НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки

Кафедра інформаційних систем та технологій

«На правах рукопису»
УДК _004.77_

До захисту допущено:
Завідувач кафедри
_____ Олександр РОЛІК
«___»_____2024 р.

Магістерська дисертація

на здобуття ступеня магістра
за освітньо-професійною програмою «Інформаційні управляючі системи та технології»
зі спеціальності 126 «Інформаційні системи та технології»
на тему: «Система оренди житла на основі мікросервісної архітектури»

Виконав:
студент 2 курсу, групи ІС-32мп
Куник Юрій Ігорович _____

Керівник:
професор каф. ІСТ, д.т.н, проф.
Теленик Сергій Федорович _____

Рецензент:
доцент каф. ОТ, к.т.н, доц.
Волокита Артем Миколайович _____

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____

Київ – 2024 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Рівень вищої освіти – другий (магістерський)

Спеціальність – 126 «Інформаційні системи та технології»

Освітньо-професійна програма «Інформаційні управляючі системи та технології»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Олександр РОЛІК
«___»_____2024 р.

ЗАВДАННЯ
на магістерську дисертацію студенту
Кунику Юрію Іговичу

1. Тема дисертації «Система оренди житла на основі мікросервісної архітектури», науковий керівник дисертації Теленик Сергій Федорович, професор каф. ІСТ, затверджені наказом по університету від «08» 11 2024 р. № 5016-с

2. Термін подання студентом дисертації «09» 12 2024 р.

3. Об’єкт дослідження: процес діяльності компанії оренди нерухомості.

4. Вихідні дані : аналіз діяльності клієна, структури організації ,функціональні вимоги до можливостей системи ,нефункціональні вимоги, технології для розробки серверного ПЗ та баз даних.

5. Перелік завдань, які потрібно розробити: провести аналіз предметної області та існуючих рішень, формування функціональних вимог, реалізація інформаційної системи.

6. Орієнтовний перелік графічного (ілюстративного) матеріалу: схеми бізнес-процесів користувачів, ER діаграми баз даних,схеми аріхітектури системи, діаграма прецедентів.

7. Орієнтовний перелік публікацій: не заплановані

8. Дата видачі завдання 02.09.2024 р.

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1	Аналіз вимог та проектування системи	08.09.2024	
2	Розробка базових мікросервісів	20.09.2024	
3	Розробка додаткових мікросервісів	28.09.2024	
4	Тестування та відладка системи	12.10.2024	
5	Деплоймент і підготовка до запуску	19.10.2024	
6	Оформлення документації	26.10.2024	
7	Подання роботи на захист	09.12.2024	

Студент Юрій КУНИК

Науковий керівник Сергій ТЕЛЕНИК

РЕФЕРАТ

Система оренди житла на основі мікросервісної архітектури: 110 с., 18 табл., 15 рис., 8 дод., 15 джерел.

МІКРОСЕРВІСИ, ОРЕНДА ЖИТЛА, АРХІТЕКТУРА СИСТЕМ, МАСШТАБОВАНІСТЬ, БЕЗПЕКА, КЛАСТЕРИЗАЦІЯ, РЕКОМЕНДАЦІЙНА СИСТЕМА.

У роботі досліджено розробку інформаційної системи оренди житла з використанням мікросервісної архітектури. Актуальність теми зумовлена необхідністю створення гнучких та надійних рішень для автоматизації процесів оренди, забезпечення безпеки та масштабованості.

Метою роботи є створення ефективної системи для управління орендою житла, що автоматизує основні процеси, забезпечує інтеграцію із зовнішніми сервісами та використовує алгоритми рекомендацій.

Об’єктом дослідження є система управління орендою житла, а предметом — застосування мікросервісної архітектури для підвищення її ефективності.

Створена система є типовим рішенням для організацій, що займаються збутом нерухомості. Вона побудована на основі мікросервісної архітектури, яка забезпечує високу гнучкість і легкість налаштування під потреби конкретного підприємства. Система надає повний перелік функцій для ефективного управління нерухомістю, включаючи організацію процесів продажу, обробку даних клієнтів, генерацію звітів та автоматизацію ключових бізнес-процесів. Це дозволяє організаціям оптимізувати свою діяльність і підвищити конкурентоспроможність на ринку.

ABSTRACT

Housing rental system based on microservice architecture: 110 p., 18 tab., 15 draw., 8 app., 15 sources.

MICROSERVICES, RENTAL HOUSING, SYSTEM ARCHITECTURE, SCALABILITY, SECURITY, CLUSTERING, RECOMMENDATION SYSTEM.

This thesis explores the development of a rental housing system using microservice architecture. The relevance of the topic is driven by the need for flexible and reliable solutions to automate rental processes, ensure data security, and enable system scalability.

The goal of the work is to create an efficient system for managing rental operations that automates core processes, integrates with external services, and incorporates recommendation algorithms.

The object of the study is a rental housing management system, and the subject is the application of microservice architecture to enhance its efficiency and reliability.

A typical system has been developed for organizations involved in real estate sales. It is built on a microservice architecture, providing high flexibility and ease of customization to meet the specific needs of a particular enterprise. The system offers a comprehensive set of functions for efficient real estate management, including the organization of sales processes, customer data processing, report generation, and automation of key business operations. This enables organizations to optimize their activities and enhance their competitiveness in the market.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ ТА ПОЗНАЧЕНЬ 8
ВСТУП 9
1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ І ПОСТАНОВКА ЗАДАЧІ 11
1.1 Загальний огляд 11
1.2 Особливості управління бізнесом оренди житла 12
1.3 Огляд існуючих рішень 13
1.4 Постановка задач на магістерську роботу 18
Висновки до розділу 1 20
2 ПРОЕКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ 21
2.1 Аналіз бізнесу 21
2.1.1 Цілі компанії 21
2.1.2 Структура організації 22
2.1.3 Послуги які надає організація 24
2.1.4 Методи надання послуг організацією та роль інформаційної системи 26
2.1.5 Опис стейкхолдерів бізнесу і як система буде сприяти їх цілям 28
2.2 Формування вимог до системи 28
2.2.1 Функціональні вимоги 29
2.2.2 Нефункціональні вимоги 31
2.3 Аналіз обмежень 32
2.4 Аналіз ризиків 34
2.5 Опис бізнес-процесів користувачів 36
2.5.1 Орендар 37
2.5.2 Орендодавець 45
2.5.3 Адміністратор 51
Висновки до розділу 2 55
3 АРХІТЕКТУРА СИСТЕМИ 56
3.1 Загальний опис архітектури системи 57
3.2 Опис схеми бази даних 60

Висновки до розділу 3 63
4 МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ 65
4.1 Постановка задачі 65
4.1.1 Формалізація задачі кластеризації 65
4.2 Алгоритм K-means 66
4.2.1 Розрахунок відповідності квартир користувачам 67
4.2.2 Рекомендаційний алгоритм 68
Висновки до розділу 4 69
5 ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ 71
5.1 Вибір програмного стеку для розробки 71
5.2 Вибір системного програмного забезпечення 76
5.2.1 Керовані сервіси Microsoft Azure 76
5.3 Реалізація бізнес-логіки 78
5.3.1 Сервер 78
5.3.2 Клієнт 82
Висновки до розділу 5 83
6 РОЗРОБЛЕННЯ ІНТЕРФЕЙСУ КОРИСТУВАЧА 85
Висновки до розділу 6 89
7 РОЗРОБЛЕННЯ СТАРТАП-ПРОЕКТУ 90
7.1 Опис ідеї стартап-проекту 90
7.2 Технологічний аудит ідеї проекту 94
7.3 Аналіз ринкових можливостей запуску стартапу 95
7.4 Розробка ринкової стратегії проекту 103
7.5 Розробка маркетингової програми стартап-проекту 106
Висновки до розідлу 7 107
ВИСНОВКИ 108
ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ 109

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ ТА ПОЗНАЧЕНЬ

API – Application Programming Interface – інтерфейс прикладного програмування, що забезпечує взаємодію між сервісами системи.

JWT – JSON Web Token – стандарт для безпечного обміну інформацією у вигляді зашифрованих токенів, використовується для аутентифікації та авторизації.

REST – Representational State Transfer – архітектурний стиль для створення веб-сервісів із використанням стандартних HTTP-запитів.

DB – Database – база даних, сховище структурованої інформації.

ERD – Entity-Relationship Diagram – діаграма "сутність-зв’язок", що відображає структуру бази даних.

UI – User Interface – інтерфейс користувача, який забезпечує взаємодію користувача із системою.

ВСТУП

Оренда житла є однією з найпопулярніших послуг у сфері нерухомості, що постійно розвивається, особливо з огляду на зростання міграційних потоків та збільшення мобільності населення [1]. Однак з розвитком технологій і цифровізацією ринку з'являються нові виклики, серед яких:

а) потреба у швидкому та безпечному доступі до інформації про об’єкти нерухомості;

б) високі вимоги до конфіденційності даних;

в) важливість зручної інтеграції платіжних систем;

г) гнучкість у масштабуванні сервісів для обслуговування великої кількості користувачів.

Ці виклики роблять сучасний ринок оренди житла ідеальною галуззю для впровадження інноваційних технологій, зокрема мікросервісної архітектури. Завдяки розподіленню функцій між окремими сервісами, мікросервісна архітектура може забезпечити необхідну масштабованість, стійкість та швидкість доступу до даних, які важливі для управління процесами оренди

Традиційні монолітні системи не завжди можуть ефективно обробляти великий потік даних і запитів, що призводить до повільної роботи, низької продуктивності та обмежених можливостей для масштабування. У сфері оренди житла це може стати значною проблемою, оскільки кількість одночасних запитів користувачів може бути дуже великою (наприклад, під час сезонного попиту на житло або святкових періодів). Також особливої уваги потребують питання безпеки, адже системи працюють із персональними даними користувачів та інформацією про платежі.

Мікросервісна архітектура дозволяє розподілити різні аспекти роботи системи між окремими сервісами, які функціонують незалежно один від одного. Це дозволяє забезпечити стабільність системи навіть при великому навантаженні та покращити безпеку, розділяючи конфіденційні дані між сервісами.

Метою цієї роботи є розробка системи оренди житла на основі мікросервісної архітектури, яка здатна забезпечити ефективне управління операціями оренди, зручний доступ користувачів до інформації про об’єкти, безпеку персональних даних і можливість інтеграції з іншими сервісами.

Завдання дослідження**:**

Для досягнення поставленої мети необхідно вирішити такі завдання:

а) провести аналіз сучасного ринку оренди житла та доступних інформаційних систем для бронювання та управління орендою;

б) визначити функціональні та нефункціональні вимоги до системи на основі мікросервісної архітектури;

в) розробити архітектуру системи з описом основних мікросервісів

г) реалізувати систему на основі обраної архітектури з використанням сучасних веб-фреймворків, СУБД та інструментів для контейнеризації;

д) провести тестування розробленої системи та оцінити її продуктивність, масштабованість і стійкість до навантаження;

е) підготувати рекомендації щодо подальшого розвитку системи та її адаптації для реального використання.

Об'єкт і предмет дослідження

а) Об'єкт дослідження: система управління орендою житла.

б) Предмет дослідження: мікросервісна архітектура для забезпечення надійності, масштабованості та безпеки інформаційної системи в галузі оренди нерухомості.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ І ПОСТАНОВКА ЗАДАЧІ

1.1 Загальний огляд

Бізнес оренди житла — це сфера нерухомості, яка надає житлові приміщення в користування на умовах оренди на певний термін, від кількох діб до кількох років. Даний сектор розділяється на дві основні категорії: довгострокову оренду та короткострокову оренду.

а) довгострокова оренда охоплює житло, яке здається на період від декількох місяців до кількох років. Цей вид оренди особливо популярний серед людей, які переїжджають на тривалий термін у зв’язку з роботою або навчанням. Довгострокова оренда передбачає стабільний дохід для орендодавців та забезпечує орендарям зручність у тривалому користуванні житлом без необхідності частих переїздів.

б) короткострокова оренда (від кількох днів до місяця) здебільшого використовується туристами, бізнес-мандрівниками та людьми, яким необхідно зняти житло на короткий термін. Ця модель швидко розвивається завдяки платформам, таким як Airbnb і Booking.com, які зробили короткострокову оренду доступною, зручною та безпечною для широкого кола користувачів. Орендодавці можуть здавати житло на короткий час і отримувати високий дохід завдяки гнучкості цієї моделі.

Попит та тенденції. Сучасний бізнес оренди житла характеризується зростанням попиту, особливо на короткострокову оренду, що пов'язано з кількома ключовими тенденціями:

а) зростання туристичної активності та підвищення попиту на короткострокове житло у зв’язку з розвитком сервісів для онлайн-бронювання.

б) підвищення мобільності населення: більше людей змінюють місце проживання на короткий термін для роботи, навчання чи мандрівок, що сприяє популярності короткострокової та середньострокової оренди.

в) потреба у зручності: сучасні користувачі очікують швидкого та зручного доступу до орендного житла, можливості онлайн-бронювання, швидких оплат та інтегрованих послуг.

1.2 Особливості управління бізнесом оренди житла

Оренда житла є складним процесом, що вимагає від власників або керуючих компаній надійного управління великим обсягом даних, постійного моніторингу ринку, а також дотримання правових норм. Основними аспектами бізнесу оренди житла є:

а) робота з клієнтами: забезпечення зручності в користуванні платформою, підтримка користувачів, обробка відгуків і вирішення спірних питань.

б) фінансове управління: облік доходів та витрат, інтеграція з платіжними системами для проведення транзакцій, контроль заборгованості орендарів.

в) безпека даних і користувачів: зберігання персональних даних орендарів та власників житла, забезпечення їхньої конфіденційності та безпеки платежів.

г) дотримання правових норм: у різних юрисдикціях існують закони та вимоги до оренди житла, і бізнес повинен відповідати цим нормам, щоб уникати юридичних проблем.

З появою цифрових платформ (наприклад, Airbnb, Booking.com, Zillow) бізнес оренди житла значно трансформувався. Сучасні платформи надають можливості для швидкого бронювання житла, проведення онлайн-платежів, залишення відгуків, а також забезпечують зручні інтерфейси для управління об'єктами нерухомості. Завдяки цифровим платформам зростає доступність оренди як для орендодавців, так і для орендарів, але разом з тим постає необхідність у надійній та гнучкій архітектурі, яка б забезпечувала стабільність та безпеку всіх операцій.

Мікросервісна архітектура стала ефективним підходом для управління великими системами, зокрема у сфері оренди житла. Вона дозволяє розділити комплексні процеси на окремі незалежні сервіси, які можуть масштабуватися та оновлюватися незалежно один від одного. Використання мікросервісів допомагає:

а) забезпечити високу продуктивність за рахунок можливості паралельної обробки різних завдань;

б) знизити ризики збоїв завдяки ізольованій роботі кожного сервісу;

в) спростити впровадження нових функцій та інтеграцію з іншими системами, такими як платіжні сервіси або сервіси авторизації.

Цей підхід до архітектури особливо корисний у секторі оренди житла, де потрібно обробляти великий обсяг запитів, а також гарантувати безпеку даних і забезпечити надійність для користувачів.

1.3 Огляд існуючих рішень

На сьогодні існує кілька великих платформ для оренди житла, які впроваджують нові технології, щоб забезпечити надійність, доступність та швидкість обслуговування. До нойбільш відомих в Україні відносяться:

Booking.com - це один з найбільших та найпопулярніших онлайн-сервісів бронювання готелів, апартаментів, гостьових будинків та інших видів тимчасового проживання по всьому світу[2]. Створений у 1996 році, цей сайт пропонує мільйони варіантів проживання у більш ніж 220 країнах, забезпечуючи користувачам широкий вибір за доступними цінами. На сайті booking.com зображеному на рисунку 1.1 та рисунку 1.2 користувачі можуть швидко та зручно знайти і забронювати проживання за допомогою різноманітних фільтрів та параметрів пошуку, таких як місцезнаходження, дати проживання, кількість гостей та багато іншого. Сервіс також пропонує велику кількість відгуків від реальних гостей, що допомагає зробити обґрунтований вибір.

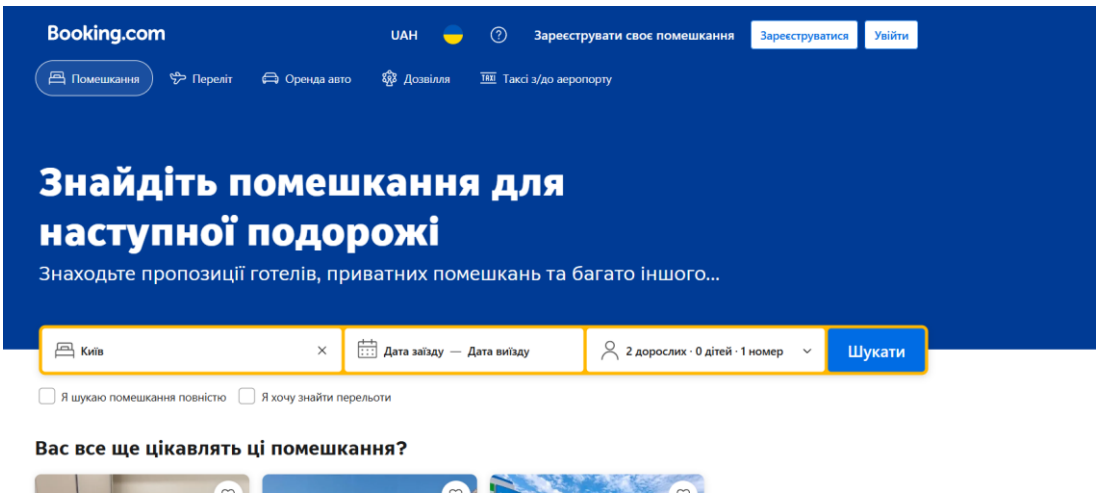

Рисунок 1.1 – Головна сторінка сервісу booking.com

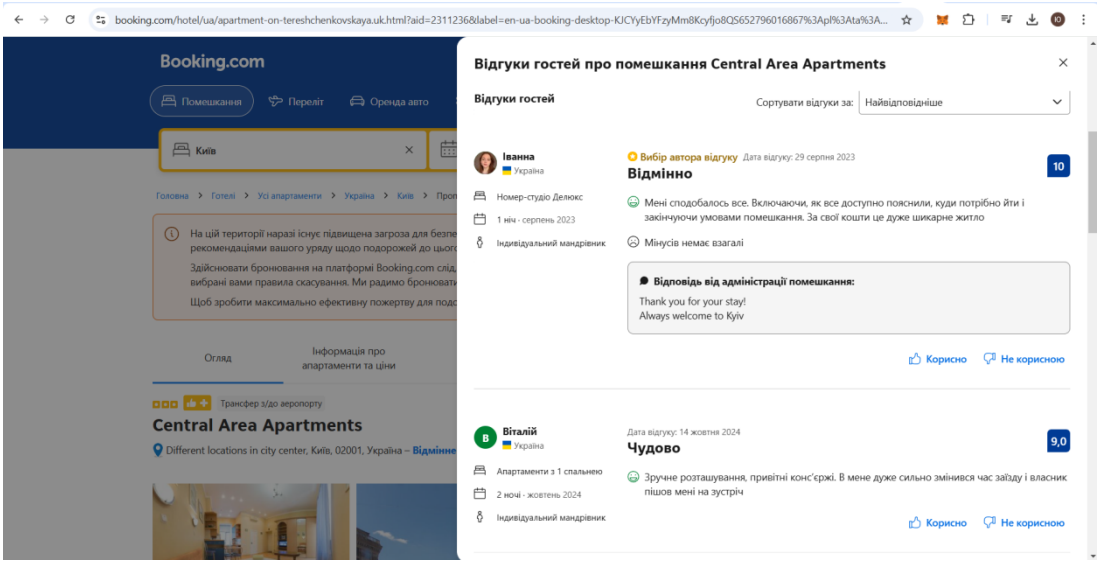

Рисунок 1.2 – Відгуки клієнтів на апартаменти

Плюси застосунку:

а) широкий вибір житла: від готелів до приватних апартаментів та будинків;

б) міжнародна доступність: підтримка багатьох мов і валюти;

в) зручний інтерфейс: інтуїтивно зрозумілий дизайн та зручна система пошуку;

г) розширені фільтри: велика кількість фільтрів для пошуку житла за різними критеріями.

Мінуси застосунку:

а) обмеженість інформації про доступність:

б) висока комісія: для власників житла комісія за використання платформи може бути високою;

в) відсутність підтримки довгострокової оренди: платформа орієнтована на короткострокові бронювання.

flatfy.ua – це онлайн-платформа, що спеціалізується на пошуку та продажу нерухомості в Україні [3]. Цей сайт надає користувачам можливість швидко та зручно знаходити квартири, будинки, кімнати та комерційну нерухомість за допомогою розширених фільтрів пошуку та зручної картографії. На flatfy.ua зображеному на рисунку 1.3 представлені тисячі актуальних оголошень від різних агентств нерухомості та приватних власників, що дозволяє користувачам знайти ідеальне житло за їхніми потребами та можливостями. Кожне оголошення містить докладну інформацію про об'єкт, включаючи фотографії, опис, параметри та контактну інформацію.

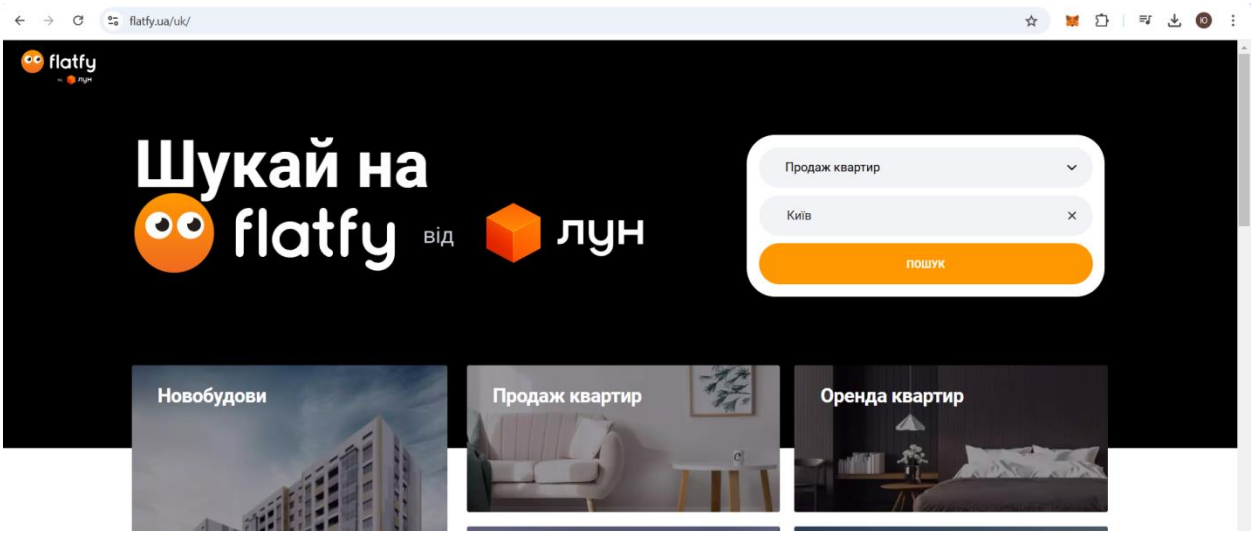

Рисунок 1.3 – Головна сторінка сайту flatfy.ua

Платформа flatfy.ua також пропонує користувачам ряд корисних інструментів, таких як калькулятори вартості нерухомості, порівняльний аналіз цін, а також новини та статті про ринок нерухомості для тих, хто цікавиться останніми тенденціями та порадами щодо купівлі або продажу житла.

