

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

ПОЯСНЮВАЛЬНА ЗАПИСКА

до кваліфікаційної роботи освітнього ступеня «бакалавр»
за спеціальністю 121 «Інженерія програмного забезпечення»
(освітня програма «Інженерія програмного забезпечення»)
на тему:

«Маркетплейс оренди житла в готелях»

Виконала студентка групи ІПЗ-22-4
ПАРОВА Вікторія Віталіївна

Керівник роботи:
ЄФРЕМОВ Юрій Миколайович

Рецензент:
ВАКАЛЮК Тетяна Анатоліївна

Житомир - 2026

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

«ЗАТВЕРДЖУЮ»

Зав. кафедри інженерії
програмного забезпечення
Тетяна Вакалюк
«27» лютого 2026 р.

ЗАВДАННЯ
на кваліфікаційну роботу

Здобувач вищої освіти: **ПАРОВА Вікторія Віталіївна**

Керівник роботи: **ЄФРЕМОВ Юрій Миколайович**

Тема роботи: **Маркетплейс оренди житла в готелях**

затверджена Наказом закладу вищої освіти від **«27» лютого 2026 р., №92/с**

Вихідні дані для роботи: **процес організації оренди житла в готелях та управління бронюваннями.**

Консультанти з бакалаврської кваліфікаційної роботи із зазначенням розділів, що їх стосуються:

Розділ	Консультант	Завдання видав	Завдання прийняв
1	Єфремов Ю.М.	28.02.2026р.	28.02.2026р.
2	Єфремов Ю.М.	28.02.2026р.	28.02.2026р.
3	Єфремов Ю.М.	28.02.2026р.	28.02.2026р.

РЕФЕРАТ

Випускна кваліфікаційна робота бакалавра складається з маркетплейсу оренди житла в готелях та пояснювальної записки. Пояснювальна записка до випускної роботи містить текстову частину, викладену на 78 сторінках друкованого тексту. Список використаних джерел містить 30 найменувань і займає 3 сторінки. У роботі наведено 41 рисунок та 29 таблиць. Також у роботі є 19 сторінок додатків.

Головною метою роботи є розробка сучасного маркетплейсу для оренди житла в готелях, який забезпечує користувачам зручний пошук і бронювання номерів, а також надає інструменти для ефективного управління готельними ресурсами із застосуванням сучасних підходів до проєктування та розробки програмного забезпечення.

У ході виконання роботи було проведено аналіз напрямків використання інформаційних технологій у сфері бронювання та оренди житла, сформульовано постановку задачі та досліджено існуючі програмні рішення й сервіси бронювання готелів. На основі проведеного аналізу було визначено вимоги до системи та обрано архітектурні рішення, що найбільш повно відповідають поставленим цілям і функціональним потребам проєкту.

У процесі проєктування було визначено функціональні та нефункціональні вимоги до системи, побудовано діаграми варіантів використання та класів, розроблено структуру бази даних і спроектовано алгоритми бронювання з перевіркою доступності номерів та підтримкою умов скасування бронювань. Також реалізовано серверну та клієнтську частини системи, включаючи функціонал керування готелями, номерами, користувачами, відгуками та систему сповіщень електронною поштою.

У результаті було розроблено програмне рішення з адаптивним інтерфейсом, яке надає користувачам можливість здійснювати пошук,

					ІПЗ.КР.Б-121-26-ПЗ				
Змн.	Арк.	№ докум.	Підпис	Дата					
Розроб.		Парова В.В.			Маркетплейс оренди житла в готелях Пояснювальна записка		Літ.	Арк.	Аркушів
Керівник		Єфремов Ю.М.						3	98
							Житомирська політехніка, група ІПЗ-22-4		
Н. контр.		Тичина О.П.							
Зав. каф.		Вакалюк Т.А.							

бронювання та управління житлом у готелях, а також оптимізує процес взаємодії між клієнтами, адміністраторами та власниками готелів. Результати тестування підтвердили коректність роботи реалізованого функціоналу та відповідність системи поставленим вимогам.

КЛЮЧОВІ СЛОВА: МАРКЕТПЛЕЙС, ОРЕНДА ЖИТЛА, ГОТЕЛІ, БРОНЮВАННЯ.

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		4

ABSTRACT

The bachelor's qualification work consists of a marketplace for hotel accommodation rentals and an explanatory note. The explanatory note contains the textual part presented on 78 pages of printed text. The list of references includes 30 sources and takes up 3 pages. The work contains 41 figures and 29 tables. There are also 19 pages of appendices.

The main goal of the work is to develop a modern marketplace for hotel accommodation, which provides users with convenient search and booking of rooms, as well as tools for effective management of hotel resources using modern approaches to software design and development.

During the work, an analysis of the areas of use of information technologies in the field of accommodation booking and rental was conducted, a problem statement was formulated and existing software solutions and hotel booking services were studied. Based on the analysis, system requirements were determined and architectural solutions were selected that most fully meet the goals and functional needs of the project.

During the design process, functional and non-functional requirements for the system were determined, use case and class diagrams were constructed, a database structure was developed, and booking algorithms were designed with room availability checking and support for cancellation conditions. The server and client parts of the system were also implemented, including the functionality of managing hotels, rooms, users, reviews and an e-mail notification system.

As a result, a software solution with an adaptive interface was developed, which provides users with the opportunity to search, book and manage accommodation in hotels, and also optimizes the process of interaction between clients, administrators and hotel owners. The testing results confirmed the correctness of the implemented functionality and the compliance of the system with the requirements.

KEYWORDS: MARKETPLACE, ACCOMMODATION RENTAL, HOTELS, BOOKING.

		Парова В.В.			ІІЗ.КР.Б-121-26-ІІЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		5

ЗМІСТ

РЕФЕРАТ	3
ABSTRACT	5
ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ.....	7
ВСТУП	8
РОЗДІЛ 1 АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ РЕАЛІЗАЦІЇ МАРКЕТПЛЕЙСУ ОРЕНДИ ЖИТЛА В ГОТЕЛЯХ.....	10
1.1 Постановка задачі.....	10
1.2 Аналіз аналогів програмного продукту	13
1.3 Вибір архітектури системи маркетплейсу оренди житла	17
1.4 Обґрунтування вибору інструментальних засобів та вимоги до апаратного забезпечення	21
Висновки до першого розділу.....	25
РОЗДІЛ 2 ПРОЄКТУВАННЯ СИСТЕМИ.....	27
2.1 Визначення варіантів використання та об'єктно-орієнтованої структури системи	27
2.2 Розробка бази даних системи.....	39
2.3 Проєктування та реалізація алгоритмів системи	46
Висновки до другого розділу	51
РОЗДІЛ 3 ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ ІЗ СИСТЕМОЮ	53
3.1 Порядок встановлення та налаштування параметрів системи	53
3.2 Структура інтерфейсу платформи.....	57
3.3 Тестування програмного системи	68
Висновки до третього розділу.....	75
ВИСНОВКИ.....	77
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	78
ДОДАТКИ.....	81

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

HTML - HyperText Markup Language

CSS - Cascading Style Sheets

Django - Вебфреймворк на мові програмування Python

MySQL - Система управління реляційними базами даних

SQL - Structured Query Language

ORM - Object-Relational Mapping

API - Application Programming Interface

HTTP - HyperText Transfer Protocol

HTTPS - HyperText Transfer Protocol Secure

MVC - Model-View-Controller

URL - Uniform Resource Locator

JSON - JavaScript Object Notation

AJAX - Asynchronous JavaScript and XML

CRUD - Create, Read, Update, Delete

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		7

ВСТУП

Актуальність теми. У сучасному світі цифрові технології стрімко змінюють підходи до організації послуг, зокрема у сфері туризму та готельного бізнесу. Зростання попиту на онлайнсервіси бронювання житла, розвиток електронної комерції та підвищення вимог користувачів до зручності й швидкості обслуговування зумовлюють необхідність створення ефективних інформаційних систем. Традиційні способи бронювання номерів поступово втрачають актуальність, поступаючись автоматизованим платформам, які забезпечують доступ до інформації в режимі реального часу.

Маркетплейс оренди житла в готелях є комплексним програмним рішенням, що дозволяє користувачам здійснювати пошук, перегляд та бронювання номерів, а також керувати власними бронюваннями. Така система забезпечує взаємодію між клієнтами, адміністраторами та власниками готелів, надає можливість автоматизувати процеси управління готелями та підвищити ефективність обслуговування. Крім того, впровадження подібної системи сприяє підвищенню прозорості ринку, покращенню якості сервісу та оптимізації бізнес-процесів у готельній сфері. Отже, розробка маркетплейсу оренди житла є актуальним завданням.

Метою випускної кваліфікаційної роботи бакалавра є розробка системи у вигляді маркетплейсу для оренди житла в готелях.

Завданнями на кваліфікаційну роботу є:

- аналіз існуючих аналогів маркетплейсів для оренди житла в готелях; вибір технологій та засобів розробки;
- проєктування архітектури системи, структури бази даних та алгоритмів її функціонування;
- реалізація механізмів реєстрації та авторизації користувачів, керування готелями, номерами та бронюваннями;
- підтримка ролей адміністратора і власника готелю;

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		8

- реалізація пошуку, фільтрації, системи відгуків та сповіщень користувачів;
- розробка користувацького інтерфейсу системи;
- проведення тестування та оцінка результатів роботи розробленого програмного забезпечення.

Об'єктом дослідження є процес організації оренди житла в готелях та управління бронюваннями.

Предметом дослідження є застосування сучасних інформаційних технологій для розробки системи маркетплейсу оренди житла, зокрема використання серверних фреймворків, баз даних та клієнтських технологій для створення ефективного програмного рішення.

