S_M.pdf	20	0.21 %
3	Kravtsov_diplom_mag_Khruslov_fkn_2023.pdf 12/3/2023 V. N. Karazin Kharkiv National University (KGNU) (Факультет комп'ютерних наук - кафедра електроніки і управляючих систем)	10	0.10 %