Плюси платформи:

а) велика база даних: широкий вибір оголошень з різних джерел;

б) розширені фільтри: можливість фільтрування за різними параметрами, включаючи ціну, район, кількість кімнат тощо;

в) місцевий фокус: платформа орієнтована на український ринок, що дозволяє знайти житло у конкретних містах та районах.

Мінуси платформи:

а) застарілий інтерфейс: дизайн сайту потребує модернізації для покращення зручності користування;

б) обмеженість підтримки: відсутність цілодобової підтримки користувачів.

Сайт rieltor.ua – це онлайн-платформа, спеціалізована на нерухомості в Україні. Незалежно від того, чи ви шукаєте квартиру для оренди, купівлі будинку чи комерційну нерухомість, rieltor.ua надає зручний інструмент для пошуку та вибору ідеального об'єкта[4]. На сайті зображеному на рисунках. 1.4 та 1.5.

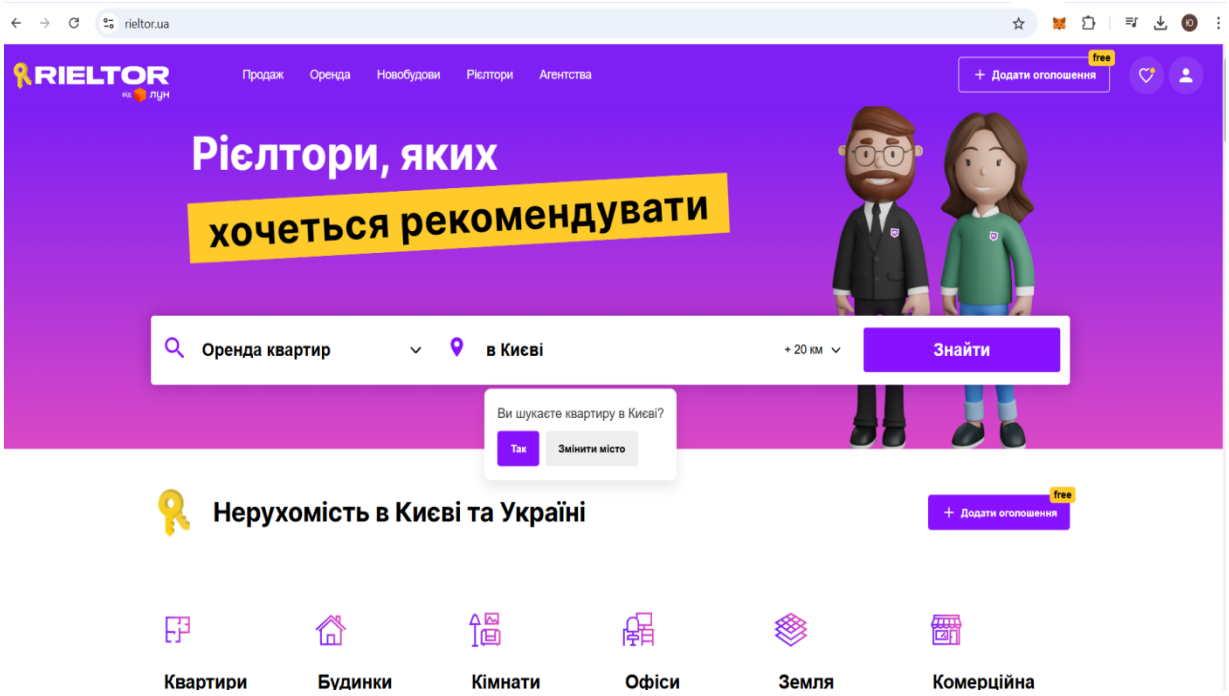

Рисунок 1.4 – Головна сторінка сайту rieltor.ua

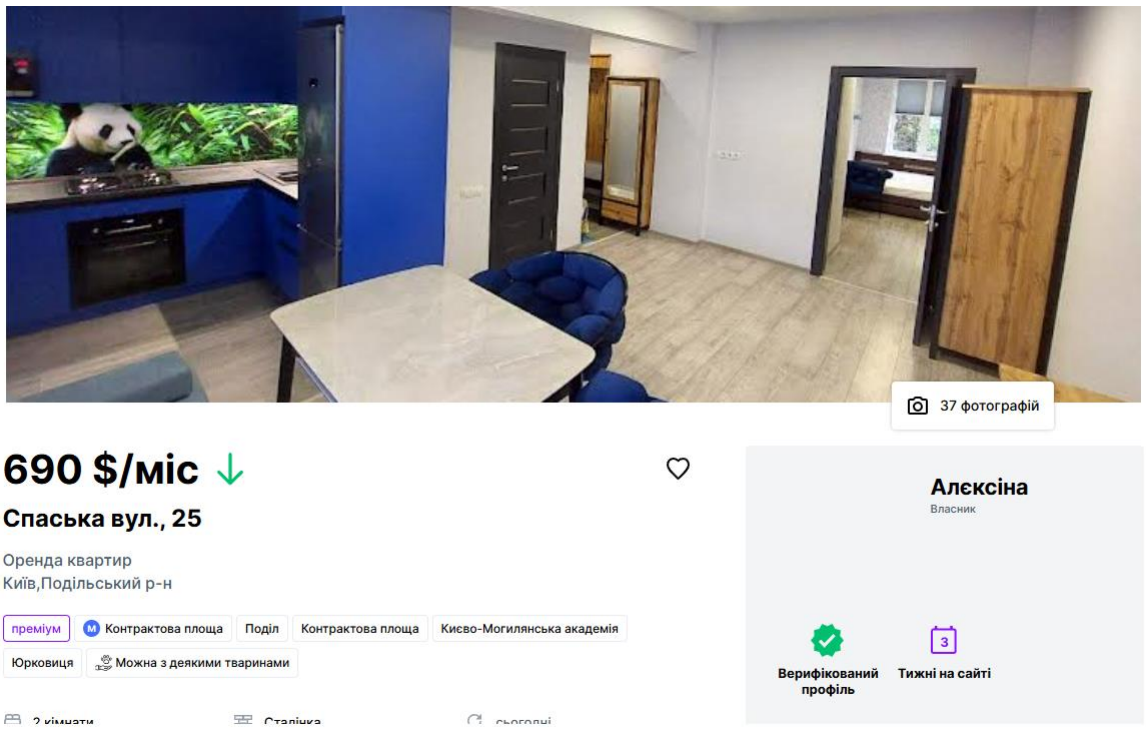

Рисунок 1.5 – Вікно апартаментів в rieltor.ua

представлені тисячі актуальних оголошень від професійних агентств нерухомості та приватних власників. Користувачі можуть використовувати різноманітні фільтри пошуку, такі як місцезнаходження, тип нерухомості, ціновий діапазон та інші параметри, щоб швидко знайти відповідні варіанти. Оголошення на rieltor.ua містять докладну інформацію про об'єкт, включаючи фотографії, характеристики, умови продажу або оренди та контактну інформацію для зв'язку з власниками або агентами. Крім того, сайт пропонує корисні інструменти, такі як калькулятори вартості нерухомості та порівняльний аналіз ринкових цін.

Плюси сайту:

а) широкий вибір варіантів: велика кількість оголошень з детальними описами;

б) підтримка ріелторів: можливість скористатися послугами професійних ріелторів;

в) розширені фільтри: можливість пошуку за різними параметрами, включаючи тип нерухомості, ціну та місцезнаходження.

1.4 Постановка задач на магістерську роботу

Метою роботи є розробка ефективної та зручної системи для онлайн-бронювання квартир з використанням мікросервісної архітектури, яка забезпечує надійність, масштабованість і безпеку обміну даними, спрощує процес пошуку й бронювання житла для орендарів, а також надає інструменти для зручного управління об’єктами оренди для власників.

Для досягнення поставленої мети необхідно вирішити наступні завдання:

а) аналіз ринку та існуючих рішень у сфері оренди житла**;**

1) провести дослідження сучасного ринку оренди житла, зокрема щодо популярних бізнес-моделей та існуючих цифрових рішень, які застосовуються для управління орендою;

2) оцінити архітектурні підходи у сучасних системах оренди житла, зокрема таких платформ, як rieltor.ua, Booking.com, для визначення сильних і слабких сторін існуючих рішень;

б) формування вимог до системи**;**

1) визначити функціональні вимоги до системи, зокрема функції пошуку житла, обробки бронювань, управління користувачами, проведення онлайн-платежів та забезпечення безпеки даних;

2) сформулювати нефункціональні вимоги, які включають продуктивність, масштабованість, стійкість до навантажень та захист конфіденційної інформації;

в) проектування архітектури системи на основі мікросервісів**;**

1) розробити загальну архітектуру системи, де основні компоненти (мікросервіси) будуть розділені за функціональністю.

2) визначити способи комунікації між мікросервісами (REST API, gRPC) та забезпечити їхню сумісність через API шлюз;

3) зпроектувати базу даних для зберігання інформації про користувачів, об’єкти оренди та бронювання з урахуванням вимог до цілісності та доступності даних;

г) реалізація та інтеграція основних компонентів системи**;**

1) розробити окремі мікросервіси, які відповідатимуть за виконання конкретних функцій системи;

2) забезпечити інтеграцію з платіжними системами для проведення транзакцій між орендодавцями та орендарями;

3) впровадити механізми аутентифікації та авторизації користувачів які гарантують безпеку доступу до ресурсів;

д) забезпечення безпеки даних та конфіденційності користувачів**;**

1) розробити заходи захисту даних для забезпечення конфіденційності та безпеки персональної інформації користувачів, у тому числі засоби шифрування даних і захищені протоколи передачі;

2) забезпечити контроль доступу до даних для різних категорій користувачів (орендарів, орендодавців, адміністраторів);

е) тестування та оцінка ефективності системи**;**

1) провести функціональне тестування для перевірки коректної роботи основних компонентів системи;

2) провести тестування безпеки для оцінки ефективності заходів захисту персональних даних та конфіденційності користувачів;

ж) аналіз отриманих результатів та розробка рекомендацій щодо покращення**;**

1) на основі результатів тестування сформувати висновки щодо ефективності системи;

2) виявити можливі напрями для покращення продуктивності, безпеки та користувацького досвіду системи.

Висновки до розділу 1

У цьому розділі було проведено загальний огляд ринку оренди житла та основних характеристик цього бізнесу, а також здійснено огляд існуючих рішень у галузі цифрових платформ для оренди житла, таких як rieltor.ua, Booking.com і flatfy.ua. Розглянуто ключові аспекти управління орендним бізнесом, сучасні тенденції та виклики, які стоять перед розробниками таких систем. На основі аналізу було виявлено, що мікросервісна архітектура є найбільш відповідним підходом для побудови системи оренди житла, оскільки вона забезпечує необхідну гнучкість, масштабованість та надійність для стабільної роботи.

Далі у розділі були сформульовані завдання, які необхідно виконати в рамках магістерської роботи. До основних задач відносяться розробка вимог до системи, проектування та реалізація архітектури на основі мікросервісів, інтеграція з платіжними сервісами.

Результати, отримані в цьому розділі, закладають основу для проектування і розробки системи оренди житла, яка відповідатиме сучасним вимогам ринку. У наступних розділах буде розглянуто конкретні технічні аспекти реалізації системи та подано детальний опис кожного з мікросервісів, які забезпечуватимуть основні функції платформи.

2 ПРОЕКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

2.1 Аналіз бізнесу

Організація, для якої розробляється система оренди житла, є онлайн-платформою з оренди нерухомості, що спеціалізується на забезпеченні доступу до житла для орендарів, а також на наданні інструментів для управління об'єктами для орендодавців. Це може бути велика компанія, подібна до Airbnb або Booking.com, або менша компанія, яка бажає автоматизувати процеси оренди для оптимізації витрат та покращення користувацького досвіду. Основна мета організації — спростити та прискорити процес оренди житла, зробити його зручним і безпечним для обох сторін, а також забезпечити високу надійність та безперебійний доступ до сервісу.

2.1.1Цілі компанії

Основна ціль компанії — максимально автоматизувати процеси, пов'язані з орендою житла, такі як розміщення оголошень, бронювання, проведення оплат і управління об'єктами. Це дозволить зменшити потребу в ручному управлінні, знизити витрати на обслуговування та підвищити ефективність роботи платформи.

а) забезпечення зручності для користувачів**.** Компанія прагне створити інтуїтивно зрозумілий і зручний інтерфейс для орендарів та орендодавців, що дозволить швидко знаходити житло, здійснювати бронювання та проводити оплату. Платформа повинна бути доступною на будь-якому пристрої (веб та мобільні додатки) і надавати користувачам позитивний досвід взаємодії;

б) підвищення безпеки та довіри користувачів. Компанія прагне забезпечити високий рівень безпеки даних користувачів, транзакцій та конфіденційної інформації. Це включає надійні методи аутентифікації, захист від шахрайства та безпечну обробку платежів, що сприятиме підвищенню довіри з боку користувачів;

в) створення надійної та масштабованої платформи. Компанія прагне створити систему, яка здатна обробляти великий потік запитів без зниження продуктивності. Платформа повинна легко масштабуватися у відповідь на зростання кількості користувачів та об'єктів, що дозволить забезпечити її стабільність і високу доступність;

г) підтримка зворотного зв’язку та поліпшення сервісу. Компанія прагне постійно вдосконалювати свою платформу на основі зворотного зв’язку від користувачів. Для цього передбачається збір відгуків, аналіз поведінки користувачів та впровадження покращень для підвищення якості обслуговування;

д) оптимізація витрат та підвищення рентабельності. Компанія ставить за мету оптимізувати витрати на обслуговування платформи, підвищити ефективність роботи співробітників та забезпечити стабільний дохід від наданих послуг. Це досягається за рахунок автоматизації процесів, зменшення ручної роботи та зниження потреби в додаткових ресурсах.

2.1.2 Структура організації

Організація, яка надає послуги з оренди житла, має типову структуру, що складається з відділів та ролей, які взаємодіють для забезпечення ефективного функціонування платформи. Основні відділи та їх функції:

а) відділ управління продуктом;

1) відповідає за стратегічне планування, розвиток функціоналу платформи, врахування потреб ринку та клієнтів, а також загальний дизайн користувацького інтерфейсу;

2) керівник продукту, менеджери продукту, UX/UI дизайнери;

3) аналіз потреб користувачів, планування нових функцій та розробка прототипів, проведення досліджень і тестування користувацького досвіду;

б) технічний відділ (розробка та підтримка платформи);

1) розробляє, підтримує та оновлює платформу, впроваджує новий функціонал, займається налаштуванням інфраструктури та забезпеченням стабільності системи;

2) технічний директор, розробники (бекенд та фронтенд), тестувальники, DevOps-інженери, спеціалісти з кібербезпеки;

3) розробка архітектури платформи, створення та тестування функціоналу, забезпечення безпеки даних користувачів, підтримка масштабованості та надійності системи;

в) відділ клієнтської підтримки;

1) забезпечує допомогу користувачам з технічних та адміністративних питань, вирішує конфліктні ситуації між орендарями та орендодавцями;

2) керівник підтримки клієнтів, фахівці підтримки (оператори), спеціалісти з обробки запитів;

3) надання технічної та адміністративної підтримки користувачам, відповіді на запитання клієнтів, допомога у вирішенні спірних ситуацій щодо бронювання та платежів;

г) відділ фінансів;

1) здійснює управління фінансами, забезпечує точний облік коштів, контроль за фінансовими операціями та інтеграцію з платіжними системами;

2) фінансовий директор, бухгалтер, спеціалісти з обробки платежів;

3) ведення обліку транзакцій, обробка платежів і виплат орендодавцям, проведення фінансових звітів, управління комісіями та податками;

д) відділ маркетингу та розвитку бізнесу;

1) розробляє та реалізує маркетингові стратегії, проводить рекламні кампанії, сприяє залученню нових користувачів та утриманню існуючих клієнтів;

2) маркетинговий директор, спеціалісти з цифрового маркетингу, контент-менеджери, аналітики;

3) просування платформи на ринку, реклама серед орендодавців і орендарів, аналіз конкурентів і ринку, розробка програми лояльності для користувачів;

е) відділ юридичних послуг:

1) забезпечує відповідність діяльності платформи вимогам законодавства, розробляє та впроваджує політики конфіденційності й використання даних;

2) юрисконсульт, спеціалісти з юридичних питань у сфері нерухомості;

3) підготовка договорів, консультування з юридичних питань, розробка політик захисту даних, робота із скаргами та претензіями користувачів.

2.1.3 Послуги які надає організація

До послуг які надає організація можна віднести:

а) організація надає доступ до каталогу об’єктів нерухомості, що здаються в оренду. Орендодавці можуть розміщувати свої об'єкти з детальними описами, фотографіями, цінами та умовами оренди. Орендарі мають змогу переглядати доступні об'єкти, використовуючи різні фільтри для пошуку відповідного житла (місцезнаходження, кількість кімнат, ціна, зручності тощо);

б) орендарі можуть здійснювати бронювання обраних об'єктів нерухомості в реальному часі. Система автоматично оновлює статус доступності об’єктів, запобігаючи подвійним бронюванням. Процес бронювання простий і швидкий, що дозволяє орендарям за кілька кліків забронювати потрібне житло;

в) управління об'єктами для орендодавців. Орендодавці отримують інструменти для управління своїми об'єктами. Вони можуть додавати, редагувати

та видаляти оголошення, оновлювати інформацію про доступність об'єктів та коригувати умови оренди. Це дозволяє їм легко управляти своїм портфоліо житла;

г) обробка онлайн-платежів. Платформа інтегрує безпечні платіжні системи для обробки транзакцій між орендарями та орендодавцями. Орендарі можуть зручно здійснювати оплату оренди через платформу, що гарантує безпеку фінансових операцій і захист коштів. Орендодавці, у свою чергу, отримують швидкі виплати на вказані рахунки;

д) система відгуків та рейтингів. Організація надає користувачам можливість залишати відгуки та оцінювати орендарів і орендодавців після завершення оренди. Це створює систему довіри, підвищує прозорість на платформі та сприяє вибору надійних партнерів для оренди житла;

е) підтримка клієнтів. Організація забезпечує клієнтську підтримку через службу підтримки. Користувачі можуть звертатися за допомогою через онлайн-чат, телефон або електронну пошту для вирішення технічних питань, спірних ситуацій або отримання додаткової інформації про процес оренди;

ж) система сповіщень. Користувачі отримують автоматичні сповіщення про важливі події, такі як підтвердження бронювання, зміна статусу оренди, нагадування про оплату та інші оновлення. Це допомагає орендарям і орендодавцям бути в курсі всіх важливих подій та своєчасно реагувати на них;

з) аналітика та звітність. Організація надає орендодавцям доступ до звітів про їхні об’єкти та фінансові операції. Це допомагає їм відстежувати свої доходи, аналізувати попит на об'єкти та оптимізувати свою діяльність. Адміністрація платформи також має доступ до аналітики для моніторингу роботи платформи та прийняття стратегічних рішень;

и) захист угод та безпека даних**.** Організація надає засоби для захисту даних користувачів і забезпечує безпеку угод. Вона гарантує конфіденційність особистих даних, захист фінансових транзакцій та впроваджує заходи для мінімізації ризиків шахрайства під час укладання угод;

2.1.4 Методи надання послуг організацією та роль інформаційної системи

а) онлайн-платформа для оренди житла;

1) організація використовує онлайн-платформу (веб-сайт та мобільний додаток), яка забезпечує доступ до каталогу житла. Орендодавці можуть розміщувати свої оголошення, а орендарі — переглядати доступні об'єкти та здійснювати бронювання;

2) інформаційна система надасть зручний та інтуїтивний інтерфейс для користувачів, який дозволить швидко шукати та фільтрувати об'єкти, переглядати їхню інформацію, обирати дати та здійснювати бронювання. Система автоматизує оновлення інформації про доступність об'єктів у режимі реального часу, що знижує ризик подвійних бронювань та забезпечує точність даних;

б) автоматизоване бронювання житла;

1) організація надає можливість орендарям здійснювати бронювання в режимі реального часу, використовуючи просту та зрозумілу форму для підтвердження дати, умов та оплати;

2) інформаційна система автоматизує процес бронювання, дозволяючи орендарям швидко здійснювати резервування житла та отримувати підтвердження у декілька кліків. Система також забезпечить перевірку доступності об'єктів у реальному часі, запобігаючи конфліктам щодо бронювання на однакові дати;

в) обробка онлайн-платежів;

1) організація використовує інтеграцію з платіжними системами для обробки онлайн-транзакцій між орендарями та орендодавцями. Оплата здійснюється безпосередньо через платформу;

2) інформаційна система забезпечить безпечну обробку платежів, використовуючи захищені платіжні шлюзи та протоколи шифрування. Вона також надасть інструменти для автоматизації обробки платежів, зменшуючи кількість ручної роботи та підвищуючи швидкість фінансових транзакцій;

г) управління об'єктами нерухомості;

1) орендодавці керують своїми об'єктами через панель управління на платформі. Вони можуть додавати нові об'єкти, оновлювати інформацію, встановлювати ціни та дати доступності;

2) система надасть зручний інтерфейс для управління об'єктами, дозволяючи орендодавцям легко редагувати інформацію, додавати фотографії та опис, а також контролювати доступність житла. Автоматизоване оновлення інформації зменшить ймовірність помилок та забезпечить своєчасне оновлення даних;

д) система відгуків та рейтингів;

1) організація дозволяє орендарям і орендодавцям залишати відгуки та оцінювати один одного після завершення оренди. Це підвищує прозорість та довіру між користувачами;

2) інформаційна система автоматизує збір відгуків, зберігає їх у базі даних та використовує для розрахунку рейтингів. Це сприяє створенню об'єктивної системи оцінок та полегшує вибір надійних партнерів для оренди;

е) система сповіщень;

1) організація використовує автоматичні сповіщення для інформування користувачів про важливі події, такі як підтвердження бронювання, оплата, зміна умов оренди або нагадування про завершення терміну оренди;

2) система автоматизує процес відправлення сповіщень через електронну пошту, SMS або push-сповіщення, що дозволяє зменшити кількість ручної роботи та підвищити ефективність комунікації. Це сприяє кращій обізнаності користувачів про актуальні події та забезпечує своєчасне реагування на важливі оновлення;

ж) аналітика та звітність;

1) організація використовує зібрані дані для аналізу ефективності роботи платформи, виявлення трендів та розробки стратегій розвитку;

2) система надає інструменти для збору, зберігання та аналізу даних про активність користувачів, фінансові операції та відгуки. Це дозволяє адміністрації платформи приймати стратегічні рішення на основі реальних даних та оптимізувати роботу бізнесу.