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		9

РОЗДІЛ 1 АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ РЕАЛІЗАЦІЇ МАРКЕТПЛЕЙСУ ОРЕНДИ ЖИТЛА В ГОТЕЛЯХ

1.1 Постановка задачі

Метою розробки є створення маркетплейсу оренди житла в готелях, який дозволить користувачам шукати, переглядати та бронювати номери через мережу Інтернет [18]. Система повинна забезпечувати зручну взаємодію між клієнтами, власниками готелів та адміністраторами, а також спрощувати процес керування інформацією про готелі, номери та бронювання.

У сучасному світі онлайнсервіси стали основним способом пошуку житла для подорожей, відряджень та відпочинку. Користувачі очікують швидкого доступу до інформації про доступні варіанти проживання, можливості порівнювати пропозиції та здійснювати бронювання в декілька кліків.

Водночас власники готелів потребують інструментів, які дозволяють ефективно керувати номерним фондом, контролювати бронювання та підтримувати актуальність інформації про свої послуги. Тому розробка спеціалізованої інформаційної системи є актуальним завданням, що сприятиме підвищенню якості обслуговування клієнтів та автоматизації бізнес-процесів у готельній сфері.

Основною перевагою запропонованої системи є централізоване зберігання інформації про готелі та номери, що забезпечить користувачам доступ до актуальних даних у будь-який час. Завдяки цьому клієнти зможуть швидко знаходити потрібне житло відповідно до власних вимог, а власники готелів отримають додатковий канал для залучення нових клієнтів та просування своїх послуг.

Для досягнення поставленої мети необхідно виконати ряд взаємопов'язаних завдань.

Першим етапом є аналіз функціональних вимог до системи. Для цього потрібно визначити потреби основних користувачів: клієнтів, власників го-

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		10

телів та адміністраторів. На основі проведеного аналізу необхідно сформува-
ти перелік функцій системи. До них належать реєстрація та авторизація ко-
ристувачів, пошук і фільтрація готелів за різними параметрами, перегляд ін-
формації про номери, бронювання житла, скасування бронювань, залишення
відгуків та оцінок, отримання сповіщень, а також керування інформацією про
готелі та користувачів [13].

Особливу увагу необхідно приділити механізму пошуку та фільтрації.
Користувачі повинні мати можливість знаходити готелі за місцем розташу-
вання, вартістю проживання, категорією готелю, рейтингом, доступністю
номерів та іншими характеристиками. Це дозволить значно скоротити час на
пошук потрібного варіанту проживання та зробить роботу із системою більш
зручною.

Наступним завданням є вибір технологій для реалізації системи. Для
створення користувацького інтерфейсу планується використовувати HTML
та CSS, які забезпечують формування структури сторінок та їх оформлення.
Серверна частина буде реалізована за допомогою фреймворку Django [1],
який містить вбудовані засоби для роботи з користувачами, маршрутизації
запитів та забезпечення безпеки вебзастосунків. Для зберігання інформації
буде використовуватися система управління базами даних MySQL [3], яка
дозволяє надійно працювати з великими обсягами даних та підтримує вико-
нання складних запитів.

Важливим етапом є проєктування архітектури системи. Для опису
структури та логіки роботи системи будуть використані UML-діаграми, зо-
крема діаграми прецедентів, класів та послідовностей. Це дозволить
візуалізувати взаємодію користувачів із системою, визначити основні сут-
ності та зв'язки між ними, а також спростити процес подальшої реалізації
програмного забезпечення. Грамотно спроектована архітектура забезпечить
можливість розширення функціоналу системи без значних змін у вже ре-
алізованих компонентах.

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		11

Після проєктування необхідно реалізувати серверну та клієнтську частини системи. Для роботи з базою даних буде використано Django ORM [10], що дозволить виконувати операції з даними без написання великої кількості SQL-запитів. Також потрібно створити вебінтерфейс користувача та забезпечити взаємодію всіх компонентів системи через програмні інтерфейси та механізми обробки запитів [28].

Основний функціонал системи повинен включати реєстрацію та авторизацію користувачів, керування готелями та номерами, бронювання житла з перевіркою доступності номерів на обрані дати, можливість скасування бронювання, особистий кабінет користувача з історією замовлень, панель керування для власників готелів, адміністративну панель, систему відгуків та оцінювання, а також механізм надсилання сповіщень. Для підвищення довіри до сервісу система повинна надавати користувачам можливість залишати коментарі та оцінки після завершення проживання, що допоможе іншим клієнтам приймати рішення під час вибору житла.

Окремо необхідно реалізувати механізми захисту даних користувачів. Система повинна забезпечувати безпечне зберігання облікових даних, захист від несанкціонованого доступу та контроль прав користувачів відповідно до їх ролей. Це дозволить гарантувати конфіденційність інформації та стабільну роботу програмного продукту.

Завершальним етапом є тестування та оптимізація розробленої системи. Необхідно перевірити правильність роботи всіх функцій, створити модульні тести та усунути виявлені помилки. Також потрібно перевірити роботу системи під навантаженням та оцінити швидкість виконання основних операцій. Для підвищення продуктивності необхідно оптимізувати роботу із базою даних, використовувати кешування та інші методи оптимізації [30].

У результаті виконання поставлених завдань має бути створена сучасна веборієнтована система бронювання готельних номерів, яка забезпечить швидкий пошук житла, зручне бронювання та ефективне управління інформацією про готелі. Використання такої системи сприятиме покращенню

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		12

якості надання готельних послуг, підвищенню рівня задоволеності користувачів та автоматизації процесів управління у сфері гостинності..

1.2 Аналіз аналогів програмного продукту

На сьогодні у сфері туризму та готельного бізнесу онлайн-платформи бронювання житла є ключовим інструментом для організації взаємодії між клієнтами та постачальниками послуг. З метою створення конкурентоздатної системи маркетплейсу оренди житла важливим етапом є аналіз існуючих аналогів.

Аналіз подібних платформ дозволяє визначити основні функціональні можливості, такі як пошук і фільтрація житла, система бронювання, управління об'єктами розміщення, система відгуків та рейтингів, а також механізми сповіщень [16]. Детальне порівняння існуючих рішень допомагає виявити їх сильні та слабкі сторони, що в подальшому впливає на проєктування власної системи.

«Booking.com» - одна з найпопулярніших платформ для бронювання житла, що надає широкий вибір готелів, апартаментів, хостелів та інших варіантів розміщення по всьому світу (Рис. 1.1). Система орієнтована як на кінцевих користувачів, так і на власників готелів, забезпечуючи повний цикл взаємодії між ними. Платформа дозволяє здійснювати детальний пошук за великою кількістю параметрів, включаючи місцезнаходження, ціну, рейтинг, тип житла, зручності та доступність номерів у конкретні дати.

Сервіс підтримує миттєве бронювання без необхідності попереднього підтвердження з боку готелю (для більшості об'єктів), а також надає користувачам можливість перегляду відгуків, оцінок і фотографій від інших клієнтів (Рис. 1.2). Для власників готелів доступна окрема панель керування, що дозволяє управляти цінами, доступністю номерів, бронюваннями та статистикою.

Платформа має низку переваг, серед яких надзвичайно широкий вибір об'єктів розміщення в різних цінових категоріях, розвинена система пошуку

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				13
Змн.	Арк.	№ докум.	Підпис	Дата		

та багаторівневої фільтрації, велика база відгуків, що підвищує довіру користувачів, автоматизована система бронювання та підтвердження, а також зручні інструменти для управління об'єктами для власників готелів. Водночас система має і певні недоліки, зокрема високу конкуренцію між готелями, що ускладнює просування нових об'єктів, складний інтерфейс через велику кількість функцій, комісійну модель, яка може бути не вигідною для власників, а також інколи надлишок інформації, що ускладнює прийняття рішення користувачем.

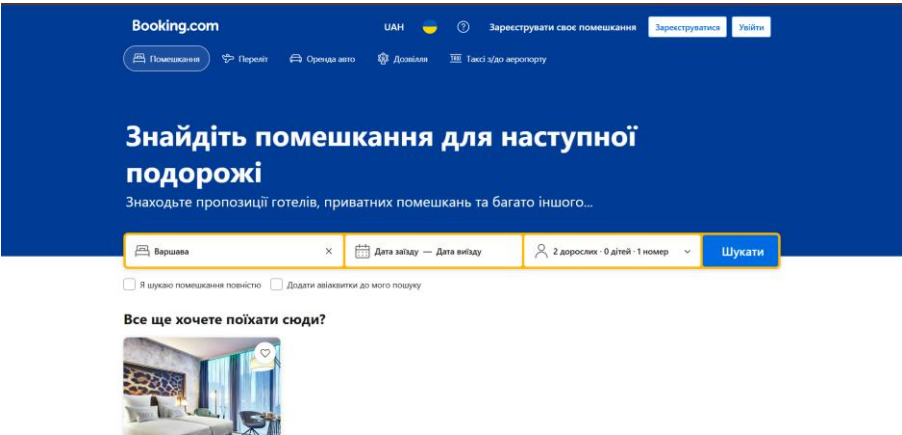


Рис. 1.1. - Інтерфейс головної сторінки сайту «Booking.com»

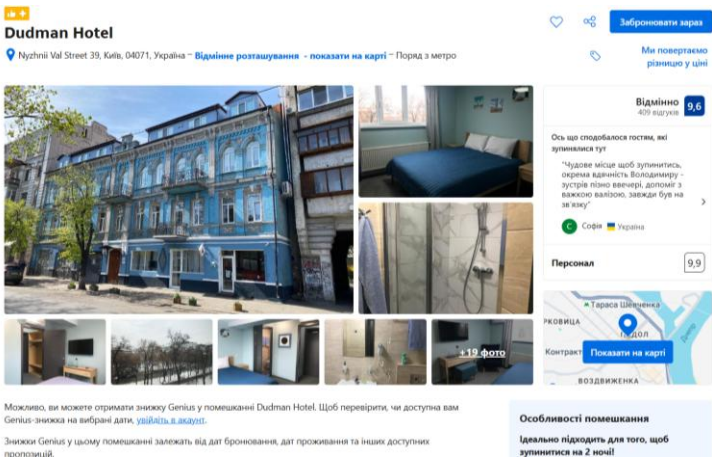


Рис. 1.2. - Інтерфейс сторінки готелю Dudman Hotel в «Booking.com»

«Airbnb» - платформа для оренди житла, яка спеціалізується на короткостроковій оренді квартир, будинків та унікальних варіантів проживання (Рис. 1.3). Основною особливістю сервісу є можливість прямої

взаємодії між орендодавцями та орендарями без посередників у класичному розумінні готельного бізнесу.