2.1.5 Опис стейкхолдерів бізнесу і як система буде сприяти їх цілям

Основними стейкхолдерами цієї системи є:

а) орендодавці**:** приватні власники або компанії, що надають житло в оренду. Система допоможе їм автоматизувати процеси управління житлом, отримувати сповіщення про бронювання і здійснювати облік доходів;

б) орендарі: особи, які шукають житло для коротко- або довгострокової оренди. Система дозволить їм швидко знаходити житло, перевіряти його доступність, переглядати умови оренди та здійснювати бронювання й оплату;

в) адміністратори платформи: особи, що відповідають за управління платформою, її технічне обслуговування та підтримку. Для них система забезпечить зручне управління даними та контроль над усіма процесами оренди та оплати;

г) служба підтримки: група, що обробляє запити користувачів, вирішує спірні питання та надає допомогу з будь-якими технічними проблемами. Система дозволить їм швидко отримувати інформацію про об'єкти, бронювання і транзакції для оперативного обслуговування клієнтів.

2.2 Формування вимог до системи

Формування вимог до системи охоплює визначення функціональності, такої як реєстрація користувачів, пошук, бронювання та оплата, а також нефункціональних критеріїв, включаючи масштабованість, продуктивність і безпеку даних.

2.2.1 Функціональні вимоги

Функціональні вимоги описують конкретні можливості та функції, які система повинна реалізувати для виконання своїх основних завдань. Для системи оренди житла на мікросервісній архітектурі до основних функціональних вимог належать:

а) управління користувачами;

1) система повинна підтримувати реєстрацію нових користувачів з підтвердженням електронної пошти або телефону;

2) реалізація аутентифікації користувачів;

3) підтримка ролей користувачів: орендарі, орендодавці та адміністратори платформи, з можливістю обмеження доступу до певних функцій залежно від ролі;

4) можливість редагування профілю, зміни особистих даних та паролів користувачів;

б) управління об'єктами оренди**;**

1) орендодавці повинні мати можливість додавати нові об'єкти до каталогу, вказуючи детальний опис, фото, ціну та умови оренди;

2) функції редагування та видалення об'єктів оренди повинні бути доступні лише для їхніх власників;

3) система повинна дозволяти перегляд об'єктів оренди з використанням фільтрів (місцезнаходження, ціна, кількість кімнат, зручності тощо);

в) пошук та фільтрація об'єктів**;**

1) система повинна підтримувати розширений пошук об'єктів за різними критеріями, такими як місце розташування, ціна, тип житла, доступність на певні дати;

2) можливість сортування результатів за ціною, рейтингом, популярністю або іншими параметрами;

г) бронювання житла**;**

1) користувачі повинні мати можливість бронювати доступні об'єкти, обираючи дати, які їх цікавлять;

2) система повинна перевіряти доступність об'єктів на задані дати та запобігати подвійним бронюванням;

3) система повинна забезпечити автоматичне підтвердження бронювання та надсилати повідомлення орендарю та орендодавцю;

д) платіжна система**;**

1) інтеграція з платіжними сервісами для обробки онлайн-платежів між орендарями та орендодавцями;

2) можливість оплати оренди через різні платіжні системи та методи;

3) забезпечення захисту транзакцій за допомогою протоколів шифрування;

е) система відгуків та рейтингів**;**

1) після завершення оренди користувачі повинні мати можливість залишати відгуки та оцінювати об'єкти, що впливатиме на їхній рейтинг;

2) орендодавці також можуть оцінювати орендарів, що допоможе створити систему довіри на платформі;

ж) система сповіщень**;**

1) автоматичні сповіщення орендарям та орендодавцям про важливі події, такі як підтвердження бронювання, зміна статусу об'єкта, нагадування про оплату;

2) підтримка відправлення сповіщень через електронну пошту, SMS та push-сповіщення;

з) адміністративна панель**;**

1) адміністратори повинні мати доступ до управління користувачами, перегляду інформації про об'єкти, модерації відгуків та управління фінансовими транзакціями;

2) інструменти моніторингу активності на платформі та доступ до звітів з використання функцій.

2.2.2 Нефункціональні вимоги

Нефункціональні вимоги визначають якісні характеристики системи та описують, як система повинна виконувати функції для забезпечення стабільності, надійності та безпеки:

а) продуктивність;

1) система повинна забезпечувати швидкий відгук на користувацькі запити. Час завантаження сторінок не повинен перевищувати 2 секунд;

2) платформа повинна бути здатна обробляти велику кількість одночасних запитів без зниження продуктивності;

б) масштабованість;

1) система повинна бути здатна до масштабування, щоб підтримувати збільшення кількості користувачів та об'єктів оренди;

2) масштабування має забезпечуватися за рахунок додавання нових серверів або мікросервісів у відповідь на зростання навантаження;

в) безпека;

1) використання протоколів шифрування для захисту даних користувачів при передачі між сервісами;

2) захист від несанкціонованого доступу за допомогою надійних методів аутентифікації;

3) забезпечення резервного копіювання даних для уникнення втрати інформації;

г) доступність;

1) платформа повинна мати високу доступність, забезпечуючи безперервний доступ до послуг у будь-який час доби. Рівень доступності повинен бути не менше 99.9% на рік;

2) система повинна бути доступною з будь-якого пристрою (десктоп, смартфон, планшет);

д) зручність використання

1) інтерфейс системи повинен бути інтуїтивно зрозумілим для користувачів з різними рівнями технічної підготовки;

2) зручна навігація, чітка структура даних та швидкий доступ до основних функцій платформи;

е) надійність;

1) система повинна бути стійкою до збоїв і мати можливість автоматичного відновлення роботи у випадку технічних проблем;

2) гарантія стабільної роботи системи навіть при великих навантаженнях або пікових періодах;

ж) сумісність;

1) платформа повинна підтримувати інтеграцію з зовнішніми сервісами, такими як платіжні системи, служби аутентифікації та інші API;

2) система повинна бути сумісною з різними веб-браузерами та операційними системами;

2.3 Аналіз обмежень

Аналіз обмежень є важливим етапом проектування інформаційної системи, оскільки дозволяє визначити фактори, які можуть впливати на розробку, впровадження та функціонування системи. У контексті системи оренди житла на основі мікросервісної архітектури були виділені наступні ключові обмеження.

Технологічні обмеження**:**

а) продуктивність системи**;**

1) необхідність забезпечення швидкого відгуку на користувацькі запити (максимальний час завантаження сторінок — 2 секунди);

2) обмеження обчислювальної потужності при обробці великої кількості запитів у пікові години;

б) хмарна інфраструктура;

1) використання Azure накладає обмеження на вибір інструментів і сервісів, які сумісні з платформою;

2) залежність від продуктивності хмарних сервісів та мережевої інфраструктури;

в) сумісність з пристроями;

1) система повинна бути доступною на різних платформах (десктоп, мобільні пристрої), що ускладнює розробку інтерфейсу.

Обмеження безпеки:

а) захист персональних даних**,** необхідність дотримання стандартів захисту даних, таких як GDPR, для забезпечення конфіденційності інформації користувачів.

б) Шифрування**,** використання SSL/TLS для шифрування передачі даних додає навантаження на сервери, що може впливати на продуктивність;

в) аутентифікація та авторизація: Реалізація безпечного зберігання токенів (JWT) та захисту від атак типу CSRF або XSS.

Обмеження бізнес-процесів:

а) різноманітність користувачів**.** Платформа орієнтована на різні типи користувачів (орендарі, орендодавці, адміністратори), що вимагає адаптації функціональності під кожну категорію;

б) висока конкуренція. Вимога до швидкої інтеграції нових функцій для підтримки конкурентоспроможності.

Організаційні обмеження:

а) обмеження бюджету: Необхідність оптимізації витрат на інфраструктуру, розробку та підтримку платформи.;

б) кваліфікація персоналу: Команда розробників повинна мати достатню кваліфікацію для роботи з мікросервісною архітектурою та хмарними сервісами Azure;

в) часові рамки: Стислі терміни розробки, впровадження та тестування системи.

2.4 Аналіз ризиків

Аналіз ризиків є ключовим етапом у розробці інформаційної системи, оскільки дозволяє ідентифікувати можливі загрози, оцінити їхній вплив і розробити заходи щодо їх мінімізації. У контексті системи оренди житла на основі мікросервісної архітектури були виділені наступні ризики.

Основні ризики для системи оренди житла можна класифікувати за категоріями:

а) технічні ризики;

1) збої у взаємодії мікросервісів через помилки в API або мережеві проблеми;

2) низька продуктивність під час високого навантаження (наприклад, пікові години використання);

3) втрата даних через помилки в базі даних або ненадійне резервне копіювання;

б) ризики безпеки**;**

1) можливість витоку персональних даних користувачів через недостатньо захищені канали зв’язку;

2) атаки типу DDoS, які можуть вивести з ладу окремі мікросервіси або всю систему;

3) незахищеність аутентифікації, що може призвести до несанкціонованого доступу;

в) організаційні ризики;

1) низька кваліфікація розробників щодо роботи з мікросервісами та хмарними платформами;

2) недостатнє фінансування, що може вплинути на якість інфраструктури або підтримки системи;

г) бізнес-ризики;

1) невідповідність системи очікуванням користувачів через недоліки в інтерфейсі чи функціоналі;

2) непередбачені затримки в розробці, що можуть призвести до втрати конкурентоспроможності.

Для оцінки ризиків використовується матриця ризиків [5]. Наведена в таблиці 2.1, яка враховує їхню імовірність (High, Medium, Low) та вплив (High, Medium, Low).

Таблиця 2.1 – Матриця оцінки ризиків

Ризик	Імовірність	Вплив	Рівень ризику
Збої у взаємодії мікросервісів	Medium	High	High
Атаки на систему (DDoS, XSS)	High	High	High
Витік персональних даних	Medium	High	High
Перевантаження бази даних	Medium	Medium	Medium
Низька кваліфікація персоналу	Low	High	Medium
Недостатнє фінансування	Low	Medium	Low

Стратегії зниження ризиків:

а) технічні заходи**;**

1) використання балансувальників навантаження для забезпечення стабільної роботи системи;

2) регулярне резервне копіювання даних з перевіркою цілісності резервних копій;

3) ретельне тестування API мікросервісів перед впровадженням;

б) безпека**;**

1) впровадження SSL/TLS для шифрування переданих даних;

2) реалізація захисту від атак DDoS через використання Azure DDoS Protection;

3) застосування багаторівневої аутентифікації користувачів (наприклад, MFA);

в) організаційні заходи;

1) навчання розробників щодо роботи з мікросервісами та хмарними платформами;

2) регулярні технічні консультації та аудит безпеки;

3) забезпечення фінансових резервів для непередбачених витрат;

г) користувацький досвід;

1) тестування інтерфейсу користувачів з потенційними клієнтами;

2) впровадження системи збору зворотного зв’язку від користувачів для швидкого виявлення недоліків.

2.5 Опис бізнес-процесів користувачів

У цьому підрозділі розглядаються ключові бізнес-процеси, які відбуваються в системі оренди житла, побудованій на мікросервісній архітектурі. Бізнес-процеси описують послідовність операцій, які виконують користувачі та система для досягнення бажаного результату. Вони включають дії, що здійснюються орендарями, орендодавцями та адміністраторами на платформі. Оскільки система оренди житла є інтегрованою, кожен з мікросервісів підтримує певні етапи бізнес-процесів, забезпечуючи взаємодію між усіма сторонами. Схеми бізнес процесів описані з використанням мови моделювання архітектури підприємств Archimate 3.2. [6].

2.5.1 Орендар

Орендарі є одними з основних користувачів системи. Їхні основні бізнес-процеси включають:

а) реєстрація та авторизація. Орендар починає свій шлях у системі з реєстрації, надаючи основні особисті дані. Після цього користувач може авторизуватися для доступу до особистого кабінету, де він має можливість керувати своїми бронюваннями та профілем що зображено на рисунку 2.1.

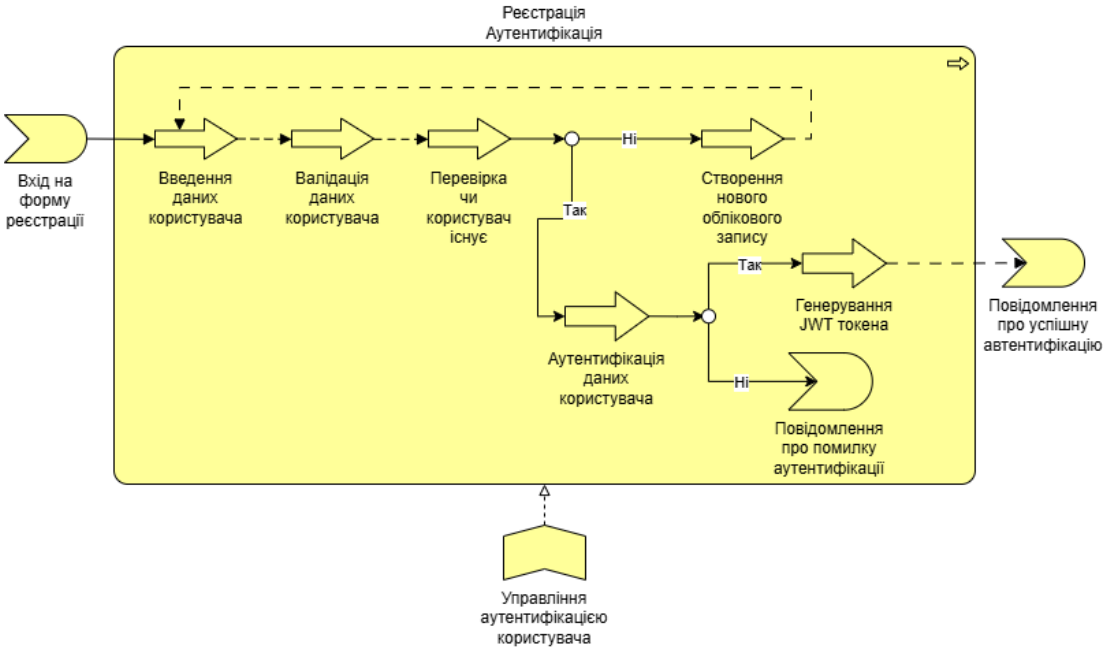

Рисунок 2.1 – Схема бізнес-процесу реєстрації/аутентифікації

Опис бізнес-процесу Реєстрації/Аутентифікації;

1) вхід на форму реєстрації, користувач заходить на форму реєстрації;

2) введення даних користувача, користувач вводить свої дані, такі як ім'я, email, пароль тощо;

3) валідація даних користувача, система перевіряє правильність введених даних (наприклад, формат email);

4) перевірка, чи користувач існує, система перевіряє, чи існує користувач з таким email. Якщо користувач існує, відбувається

аутентифікація користувача. Якщо користувач не існує, система переходить до етапу створення нового облікового запису;

5) створення нового облікового запису, якщо користувач не існує, система створює новий обліковий запис користувача;

6) аутентифікація даних користувача, якщо користувач існує, система перевіряє правильність його даних (наприклад, пароль);

7) генерація JWT токена, після успішної аутентифікації генерується JWT токен, який дозволяє користувачу отримати доступ до системи;

8) повідомлення про успішну аутентифікацію, користувач отримує повідомлення про успішну аутентифікацію.

9) повідомлення про помилку аутентифікації, якщо дані неправильні, користувач отримує повідомлення про помилку.

б) Пошук квартири: Орендар використовує функціонал пошуку на платформі для знаходження доступних квартир. Пошук здійснюється за різними критеріями: ціна, місцезнаходження, кількість кімнат, зручності тощо. Після вибору квартири користувач може переглянути деталі оголошення на рисунку 2.2.

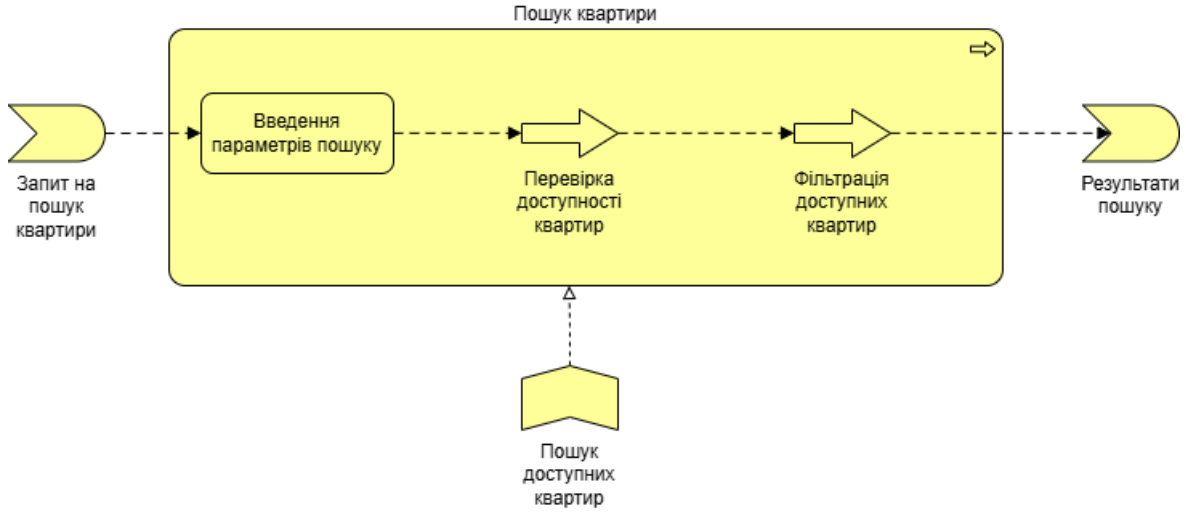

Рисунок 2.2 – Схема бізнес-процесу пошуку квартири

Опис бізнес-процесу "Пошук квартири";

1) запит на пошук квартири: Користувач ініціює процес пошуку, активуючи відповідну функцію через інтерфейс (наприклад, кнопка пошуку або відображення форми для введення параметрів). Це є початковою подією в процесі, яка ініціює подальші етапи пошуку;

2) введення параметрів пошуку: Користувач вводить параметри для пошуку, такі як локація, ціна, кількість кімнат, тип квартири та інші фільтри. Цей етап є критичним для визначення вимог до квартир, які користувач хоче знайти;

3) перевірка доступності квартир: Система перевіряє, чи є в базі даних доступні квартири, що відповідають введеним параметрам пошуку. Цей етап включає взаємодію з базою даних або іншими джерелами для перевірки наявності квартир;

4) фільтрація доступних квартир: Після перевірки доступності система фільтрує варіанти квартир за додатковими критеріями, такими як ціна, локація, площа, наявність додаткових зручностей і т. д. Це дозволяє звузити пошук і відобразити тільки ті варіанти, які найбільше відповідають вимогам користувача;

5) пошук доступних квартир: Система проводить сам пошук, використовуючи введені параметри та застосовуючи фільтри для отримання актуальних варіантів квартир;

6) результати пошуку: Система виводить список доступних квартир, що відповідають заданим критеріям. Користувач бачить перелік варіантів, який може переглядати, порівнювати та обирати найкращі квартири. Це є кінцевим результатом процесу пошуку, який завершився виведенням результатів для користувача;

в) перегляд деталей квартири: Орендар вибирає конкретну квартиру зі списку доступних варіантів, щоб переглянути деталі оголошення. Деталі квартири відображаються на платформі, включаючи фотографії, опис, ціну, площу,

розташування та інші характеристики. Після перегляду інформації користувач може оцінити варіант квартири, додати її до списку обраних або продовжити пошук інших варіантів, що зображено на рисунку 2.3 .

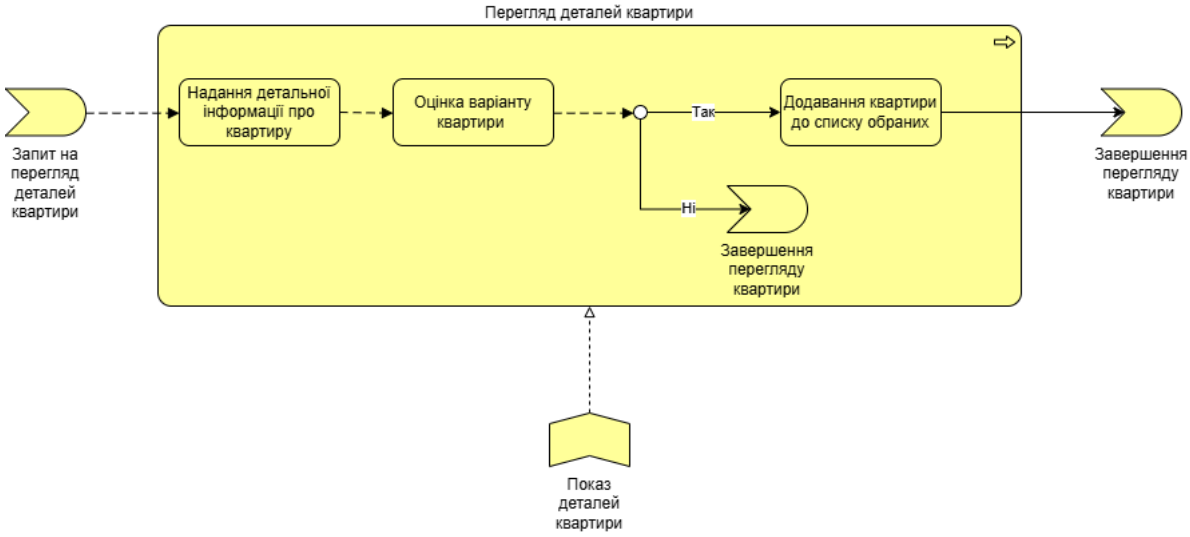

Рисунок 2.3 – Схема бізнес-процесу перегляд деталей квартири

1) запит на перегляд деталей квартири: Користувач ініціює перегляд конкретної квартири, вибираючи її зі списку результатів пошуку або натискаючи на конкретну квартиру, щоб побачити більше деталей. Це є початковою подією, що ініціює процес перегляду квартири;

2) надання детальної інформації про квартиру: Система відображає всю необхідну інформацію про квартиру, включаючи опис, фото, ціну, площу, кількість кімнат, розташування, додаткові зручності тощо. Після запиту на перегляд, система надає повну інформацію про квартиру;

3) оцінка варіанту квартири: Користувач оцінює квартиру на основі наданої інформації. Це може включати порівняння з іншими варіантами квартир та розгляд переваг або недоліків. Користувач приймає рішення, чи підходить йому ця квартира для подальшого перегляду або бронювання;

4) додавання квартири до списку обраних: Якщо квартира підходить, користувач може додати її до списку обраних для подальшого

перегляду або бронювання. Цей етап дозволяє користувачу зберігати варіанти для подальших дій;

5) завершення перегляду квартири. Після того, як користувач переглянув всі деталі квартири або додав її до списку обраних, процес перегляду завершується. Це є кінцевою подією, що сигналізує про завершення процесу перегляду, і користувач може перейти до інших варіантів або дій (наприклад, бронювання);

г) бронювання квартири. Після вибору бажаного об'єкта оренди, орендар здійснює бронювання на платформі. Система перевіряє доступність об'єкта на потрібні дати, автоматично підтверджує бронювання або пропонує альтернативи, що зображено на рисунку 2.4.

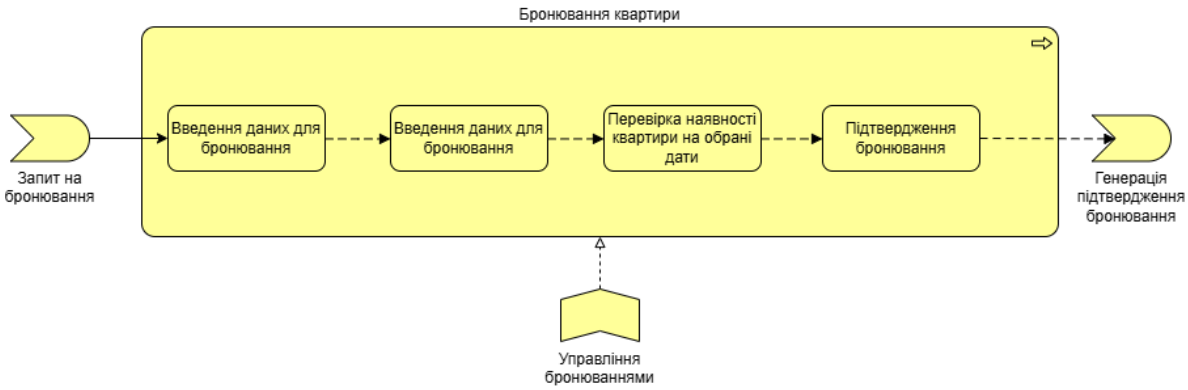

Рисунок 2.4 – Схема бізнес-процесу бронювання

Опис бізнес-процесу "Бронювання квартири":

1) запит на бронювання: Користувач ініціює процес бронювання квартири через інтерфейс (веб-сайт, додаток), натискаючи на кнопку або вибираючи функцію для бронювання. Це перша подія в процесі, яка сигналізує про намір користувача забронювати квартиру;

2) введення даних для бронювання: Користувач вводить необхідні дані для бронювання, такі як дати заїзду та виїзду, кількість осіб, тощо;

3) перевірка наявності квартири на обрані дати: Система перевіряє наявність квартири на зазначені користувачем дати, щоб переконатися, що

квартира доступна для бронювання в цей період. Система взаємодіє з базою даних, щоб перевірити доступність квартири на вибрані дати;

4) підтвердження бронювання: Якщо квартира доступна на обрані дати, система підтверджує бронювання і інформує користувача про успішне бронювання. Цей етап завершується підтвердженням бронювання, що дає користувачу відчуття безпеки і гарантії щодо забронюваної квартири;

5) генерація підтвердження бронювання: Після успішного підтвердження бронювання, система генерує документ або електронне підтвердження, яке користувач може зберегти для подальших дій (наприклад, для поїздки). Це кінцева подія бізнес-процесу, яка сигналізує, що процес бронювання завершено успішно;

д) оплата оренди: Після підтвердження бронювання, орендар здійснює платіж через інтегровану платіжну систему. Оплата може бути здійснена за допомогою різних платіжних методів (банківські картки, електронні гаманці тощо) як зображено на рисунку 2.5.

Рисунок 2.5 – Схема бізнес-процесу оплати

Опис бізнес-процесу "Оплата оренди"**:**

1) запит на оплату оренди: Користувач ініціює оплату оренди після підтвердження бронювання квартири. Це може бути зроблено через інтерфейс платіжної системи на платформі. Це початкова подія в процесі

оплати, яка сигналізує про намір користувача здійснити платіж за оренду квартири;

2) введення платіжних даних: Користувач вводить свої платіжні дані, такі як номер картки, дата закінчення терміну дії, CVV код, або обирає інший метод оплати, наприклад, через електронний гаманець або банківський переказ. Це етап, на якому користувач надає дані для оплати, що є необхідними для здійснення фінансової операції;

3) обробка платежу: Система перевіряє платіжні дані та обробляє платіж через платіжний шлюз. Це може включати перевірку валідності картки, наявності достатніх коштів на рахунку та обробку самого платіжного запиту. Система взаємодіє з платіжною системою для перевірки та обробки запиту на оплату оренди;

4) підтвердження успішної оплати: Після успішної обробки платежу система підтверджує користувачу, що платіж був успішно проведений. Це може бути повідомлення через інтерфейс платформи, на e-mail, або через SMS;

5) генерація квитанції: Після підтвердження успішної оплати система генерує електронну квитанцію або рахунок-фактуру, який містить інформацію про суму, дату платежу та інші деталі;

6) завершення процесу оплати: Після генерування квитанції процес оплати оренди завершено. Користувач отримує доступ до підтвердження і може перейти до інших етапів (наприклад, отримати доступ до квартири або продовжити взаємодію з платформою). Це кінцева подія процесу оплати, яка сигналізує про успішне завершення бізнес-процесу;

е) Завершення оренди та відгук: Після завершення оренди, орендар залишає відгук про квартиру та орендодавця, що допомагає створити довіру до платформи та підвищити прозорість процесу для майбутніх користувачів як зображено на рисунку 2.6.

Рисунок 2.6 – Схема бізнес-процесу завершення оренди

Опис бізнес-процесу "Завершення оренди і відгук":

1) запит на завершення оренди. Користувач ініціює процес завершення оренди після того, як термін оренди закінчився, або квартира повертається після використання. Це перша подія в процесі, яка сигналізує про завершення оренди;

2) перевірка стану квартири. Система або власник квартири перевіряють стан квартири після закінчення оренди. Це може включати перевірку пошкоджень або виконання інших умов оренди (наприклад, повернення ключів, чистота). Це етап, де перевіряється фізичний стан квартири після використання;

3) оцінка апартаментів користувачем. Користувач оцінює квартиру після оренди, заповнюючи відгук про стан квартири, надані зручності, взаємодію з власником та інші аспекти оренди. Користувач надає зворотний зв'язок, що дозволяє власнику квартири або платформі покращити якість послуг;

4) додавання відгуку: Користувач додає свій відгук до системи після оцінки квартири. Це може включати рейтинг квартири та опис досвіду. Відгук додається в систему, де його можуть переглядати інші користувачі або адміністратори для подальшої обробки;

5) генерація звіту для адміністратора: Після завершення процесу користувач отримує підтвердження про додавання відгуку, а система генерує

звіт для адміністратора з детальною інформацією про оренду та відгуки. Цей етап завершує процес, генеруючи звіт для подальшого аналізу та обробки адміністратором платформи;

6) завершення процесу оренди: Після створення звіту для адміністратора, процес завершення оренди завершується. Користувач може повернутися до основної платформи або виконати інші дії. Це фінальна подія, що сигналізує про успішне завершення всіх етапів процесу оренди.

Трьохрівневу схему [7] бізнес-процесів орендаря можна переглянути в додатку Е.

2.5.2 Орендодавець

Орендодавець є важливим стейкхолдером системи оренди житла, який відповідає за надання та управління житловими об'єктами для оренди. Бізнес-процеси орендодавця передбачають виконання ряду дій на платформі для розміщення, редагування і контролю оголошень, а також для взаємодії з орендарями.

Основні етапи бізнес-процесу орендодавця:

а) реєстрація та авторизація: Орендодавець реєструється на платформі, надаючи особисті дані для створення облікового запису. Після реєстрації, орендодавець авторизується для доступу до системи, де може керувати своїми об'єктами. Процес реєстрації\авторизації аноалогічний бізнес-процесу наведеному на рисунку 2.1 ;

б) Додавання квартири: Орендодавець додає свої квартири до системи, вказуючи важливі деталі: ціна, опис, фотографії, умови оренди, доступність. Платформа перевіряє надані дані і публікує оголошення після модерації як зображено на рисунку 2.7.

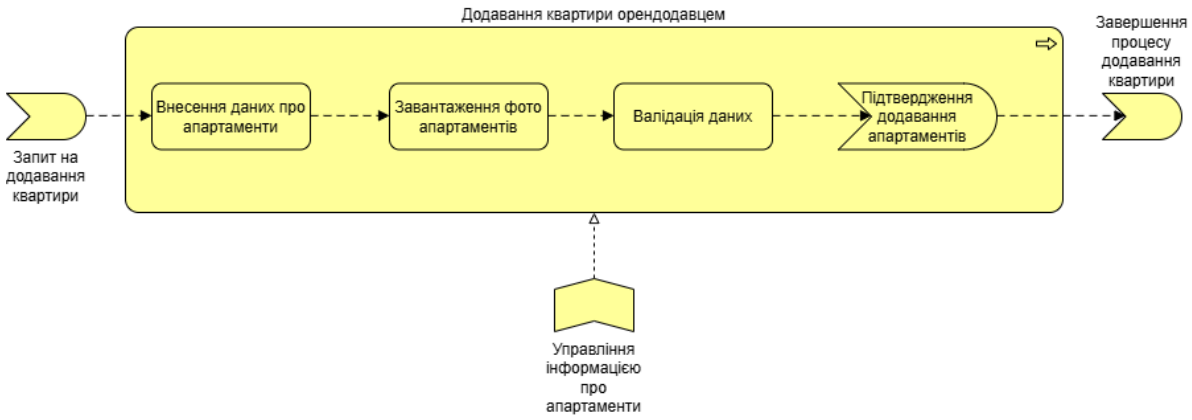

Рисунок 2.7 – Схема бізнес-процесу додавання апартаментів в систему

Опис бізнес-процесу "Додавання квартири орендодавцем":

1) запит на додавання квартири. Орендодавець ініціює процес додавання квартири на платформу для оренди. Цей крок є початковою подією процесу. Це подія, яка сигналізує про початок процесу додавання нової квартири;

2) введення даних про апартаменти. Орендодавець вводить основну інформацію про квартиру, таку як адреса, кількість кімнат, площа, ціна оренди, тип квартири та інші характеристики. Це етап, на якому орендодавець надає всю необхідну інформацію про квартиру для її публікації на платформі;

3) завантаження фото апартаментів. Орендодавець завантажує фотографії квартири, щоб забезпечити візуальне представлення об'єкта на платформі. Це етап, коли орендодавець додає зображення для створення повного оголошення про квартиру;

4) валідація даних: Система перевіряє введені орендодавцем дані на коректність і відповідність вимогам платформи (наприклад, перевірка формату ціни, наявності фотографій, правильно введеної адреси). Це етап перевірки, щоб переконатися, що введена інформація правильна і відповідає вимогам платформи;

5) підтвердження додавання апартаментів: Якщо всі дані коректні, система підтверджує успішне додавання квартири на платформу. Квартира стає доступною для перегляду іншими користувачами. Це подія, яка сигналізує про успішне додавання квартири в систему, і квартира стає доступною для інших користувачів платформи;

6) завершення процесу додавання квартири: Процес додавання квартири завершено, орендодавець може продовжити додавати інші квартири або управляти своїми об'єктами. Це кінцева подія, що позначає завершення процесу додавання квартири на платформу;

в) Управління оголошеннями**:** Орендодавець може редагувати оголошення, змінювати ціни, додаючи нові фотографії чи змінюючи умови оренди. Платформа дозволяє орендодавцю оновлювати інформацію в реальному часі як показано на рисунку 2.8.

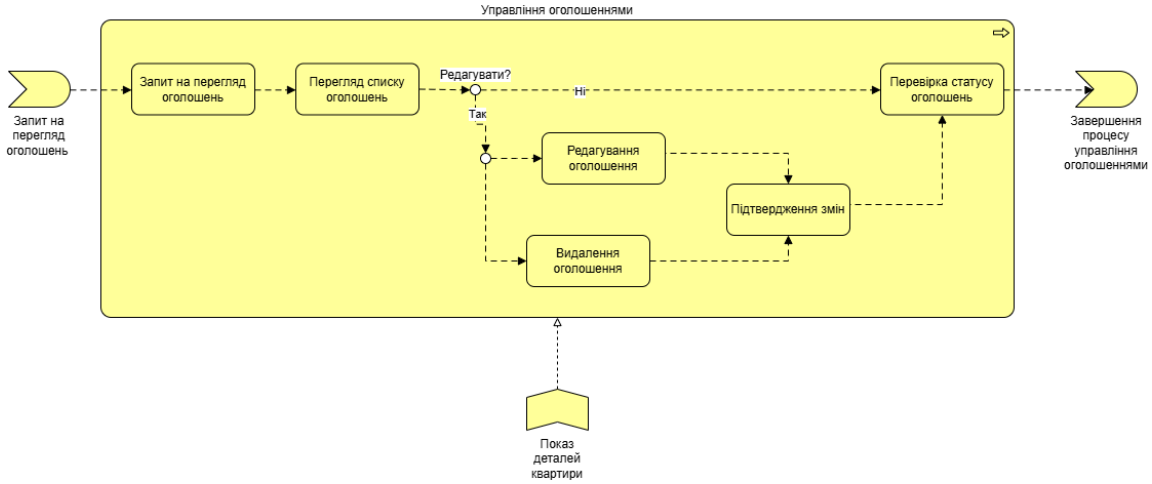

Рисунок 2.8 – Схема бізнес-процесу управління оголошеннями

Опис бізнес-процесу "Управління оголошеннями"**:**

1) запит на перегляд оголошень: Орендодавець ініціює процес перегляду своїх оголошень на платформі для того, щоб переглянути список своїх об'єктів (квартир);

2) перегляд списку оголошень: Орендодавець переглядає список своїх оголошень, включаючи їх статуси, ціни, доступність та інші деталі. Це

етап, на якому орендодавець бачить загальний стан своїх оголошень і може вибрати, яке з них потрібно редагувати або видалити;

3) дії над оголошеннями: Орендодавець вирішує, чи потрібно редагувати оголошення або ж залишити його без змін. Користувач може або редагувати наявні оголошення, або видалити не актуальні, або лишити все як є;

4) редагування оголошення: Орендодавець вносить зміни до оголошення, наприклад, редагує ціну, опис квартири, завантажує нові фотографії або оновлює інші деталі;

5) видалення оголошення: Якщо орендодавець вирішує, що оголошення більше не актуальне, він може вибрати варіант видалення цього оголошення з платформи;

6) підтвердження змін: Після редагування або видалення оголошення орендодавець підтверджує зміни, і система оновлює або видаляє оголошення на платформі;

7) перевірка статусу оголошення: Орендодавець перевіряє статус своїх оголошень, щоб оцінити, чи є бронювання, активні оголошення чи чи потрібно оновити деякі дані;

8) завершення процесу управління оголошеннями: Після завершення всіх необхідних дій орендодавець завершує процес управління оголошеннями і може повернутися до інших функцій платформи;

г) управління бронюваннями. Орендодавець отримує сповіщення про нові запити на бронювання. Він може підтвердити або відхилити запит на бронювання залежно від доступності квартири. Система автоматично оновлює статус доступності квартир, як наведено на рисунку 2.9.

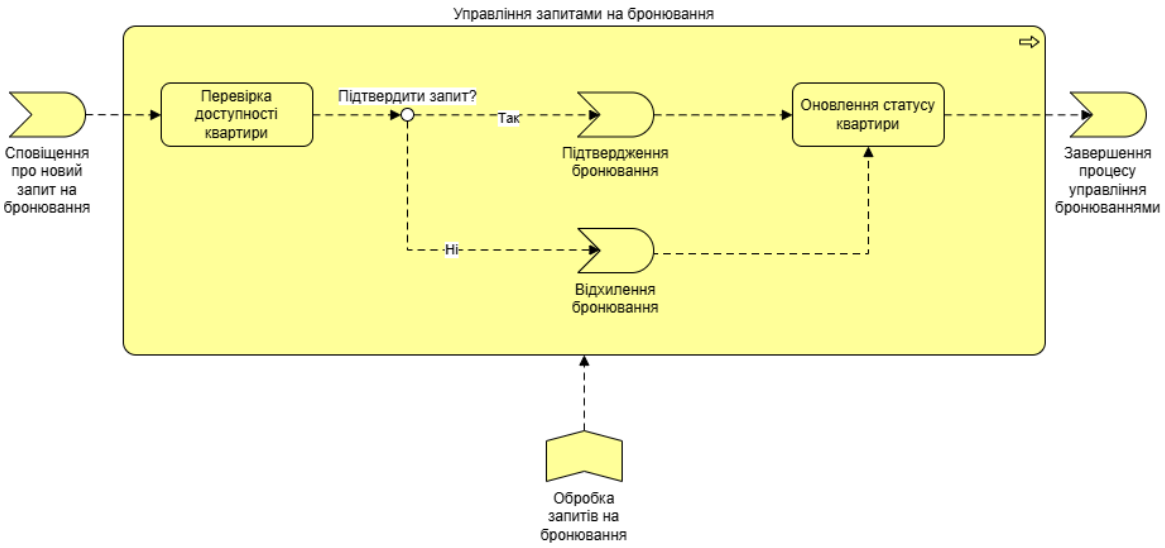

Рисунок 2.9 – Схема бізнес-процесу управління запитами на бронювання

Опис бізнес-процесу "Управління запитами на бронювання":

1) сповіщення про новий запит на бронювання.Орендодавець отримує сповіщення про новий запит на бронювання від орендаря для певної квартири;

2) перевірка доступності квартири. Орендодавець перевіряє, чи є квартира доступною на запитувані дати для бронювання;

3) підтвердження запиту: Орендодавець приймає рішення, чи підтверджувати запит на бронювання, якщо квартира доступна, орендодавець підтверджує запит на бронювання, або якщо квартира недоступна, орендодавець відхиляє запит на бронювання;

4) підтвердження бронювання. Якщо орендодавець підтверджує бронювання, система оновлює статус бронювання і сповіщає орендаря про успішне бронювання;

5) відхилення бронювання. Якщо орендодавець відхиляє запит, система сповіщає орендаря, що бронювання не підтверджене через відсутність доступності;

6) оновлення статусу квартири. Після підтвердження або відхилення запиту на бронювання система автоматично оновлює статус доступності квартири в базі даних;

7) завершення процесу управління бронюваннями. Після оновлення статусу доступності квартири процес управління бронюваннями завершується;

д) отримання платежів. Орендодавець отримує повідомлення про здійснення платежу орендарем. Платіж через систему обробляється через інтеграцію з платіжними системами, і орендодавець отримує кошти на свій рахунок як наведено на рисунку 2.10;

Рисунок 2.10 – Схема бізнес-процесу отримання платежів

Опис бізнес-процесу "Отримання платежів":

1) повідомлення про здійснення платежу. Орендодавець отримує сповіщення про те, що орендар здійснив платіж за оренду через платформу;

2) перевірка інформації про оплату. Система перевіряє деталі платіжної транзакції, щоб переконатися, що сума правильна, платіж пройшов успішно і не було помилок у даних;

3) обробка платежу. Система обробляє платіж за допомогою платіжної системи або шлюзу, для підтвердження того, що кошти насправді були отримані;

4) сповіщення про успішну транзакцію. Після успішної обробки платежу система сповіщає орендодавця про успішну транзакцію;

5) переведення коштів орендодавцю. Після підтвердження успішного платежу система переведе кошти на рахунок орендодавця, завершивши фінансову транзакцію;

6) завершення процесу отримання платежу. Після того як кошти переведені на рахунок орендодавця, процес отримання платежу завершено;

2.5.3 Адміністратор

Адміністратор є важливою роллю в системі оренди житла, відповідаючи за управління всіма аспектами платформи, включаючи моніторинг користувачів, оголошень, транзакцій та забезпечення правил використання. Бізнес-процеси адміністратора включають управління акаунтами користувачів, модерацію контенту, обробку фінансових операцій та аналіз системи.

До бізнес-процесів адміністратора входять:

а) аутентифікація. Адміністратор проходить через процес реєстрації\аутентифікації наведений на Рисунку 2.1 аналогічний орендарю та орендодавцю;

б) управління користувачами. Адміністратор керує користувачами системи, включаючи орендодавців та орендарів, включає реєстрацію, редагування даних, блокування або видалення користувачів як зображено на рисунку 2.11.