Платформа надає користувачам можливість створювати власні пропозиції, встановлювати правила проживання, ціни, а також спілкуватися з потенційними клієнтами через внутрішню систему повідомлень. Значна увага приділяється персоналізації сервісу, включаючи рекомендації, відгуки та рейтинги.

Платформа має низку переваг, серед яких велика кількість унікальних і нестандартних варіантів житла, гнучкі умови бронювання, які визначаються безпосередньо власником, розвинена система рейтингів і відгуків, що формує довіру між користувачами, можливість прямої комунікації між гостем і господарем, а також персоналізовані рекомендації на основі попереднього досвіду. Водночас система має і недоліки: нестабільну якість житла через відсутність єдиних стандартів, ризики безпеки та надійності угод, складність контролю якості сервісу, можливі скасування бронювань з боку власників і значну залежність від людського фактору.

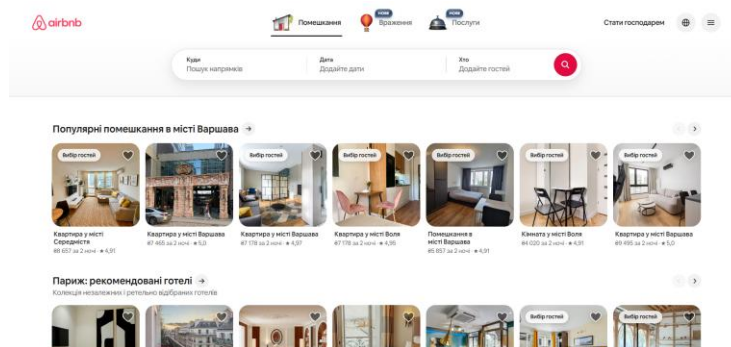


Рис. 1.3. - Інтерфейс головної сторінки сайту «Airbnb»

«Expedia» - онлайнсервіс для бронювання туристичних послуг, який пропонує не лише готелі, а й авіаквитки, оренду автомобілів та туристичні пакети (Рис. 1.4). Платформа орієнтована на користувачів, які планують комплексні подорожі та хочуть отримати всі послуги в одному місці.

Сервіс дозволяє бронювати готелі разом із транспортом, що спрощує організацію поїздок. Крім того, Expedia пропонує різноманітні акції, знижки

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				15
Змн.	Арк.	№ докум.	Підпис	Дата		

та бонусні програми для постійних клієнтів. Інтерфейс включає велику кількість функцій, що дозволяють гнучко налаштовувати параметри пошуку.

Платформа має переваги у вигляді інтеграції різних туристичних послуг в одному сервісі, можливості пакетного бронювання, наявності спеціальних пропозицій і знижок, розширеного функціоналу для планування подорожей та бонусних програм для користувачів. Водночас система має недоліки: перевантажений інтерфейс через велику кількість функцій, складну навігацію для нових користувачів, можливі неточності або затримки в оновленні інформації, незручність для швидкого бронювання одного об'єкта та надлишок додаткових опцій, що ускладнює користування.

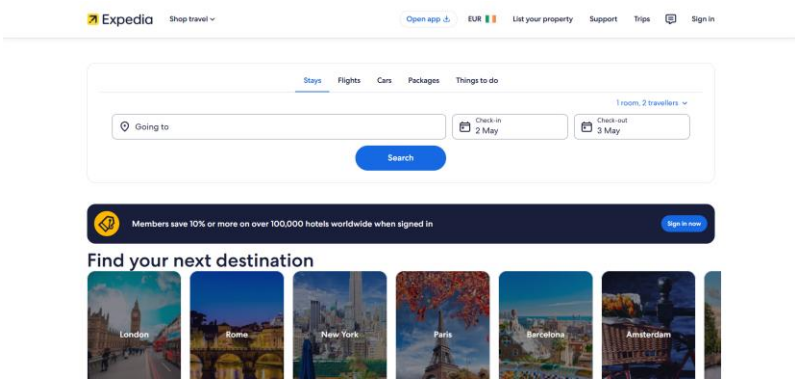


Рис. 1.4. - Інтерфейс головної сторінки сайту «Expedia»

Після аналізу платформ Booking.com, Airbnb та Expedia було визначено ключові аспекти та проведено порівняння (Табл. 1.1), які необхідно врахувати при розробці власного маркетплейсу. Основними з них є реалізація ефективної системи пошуку та фільтрації, зручного механізму бронювання, можливість залишення відгуків, а також створення зрозумілого інтерфейсу для різних категорій користувачів. Окрім цього, важливим є забезпечення функціоналу для адміністрування системи та управління готелями і номерами.

Таблиця 1.1

Порівняння аналогів

Характеристика	Booking.com	Airbnb	Expedia
Пошук і фільтрація	+	+	+

Система бронювання	+	+	+
Система відгуків	+	+	+
Сповіщення користувачів	+	±	+
Зручність інтерфейсу	±	+	±
Управління об'єктами	+	+	+

1.3 Вибір архітектури системи маркетплейсу оренди житла

Вибір архітектури для системи маркетплейсу оренди житла є одним із ключових етапів проєктування, оскільки саме він визначає здатність системи до масштабування, продуктивність, зручність підтримки та подальшого розвитку. Оскільки система передбачає роботу з різними типами користувачів (анонімні користувачі, авторизовані клієнти, власники житла та адміністратори), а також обробку транзакцій бронювання, відгуків та модерації, архітектура повинна забезпечувати баланс між гнучкістю та складністю реалізації [25]. У межах даної роботи розглядаються три основні підходи: монолітна архітектура, мікросервісна архітектура та сервісно-орієнтована архітектура (SOA).

Монолітна архітектура (Рис. 1.5) передбачає побудову системи як єдиного застосунку [17], у якому всі функціональні модулі (керування користувачами, готелями, бронюваннями, відгуками, адміністрування) реалізовані в межах одного середовища виконання та використовують спільну базу даних.

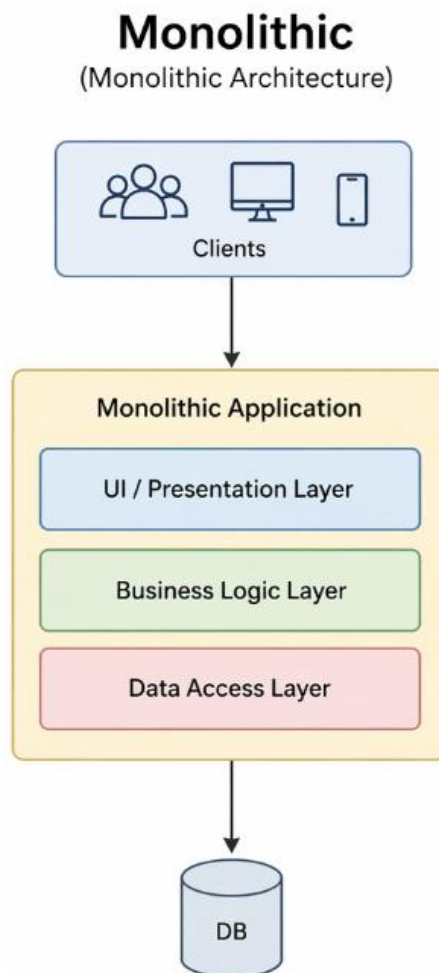


Рис. 1.5. - Діаграма монолітної архітектури

Монолітна архітектура характеризується єдиною кодовою базою, централізованою бізнес-логікою, спільною базою даних, тісною інтеграцією всіх компонентів та синхронною взаємодією між модулями[26]. Її основними перевагами є простота розробки на початковому етапі, легкість розгортання та налагодження, відсутність складної міжсервісної взаємодії і можливість швидкої реалізації базового функціоналу. Водночас зі зростанням системи виникають труднощі з підтримкою та масштабуванням окремих модулів, підвищується зв'язаність компонентів, збільшується ризик впливу помилки в одному модулі на роботу всієї системи та зменшується гнучкість під час впровадження нових функціональних можливостей.

Цей вид архітектури підходить для невеликих або середніх проєктів, однак у випадку масштабних систем може створювати обмеження для розвитку.

Сервісно-орієнтована архітектура (SOA) (Рис. 1.6) передбачає розподіл системи на окремі сервіси, кожен з яких виконує конкретну бізнес-функцію [17]. Взаємодія між сервісами здійснюється через стандартизовані протоколи.