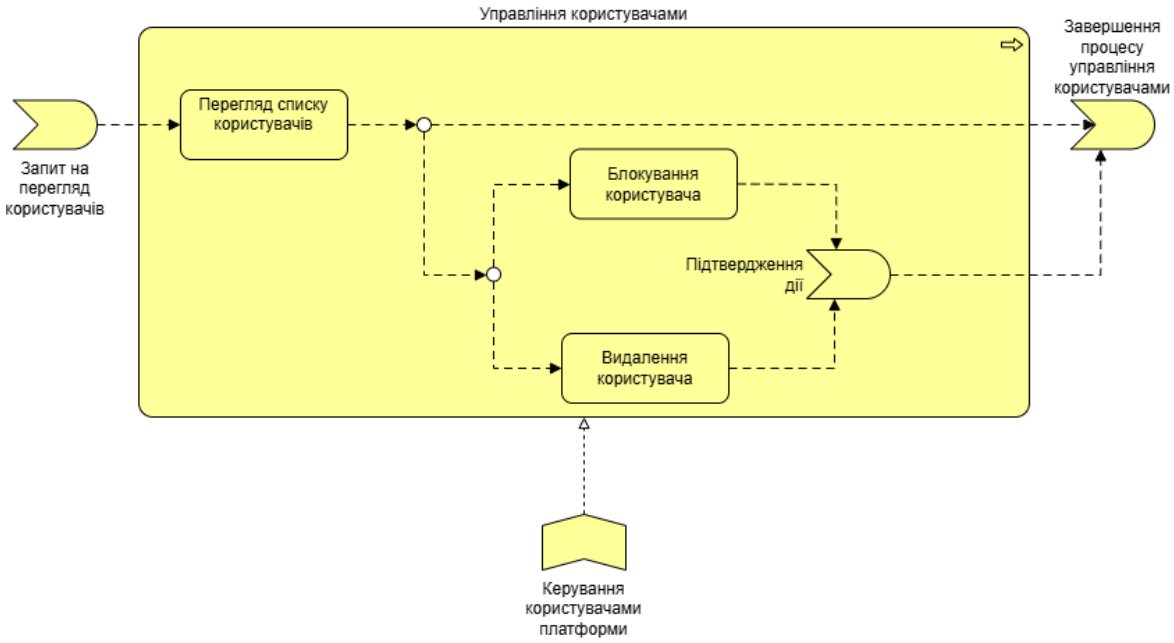

Рисунок 2.11 – Схема бізнес-процесу управління користувачами

Опис бізнес-процесу "Управління користувачами":

1) запит на перегляд користувачів: Адміністратор ініціює процес перегляду списку користувачів на платформі, щоб побачити інформацію про всіх зареєстрованих користувачів;

2) перегляд списку користувачів: Адміністратор переглядає всі зареєстровані облікові записи користувачів, включаючи їх статуси (активний, заблокований тощо);

3) дії над користувачами: Адміністратор приймає рішення щодо того, чи потрібно блокувати або видаляти користувача.Якщо користувач порушує правила або має неприпустиму активність, адміністратор блокує його. Якщо користувач більше не є активним або потребує видалення, адміністратор видаляє його з платформи;

4) блокування користувача: Якщо адміністратор вирішує заблокувати користувача, система змінює статус користувача на "заблокований", що обмежує його доступ до платформи;

5) видалення користувача: Якщо адміністратор вирішує видалити користувача, всі його дані видаляються з платформи, і обліковий запис стає недоступним;

6) підтвердження дії: Після блокування або видалення користувача система підтверджує, що операція була успішно виконана;

7) завершення процесу управління користувачами: Після виконання всіх дій з користувачами (перегляд, блокування, видалення) процес завершено;

в) управління оголошеннями: Адміністратор має можливість переглядати всі оголошення на платформі та приймати рішення про публікацію, редагування або видалення оголошень як на рисунку 2.12.

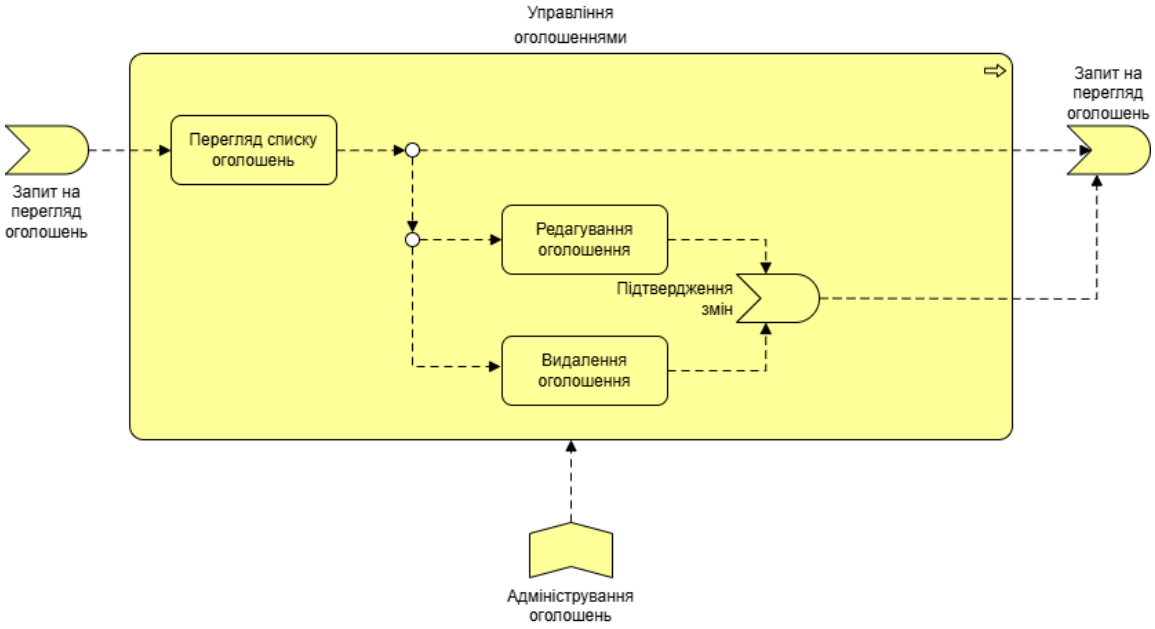

Рисунок 2.12 – Схема бізнес-процесу управління оголошеннями

Опис бізнес-процесу "Управління оголошеннями":

1) запит на перегляд оголошень: Орендодавець або адміністратор ініціює процес перегляду списку оголошень на платформі;

2) перегляд списку оголошень: Орендодавець або адміністратор переглядає список оголошень, включаючи деталі, фото, опис, ціну та статус доступності;

3) дії над оголошеннями: Після перегляду оголошень адміністратор може або редагувати оголошення якщо потрібно змінити якусь

інформацію, або видалити якщо публікація втратила актуальність чи не відповідає нормам публікації на платформі;

4) редагування оголошення: Адміністратор редагує оголошення, змінюючи його ціну, опис, фото чи інші характеристики за необхідності.

5) видалення оголошення: Якщо оголошення більше не актуальне, адміністратор може видалити його з платформи;

г) управління платежами: Адміністратор відповідає за моніторинг платіжних транзакцій і може втручатися у випадку проблем із платіжними операціями я на рисунку 2.13.

Рисунок 2.13 – Схема бізнес-процесу управління платежами

Опис бізнес-процесу "Управління платежами":

1) повідомлення про здійснення платежу: Орендодавець отримує повідомлення від платіжної системи або платформи про те, що орендар здійснив платіж за оренду. Це може включати суму платежу, ідентифікатор транзакції та іншу інформацію;

2) перевірка платіжної інформації: Адміністратор або система перевіряє деталі платіжної транзакції, щоб переконатися, що платіж був успішним, сума правильна, а платіжні реквізити відповідають вимогам;

3) обробка платежу: Платіж обробляється через платіжний шлюз або платіжну систему, яка підтверджує успішну транзакцію та ініціює переведення коштів на рахунок орендодавця;

4) сповіщення про успішний платіж: Після успішної обробки транзакції система або платіжна система надсилає сповіщення орендодавцю про те, що платіж був успішно здійснений, і кошти будуть зараховані на його рахунок;

5) переведення коштів на рахунок орендодавця: Після підтвердження платежу система переведе кошти на рахунок орендодавця, і він отримає їх на своєму банківському рахунку або в іншій платіжній системі;

6) завершення процесу управління платежами: Після того, як кошти переведені на рахунок орендодавця, процес завершується.

Висновки до розділу 2

У цьому розділі було проведено детальне проектування інформаційної системи оренди житла з урахуванням сучасних вимог до ефективності, масштабованості та безпеки. Визначено функціональні вимоги, які охоплюють всі основні аспекти роботи системи. Окрім цього, сформульовано нефункціональні вимоги, які гарантують стабільну, продуктивну та безпечну роботу системи.

Проведено аналіз обмежень, таких як продуктивність, безпека даних, час розробки та адаптація системи до різних типів користувачів. Особливу увагу приділено ризикам, які можуть виникнути під час впровадження, зокрема технічним, організаційним та бізнес-ризикам.

Описані ключові бізнес-процеси, такі як реєстрація, пошук житла, бронювання, управління об'єктами та обробка транзакцій, а також їхня інтеграція із системою. Взаємодія між користувачами та системою була представлена у вигляді послідовностей дій, що допомогло визначити вимоги до кожного з компонентів.

Розділ закладає основу для подальшої розробки системи, зокрема проектування її архітектури, вибору технологічного стеку та реалізації функціональності.

3 АРХІТЕКТУРА СИСТЕМИ

Архітектура інформаційної системи є ключовим аспектом її розробки, оскільки визначає структуру, компоненти та взаємодію між ними. Для побудови системи оренди житла було обрано мікросервісну архітектуру [8], яка забезпечує гнучкість, масштабованість та надійність. Мікросервісна архітектура передбачає розподіл системи на незалежні сервіси, кожен з яких виконує чітко визначені завдання. Це дозволяє знижувати складність розробки, підвищувати стабільність і забезпечувати можливість паралельного розвитку різних компонентів.

Переваги мікросервісної архітектури для системи онлайн-бронювання квартир:

а) масштабованість. Кожен мікросервіс можна масштабувати незалежно, що дозволяє системі легко витримувати збільшення навантаження на окремі сервіси, такі як пошук квартир чи обробка платежів;

б) гнучкість розробки: Різні команди можуть працювати над окремими мікросервісами (наприклад, сервісом користувачів чи сервісом бронювання), що полегшує додавання нових функцій та покращує швидкість розробки;

в) надійність: Вихід з ладу одного мікросервісу не зупиняє роботу всієї системи. Наприклад, помилка в платіжному сервісі не заважає користувачам переглядати доступні квартири;

г) можливість використання різних технологій: Кожен мікросервіс може використовувати оптимальну для нього технологію, що додає системі гнучкості та знижує технологічні обмеження.

Недоліки мікросервісної архітектури для системи онлайн-бронювання квартир:

а) складність управління**:** Велика кількість мікросервісів ускладнює моніторинг, налаштування і підтримку системи, що може вимагати додаткових ресурсів та часу для адміністрування;

б) ускладнення комунікації між сервісами: Для стабільної взаємодії мікросервісів через мережу необхідно ретельно налаштовувати мережеві протоколи, що може призводити до затримок і потребує додаткових ресурсів;

в) безпека та автентифікація: Забезпечення безпеки кожного мікросервісу є складним завданням, оскільки необхідно впроваджувати аутентифікацію та авторизацію окремо для різних компонентів системи;

г) ресурсоємність: Мікросервіси зазвичай потребують окремих середовищ виконання (часто в контейнерах), що збільшує витрати на інфраструктуру і ресурси.

3.1 Загальний опис архітектури системи

Архітектура системи оренди житла складається з ряду компонентів, кожен з яких відповідає за конкретну функціональність системи. Основні компоненти системи включають:

а) клієнтський веб-застосунок: Інтерфейс користувача, через який орендарі та орендодавці взаємодіють із системою. Веб-застосунок забезпечує доступ до функцій платформи, таких як пошук, перегляд, бронювання апартаментів, управління профілем користувача та проведення платежів;

б) API-шлюз: Централізована точка доступу для клієнтських запитів, яка маршрутизує їх до відповідних мікросервісів. API-шлюз забезпечує аутентифікацію, авторизацію та балансування навантаження;

в) мікросервіси для реалізації бізнес-логіки системи: Автономні компоненти системи, які можуть бути розгорнуті на сервері системи незалежно один від одного. Список всіх мікросервісів наведений в таблиці 3.1;

г) бази даних: Сховища інформації про користувача, об’єктів нерухомості, та решти інформації необхідної для роботи бізнесу.

Таблиця 3.1 – Перелік мікросервісів системи

№	Назва мікросервісу	Призначення	Взаємодія з іншими компонентами
1	API Шлюз	Централізована точка входу для маршрутизації запитів.	Взаємодіє з усіма мікросервісами.
2	Мікросервіс авторизації	Відповідає за автентифікацію та авторизацію користувачів.	Спілкується з API Шлюзом та іншими сервісами для перевірки доступу.
3	Мікросервіс публікації оголошень	Управляє створенням, редагуванням та переглядом оголошень.	Взаємодіє з API Шлюзом та Мікросервісом апартаментів.
4	Мікросервіс апартаментів	Управляє даними про апартаменти, їх статус та доступність.	Взаємодіє з БД апартаментів, Мікросервісом публікації оголошень і Мікросервісом оренди.

№	Назва мікросервісу	Призначення	Взаємодія з іншими компонентами
5	Мікросервіс оренди	Відповідає за обробку заявок на оренду.	Взаємодіє з Мікросервісом користувачів, Мікросервісом апартаментів і Мікросервісом платежів.
6	Мікросервіс платежів	Відповідає за обробку транзакцій, підтвердження оплати та генерацію квитанцій.	Інтегрується з API Шлюзом та Мікросервісом оренди.
7	Мікросервіс користувача	Управляє профілями користувачів, їх даними та історією оренд.	Взаємодіє з API Шлюзом, Мікросервісом оренди та БД користувачів.

Бази даних передбачених в системі наведені в таблиці 3.2.

Таблиця 3.2 – Сховища даних системи

№	Назва бази даних	Призначення	Дані, що зберігаються
1	База даних Користувачів	Призначена для зберігання всієї інформації, пов'язаної з користувачами та їхньою взаємодією з системою.	Дані про користувачів: профілі (ім'я, контактна інформація), ролі, історія бронювань, статуси оплат, відгуки.
2	База даних апартаментів	Зберігає інформацію про об'єкти нерухомості, оголошення та їхній статус.	Інформація про апартаменти: назва, опис, розташування, ціна, статус доступності, фото; дані про оголошення: заголовки, описи, дати публікації.

3.2 Опис схеми бази даних

Схема бази даних клієнтів побудована на основі ASP.NET Core Identity[10], що дозволяє управляти автентифікацією, ролями та токенами користувачів. Основна таблиця AspNetUsers містить основну інформацію про користувачів, а додаткові таблиці забезпечують збереження ролей, claims, токенів та зовнішніх логінів.Детальний опис можна побачити в таблиці 3.3

Таблиця 3.3 – Опис бази даних клієнтів

№	Таблиця	Поля	Типи даних	Роль таблиці
1	AspNetUsers	Id, Name, Email, UserName, PasswordHash, PhoneNumber, EmailConfirmed, TwoFactorEnabled, LockoutEnabled, LockoutEnd, AccessFailedCount	Text, Character varying(256), Boolean, Timestamp, Integer,	Зберігає основні дані користувачів
2	AspNetRoles	Id, Name, NormalizedName, ConcurrencyStamp	Text, Character varying(256),	Містить ролі, що визначають права доступу користувачів.
3	AspNetUserRoles	UserId, RoleId	Text,	Зберігає зв'язок між користувачами та ролями
4	AspNetUserTokens	UserId, LoginProvider, Name, Value	Text	Зберігає токени для користувачів
5	AspNetUserClaims	Id, UserId, ClaimType, ClaimValue	Integer, Text	

№	Таблиця	Поля	Типи даних	Роль таблиці
6	AspNetRoleClaims	Id, RoleId, ClaimType, ClaimValue	Integer, Text	
7	AspNetUserLogins	LoginProvider, ProviderKey, ProviderDisplayName, UserId	Text	

База даних апартаментів складається з таблиць, що дозволяють зберігати інформацію про апартаменти, бронювання, відгуки та інші пов'язані дані. Основна таблиця Apartments містить загальну інформацію про об'єкти нерухомості. Опис бази даних можна розглянути в таблиці 3.4.

Таблиця 3.4 – Опис бази даних апартаментів

№	Таблиця	Поля	Типи даних	Роль таблиці
1	Apartments	Id, City, Street, ApartmentNumber, BuildingNumber, NumberOfRooms, Price, IsRented, OwnerId, CreatedAt	UUID, TEXT, INTEGER, BOOLEAN, NUMERIC	Зберігає інформацію про апартаменти
2	Bookings	Id, ApartmentId, UserId, StartDate, EndDate, Status	UUID, DATE, TEXT	Зберігає дані про бронювання

№	Таблиця	Поля	Типи даних	Роль таблиці
3	Reviews	Id, ApartmentId, UserId, Rating, Comment, CreatedAt	UUID, INTEGER, TEXT, TIMESTAMP	Зберігає дані про відгуки
4	Photos	Id, ApartmentId, Url	UUID, TEXT	Зберігає URL-адреси зображень апартаментів.
5	Features	Id, ApartmentId, FeatureName	UUID, TEXT	Описує додаткові характеристики апартаментів (наприклад, наявність кондиціонера або балкона).

ER Діаграми таблиць наведені в додатках В та Г.

Висновки до розділу 3

У цьому розділі детально розглянуто архітектуру системи оренди житла, побудованої на основі мікросервісної архітектури, яка відповідає сучасним вимогам до гнучкості, надійності та масштабованості. Проаналізовано ключові компоненти системи, включаючи API Gateway для маршрутизації запитів, а також мікросервіси, що відповідають за управління користувачами, об’єктами оренди, процесами бронювання та обробкою платежів. Кожен з цих компонентів функціонує незалежно, що дозволяє розподілити навантаження та забезпечити стабільність роботи платформи навіть за високої кількості запитів.

Архітектура підтримує інтеграцію з базами даних для зберігання інформації про користувачів і апартаменти, а також із зовнішніми сервісами, такими як платіжні системи. Використання сучасних технологій, зокрема REST API для комунікації між сервісами, JWT для безпечної авторизації та Docker для контейнеризації, сприяє підвищенню ефективності роботи платформи. Завдяки мікросервісному підходу система дозволяє проводити оновлення чи вдосконалення окремих компонентів без ризику порушення роботи інших частин, що робить її надзвичайно гнучкою і зручною для адміністрування.

Завдяки продуманій архітектурі створено основу для розробки програмного забезпечення, яке відповідає високим стандартам ринку, забезпечуючи зручність для користувачів, стабільну роботу платформи та легкість її адаптації до нових викликів і потреб.

4 МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ

4.1 Постановка задачі

Метою даного розділу є побудова алгоритму, який дозволяє класифікувати користувачів системи на основі їх характеристик і забезпечувати їм релевантні рекомендації щодо оренди квартир. Для досягнення цієї мети використовується метод кластеризації даних, що дозволяє групувати користувачів за спільними характеристиками. Після кластеризації для кожного кластеру обчислюється відповідність квартир, щоб забезпечити користувачам максимально релевантні варіанти.

4.1.1 Формалізація задачі кластеризації

Для кожного користувача системи визначається набір характеристик, які можна подати у вигляді вектора:

$$x_i = (x_{i1}, x_{i2}, \dots, x_{in}) , \tag{4.1}$$

де i — індекс користувача;

x_{ij} — значення j-ї характеристики користувача i;

n — загальна кількість характеристик (наприклад, вік, бюджет, кількість кімнат, бажана відстань до центру міста тощо).

Метою є згрупувати користувачів на основі цих характеристик таким чином, щоб користувачі зі схожими вимогами опинилися в одній групі (кластері).

Для реалізації кластеризації даних використовується алгоритм K-means [9], який розподіляє множину користувачів на k кластерів. Кожен кластер C_j описується його центроїдом - середнім вектором характеристик користувачів у цьому кластері:

$$\mu_j = \frac{1}{|C_j|}\sum_{x_i \in C_j} x_i, \tag{4.2}$$

де μ_j - центроїд кластера C_j;

$|C_j|$ - кількість користувачів у кластері j.

Основна мета алгоритму K-means — мінімізувати суму квадратів відстаней від кожного користувача до його центроїда. Це можна записати у вигляді оптимізаційної задачі:

$$J = \sum_{j=1}^{k}\sum_{x_i \in C_j} \left\| x_j + \mu_j \right\|^2, \tag{4.3}$$

де $\left\| x_j + \mu_j \right\|$ - евклідова відстань між користувачем та центроїдом.

4.2 Алгоритм K-means

Для ініціалізації потрібно вибрати кількість кластерів k, що визначає, на скільки груп буде розподілена вибірка користувачів. Випадковим чином ініціалізувати положення k центроїдів μ_j.

На кожній ітерації алгоритму кожен користувач призначається до найближчого центроїда, використовуючи евклідову відстань як міру схожості:

$$assign(x_i) = arg\min_j \left\| x_j + \mu_j \right\| \tag{4.4}$$

Після призначення всіх користувачів до кластерів обчислюється нове положення центроїдів як середнє значення характеристик усіх користувачів у кластері:

$$\mu_j = \frac{1}{|C_j|} \sum_{x_i \in C_j} x_i \quad (4.5)$$

Алгоритм повторює призначення користувачів та оновлення центроїдів, доки зміни у положенні центроїдів не стануть незначними або не буде досягнута максимальна кількість ітерацій.

4.2.1 Розрахунок відповідності квартир користувачам

Після кластеризації користувачів необхідно визначити, наскільки кожна квартира відповідає характеристикам користувачів у конкретному кластері. Для цього вводиться метрика відповідності $S(y, C_j)$, яка обчислюється для кожної квартири у відносно середніх характеристик користувачів у кластері C_j:

$$S(y, C_j) = \sum_{l=1}^{m} \omega_l * (1 - \frac{|y_l - \mu_{jl}|}{Max_l}), \quad (4.6)$$

де $S(y, C_j)$ - коефіцієнт відповідності квартири y кластеру C_j;

y_l - значення $l-$ї характеристики квартири;

μ_{jl} - середнє значення $l-$ї характеристики користувачів у кластері C_j;

Max_l - максимальна можлива різниця для характеристики l;

ω_l - ваговий коефіцієнт, що визначає важливість характеристики l.

Для забезпечення коректності розрахунків різні характеристики квартири повинні бути нормалізовані. Наприклад, характеристики, такі як бюджет або відстань до центру, можуть мати різні шкали, тому необхідно використовувати нормалізацію, щоб уникнути впливу різниць у шкалах.

4.2.2 Рекомендаційний алгоритм

При реєстрації нового користувача в системі або оновленні його даних, система визначає, до якого кластера він належить, на основі його характеристик. Це дозволяє швидко отримати групу користувачів з подібними потребами. На основі даних кластеру користувачів система обчислює коефіцієнт відповідності кожної доступної квартири для даного кластера, використовуючи формулу відповідності $S(y, C_j)$. Всі квартири, які оцінюються, ранжуються за зменшенням коефіцієнта відповідності. Таким чином, користувачам пропонуються найбільш релевантні варіанти оренди на основі їхніх уподобань.

Для оцінки точності рекомендаційної моделі використовуються такі метрики:

а) Точність (Precision) [13]**:** частка рекомендованих квартир, які справді були орендовані користувачами**.**

$$P = \frac{TP}{TP+FP} \tag{4.7}$$

де TP — кількість релевантних об'єктів, правильно рекомендованих системою.
FP — кількість нерелевантних об'єктів, рекомендованих системою;

б) Recall [14] : частка орендованих квартир, які були рекомендовані системою.

$$Recall = \frac{TP}{TP+FN} \tag{4.8}$$

де FN — кількість релевантних об'єктів, які не були рекомендовані;

в) F1-міра [15]: зважена середня метрик Precision та Recall, яка надає загальну оцінку ефективності алгоритму.