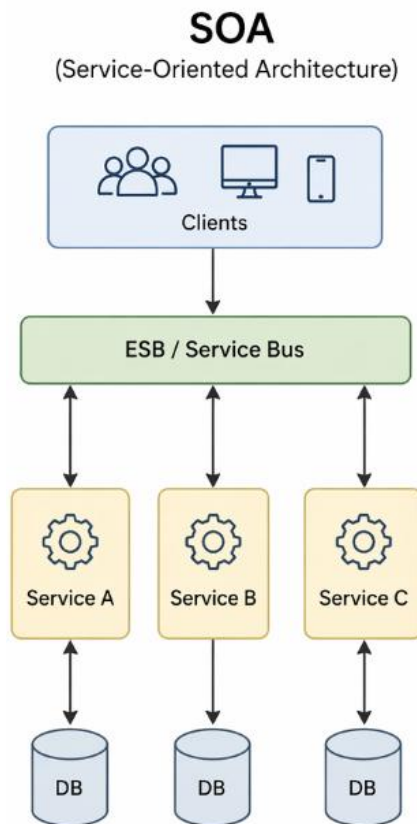


Рис. 1.6. - Діаграма сервісно-орієнтованої архітектури

Сервіс-орієнтована архітектура (SOA) базується на використанні набору незалежних сервісів, що взаємодіють через інтеграційний шар за допомогою стандартизованих протоколів. Такий підхід забезпечує модульність системи, можливість повторного використання функціоналу, гнучку інтеграцію з іншими системами та підтримку різних технологічних рішень. Водночас SOA має певні недоліки, серед яких складність конфігурації та тестування, необхідність використання додаткових механізмів інтеграції, підвищені витрати на підтримку та надлишковість для систем середнього масштабу.

У контексті маркетплейсу оренди житла SOA може бути надмірно складною, оскільки потребує значних ресурсів для реалізації інтеграційного шару та управління сервісами.

Мікросервісна архітектура (Рис. 1.7) передбачає поділ системи на незалежні сервіси [17], кожен з яких відповідає за окрему бізнес-функцію та може розгортатися автономно.

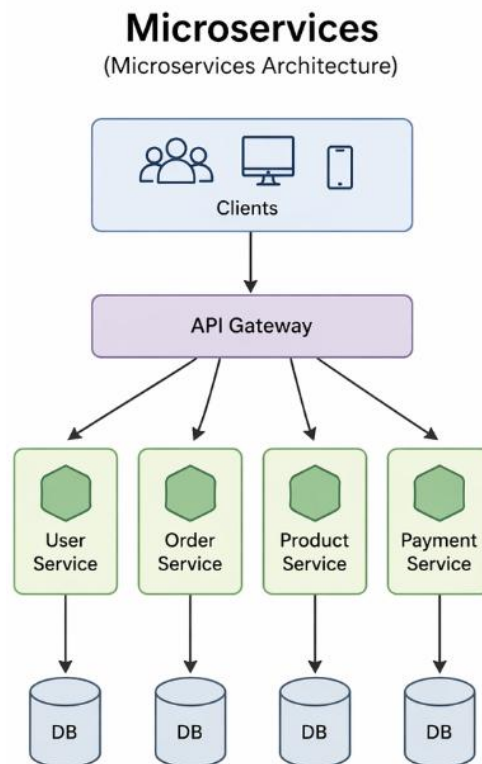


Рис 1.7. - Діаграма мікросервісної архітектури

Такий підхід забезпечує високу масштабованість, гнучкість розвитку системи, ізоляцію відмов, можливість незалежного розгортання сервісів і ефективну роботу при високих навантаженнях. Разом із тим мікросервісна архітектура характеризується складністю проєктування, необхідністю організації міжсервісної взаємодії, ускладненим тестуванням і моніторингом, потребою в додатковій інфраструктурі та вищими витратами на підтримку системи.

Для даного проєкту використання повноцінної мікросервісної архітектури є передчасним, оскільки система не потребує складної розподіленої обробки або незалежного масштабування великої кількості сервісів.

З урахуванням особливостей системи маркетплейсу оренди житла було обрано монолітну архітектуру з модульною організацією (modular monolith)

[17], оскільки вона найкраще відповідає масштабу та складності проєкту. Система охоплює обмежений набір доменів, таких як користувачі, готелі, номери, бронювання, відгуки та модерація, які доцільно реалізовувати в межах одного застосунку без розподілення на окремі сервіси. Це дозволяє спростити розробку та підтримку, оскільки єдина кодова база зменшує складність взаємодії між компонентами та полегшує тестування.

Попри монолітну структуру, система логічно поділена на окремі модулі: автентифікації та користувачів, готелів, номерів, бронювань, відгуків та адміністрування. Такий підхід забезпечує чіткий розподіл відповідальностей [17,25] і наближає архітектуру до мікросервісного стилю без його ускладнень. Додатковою перевагою є використання централізованої бази даних, що виключає проблеми узгодженості даних між сервісами, характерні для розподілених систем.

Важливою перевагою також є продуктивність, оскільки взаємодія між модулями відбувається без мережових викликів, що зменшує затримки та підвищує швидкодію системи. Крім того, така архітектура забезпечує можливість подальшого розвитку: у разі зростання навантаження окремі модулі можуть бути винесені в мікросервіси без повного перепроєктування системи.

Таким чином, монолітна модульна архітектура є оптимальним рішенням для даної системи, оскільки забезпечує простоту реалізації, зручність підтримки, достатню продуктивність і потенціал для подальшого масштабування, уникаючи при цьому надмірної складності мікросервісних і сервісно-орієнтованих підходів.

1.4 Обґрунтування вибору інструментальних засобів та вимоги до апаратного забезпечення

Для розробки системи маркетплейсу оренди житла необхідно обрати програмні засоби, які забезпечать надійну роботу системи, зручність розробки та можливість подальшого масштабування. Вибір технологій

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		21

здійснювався з урахуванням особливостей предметної області, функціональних вимог до системи та необхідності забезпечення безпеки і продуктивності.

У якості системи управління базами даних було обрано реляційну СУБД MySQL. Такий вибір зумовлений високою продуктивністю системи під час виконання транзакційних операцій, зокрема створення бронювань, обробки інформації про користувачів та оновлення даних про доступність номерів. Реляційна модель даних добре підходить для предметної області готельного бізнесу, оскільки дозволяє ефективно організувати зв'язки між користувачами, готелями, номерами, бронюваннями та відгуками.

Однією з важливих переваг MySQL є підтримка механізмів забезпечення цілісності даних [3]. Використання зовнішніх ключів дозволяє контролювати зв'язки між таблицями та запобігати виникненню помилок у базі даних. Підтримка транзакцій забезпечує правильне виконання критично важливих операцій, наприклад під час створення бронювання, коли необхідно одночасно змінити декілька записів у базі даних. Також MySQL має високу швидкодію, стабільність роботи та широку підтримку з боку спільноти розробників, що робить її одним із найпоширеніших рішень для вебзастосунків.

У розроблюваній системі MySQL використовується для зберігання інформації про користувачів, готелі, номери, бронювання, відгуки та оцінки. Крім того, база даних забезпечує швидкий пошук інформації, сортування та фільтрацію номерів за різними параметрами, що є важливим для комфортної роботи користувачів із системою.

Для реалізації серверної частини застосунку використовується фреймворк Django, який є одним із найпопулярніших фреймворків для розробки вебзастосунків мовою Python. Django реалізує архітектурний підхід Model-View-Template (MVT), що є адаптацією архітектури MVC та дозволяє розділити логіку роботи системи на окремі компоненти. Це спрощує розробку, тестування та подальшу підтримку програмного забезпечення.

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		22

Однією з головних переваг Django є наявність великої кількості готових інструментів, які значно скорочують час розробки. До них належать система автентифікації користувачів, адміністративна панель, механізми роботи з формами та вбудована ORM [10]. Завдяки цьому розробнику не потрібно створювати багато базових компонентів з нуля, що дозволяє зосередитися на реалізації бізнес-логіки системи.

Важливою перевагою фреймворку є його безпека. Django містить вбудовані механізми захисту від поширених вебзагроз, зокрема CSRF-атак, XSS-атак та SQL-ін'єкцій [19][29]. Також фреймворк забезпечує перевірку введених даних через систему форм і механізми валідації [11], що знижує ризик виникнення помилок та підвищує надійність роботи застосунку.

За допомогою Django реалізуються основні функції системи, серед яких реєстрація та авторизація користувачів [9], керування профілями, створення та редагування інформації про готелі й номери, бронювання житла, система відгуків та оцінювання, а також адміністративна панель для контролю роботи всієї платформи.

Для взаємодії з базою даних використовується Django ORM [10]. Цей інструмент дозволяє працювати з даними у вигляді об'єктів Python без необхідності написання великої кількості SQL-запитів. Такий підхід підвищує продуктивність розробки, спрощує підтримку програмного коду та зменшує ймовірність помилок під час роботи з базою даних. Крім того, використання ORM сприяє кращій масштабованості системи та полегшує внесення змін до структури бази даних у майбутньому.

Клієнтська частина системи реалізована за допомогою мов HTML [6], CSS [7] та JavaScript [8]. HTML використовується для створення структури вебсторінок, CSS - для їхнього оформлення та забезпечення адаптивного дизайну, а JavaScript - для реалізації інтерактивних елементів інтерфейсу та покращення взаємодії користувача із системою.

Також використовується шаблонізатор Django, який дозволяє формувати сторінки на сервері та повторно використовувати спільні

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		23

елементи інтерфейсу, такі як меню навігації, шапка сайту, футер та інші компоненти. Це сприяє зменшенню дублювання коду та спрощує внесення змін до дизайну системи. Обраний підхід є оптимальним для даного проєкту, оскільки дозволяє створити функціональний вебзастосунок без використання складної SPA-архітектури та додаткових JavaScript-фреймворків.

Для забезпечення безпеки користувацьких даних використовуються вбудовані механізми Django. Система підтримує безпечне зберігання паролів за допомогою сучасних алгоритмів хешування [9], розмежування прав доступу відповідно до ролей користувачів та захист форм від несанкціонованих дій. У системі передбачено три основні ролі: клієнт, власник готелю та адміністратор. Кожна роль має власний набір прав доступу до функцій системи, що забезпечує контроль над виконанням операцій та підвищує рівень безпеки [19, 20].

Для керування вихідним кодом використовується система контролю версій Git [21]. Її використання дозволяє зберігати історію змін проєкту, повертатися до попередніх версій програмного коду, працювати над декількома функціями одночасно та спрощує командну розробку. Крім того, Git допомагає контролювати процес внесення змін та забезпечує надійне зберігання вихідного коду.

Оскільки система реалізується як монолітний вебзастосунок, вимоги до апаратного забезпечення є відносно невисокими. Для розміщення серверної частини достатньо сервера з 2-4 ядрами процесора, оперативною пам'яттю обсягом 4-8 ГБ та SSD-накопичувачем обсягом від 20 до 40 ГБ. Такі характеристики дозволять забезпечити стабільну роботу системи та обробку одночасних запитів від користувачів.

Для сервера бази даних рекомендовано використовувати аналогічні характеристики, оскільки основне навантаження припадає на виконання операцій читання та запису даних. Використання SSD-накопичувачів дозволяє значно прискорити доступ до інформації та скоротити час виконання запитів [27].

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		24

З боку клієнта спеціальних вимог до апаратного забезпечення немає. Для роботи із системою достатньо персонального комп'ютера, ноутбука, планшета або смартфона з встановленим сучасним веббраузером та доступом до мережі Інтернет. Завдяки використанню вебтехнологій користувачі можуть працювати із системою незалежно від операційної системи та типу пристрою.

Таким чином, обраний стек технологій є оптимальним для реалізації маркетплейсу оренди житла. Використання MySQL, Django, HTML, CSS та JavaScript забезпечує необхідний рівень продуктивності, безпеки та зручності розробки. Крім того, такі технології мають широку підтримку, добре документовані та дозволяють у майбутньому розширювати функціональні можливості системи без суттєвих змін її архітектури.

Висновки до першого розділу

Було сформульовано задачу розробки маркетплейсу для оренди житла в готелях та визначено основні вимоги до його функціоналу. Зокрема, передбачено реалізацію реєстрації та авторизації користувачів, керування готелями й номерами, бронювання житла, системи відгуків, а також підтримку ролей клієнта, власника готелю та адміністратора.

Проведено аналіз існуючих сервісів онлайн-бронювання, що дозволило визначити їх переваги та недоліки, а також сформувати вимоги до майбутньої системи, зокрема щодо зручності використання, надійності бронювання та ефективного пошуку житла.

У результаті аналізу архітектурних підходів було обрано монолітну архітектуру з модульною організацією (Modular Monolith), яка забезпечує простоту реалізації та можливість подальшого розвитку системи.

Для розробки серверної частини використано фреймворк Django, для зберігання даних - реляційну базу даних MySQL. Клієнтську частину реалізовано за допомогою HTML, CSS та JavaScript із використанням

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		25

шаблонізатора, що дозволило створити зручний та інтуїтивно зрозумілий інтерфейс користувача.

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		26

РОЗДІЛ 2 ПРОЄКТУВАННЯ СИСТЕМИ

2.1 Визначення варіантів використання та об'єктно-орієнтованої структури системи

Проектування інформаційної системи маркетплейсу оренди житла передбачає формалізацію функціональних вимог у вигляді варіантів використання (Use Case Model) та побудову об'єктно-орієнтованої моделі предметної області. Цей етап є важливим для аналізу системи, оскільки визначає взаємодію між користувачами та системними компонентами і формує основу для подальшого проектування програмної архітектури. У межах розроблюваної системи виділено такі основні актори:

1. Гість (Anonymous User) - неавторизований користувач, який має доступ до публічної інформації;
2. Клієнт (Client) - зареєстрований користувач, що здійснює пошук, бронювання та оцінювання житла;
3. Власник (Owner) - користувач, який створює та адмініструє об'єкти оренди;
4. Адміністратор (Administrator) - користувач, що здійснює модерацію та контроль системи.

Розмежування ролей дозволяє реалізувати принципи Role-Based Access Control (RBAC), що забезпечує безпеку та контроль доступу до функціональних можливостей системи (Рис. 2.1).

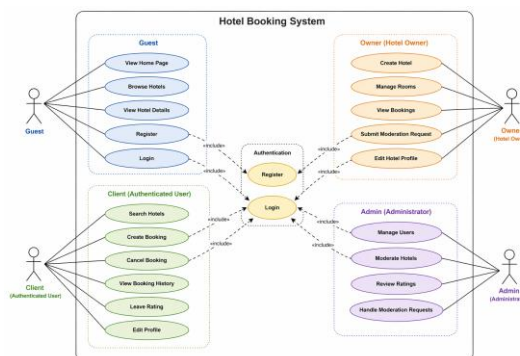


Рис 2.1. - Діаграма використання

Детальний опис прецедентів наведено в таблицях 2.1-2.18:

Таблиця 2.1

Перегляд головної сторінки

Назва	Перегляд головної сторінки
Актор	Гість
Опис	Перегляд загальної інформації про платформу та доступних об'єктів

Даний прецедент (Табл. 2.1) описує взаємодію неавторизованого користувача із системою на початковому етапі роботи. Гість має можливість переглянути загальну інформацію про платформу, ознайомитися з доступними пропозиціями житла та основними можливостями сервісу. Головна сторінка виконує інформаційну функцію та допомагає користувачу прийняти рішення щодо подальшого використання системи.

Таблиця 2.2

Перегляд деталей житла

Назва	Перегляд деталей житла
Актор	Гість
Опис	Ознайомлення з інформацією про житло (опис, фото, ціна, рейтинг)

Після вибору конкретного об'єкта користувач може переглянути його детальну інформацію (Табл. 2.2). До неї належать опис житла, фотографії, ціна проживання, рейтинг та відгуки інших клієнтів. Надання повної інформації дозволяє користувачу оцінити якість пропозиції та прийняти рішення щодо бронювання.

Таблиця 2.3

Реєстрація / Вхід

Назва	Реєстрація/ Вхід
Актор	Гість
Опис	Створення облікового запису або авторизація у системі

Цей прецедент забезпечує можливість створення нового облікового запису або входу до системи (Табл. 2.3). Реєстрація необхідна для отримання доступу до функцій бронювання та керування власними даними. Авторизація дозволяє ідентифікувати користувача та надати йому відповідні права доступу залежно від ролі.

Таблиця 2.4

Керування профілем

Назва	Керування профілем
Актор	Клієнт / Власник
Опис	Перегляд та редагування персональних даних

Авторизований користувач має можливість переглядати та редагувати власні персональні дані (Табл. 2.4). Зокрема, він може змінювати контактну інформацію або оновлювати налаштування профілю. Наявність такого функціоналу забезпечує актуальність даних та покращує зручність використання системи.

Таблиця 2.5

Пошук житла

Назва	Пошук житла
Актор	Клієнт

продовж. табл. 2.5

Опис	Пошук об'єктів за містом, областю, ціною та іншими параметрами
------	--

Функція пошуку дозволяє клієнту знаходити житло відповідно до заданих критеріїв (Табл. 2.5). Для цього можуть використовуватися фільтри за містом, областю, вартістю проживання, рейтингом та іншими характеристиками. Такий підхід значно спрощує пошук потрібного варіанту серед великої кількості пропозицій.

Таблиця 2.6

Перегляд списку житла

Назва	Перегляд списку житла
Актор	Клієнт
Опис	Отримання переліку доступних варіантів оренди

Після виконання пошуку система відображає список доступних варіантів проживання (Табл. 2.6). Користувач може переглянути основну інформацію про кожен об'єкт та перейти до його детального опису. Це дозволяє швидко порівнювати різні пропозиції та обирати найбільш підходящу.

Таблиця 2.7

Бронювання житла

Назва	Бронювання житла
Актор	Клієнт
Опис	Створення бронювання із зазначенням дат заїзду та виїзду

Даний прецедент є одним із ключових у роботі системи (Табл. 2.7). Клієнт обирає житло та вказує дати заїзду і виїзду, після чого система

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				30
Змн.	Арк.	№ докум.	Підпис	Дата		

перевіряє доступність номера. У разі успішної перевірки створюється нове бронювання та зберігається в базі даних.

Таблиця 2.8

Перегляд власних бронювань

Назва	Перегляд власних бронювань
Актор	Клієнт
Опис	Перегляд історії бронювань та їх статусів

Функціонал дозволяє клієнту отримати доступ до історії своїх бронювань (Табл. 2.8). Користувач може переглядати інформацію про активні, завершені та скасовані замовлення. Це забезпечує зручний контроль над усіма здійсненими бронюваннями.

Таблиця 2.9

Скасування бронювання

Назва	Скасування бронювання
Актор	Клієнт
Опис	Скасування активного бронювання

Клієнт має можливість скасувати активне бронювання відповідно до встановлених правил системи (Табл. 2.9). Після скасування інформація про доступність номера оновлюється автоматично. Така функція забезпечує гнучкість у разі зміни планів користувача.