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall} \quad (4.9)$$

Виділимо переваги та недоліки методу.

Переваги:

а) простота реалізації та обчислень;

б) можливість швидкої класифікації користувачів на основі їх характеристик;

в) висока ефективність при відносно невеликій кількості характеристик.

Недоліки:

а) чутливість до вибору кількості кластерів;

б) модель не враховує складні взаємозв'язки між характеристиками;

в) початкове положення центроїдів впливає на результат, що може призводити до локальних мінімумів.

Висновки до розділу 4

У даному розділі було розроблено математичну модель для класифікації користувачів і надання їм рекомендацій щодо оренди квартир на основі їхніх характеристик. Для цього використано метод кластеризації K-means, який дозволив групувати користувачів зі схожими потребами в окремі кластери. Це забезпечує ефективне використання даних для пошуку релевантних варіантів оренди.

Запропонована модель фільтрації використовує метод оцінки відповідності, який базується на зіставленні характеристик квартир із середніми значеннями характеристик користувачів у кожному кластері. Використання нормалізованих значень і вагових коефіцієнтів для кожної характеристики дозволяє врахувати важливість різних параметрів при оцінюванні відповідності квартир потребам користувачів.

Результати класифікації та фільтрації дозволяють системі автоматизовано надавати персоналізовані рекомендації, що підвищує зручність і задоволеність користувачів системи. Застосування запропонованої моделі також дозволяє зменшити час, необхідний для пошуку відповідного житла, оскільки система пропонує лише ті варіанти, які найбільш ймовірно відповідають потребам користувача.

У перспективі, запропонований алгоритм може бути вдосконалений шляхом використання більш складних методів машинного навчання для аналізу даних, таких як нейронні мережі або методи зворотного зв'язку від користувачів, що дозволить ще точніше враховувати індивідуальні переваги та підвищити якість рекомендацій. Таке вдосконалення сприятиме створенню більш персоналізованого досвіду для користувачів, забезпечуючи підвищення їхньої задоволеності платформою. Це також дозволить збільшити конкурентоспроможність системи на ринку цифрових сервісів.

5 ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

5.1 Вибір програмного стеку для розробки

Для успішної реалізації системи оренди житла на мікросервісній архітектурі необхідно було підібрати оптимальний програмний стек, який забезпечить гнучкість, масштабованість та надійність. Нижче наведено огляд обраного стеку технологій для реалізації кожного з компонентів системи.

Основною мовою програмування для розробки серверної частини було обрано **C#**. Це обумовлено наступними факторами:

а) продуктивність і стабільність**;**

1) ефективне управління ресурсами: Завдяки оптимізованій роботі з пам’яттю та підтримці асинхронного програмування (через async/await), C# забезпечує високу продуктивність навіть за значного навантаження;

2) масштабованість: У поєднанні з ASP.NET Core, C# дозволяє легко масштабувати кожен мікросервіс, підтримуючи тисячі паралельних з’єднань;

б) підтримка мікросервісної архітектури;

1) декомпозиція додатку: C# забезпечує модульність та підтримку незалежного розвитку окремих компонентів завдяки об'єктно-орієнтованому підходу;

2) комунікація між мікросервісами: У C# легко реалізувати HTTP/REST API або використовувати протоколи, такі як gRPC, для ефективної взаємодії між мікросервісами;

3) робота з контейнерами: C# добре інтегрується з Docker, що спрощує контейнеризацію кожного мікросервісу;

в) інтеграція з хмарними сервісами;

1) Azure: C# є "рідною" мовою для Microsoft Azure, що забезпечує глибоку інтеграцію з такими сервісами, як Azure Kubernetes Service (AKS), Azure Functions, Azure App Service, Azure Storage та іншими.

2) гнучке управління: Використовуючи SDK для Azure на C#, можна безпосередньо взаємодіяти з хмарними ресурсами з коду;

г) розширені можливості безпеки;

1) вбудовані засоби захисту: C# підтримує шифрування, захист конфіденційних даних і вбудовані бібліотеки для роботи з аутентифікацією (зокрема, через JWT);

2) розмежування доступу: Завдяки вбудованій підтримці ролей і політик безпеки в ASP.NET Core, C# дозволяє зручно реалізовувати захист API на рівні мікросервісів;

д) інструменти розробки та підтримка;

1) Visual Studio та Visual Studio Code: Забезпечують передові засоби для розробки, тестування та налагодження;

2) активна спільнота: C# має велику спільноту розробників і постійно оновлюється, що гарантує актуальність і підтримку найновіших стандартів;

е) гнучкість у роботі з даними;

1) Entity Framework Core: Цей ORM (Object-Relational Mapping) спрощує роботу з базами даних, такими як PostgreSQL, забезпечуючи швидку розробку та підтримку даних;

2) робота з JSON: C# має вбудовану підтримку для роботи з JSON, що є стандартом для обміну даними в мікросервісах.

ASP.NET Core це кросплатформенний, модульний, високопродуктивний фреймворк для створення веб-додатків, API та мікросервісів від Microsoft. Ось кілька ключових аспектів ASP.NET Core, які є особливо корисними для проекту з онлайн-бронювання квартир:

а) кросплатформеність: Працює на Windows, Linux, macOS, дозволяючи гнучко розгортати систему;

б) REST API: фреймворк підтримує цей архітектурний підхід який можна використовувати для обміну даними між клієнтськими додатками та

сервісами. Він підтримує JSON як основний формат передачі даних, що робить його зручним для інтеграції з сучасними фронт-енд застосунками;

в) аутентифікація та авторизація: Використовує ASP.NET Core Identity та підтримує JWT, OAuth для захисту API й ролей користувачів;

г) Entity Framework Core: ORM для зручної роботи з базами даних.

д) підтримка мікросервісів: Легко інтегрується з Docker і Kubernetes, що робить архітектуру гнучкою й масштабованою;

е) продуктивність: Оптимізований для швидкого виконання запитів.

Ocelot — це потужна бібліотека, яка використовується для створення API Gateway у мікросервісній архітектурі[11]. У проекті вона виконує ключову роль централізованого маршрутизації запитів від клієнтів до відповідних мікросервісів, таких як сервіси користувачів, квартир або бронювання. Ocelot забезпечує зручне управління маршрутами, балансування навантаження та підтримку кешування. Також бібліотека інтегрується з JWT для перевірки авторизації, що дозволяє обмежувати доступ до ресурсів і захищати систему від несанкціонованих запитів. Простота налаштування і висока продуктивність роблять Ocelot ідеальним вибором для API Gateway у даному проекті;

JWT (JSON Web Tokens) використовується як основний механізм аутентифікації та авторизації в системі [12]. Після успішного входу користувач отримує токен, який містить закодовану інформацію, таку як ідентифікатор користувача, ролі та права доступу. JWT забезпечує безпечну передачу даних між клієнтами та сервером через HTTP-заголовки, не потребуючи постійних звернень до бази даних для перевірки сесії. Використання JWT дозволяє ефективно контролювати доступ до мікросервісів, спрощує розподілену аутентифікацію в мікросервісній архітектурі та гарантує, що кожен запит перевіряється на валідність і відповідність ролям користувача.

ASP.NET Core Identity — це бібліотека для аутентифікації та авторизації, яка допомагає створювати безпечні системи керування користувачами. Вона забезпечує реєстрацію, вхід, управління ролями та захист даних. У проєкті ASP.NET Core Identity дозволяє легко розмежувати доступ для різних ролей

(орендар, власник, адміністратор) та забезпечує захищений доступ до API й особистих даних.

EF Core — це ORM (Object-Relational Mapping), яка дозволяє працювати з базами даних за допомогою об'єктів C# замість SQL-запитів. Вона підтримує автоматичну міграцію бази даних і дозволяє використовувати LINQ для простих і складних запитів. EF Core полегшує роботу з базою даних PostgreSQL у проекті, надаючи зручний спосіб взаємодії з даними про користувачів, квартири та бронювання.

PosgtreSQL - це об'єктно-реляційна база даних з відкритим кодом, яка забезпечує стабільне збереження та обробку даних. В рамках проекту використання PostgreSQL було обрано як основне рішення для зберігання даних з наступних причин:

а) надійність та відповідність ACID: PostgreSQL забезпечує атомарність, консистентність, ізольованість та стійкість (ACID), що важливо для транзакцій, як-от бронювання квартир і обробка платежів;

б) підтримка складних запитів: PostgreSQL оптимально працює зі складними запитами та обробляє великі обсяги даних, що дозволяє швидко отримувати й фільтрувати інформацію про квартири та користувачів;

в) розширюваність: Підтримка JSON дозволяє працювати зі складними структурами даних, що може знадобитися для зберігання гнучких атрибутів квартир або профілів користувачів;

г) масштабованість: PostgreSQL добре працює як для невеликих проектів, так і для високонавантажених систем, що дозволяє масштабувати систему бронювання при зростанні кількості користувачів;

д) безпека: PostgreSQL має вбудовані засоби для контролю доступу та шифрування даних, що підвищує безпеку особистої інформації та транзакцій;

е) контейнери з Azure Database for PostgreSQL: Azure надає керовану базу даних PostgreSQL, що дозволяє знизити витрати на адміністрування та забезпечує автоматичне резервне копіювання, масштабування та захист від збоїв. Це робить PostgreSQL ідеальним рішенням для хмарних розгортань.

Для розробки інтерфейсу користувача було обрано Angular — популярний фреймворк для створення динамічних SPA-додатків, що забезпечує швидкість, адаптивність і зручність. Angular дозволяє реалізувати односторінковий додаток, де дані оновлюються без перезавантаження сторінок, використовує компонентний підхід для модульності, підтримує двобічне зв'язування даних для динамічного оновлення та інтеграцію з API. Вбудована маршрутизація та оптимізація продуктивності забезпечують плавну навігацію і швидку роботу додатка навіть із великими обсягами даних.

Microsoft Azure було обрано як хмарну платформу для розгортання системи оренди житла з урахуванням специфічних потреб проекту. Основні переваги, які зробили Azure оптимальним вибором для даної системи:

а) хмарні ресурси та масштабованість: Azure надає велику кількість хмарних сервісів, що дозволяють розгортати додаток без необхідності керувати фізичним обладнанням. Це знижує витрати на інфраструктуру та забезпечує гнучкість у налаштуванні;

б) автоматичне масштабування: Azure забезпечує автоматичне горизонтальне та вертикальне масштабування додатку. Це означає, що під час пікових навантажень (наприклад, під час збільшення кількості бронювань) ресурси можуть бути автоматично збільшені без втручання адміністратора;

в) Azure Kubernetes Service (AKS)**:** Цей сервіс дозволяє легко керувати оркестрацією контейнерів за допомогою Kubernetes, автоматизуючи розгортання, управління та масштабування додатку. Завдяки AKS можна забезпечити високу доступність системи та легко додавати нові мікросервіси;

г) CI/CD з Azure DevOps: Використання Azure DevOps для налаштування безперервної інтеграції та доставки (CI/CD) дозволяє автоматизувати процес розгортання оновлень додатку, зменшуючи ризик помилок і прискорюючи випуск нових версій;

д) безпека: Azure пропонує високий рівень безпеки даних, включаючи підтримку шифрування, багатофакторної аутентифікації та захисту від DDoS-атак;

е) резервне копіювання та відновлення**:** Azure пропонує вбудовані можливості резервного копіювання даних та відновлення системи, що є важливим для забезпечення надійності та безперервності бізнесу;

5.2 Вибір системного програмного забезпечення

Для системи оренди житла на мікросервісній архітектурі було вирішено використовувати керовані сервіси Microsoft Azure**.** Це дозволяє уникнути необхідності налаштування традиційного серверного середовища та зосередитися на розробці функціональності системи.

5.2.1 Керовані сервіси Microsoft Azure

Розгортання додатків та управління мікросервісами здійснюватиметься за допомогою таких керованих сервісів Azure:

а) Azure Kubernetes Service (AKS): Використовується для оркестрації контейнерів. AKS автоматизує управління контейнерами, включаючи розгортання, масштабування, балансування навантаження та моніторинг. Це забезпечує високу продуктивність та доступність системи без необхідності адміністрування серверної ОС;

б) Azure Database for PostgreSQL: Для зберігання даних використовується керована база даних PostgreSQL, яка забезпечує автоматичне резервне копіювання, оновлення та відновлення у разі збоїв. Це дозволяє зосередитися на структурі даних, а не на технічному обслуговуванні бази;

в) Azure App Service: Цей сервіс забезпечує швидке розгортання веб-додатків і REST API без необхідності управління інфраструктурою. Він підтримує автоматичне масштабування залежно від навантаження.

Контейнеризація мікросервісів виконується за допомогою Docker:

а) Кожен мікросервіс буде упакований у контейнер для ізоляції залежностей та забезпечення портативності додатку;

б) Контейнери будуть розгортатися через Azure Kubernetes Service, що дозволяє масштабувати окремі компоненти системи незалежно.

Керовані сервіси Azure забезпечують інтеграцію з іншими хмарними інструментами:

а) Azure Monitor: Використовується для моніторингу стану сервісів, збору метрик продуктивності та створення попереджень у разі виникнення проблем;

б) Azure DevOps: Для налаштування CI/CD, що дозволяє автоматизувати процеси тестування та розгортання нових версій додатків;

в) Azure Security Center: Забезпечує захист додатку, виявлення потенційних загроз та рекомендації з підвищення безпеки.

У проєкті адміністрування автоматизовано завдяки хмарним сервісам Azure, таким як AKS, Azure Database for PostgreSQL та Azure Monitor, які забезпечують масштабування, моніторинг, резервне копіювання та оновлення. Використання CI/CD-пайплайнів в Azure DevOps прискорює розгортання та оновлення системи, а безпека доступу до ресурсів контролюється через Azure Active Directory.

Для управління кодом використовується Git разом із GitHub, що забезпечує централізоване зберігання, спільну роботу та інтеграцію з Azure. GitHub слугує репозиторієм для мікросервісів, підтримуючи гілки, pull request'и та рев'ю коду. Інтеграція з Azure дозволяє автоматизувати збірку Docker-образів, їх публікацію в Azure Container Registry та деплоймент через GitHub Actions, спрощуючи розробку, тестування й розгортання.

5.3 Реалізація бізнес-логіки

У цьому підрозділі описано процеси реалізації бізнес-логіки системи оренди житла, що були здійснені для забезпечення коректного функціонування системи згідно з вимогами та специфікацією.

5.3.1 Сервер

Проект передбачено для випуску версіями, що дозволяє поетапно запускати нові функції та розширювати функціонал системи. У початковій версії реалізовано три основні мікросервіси: мікросервіс клієнта, мікросервіс апартаментів та API Gateway які реалізовують базовий функціонал системи. Структура бек-енду проекту наведена на рисунку 5.1.

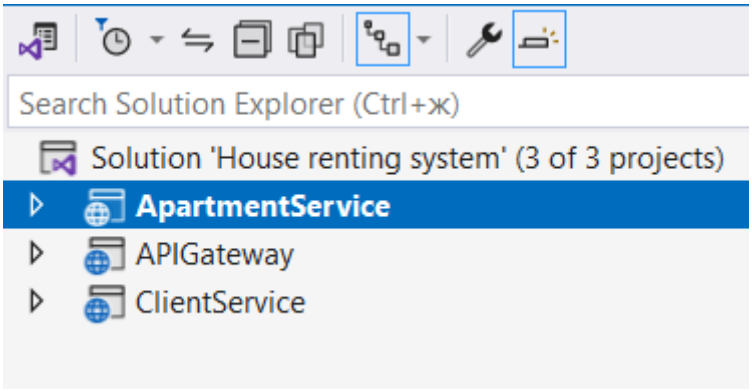

Рисунок 5.1 – Структура серверної частини системи

Першим в списку компонентом проекту є мікросервіс апартаментів (Apartment Service) який відповідає за роботу з даними про нерухомість, обробку заявок. Його структура наведена на рисунку 5.2 .

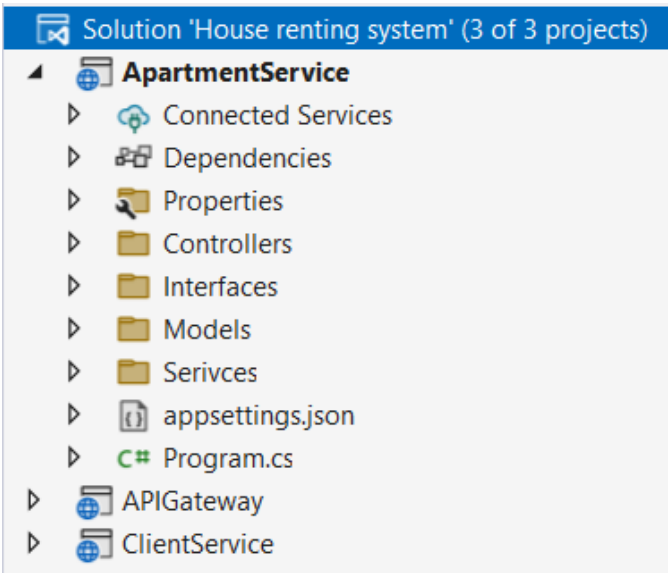

Рисунок 5.2 – Структура мікросервісу апартаментів

Наступним в списку є API Шлюз, реалізований як Asp.Net Core застосунок з використанням бібліотеки Ocelot. Логіка направлення запитів вказана в конфігураційному файлі Ocelot.json. Структура наведена на рисунку 5.3.

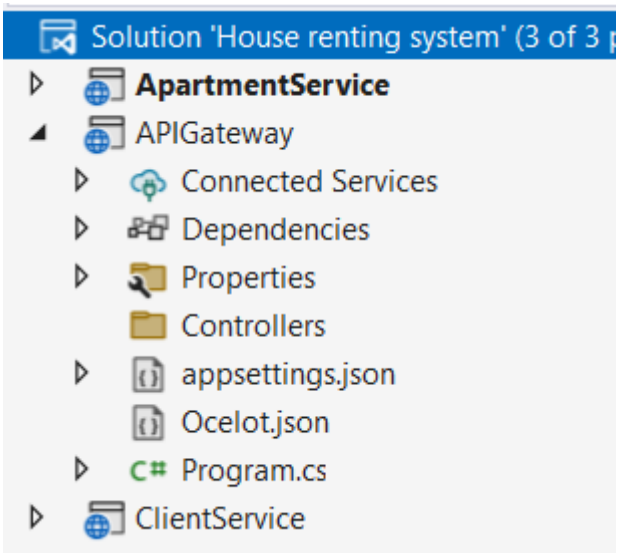

Рисунок 5.3 – Структура API шлюзу

Третім компонентом є мікросервіс клієнтів (ClientService) де реалізована робота з даними клієнтів а також логіка для аутентифікації користувачів. Структура наведена на рисунку 5.4.

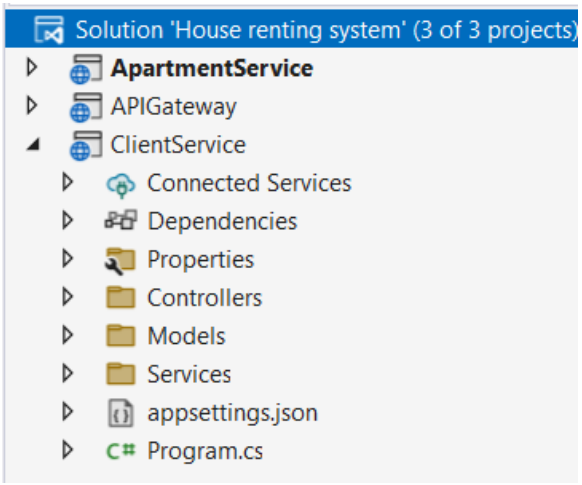

Рисунок 5.4 – Структура мікросервісу клієнта

Як приклад реалізації бізнес-логіки наведемо процес пошуку квартири.

Спочатку користувачу потрібно надіслати запит на API шлюз, який перенаправить запит користувача на контроллер, і обробить відповідним методом який наведений на рисунку 5.5

```
0 references
public async Task<ActionResult<IEnumerable<Apartment>>> GetApartment([FromQuery] ApartmentQueryParams parameters)
{
    var result = await _apartmentService.GetApartments(parameters);
    Console.WriteLine("Response: \n" + result);
    return Ok(result);
}
```

Рисунок 5.5 – Метод обробки запиту на список квартир

Логіка обробки запитів винесена в клас ApartmentDataService який віповідає за формування параметрів запиту в БД. Метод класу для отриманння списку квартир які цікавлять користувача наведений на рисунку 5.6.

```
1 reference
public async Task<IEnumerable<Apartment>> GetApartments(ApartmentQueryParams queryParams)
{
    var parameter = Expression.Parameter(typeof(Apartment), "r");

    Expression predicate = Expression.Constant(true);

    if (!string.IsNullOrEmpty(queryParams.City))
    {
        var cityCondition = Expression.Equal(
            Expression.Property(parameter, nameof(Apartment.City)),
            Expression.Constant(queryParams.City)
            );
        predicate = Expression.AndAlso(predicate, cityCondition);
    }
    if (queryParams.Rooms.HasValue)
    {
        var roomsCondition = Expression.Equal(
            Expression.Property(parameter, nameof(Apartment.NumberOfRooms)),
            Expression.Constant(queryParams.Rooms.Value)
            );
        predicate = Expression.AndAlso(predicate, roomsCondition);
    }

    var publishedCondition = Expression.NotEqual(
        Expression.Property(parameter, nameof(Apartment.IsRented)),
        Expression.Constant(true)
        );
    predicate = Expression.AndAlso(predicate, publishedCondition);

    var lambda = Expression.Lambda<Func<Apartment, bool>>(predicate, parameter);

    var result = await _repository.GetByCondition(lambda);

    if (result == null)
    {
        return new List<Apartment>();
    }

    return result;
}
```

Рисунок 5.6 – Метод для роботи з джерелом даних.

В свою чергу клас ApartmentDataService після формування черги запиту звертається до класу ApartmentRepository який відповідає за роботу з БД. Код класу наведений на рисунку 5.7 .

```
2 rererences
public class ApartmentRepository: IRepository<Apartment>
{
    private readonly ApartmentDbContext _db;
    0 references
    public ApartmentRepository(ApartmentDbContext db)
    {
        _db = db;
    }

    3 references
    public async Task<IEnumerable<Apartment>> GetByCondition(Expression<Func<Apartment, bool>> condition)
    {
        var query = _db.Apartments.AsQueryable();
        query = query.Where(condition);
        var result = await query.ToListAsync();

        return result;
    }
    2 references
    public async Task AddAsync(Apartment user)
    {
        await _db.Apartments.AddAsync(user);
        await _db.SaveChangesAsync();
    }

    2 references
    public async Task UpdateAsync(Apartment user)
    {
        _db.Apartments.Update(user);
        await _db.SaveChangesAsync();
    }

    2 references
    public async Task DeleteAsync(Guid id)
    {
```

Рисунок 5.7 – Код класу ApartmentRepository

Для решти сутностей, таких як оголошення, користувачі логіка роботи аналогічна.