Таблиця 2.10

Залишення відгуку

Назва	Залишення відгуку
Актор	Клієнт
Опис	Оцінювання житла та додавання коментаря

Після завершення проживання клієнт може залишити оцінку та текстовий відгук про житло (Табл. 2.10). Відгуки допомагають іншим користувачам оцінити якість послуг та зробити правильний вибір. Також вони сприяють підвищенню довіри до платформи.

Таблиця 2.11

Створення профілю готелю/ житла

Назва	Створення профілю готелю/ житла
Актор	Власник
Опис	Додавання нового об'єкта оренди

Власник має можливість додати новий об'єкт до системи (Табл. 2.11). Під час створення профілю вказується основна інформація про готель або житло, його місцезнаходження, опис та фотографії. Це дозволяє зробити об'єкт доступним для пошуку та бронювання клієнтами.

Таблиця 2.12

Редагування житла

Назва	Редагування житла
Актор	Власник
Опис	Оновлення інформації про об'єкт

Даний прецедент забезпечує можливість оновлення інформації про об'єкт (Табл. 2.12). Власник може змінювати опис, фотографії, ціни та інші характеристики житла. Це дозволяє підтримувати актуальність інформації для потенційних клієнтів.

Таблиця 2.13

Керування номерами

Назва	Керування номерами
Актор	Власник
Опис	Додавання, редагування та видалення кімнат

Власник готелю може додавати нові номери, редагувати наявні або видаляти застарілі записи (Табл. 2.13). Для кожного номера задаються характеристики, ціна та інформація про доступність. Завдяки цьому забезпечується ефективне управління номерним фондом.

Таблиця 2.14

Керування бронюваннями

Назва	Керування бронюваннями
Актор	Власник
Опис	Підтвердження або відхилення бронювань

Після отримання заявки на бронювання власник має можливість її переглянути та прийняти відповідне рішення (Табл. 2.14). За необхідності бронювання може бути підтверджене або відхилене. Такий механізм дозволяє контролювати процес заселення клієнтів.

Таблиця 2.15

Перегляд власних готелей

Назва	Перегляд власних готелей
Актор	Власник
Опис	Перегляд списку створених готелей

Функція надає власнику доступ до списку всіх створених ним об'єктів (Табл. 2.15). Користувач може переглядати їхній поточний стан, статистику та основну інформацію. Це спрощує керування декількома готелями або об'єктами житла одночасно.

Таблиця 2.16

Модерація готелей

Назва	Модерація готелей
Актор	Адміністратор
Опис	Перевірка та підтвердження власника готеля

Адміністратор виконує перевірку інформації про готелі та власників перед їх публікацією на платформі (Табл. 2.16). Метою модерації є забезпечення достовірності даних та дотримання правил сервісу. Після перевірки об'єкт може бути схвалений або відхилений.

Таблиця 2.17

Керування користувачами

Назва	Керування користувачами
Актор	Адміністратор
Опис	Перегляд, блокування або редагування користувачів

Адміністратор має доступ до списку зареєстрованих користувачів системи (Табл. 2.17). Він може переглядати їхні профілі, змінювати статус облікових записів або блокувати порушників. Це дозволяє підтримувати безпечне та стабільне функціонування платформи.

Таблиця 2.18

Обробка запитів на модерацію

Назва	Обробка запитів на модерацію
Актор	Адміністратор

Опис	Розгляд заявок на верифікацію або встановлення зірок
------	--

Даний прецедент передбачає розгляд заявок від власників щодо верифікації облікових записів або присвоєння категорії готелю (Табл. 2.18). Адміністратор перевіряє надані документи та приймає рішення щодо схвалення або відхилення заявки. Такий процес сприяє підвищенню довіри користувачів до сервісу та достовірності розміщеної інформації.

Відповідно до аналізу варіантів використання було сформовано узагальнений перелік функціональних вимог, які визначають основні можливості інформаційної системи маркетплейсу оренди житла.

Система повинна забезпечувати реалізацію широкого набору функціональних можливостей, які охоплюють як загальнодоступні функції, так і ролі користувачів різних рівнів доступу.

До загальнодоступних функцій належать перегляд головної сторінки з інформацією про платформу, перегляд детальної інформації про доступні об'єкти житла, а також пошук і фільтрація житла за різними критеріями. Це забезпечує базову навігацію та ознайомлення з пропозиціями без необхідності авторизації.

Функціонал користувача (Client) включає реєстрацію та авторизацію в системі, управління особистим профілем, створення, перегляд і скасування бронювань, доступ до історії бронювань, а також можливість залишати рейтинги та відгуки щодо житла. Це формує повний цикл взаємодії користувача з платформою.

Функції власника (Owner) передбачають створення та редагування об'єктів житла (готелів), управління кімнатами та їх характеристиками, перегляд і обробку бронювань користувачів, а також контроль інформації про власні об'єкти. Таким чином забезпечується повний контроль над пропозиціями оренди.

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		35

Адміністративні функції включають управління користувачами системи, модерацію готелів і перевірку контенту, обробку запитів на верифікацію та модерацію, контроль рейтингових даних і відгуків, а також моніторинг стану системи та активності користувачів. Це забезпечує стабільність і якість роботи платформи.

Нефункціональні вимоги визначають якість роботи системи, її продуктивність, надійність та експлуатаційні обмеження. До вимог продуктивності належать середній час відповіді системи не більше 5 секунд, виконання операцій бронювання до 10 секунд, а також підтримка щонайменше 50–100 одночасних користувачів.

Вимоги надійності та стабільності передбачають рівень доступності системи не менше 99%, допустиму кількість критичних помилок не більше 1 на 1000 запитів, а також забезпечення атомарності та захищеності всіх транзакційних операцій. Відмовостійкість системи реалізується через автоматичне відновлення після збоїв, підтримку retry-механізмів і гарантію відсутності втрати даних при аварійних ситуаціях.

До вимог зручності використання належать час навчання користувача не більше 1 години, інтуїтивно зрозумілий інтерфейс і мінімальна кількість кроків для виконання основних операцій. Обмеження відмов передбачають максимальний рівень відмов операцій 2–3%, успішність бронювань на рівні 97–98% та пріоритетну обробку критичних операцій із контролем помилок.

Доменна модель інформаційної системи побудована на принципах об'єктно-орієнтованого проектування. Основна ідея такого підходу полягає у відображенні предметної області через чітко визначені бізнес-сутності та їх взаємозв'язки, що дозволяє забезпечити узгодженість між програмною реалізацією та реальними процесами предметної області.

У межах розроблюваної системи виділено набір ключових доменних сутностей (Табл. 2.19), кожна з яких відповідає окремому бізнес-процесу або об'єкту предметної області:

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		36

Характеристика основних сутностей системи

Сутність	Опис
CustomUser	Базова сутність користувача системи, яка розширює стандартну модель користувача та реалізує рольову модель доступу. Підтримує три основні ролі: client, owner та admin, що дозволяє розмежувати функціональні можливості між різними типами користувачів.
Hotel	Агрегуюча сутність, що представляє основний об'єкт оренди. Вона містить інформацію про житло, його розташування, контактні дані, рівень верифікації та належить конкретному власнику. Hotel виступає центральним об'єктом у доменній моделі та об'єднує пов'язані сутності нижчого рівня.
Room	Сутність, що описує конкретні варіанти розміщення в межах готелю. Вона містить характеристики житлової одиниці, включаючи ціну, площу, кількість кімнат та категорію комфорту. Room є деталізацією готелю та безпосереднім об'єктом бронювання.
Booking	Сутність, що реалізує бізнес-процес бронювання. Вона забезпечує зв'язок між користувачем (client) та конкретною кімнатою, а також містить інформацію про період проживання, загальну вартість та статус бронювання протягом його життєвого циклу.
Rating	Сутність системи оцінювання, яка дозволяє користувачам залишати відгуки та рейтинги щодо конкретних кімнат. Вона забезпечує механізм зворотно-го зв'язку та підвищує довіру до платформи.
Moderation Request	Службова сутність, що відповідає за процес модерації контенту. Використовується адміністраторами для перевірки, верифікації та контролю готелів, а також інших змін, що потребують підтвердження.

Наявність добре сформованої доменної моделі також створює основу для подальшого розширення функціональних можливостей, інтеграції нових сервісів і масштабування системи відповідно до зростання кількості користувачів та об'єктів оренди. У результаті це забезпечує логічну цілісність, гнучкість та ефективність роботи інформаційної системи в довгостроковій перспективі.

2.2 Розробка бази даних системи

База даних інформаційної системи маркетплейсу оренди житла розроблена відповідно до принципів реляційного моделювання та тісно узгоджена з доменною моделлю системи. Оскільки система реалізована на основі монолітної архітектури Django, база даних виступає єдиним централізованим сховищем усіх бізнес-даних.

Основними вимогами до проєктування бази даних є забезпечення цілісності даних, підтримка зв'язків між сутностями, а також можливість ефективного виконання операцій пошуку, фільтрації та транзакційних процесів (зокрема бронювання та рейтингової системи).

Для реалізації бази даних використовується ORM Django, що забезпечує автоматичне відображення об'єктно-орієнтованої моделі на реляційну структуру таблиць. До складу бази даних входять такі основні сутності (Табл. 2.20).

Таблиця 2.20

Основні сутності

Таблиця	Опис
CustomUser	Користувачі системи з підтримкою ролей
Hotel	Об'єкти оренди (готелі/житло);
Room	Кімнати або житлові одиниці
Booking	Бронювання житла

Rating	Відгуки та оцінки користувачів
ModerationRequest	Адміністративні запити на модерацію

Вище наведено перелік основних сутностей бази даних (Табл. 2.20), які використовуються для реалізації функціоналу системи маркетплейсу оренди житла. Кожна сутність відповідає певному об'єкту предметної області та забезпечує зберігання необхідної інформації для роботи системи. Разом вони формують структуру бази даних та забезпечують реалізацію бізнес-процесів, пов'язаних із бронюванням житла, керуванням користувачами та модерацією контенту.

Таблиця 2.21

CustomUser

Поле	Тип	Призначення
id	bigint	Унікальний ідентифікатор користувача
email	string	Email (логін)
password	string	Хеш пароля
first_name	string	Ім'я
last_name	string	Прізвище
phone_number	string	Номер телефону
avatar	string	Фото профілю
role	string	Роль користувача

Сутність CustomUser (Табл. 2.21) призначена для зберігання облікових записів користувачів та інформації про них. Вона забезпечує реалізацію механізмів автентифікації та авторизації, а також підтримує рольову модель

доступу. Наявність ролей дозволяє розмежувати функціональні можливості між клієнтами, власниками житла та адміністраторами системи.

Таблиця 2.22

Hotel

Поле	Тип	Призначення
id	bigint	Ідентифікатор готелю
name	string	Назва
description	text	Опис
main_photo	string	Головне фото
phone_number	string	Контактний номер
city	string	Місто
oblast	string	Область
address	string	Адреса
certificate_link	string	Документ/сертифікація
stars	int	Зірковість готелю
is_approved	Boolean	Статус перевірки готелю
owner_id	bigint FK	Власник

Сутність Hotel (Табл. 2.22) призначена для зберігання інформації про готелі та інші об'єкти оренди, які розміщуються на платформі. Вона містить дані про місцезнаходження, контактну інформацію, опис об'єкта та його статус перевірки. Крім того, сутність забезпечує зв'язок між власником житла та номерами, які належать конкретному готелю.

Таблиця 2.23

Room

Поле	Тип	Призначення
id	bigint	Ідентифікатор кімнати
hotel_id	bigint FK	Готель
name	string	Назва кімнати
price_per_night	decimal	Ціна
description	text	Опис
main_photo	string	Фото
number_of_rooms	int	Кількість кімнат
square_meters	float	Площа
grade	string	Категорія

Сутність Room (Табл. 2.23) призначена для опису окремих номерів або житлових одиниць, доступних для бронювання. Вона зберігає характеристики житла, такі як ціна, площа, кількість кімнат та категорія комфорту. Саме ця сутність використовується користувачами під час пошуку житла та створення бронювання.

Таблиця 2.24

Booking

Поле	Тип	Призначення
id	bigint	Ідентифікатор бронювання
user_id	bigint FK	Користувач
room_id	bigint FK	Кімната

check_in	date	Дата заїзду
check_out	date	Дата виїзду
total_price	decimal	Загальна вартість
status	string	Статус бронювання

Сутність Booking (Табл. 2.24) призначена для реалізації процесу бронювання житла в системі. Вона забезпечує зв'язок між користувачем та обраним номером, а також зберігає інформацію про дати проживання, вартість послуги та поточний статус бронювання. Дані цієї сутності використовуються для контролю доступності номерів та ведення історії замовлень.

Таблиця 2.25

Rating

Поле	Тип	Призначення
id	bigint	Ідентифікатор
user_id	bigint FK	Автор
room_id	bigint FK	Кімната
score	int	Оцінка (1-5)
comment	text	Коментар
created_at	datetime	Дата створення

Сутність Rating (Табл. 2.25) призначена для реалізації механізму оцінювання житла та збору відгуків користувачів. Вона дозволяє зберігати рейтинги та текстові коментарі щодо номерів після завершення проживання.

Інформація про оцінки використовується для формування репутації об'єктів та допомагає іншим користувачам під час вибору житла.

Таблиця 2.26

ModerationRequest

Поле	Тип	Призначення
id	bigint	Ідентифікатор
type	string	Тип запиту
status	string	Статус
payload	json	Дані запиту
created_at	datetime	Дата створення

Сутність ModerationRequest (Табл. 2.26) призначена для організації процесу модерації в системі. Вона використовується для зберігання заявок на перевірку готелів, підтвердження інформації або виконання інших адміністративних дій. Збереження статусу та додаткових даних запиту дозволяє адміністраторам контролювати процес модерації та приймати відповідні рішення.

Далі наведено діаграму бази даних інформаційної системи маркетплейсу оренди житла (Рис. 2.3), яка відображає структуру основних таблиць, їх атрибути та взаємозв'язки між ними.

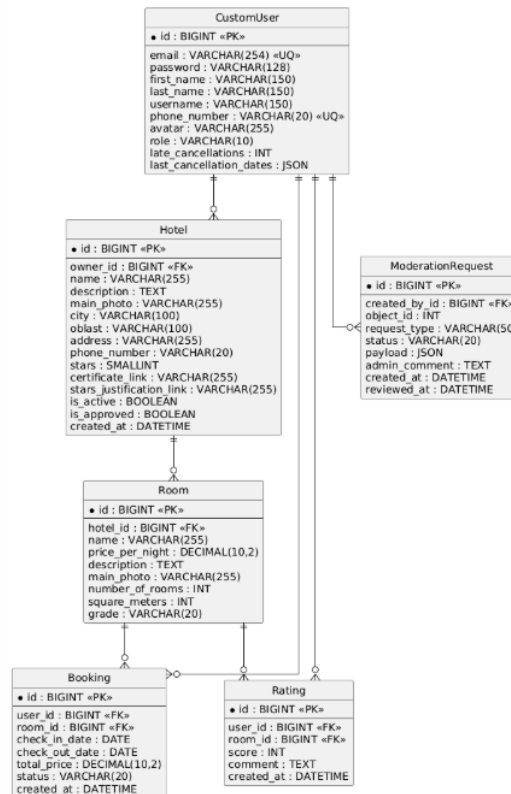


Рис. 2.3. - Діаграма бази даних

Дана діаграма побудована відповідно до реляційної моделі даних та відображає як доменні сутності системи (CustomUser, Hotel, Room, Booking, Rating, ModerationRequest), так і службові таблиці Django, що забезпечують підтримку роботи фреймворку, зокрема механізми сесійної авторизації (django_session) та систему міграцій бази даних (django_migrations).

Зв'язки між таблицями реалізовані за допомогою зовнішніх ключів (Foreign Key).

У процесі проєктування бази даних було прийнято низку ключових рішень, що забезпечують ефективну роботу системи маркетплейсу оренди житла. Зокрема, використання ORM Django дозволяє автоматично синхронізувати доменну модель із структурою бази даних, що спрощує розробку та зменшує кількість ручних SQL-запитів. Застосування зовнішніх ключів забезпечує цілісність зв'язків між основними сутностями системи, такими як користувачі, готелі, номери та бронювання.

Для підвищення продуктивності було впроваджено індексування полів, які найчастіше використовуються у пошуку та фільтрації, зокрема місто, ціна

та дати бронювання. Це дозволяє суттєво пришвидшити виконання запитів при великій кількості даних. Транзакційна модель обробки бронювань гарантує коректність роботи системи в умовах одночасних запитів, запобігаючи конфліктам і подвійним бронюванням.

Окремі бізнес-процеси реалізовано через розділення сутностей Booking, Rating та ModerationRequest, що дозволяє чітко структурувати логіку системи та спростити її подальший супровід і масштабування. Такий підхід забезпечує баланс між продуктивністю, цілісністю даних та зручністю розробки, що є критично важливим для подібного типу веб-платформ.

Розроблена база даних забезпечує повну підтримку бізнес-логіки системи маркетплейсу оренди житла. Вона гарантує цілісність даних, ефективну обробку запитів та стабільну роботу ключових процесів, таких як бронювання, оцінювання та управління об'єктами оренди.

2.3 Проектування та реалізація алгоритмів системи

При розробці інформаційної системи маркетплейсу оренди житла важливим етапом є проектування та формалізація основних алгоритмів взаємодії користувачів із системою. Алгоритми визначають порядок виконання операцій, правила обробки даних та логіку переходу між різними станами системи. Від якості їх проектування залежить коректність роботи програмного продукту, швидкість обробки запитів користувачів та надійність функціонування всієї системи.

У розроблюваній системі алгоритми реалізують основні бізнес-процеси, пов'язані з пошуком житла, переглядом інформації про об'єкти, створенням та обробкою бронювань, роботою з відгуками, а також адмініструванням і модерацією контенту. Формалізація цих процесів за допомогою діаграм дозволяє наочно представити логіку роботи системи та спростити подальшу реалізацію програмного коду.

Однією з головних особливостей маркетплейсу оренди житла є необхідність забезпечення узгодженості даних між різними категоріями

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		46

користувачів. У системі одночасно взаємодіють клієнти, власники житла та адміністратори, тому кожна дія повинна виконуватися відповідно до встановлених правил доступу та не призводити до порушення цілісності даних. Саме тому алгоритми системи враховують перевірку прав користувачів, валідацію введеної інформації та контроль допустимих переходів між станами об'єктів.

Першим етапом взаємодії користувача із системою зазвичай є пошук житла. Клієнт задає параметри пошуку, такі як місто, область, діапазон цін або інші характеристики. Після отримання запиту система виконує звернення до бази даних та формує перелік об'єктів, які відповідають заданим критеріям. Далі результати сортуються та відображаються користувачу у зручному форматі. У разі відсутності відповідних варіантів система повідомляє користувача про необхідність зміни параметрів пошуку.