5.3.2 Клієнт

Клієнтська частина системи оренди житла реалізована як веб-додаток, що забезпечує користувачам зручний інтерфейс для взаємодії з платформою. Основними завданнями клієнтської частини є забезпечення доступу до функціональності системи, інтуїтивно зрозумілої навігації та візуального представлення даних.

Загальна струкутруа застосунку наведена на рисунку 5.8 .

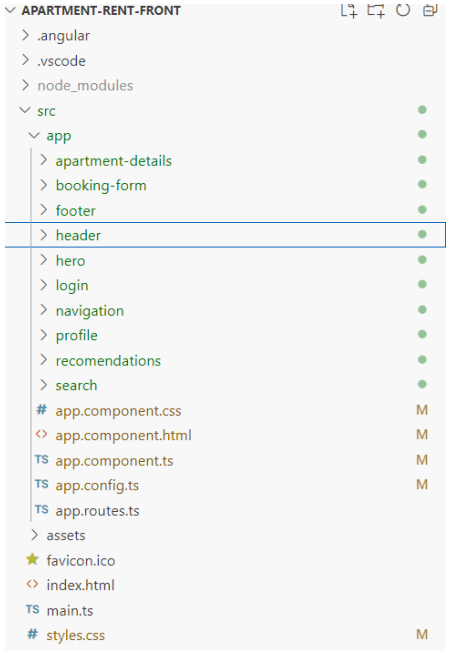

Рисунок 5.8 – Загальна структура користувацького застосунку

Проект поділений на компоненти кожен з яких відповідає за відображення певної частини функціоналу. На рисунку 5.9 наведений компонент реєстрації в системі.

Рисунок 5.9 – Компонент аутентифікації в системі

Кожен компонент поділений на три частини: html фрагмент, TS фрагмент який відповідає за поведінку інтерфейсу, і css фрагмент що описує специфічні для компонента стилі.

Висновки до розділу 5

У розділі програмної реалізації детально описано використання сучасних технологій та інструментів для створення системи оренди житла на основі

мікросервісної архітектури. Кожен компонент системи було розроблено з урахуванням особливостей обраних технологій, що дозволило забезпечити стабільну роботу, ефективну взаємодію між сервісами та високу продуктивність.

Реалізація системи включала інтеграцію баз даних, і механізмів безпеки, таких як шифрування даних та захищена авторизація користувачів. Використання розподіленої архітектури дозволило досягти масштабованості та гнучкості у додаванні нових функцій.

Результати тестування підтвердили, що система відповідає визначеним вимогам і готова до подальшого впровадження та використання у реальному середовищі, забезпечуючи зручність і безпеку для користувачів.

6 РОЗРОБЛЕННЯ ІНТЕРФЕЙСУ КОРИСТУВАЧА

У цьому розділі розглянуто особливості взаємодії користувача з веб-інтерфейсом системи оренди житла, розробленої на основі мікросервісної архітектури. Описано основні функції, які забезпечують зручний доступ до функціональності системи, а також технічні аспекти реалізації інтерфейсу.

Інтерфейс інформаційної системи було розроблено як веб-додаток на базі Angular. Даний фреймворк забезпечує швидке завантаження сторінок, високу продуктивність і зручну роботу з компонентами, що дозволяє користувачу легко орієнтуватися у функціональності системи.

Основні елементи інтерфейсу включають:

а) головна сторінка – інформація про доступні квартири, загальні рекомендації, фільтри пошуку. Зображено на рисунку 6.1;

б) сторінка пошуку квартир – можливість пошуку з використанням фільтрів (розташування, ціна, кількість кімнат тощо). Зображено на рисунку 6.2;

в) форма реєстрації та авторизації – надання доступу до особистого кабінету користувача. Зображено на рисунку 6.3;

г) сторінка бронювання – функціонал для вибору та оренди квартири. Наведено на рисунку 6.4;

д) особистий кабінет – функції управління обліковим записом, історія оренди, налаштування профілю. Зображено на рисунку 6.5;

е) деталі квартири – функціонал для перегляду детальної інформації про квартиру. Наведено на рисунку 6.6 ;

ж) панель адміністратора – доступна користувачам із роллю "адміністратор"; дозволяє додавати, редагувати та видаляти інформацію про квартири. Зображено на рисунку 6.7 .

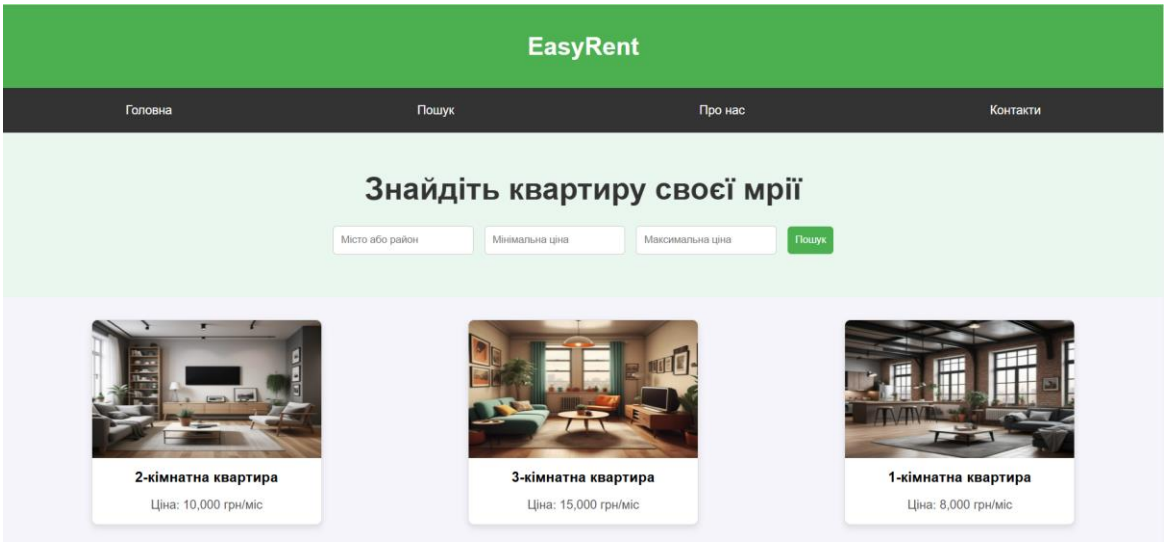

Рисунок 6.1 – Головна сторінка системи

Цей інтерфейс демонструє головну сторінку системи, де користувачі можуть переглядати список доступних квартир, фільтрувати їх за основними параметрами (ціна, розташування, кількість кімнат) та переходити до перегляду детальної інформації про обрані об’єкти.

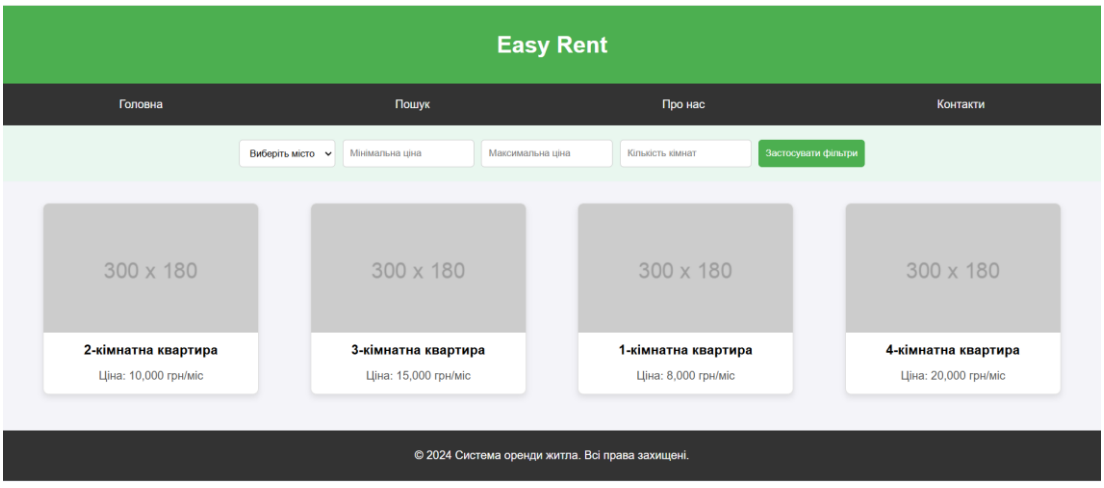

Рисунок 6.2 – Сторінка пошуку квартир

Інтерфейс пошуку квартир дозволяє користувачам швидко знайти житло, що відповідає їхнім вимогам.

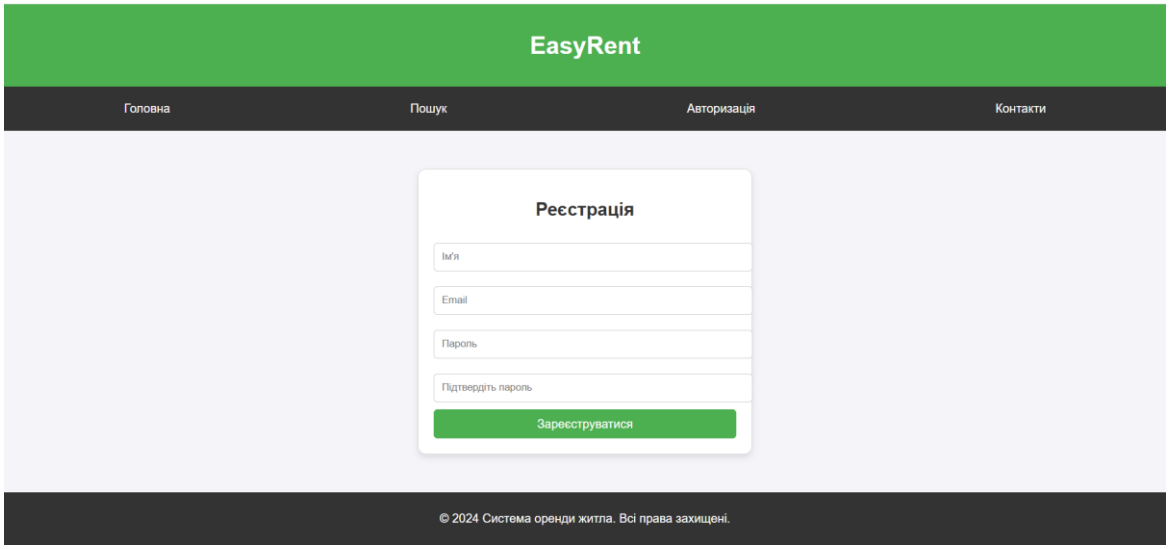

Рисунок 6.3 – Сторінка реєстрації

Форма для реєстрації в системі.

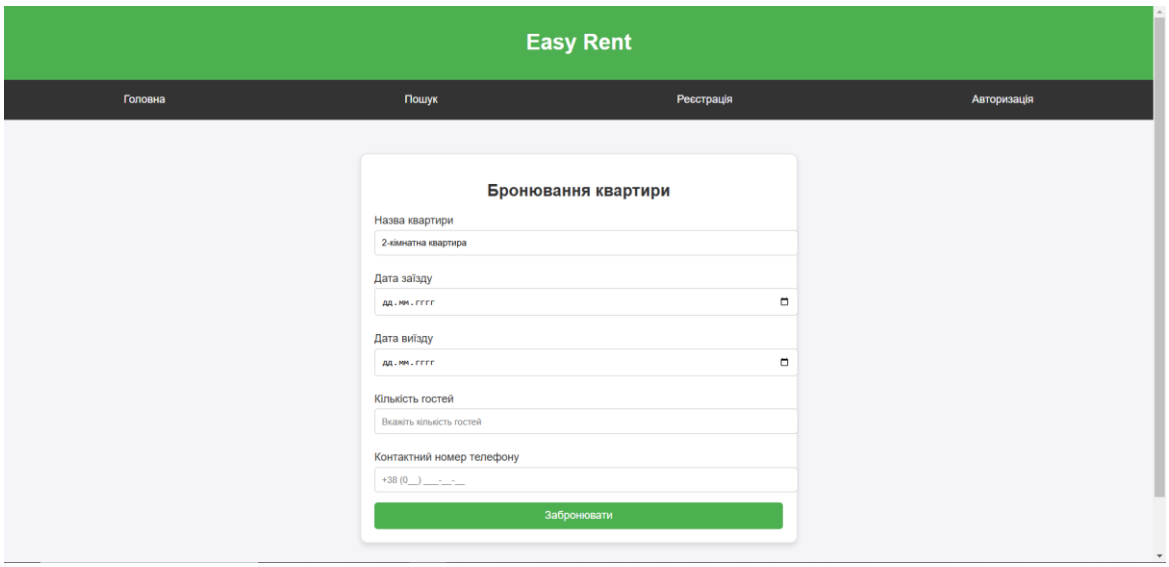

Рисунок 6.4 – Сторінка бронювання

На сторінці бронювання користувач вводить необхідні дані, такі як дати заїзду та виїзду, після чого підтверджує бронювання. Додатково відображається інформація про квартиру, загальна сума оренди та кнопка для завершення оплати.

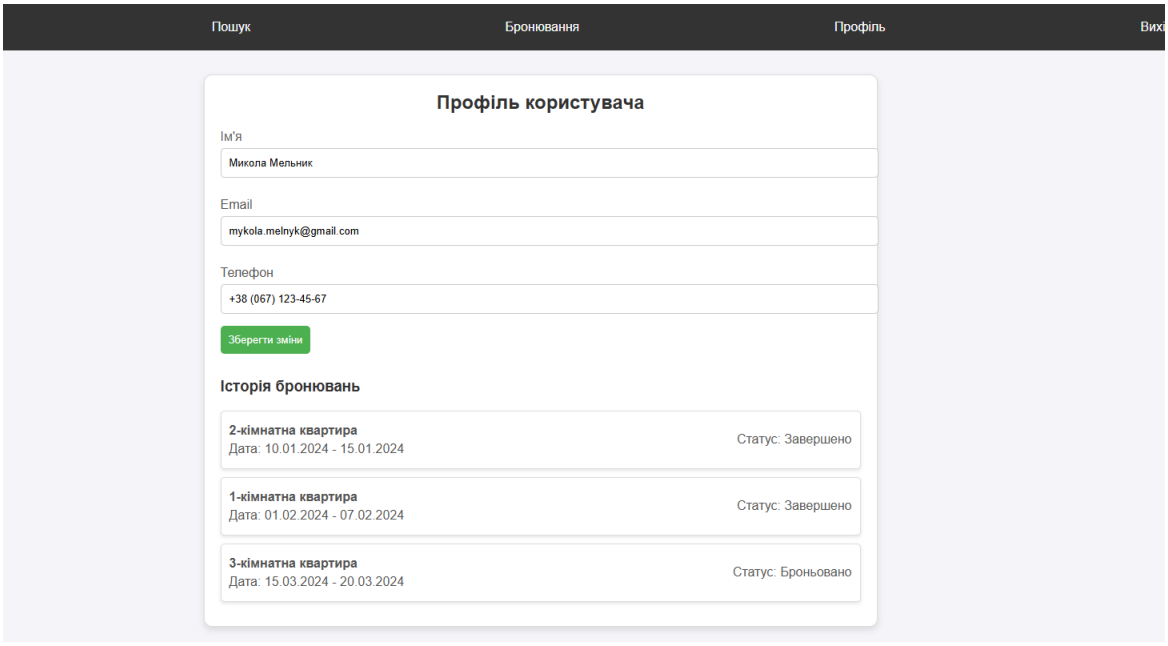

Рисунок 6.5 – Сторінка профілю користувача

Сторінка профілю дозволяє користувачеві переглядати та редагувати персональні дані, змінювати пароль, а також переглядати історію бронювань і залишені відгуки.

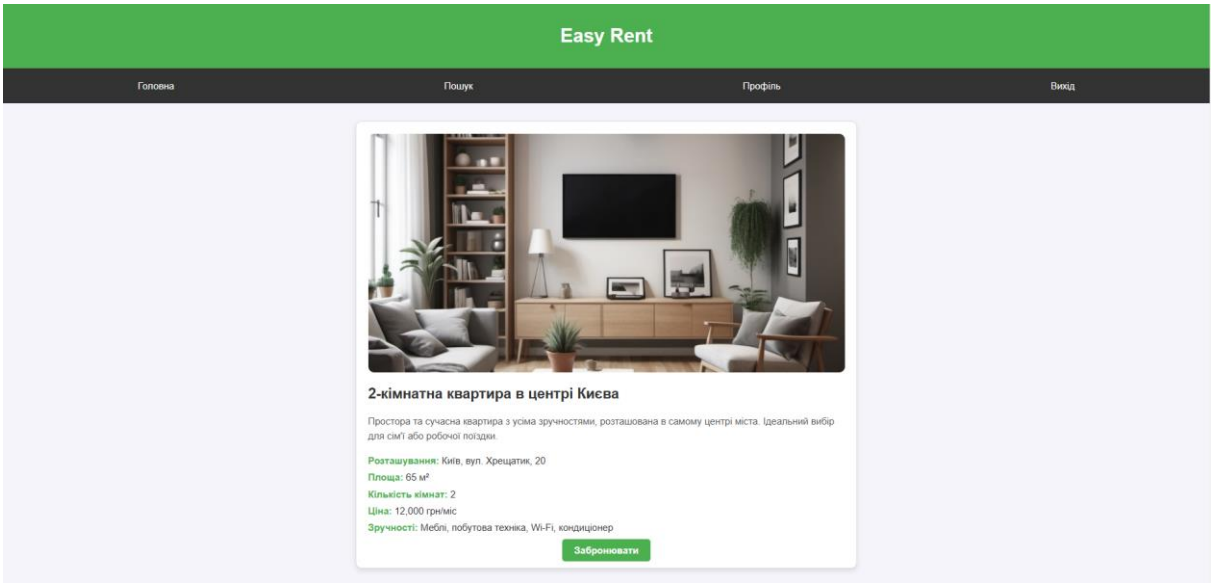

Рисунок 6.6 – Сторінка деталей квартири

На сторінці відображається повна інформація про обрану квартиру, включно з фото, описом, характеристиками, ціною та розташуванням. Також

передбачено кнопку для бронювання житла та можливість додати квартиру до обраних.

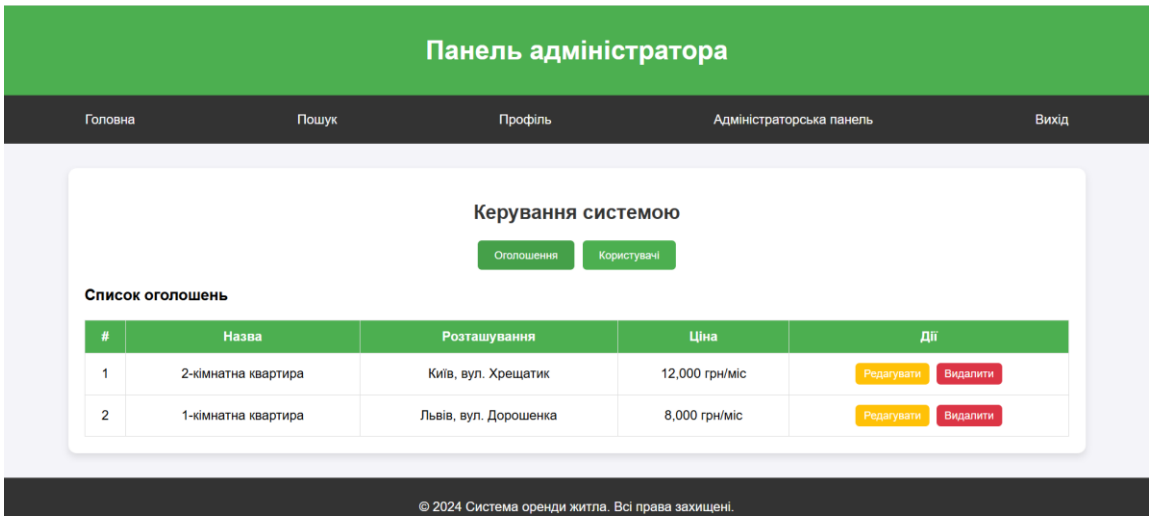

Рисунок 6.7 – Сторінка панелі адіміністратора

Панель адміністратора надає інструменти для керування системою**:** перегляд і модерація оголошень, управління користувачами, моніторинг бронювань та відгуків. Адміністратор має можливість блокувати облікові записи, видаляти некоректні оголошення та контролювати загальну активність у системі.

Висновки до розділу 6

У цьому розділі було представлено основні інтерфейси системи оренди житла, які забезпечують зручну взаємодію користувача з функціоналом платформи. Головна увага приділена інтуїтивно зрозумілому дизайну та ефективному користувацькому досвіду для різних категорій користувачів: орендарів, орендодавців та адміністраторів. Розроблені інтерфейси включають можливість пошуку та фільтрації житла, бронювання об’єктів, керування об’єктами оренди та перегляду історії взаємодій користувача із системою.

Усі функціональні елементи були розроблені з урахуванням сучасних принципів UI/UX-дизайну та оптимізовані для роботи в умовах великої кількості даних.

7 РОЗРОБЛЕННЯ СТАРТАП-ПРОЕКТУ

7.1 Опис ідеї стартап-проекту

Ідея стартапу полягає в створенні цифрової платформи для оренди житла, Система повинна забезпечити зручне бронювання квартир, автоматизовану систему обробки запитів, інтеграцію з платіжними системами та високий рівень безпеки для персональних даних користувачів.

Платформа забезпечує зручний і функціональний інструмент для взаємодії орендарів та орендодавців, автоматизуючи ключові процеси:

а) пошук та бронювання житла (як короткострокового, так і довгострокового);

б) управління оголошеннями, статусами та об’єктами нерухомості;

в) здійснення безпечних платежів через інтеграцію з платіжними системами;

г) розширення охоплення через інтеграцію з великими цифровими платформами;

Основною архітектурною особливістю є мікросервісний підхід, що дозволяє підвищити продуктивність, масштабованість та надійність системи.

Можливі напрямки застосування:

а) агентства нерухомості: Платформа дозволяє автоматизувати процес управління об’єктами нерухомості, оптимізуючи роботу агентів та полегшуючи комунікацію з клієнтами;

б) туризм: Мандрівники можуть швидко знайти відповідне житло для короткострокового перебування, використовуючи зручний інтерфейс для бронювання та оплати;

в) великі цифрові платформи: Платформа може інтегруватися з глобальними сервісами для розширення аудиторії та підвищення впізнаваності бренду.

За кожним напрямком застосування користувачі отримують конкретні переваги:

а) агентства нерухомості**:** Автоматизація процесів, зменшення витрат часу на управління об’єктами та клієнтами;

б) туризм: Зручний пошук житла, прозоре бронювання та швидкі платежі, що покращує досвід користувачів;

в) великі цифрові платформи: Розширення охоплення цільової аудиторії та підвищення впізнаваності бренду через інтеграцію на глобальному рівні.

На відміну від існуючих платформ, таких як Airbnb чи Booking.com, запропонована система має низку відмінних характеристик:

а) мікросервісна архітектура забезпечує високу продуктивність і гнучкість при масштабуванні;

б) рекомендаційна система, що адаптується до вподобань користувачів, забезпечує персоналізовані пропозиції житла;

в) інтеграція з платіжними системами для безпечних транзакцій;

г) підтримка як короткострокової, так і довгострокової оренди на єдиній платформі, що розширює цільову аудиторію.

Підсумком проведеного аналізу є таблиця 7.1, що систематизує зміст ідеї, напрямки застосування та вигоди для користувачів.

Таблиця 7.1 – Опис ідеї стартапу

Зміст ідеї	Напрямки застосування	Вигода для користувачів
Розробка платформи для оренди житла на основі мікросервісної архітектури	Агентства нерухомості	Автоматизація роботи агентів, управління об'єктами та клієнтами.
	Туризм	Легкий пошук житла для подорожей, прозоре бронювання, швидкі платежі.
	Великі цифрові платформи	Розширення аудиторії та підвищення впізнаваності бренду.

Для оцінки потенційних переваг стартап-проєкту визначено перелік техніко-економічних властивостей та характеристик ідеї платформи для оренди житла. Орієнтовний перелік показників включає:

а) масштабованість (здатність до обробки зростаючої кількості користувачів);

б) функціональність (обсяг можливостей для орендарів та орендодавців);

в) зручність користувацького інтерфейсу (UI/UX);

г) інтеграція з платіжними системами;

д) персоналізація (впровадження рекомендаційної системи для підбору житла).

Основними конкурентами запропонованої платформи на ринку оренди житла є:

а) Flatfy.ua – агрегатор оголошень про нерухомість, що пропонує оголошення з різних джерел;

б) Rieltor.ua – платформа для публікації оголошень про оренду та продаж житла;

в) Booking.com – сервіс для короткострокової оренди житла (включно з апартаментами).

Збір інформації щодо техніко-економічних показників проведено для зазначених конкурентів та власної ідеї платформи. Порівняльний аналіз подано у таблиці 7.2 .

Таблиця 7.2 – Визначення сильних, слабких та нейтральних характеристик

№ п/п	Техніко-економічні характеристики ідеї	(потенційні) товари/концепції конкурентів				W (слабка сторона)	N (нейтральна сторона	S (сильна сторона)
		Мій проект	Flatfy .ua	Rieltor .ua	Booking. com			
1	Масштабованість	Присутнє	Присутнє	Відсутнє	Присутнє	-	-	+
2	Функціональність	Присутнє	Присутнє	Присутнє	Присутнє	-	+	-
3	Зручність користувацького інтерфейсу	Присутнє	Присутнє	Присутнє	Присутнє	-	-	+
4	Інтеграція з платіжними системами	Присутнє	Відсутнє	Відсутнє	Присутнє	-	+	-
5	Персоналізація (рекомендаційна система)	Присутнє	Відсутнє	Відсутнє	Присутнє	-	-	+

7.2 Технологічний аудит ідеї проекту

Для розробки стартапу використовуватимуться передові технології, що забезпечують масштабованість, надійність і безпеку. Мікросервісна архітектура дозволяє ефективно масштабувати систему відповідно до зростаючих вимог ринку оренди житла.

Список технологій які планується використовувати наведений в таблиці 7.3

Таблиця 7.3 – Технології для реалізації проекту

№	Ідея проекту	Технології її реалізації	Наявність технології	Доступність технологій
1	Для створення API-шлюзу та мікросервісів	ASP.NET Core	Наявна	Доступна
2	Сховища даних, та інструменти роботи з ними	PostgreSQL, Entity Framework Core	Необхідно розробити модуль для роботи з сховищами даних	Доступна
3	Платформа для розгортання системи в мережі	Azure	Наявна	Доступна, потребує додаткової конфігурації
4	Користувацький додаток	TypeScript, Angular	Наявна	Доступна
5	Платіжна система	LiqPay	Наявна	Доступна , але вимагає комісії за транзакції

7.3 Аналіз ринкових можливостей запуску стартапу

Для визначення ринкових можливостей та загроз, пов'язаних із запуском платформи для оренди житла, було проведено аналіз поточного стану ринку. Характеристика потенційного ринку наведена в таблиці 7.4.

Таблиця 7.4 – Попередня характеристика потенційного ринку стартап-проекту

№ п/п	Показники стану ринку (найменування)	Характеристика
1	Кількість головних гравців, од	10–12 основних платформ, зокрема Flatfy.ua, Rieltor.ua, Booking.com, а також локальні сервіси.
2	Загальний обсяг продаж, грн/ум.од	Від 3 до 5 мільярдів грн на рік для українського ринку оренди нерухомості.
3	Динаміка ринку (якісна оцінка)	Ринок зростає через збільшення попиту на онлайн-сервіси для оренди житла та цифровізацію галузі.
4	Наявність обмежень для входу	Висока конкуренція з глобальними платформами;
5	Специфічні вимоги до стандартизації та сертифікації	Відповідність законодавчим вимогам щодо захисту персональних даних (GDPR, Закон України про захист даних).
6	Середня норма рентабельності в галузі (%)	15–20% залежно від моделі монетизації (комісія від оренди, платні послуги для орендодавців).

Впровадження стартап-проєкту є перспективним, за умови впровадження унікальних конкурентних переваг, таких як рекомендаційна система, інтеграція з локальними платіжними сервісами та персоналізований підхід до клієнтів.

На основі аналізу ринку та потреб користувачів визначено ключові групи клієнтів, для яких платформа оренди житла буде актуальною. Кожна група має свої вимоги які потрібно врахувати при розробці.Потреби потенційних груп клієнтів, їхні особливості та вимоги до платформи систематизовано у таблиці 7.5 .

Таблиця 7.5 – Характеристика потенційних клієнтів стартап-проекту

№ п/п	Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
1	Пошук і бронювання житла для короткострокової оренди	Туристи та мандрівники	Швидкий пошук житла, гнучкі умови оплати та бронювання, наявність відгуків.	- Простий і швидкий інтерфейс - Безпечні платежі
2	Пошук житла для довгострокової оренди	Студенти, молоді фахівці, сім’ї	Пошук доступного житла, можливість вибору за ціною та локацією.	- Фільтри пошуку за ціною, локацією, строками оренди
3	Управління житлом і оголошеннями	Власники житла та агентства нерухомості	Потреба в інструменті для розміщення та управління оголошеннями.	- Зручна публікація оголошень - Аналітика переглядів

Для успішного впровадження стартап-проєкту на ринок необхідно визначити основні загрози, які можуть перешкоджати реалізації платформи для оренди житла. Такі фактори проаналізовані та подані у порядку зменшення їхньої значущості у таблиці 7.6 .

Таблиця 7.6 – Фактори загроз

№ п/п	Фактор	Зміст загрози	Можлива реакція компанії
1	Висока конкуренція	Сильна присутність глобальних гравців (Booking.com, Airbnb).	Розробка унікальних функцій: рекомендаційна система, локалізація.
2	Законодавчі обмеження	Регулювання обробки персональних даних (GDPR) та фінансових транзакцій.	Використання сертифікованих технологій безпеки та відповідності.
3	Технологічні бар'єри	Можливі проблеми інтеграції з платіжними системами або зовнішніми API.	Поетапне тестування інтеграцій та вибір перевірених рішень.
4	Низька лояльність користувачів	Перевага користувачів до відомих платформ-конкурентів.	Впровадження програм лояльності та бонусів для перших користувачів.

Поряд із загрозами ринок надає можливості, які можуть сприяти успішному впровадженню платформи для оренди житла. Основні фактори можливостей систематизовано у таблиці 7.7 .

Таблиця 7.7 – Фактори можливостей

№ п/п	Фактор	Зміст можливості	Можлива реакція компанії
1	Зростання попиту на оренду житла	Збільшення мобільності населення та потреби в онлайн-сервісах для оренди.	Швидкий запуск платформи з орієнтацією на потреби цільових груп.
2	Цифровізація ринку нерухомості	Поступовий перехід до онлайн-управління нерухомістю.	Розробка інструментів для орендодавців та агентств нерухомості.
3	Підтримка локальних сервісів	Перевага користувачів до національних платформ у регіонах.	Локалізація інтерфейсу та підтримка локальних платіжних систем.
4	Розвиток платіжних технологій	Підвищення доступності безготівкових та онлайн-платежів.	Інтеграція з популярними платіжними системами (Stripe, LiqPay).
5	Партнерства з іншими сервісами	Можливість інтеграції з туристичними та корпоративними платформами.	Розробка API для інтеграції зі сторонніми сервісами.

Аналіз конкуренції на ринку наведений в таблиці 7.8 .

Таблиця 7.8 – Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)
Тип конкуренції - монопольна	За монополістична конкуренції ринок має багато гравців з подібними, але не ідентичними послугами.	Розробка унікальних функцій, таких як рекомендаційна система та інтеграція локальних рішень.
За рівнем конкурентної боротьби – національний	Продукт планується постачати на національний ринок	Орієнтація на локальні ринки, співпраця з національними партнерами для підвищення лояльності.
За галузевою ознакою - внутрішньогалузева	Конкуренція між платформами оренди житла.	Забезпечення доступного функціоналу, швидкої обробки запитів та прозорості операцій.
Конкуренція за видами товарів - товарно-видова	Різні платформи пропонують житло для короткострокової або довгострокової оренди.	Підтримка обох типів оренди на одній платформі для охоплення більшої аудиторії.
За характером конкурентних переваг - нецінова	Ключовими факторами є зручність, швидкість сервісу та підтримка клієнтів.	Покращення UI/UX, інтеграція безпечних платежів та розвиток системи підтримки користувачів.
За інтенсивністю - марочна	Відомі бренди (Booking.com, Airbnb) домінують на ринку.	Створення бренду з акцентом на локальні переваги, програми лояльності для залучення користувачів.

Результат аналізу конкуренції за методом Портера наведений в таблиці 7.9 конкуренції.

Таблиця 7.9 – Аналіз конкуренції в галузі за М. Портером

Складові аналізу	Прямі конкуренти в галузі	Потенційні конкуренти	Постачальники	Клієнти	Товари-замінники
	Booking.com, Airbnb, Flatfy.ua, Rieltor.ua	Відсутні	Відсутні	Клієнти (орендарі, орендодавці) мають значний вплив на ринок через високі вимоги до функціональності, зручності сервісу та безпеки	Часткова заміна присутня
Висновок	Інтенсивність конкурентної боротьби є високою через наявність глобальних гравців та сильну конкуренцію локальних сервісів.	Потенційні конкуренти можуть легко увійти на ринок завдяки технологіям, однак бар'єром є маркетингові витрати.	Постачальники мають обмежений вплив, оскільки ринок технологічних сервісів є конкурентним.	Вплив клієнтів значний через їхні високі вимоги до функціональності та зручності.	Загроза товарів-замінників помірна, оскільки альтернативні канали соцмережі і т.д.

Проєкт є конкурентоспроможним, оскільки поєднує унікальні функції (рекомендаційна система), орієнтується на локальні особливості ринку. В таблиці 7.10 наведено обґрунтування факторів конкурентноспроможності.

Таблиця 7.10 – Обґрунтування факторів конкурентоспроможності

№ п/п	Фактор конкурентоспроможності	Обґрунтування (наведення чинників, що роблять фактор для порівняння конкурентних проектів значущим)
1	Рекомендаційна система	Забезпечує персоналізований підбір житла відповідно до уподобань користувача, що відсутнє у багатьох конкурентів.
2	Інтеграція з платіжними системами	Підтримка локальних та міжнародних платіжних систем (Stripe, LiqPay) для зручності користувачів.
3	Гнучка підтримка коротко- та довгострокової оренди	Відмінність від конкурентів, що переважно орієнтовані лише на короткострокову оренду.
4	Локалізація платформи	Орієнтація на місцевий ринок (мовна підтримка, локальні особливості) для залучення регіональної аудиторії.
5	Висока продуктивність і масштабованість	Використання мікросервісної архітектури забезпечує швидкість роботи та можливість обслуговування великої кількості користувачів.
6	Зручний користувацький інтерфейс (UI/UX)	Спрощення навігації, фільтрів пошуку та процесу бронювання підвищує лояльність клієнтів.

За даними таблиці 7.10 аналізуємо сильні та слабкі сторони проекту. Результати наведені в таблиці 7.11 .

Таблиця 7.11 – Порівняльний аналіз сильних та слабких сторін проекту

№ п/п	Фактор конкурентоспроможності	Бали 1-20	Рейтинг конкурентів						
			-3	-2	-1	0	+1	+2	+3
1	Рекомендаційна система	18					+		
2	Інтеграція з платіжними системами	17			+				
3	Гнучка підтримка коротко- та довгострокової оренди	15		+					
4	Локалізація платформи	16			+				
5	Висока продуктивність і масштабованість	16			+				
6	Зручний користувацький інтерфейс (UI/UX)	18					+		

В таблиці 7.12 наведений SWOT-аналіз проекту.

Таблиця 7.12 SWOT – аналіз стартап-проекту

Сильні сторони: рекомендаційна система, Інтеграція з платіжними системами, Локалізація платформи для регіональних користувачів.	Слабкі сторони: Висока залежність від технологічних постачальників (платіжні системи, хмарна інфраструктура), Обмежена початкова впізнаваність бренду.
Можливості: Зростання попиту на онлайн-сервіси оренди житла, Розвиток партнерств з агентствами нерухомості та туристичними компаніями.	Загрози: Висока конкуренція з глобальними платформами (Booking.com, Airbnb), Нестабільна економічна ситуація, що впливає на платоспроможність клієнтів.

Після проведення SWOT-аналізу проекту також необхідно визначити альтернативи впровадження проекту на ринок. Результати наведено в таблиці 7.13

Таблиця 7.13 – Альтернативи ринкового впровадження стартап-проекту

№ п/п	Альтернатива (орієнтовний комплекс заходів) ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
1	Запуск платформи з базовим функціоналом: короткострокова оренда, інтеграція з платіжними системами.	Висока (готовність технологій та ресурсів).	3-6 місяців

7.4 Розробка ринкової стратегії проекту

Розробка ринкової стратегії є важливим етапом впровадження стартап-проєкту, що визначає напрями охоплення цільового ринку та забезпечує ефективну взаємодію з потенційними споживачами. Першим кроком є визначення стратегії охоплення ринку**,** що передбачає ідентифікацію та детальний опис цільових груп потенційних споживачів, їхніх потреб та очікувань, які відображені **у** таблиці 7.14.

Таблиця 7.14 – Вибір цільових груп потенційних споживачів

№ п/п	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
1	Туристи та мандрівники	Висока	Високий у туристичних містах	Висока (Booking.com, Airbnb)	Середня

№ п/п	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
2	Студенти та молоді фахівці	Висока	Стабільний, піковий восени	Середня (OLX, Flatfy.ua)	Висока
3	Власники житла та агентства	Висока	Високий для управління орендою	Середня (локальні платформи)	Висока
Обрана цільова група – Власникам житла					

Як видно з таблиці перевага була надана категорії власників житла. Хоча сегмент туристів демонструє високий попит, саме власники житла та агентства є найбільш пріоритетною категорією через стабільний попит, меншу конкуренцію та простіший вихід на ринок. Далі сформуємо базовоу стратегію розвитку на ринку. Результат наведено в таблиці 7.15.

Таблиця 7.15 – Визначення базової стратегії розвитку

№ п/п	Обрана альтернатива розвитку проекту	Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку
1	Запуск платформи з базовим функціоналом	Масове охоплення цільової аудиторії	Гнучка підтримка коротко- та довгострокової оренди, Зручний інтерфейс, Інтеграція платіжних систем	Стратегія диференціації

Далі потрібно визначити стратегію конкурентної поведінки. Опис стратегії наведений в таблиці 7.16.

Таблиця 7.16 – Визначення базової стратегії конкурентної поведінки

№ п/п	Чи є проект «першопрохідцем» на ринку?	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, і які?	Стратегія конкурентної поведінки
1	Ні, ринок вже зайнятий конкурентами.	Залучення нових клієнтів через унікальні функції.	Так, часткове копіювання пошуку та бронювання	Диференціація через унікальні можливості.

Тепер визначимо стратегію позиціонування згідно якої проект буде просуватися на ринку. Стратегія наведена в таблиці 7.17.

Таблиця 7.17 – Визначення стратегії позиціонування

№ п/п	Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проекту	Вибір асоціацій, які мають сформувати комплексну позицію власного проекту (три ключових)
1	Зручний пошук, прозоре бронювання, безпечні платежі	Стратегія диференціації	Рекомендаційна система, Гнучка підтримка оренди, Локалізація платформи	Персоналізований підбір житла, Зручність та безпека, Орієнтація на локальні потреби

7.5 Розробка маркетингової програми стартап-проекту

Для успішного просування проекту на ринку необхідно розробити маркетингову стратегію. Для цього спочатку підсумуємо результати попереднього аналізу конкурентноспроможності продукту, які наведені в таблиці 7.18.

Таблиця 7.18 – Визначення ключових переваг концепції потенційного товару

№ п/п	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити
1	Зручний пошук та бронювання житла	Швидкий та персоналізований підбір житла	- Рекомендаційна система для індивідуальних пропозицій
2	Безпечні та зручні платежі	Швидкі та прозорі онлайн-транзакції	Інтеграція з платіжними системами (Stripe, LiqPay), Підтримка локальних валют
3	Гнучкість умов оренди	Можливість обирати між коротко- та довгостроковою орендою	Гнучка підтримка обох типів оренди, на відміну від конкурентів, що обмежені короткостроковою орендою
4	Потреба у локалізованих сервісах	Орієнтація на місцевий ринок	Локалізація платформи (мовна підтримка, регіональні особливості)
5	Зручність користування	Інтуїтивно зрозумілий та сучасний інтерфейс	Сучасний UI/UX, що спрощує навігацію та процес бронювання

Висновки до розідлу 7

У цьому розділі було розглянуто основні етапи розробки стартап-проєкту цифрової платформи для оренди житла на основі мікросервісної архітектури. Проведено аналіз ідеї проєкту, технологічний аудит, а також оцінено ринкові можливості та конкурентне середовище. Запропонована стратегія розвитку та унікальні функціональні можливості платформи, такі як зручність пошуку та бронювання, інтеграція з платіжними системами та підтримка коротко- та довгострокової оренди, демонструють потенціал для успішного впровадження. Результати аналізу підтвердили актуальність стартапу на ринку та створили основу для подальшої оптимізації, тестування й маркетингового просування платформи.

ВИСНОВКИ

У магістерській роботі досліджено проблему розробки сучасної інформаційної системи для оренди житла на основі мікросервісної архітектури. У результаті проведеного аналізу предметної області, визначення вимог до системи та реалізації її компонентів досягнуто поставленої мети – створено масштабовану, надійну та безпечну платформу для автоматизації процесів оренди житла.

Основні досягнення роботи:

Проведено аналіз сучасного ринку оренди житла та огляд існуючих рішень, що дозволило визначити ключові вимоги до системи.

а) запроектовано архітектуру системи з використанням принципів мікросервісної архітектури, яка забезпечує розподілену роботу сервісів, масштабованість та гнучкість у подальшому розвитку;

б) реалізовано основні функціональні модулі системи, такі як управління користувачами, бронювання, інтеграція з платіжними сервісами та система відгуків;

в) виконано тестування функціоналу та продуктивності системи, яке підтвердило її відповідність вимогам та стабільну роботу при високих навантаженнях;

г) розроблено рекомендації щодо подальшого розвитку системи, включаючи впровадження нових функцій та розширення алгоритмів рекомендацій.

Практична цінність роботи полягає у створенні платформи, яка дозволяє автоматизувати процеси оренди житла, знижувати витрати на адміністрування та покращувати досвід взаємодії користувачів. Запропоновані рішення можуть бути інтегровані в реальні бізнес-процеси у сфері оренди житла.

Результати дослідження та розробки підтверджують, що обрані методи і технології є ефективними та можуть бути використані для створення аналогічних систем у інших галузях.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Оренда житлових приміщень. URL:http://surl.li/husjfb (дата звернення: 10.09.2024).

2. Booking.com. URL: https://www.booking.com (дата звернення: 11.09.2024).

3. Flatify. URL:https://flatfy.ua/uk/ (дата звернення: 11.09.2024).

4. Rieltor.ua. URL: https://rieltor.ua/ (дата звернення: 11.09.2024).

5. What Is Risk Matrix. URL: https://blog.falcony.io/en/what-is-risk-matrix (дата звернення: 01.10.2024).

6. Мова моделювання архітектури підприємств Archimate 3.2 . URL: https://pubs.opengroup.org/architecture/archimate3-doc/ (дата звернення: 05.10.2024).

7. What is Layers and Aspects in ArchiMate? URL:https://archimate.visual-paradigm.com/what-is-layers-and-aspects-in-archimate-core-framework/ (дата звернення: 10.10.2024)

8. Microservice Architecture pattern. URL:https://microservices.io/patterns/microservices.html (дата звернення: 15.10.2024)

9. Кластеризація методом к–середніх (k-means). URL : https://uk.wikipedia.org/wiki (дата звернення: 20.10.2024)

10. Asp Net Core Identity. URL: https://learn.microsoft.com/en-us/aspnet/core/security/authentication/identity?view=aspnetcore-9.0&tabs=visual-studio (дата звернення: 25.10.2024)

11. GitHub Repository: Ocelot API Gateway. URL: https://github.com/ThreeMammals/Ocelot (дата звернення: 25.10.2024)

12. JSON Web Token (JWT). URL: https://jwt.io/ (дата звернення: 25.10.2024)

13. Точність в класифікації. URL: https://uk.wikipedia.org/wiki (дата звернення: 15.09.2024)

14. Повнота в класифікаці. URL:https://builtin.com/data-science/precision-and-recall (дата звернення: 15.09.2024)

15. F-measure . URL : https://nlp.stanford.edu/IR-book/html/htmledition/evaluation-of-ranked-retrieval-results-1.html (дата звернення: 16.09.2024)