Після вибору конкретного об'єкта користувач переходить до перегляду детальної інформації про житло. На цьому етапі система отримує дані про готель та пов'язані з ним номери, фотографії, рейтинги й відгуки. Відображення повної інформації дозволяє користувачу оцінити якість пропозиції та прийняти рішення щодо подальшого бронювання.

Основним сценарієм роботи системи є створення бронювання клієнтом (Рис. 2.4). Саме цей процес забезпечує реалізацію головної функції маркетплейсу та є центральним елементом усієї бізнес-логіки. Алгоритм створення бронювання складається з декількох послідовних етапів, кожен із яких має важливе значення для забезпечення коректності роботи системи.

На початку процесу користувач обирає номер та вказує дати заїзду і виїзду. Після отримання запиту система виконує перевірку введених даних. Насамперед перевіряється коректність дат, тобто дата виїзду повинна бути пізнішою за дату заїзду. Додатково перевіряється, чи не належать обрані дати до минулого періоду та чи відповідають вони встановленим обмеженням системи.

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		47

Після успішного проходження валідації виконується перевірка доступності номера. Для цього система аналізує вже існуючі бронювання та визначає, чи не перетинаються обрані дати з іншими активними замовленнями. Така перевірка дозволяє уникнути подвійного бронювання одного й того самого номера, що є одним із найважливіших аспектів роботи будь-якої системи бронювання.

Якщо номер доступний у зазначений період, система переходить до розрахунку вартості проживання. Загальна сума визначається на основі кількості днів проживання та вартості номера за одну ніч. За необхідності до розрахунку можуть включатися додаткові платежі або знижки. Після завершення обчислень користувачу відображається підсумкова інформація про майбутнє бронювання.

Наступним етапом є створення запису в базі даних. Система формує нову сутність бронювання та встановлює для неї початковий статус *created*. До запису зберігаються дані про користувача, номер, дати проживання, вартість та час створення заявки. Після успішного збереження інформації користувач отримує повідомлення про створення бронювання.

Результатом виконання даного алгоритму є поява нового бронювання в системі та передача інформації власнику житла для подальшого розгляду. Таким чином забезпечується початок життєвого циклу бронювання та створюються умови для подальшої взаємодії між сторонами.

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		48

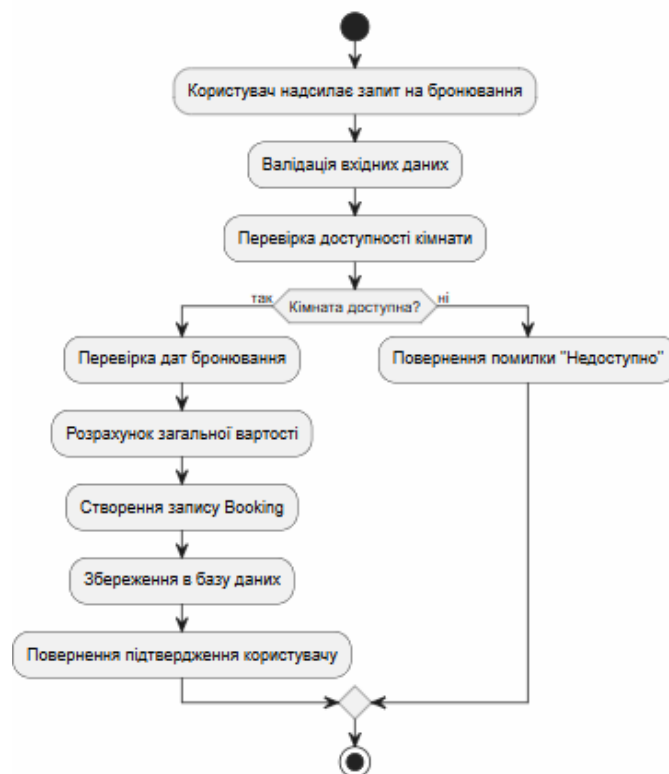


Рис. 2.4. - Діаграма процесу створення бронювання

Другим важливим процесом є обробка бронювання власником житла (Рис. 2.5). Після створення заявки інформація автоматично стає доступною власнику відповідного об'єкта. Основним завданням власника є перевірка отриманого запиту та прийняття рішення щодо можливості надання послуги.

Під час перегляду заявки власник отримує доступ до інформації про клієнта, обраний номер та дати проживання. На основі цих даних він може оцінити можливість виконання бронювання. Якщо перешкод для заселення немає, власник підтверджує заявку. У цьому випадку статус бронювання змінюється на confirmed, а клієнту надсилається відповідне повідомлення.

Якщо з певних причин власник не може надати послугу у зазначений період, бронювання може бути відхилене. У такому випадку його статус змінюється на cancelled. Система автоматично повідомляє користувача про результат розгляду заявки та оновлює інформацію про доступність номера.

Важливою особливістю цього алгоритму є контроль допустимих переходів між станами бронювання. Власник може змінювати статус лише

відповідно до визначених правил, що запобігає виникненню помилок та неузгодженостей у даних.

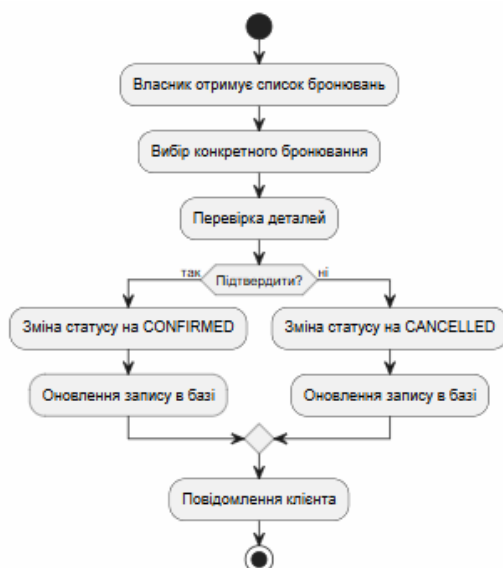


Рис. 2.5. - Діаграма обробки бронювання власником

Окрім створення та підтвердження бронювань, система підтримує механізм їх подальшого супроводу. Після настання дати заїзду бронювання переходить до стану активного виконання. На цьому етапі клієнт фактично користується послугою проживання, а система зберігає інформацію про поточний статус замовлення.

Після завершення терміну проживання бронювання автоматично переводиться до завершеного стану. Завершені бронювання використовуються для формування історії замовлень користувача та можуть бути основою для залишення відгуків і оцінок. Це дозволяє забезпечити накопичення статистичних даних та формування рейтингу житла.

Бронювання у системі проходить кілька етапів життєвого циклу: створення, підтвердження, виконання та завершення. Кожен із цих станів відображає поточний етап обробки замовлення та визначає набір доступних дій для користувачів системи (Рис.2.6).

Початковим станом є created. У цьому стані бронювання вже створене клієнтом, але ще не було розглянуте власником житла. Наступним можливим

станом є confirmed, який означає успішне підтвердження заявки. Якщо бронювання не може бути виконане, воно переводиться у стан cancelled.

Після настання дати заселення підтверджене бронювання переходить у стан active або in progress, що свідчить про фактичне використання послуги проживання. Після завершення періоду проживання бронювання отримує статус completed, який означає успішне завершення всіх етапів надання послуги.

Такий підхід дозволяє чітко контролювати життєвий цикл замовлення та забезпечує прозорість взаємодії між клієнтом і власником житла. Крім того, використання статусної моделі спрощує реалізацію бізнес-логіки та дозволяє легко розширювати функціональні можливості системи в майбутньому.

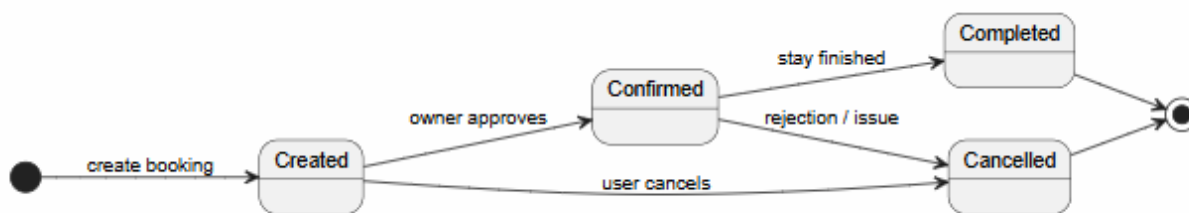


Рис. 2.6. - Діаграма життєвого циклу бронювання

Таким чином, розроблені алгоритми забезпечують реалізацію основних функцій маркетплейсу оренди житла та визначають порядок взаємодії між усіма учасниками системи. Використання формалізованих сценаріїв дозволяє гарантувати коректність виконання бізнес-процесів, підтримувати цілісність даних та забезпечувати стабільну роботу програмного забезпечення. Реалізована алгоритмічна логіка створює основу для подальшого розвитку системи та впровадження додаткових можливостей, пов'язаних із автоматизацією процесів бронювання та управління житлом.

Висновки до другого розділу

У процесі проектування інформаційної системи маркетплейсу оренди житла було визначено та формалізовано ключові функціональні вимоги, що

описують основні сценарії взаємодії користувачів із системою, включаючи пошук житла, бронювання, управління об'єктами оренди та модерацію контенту.

На основі аналізу вимог була розроблена доменна модель системи, в якій виділено основні сутності (CustomUser, Hotel, Room, Booking, Rating, ModerationRequest) та встановлено зв'язки між ними. Це дозволило забезпечити чітку структуру предметної області та закласти основу для подальшої реалізації бізнес-логіки.

Також було спроектовано реляційну базу даних, яка відображає доменну модель системи та забезпечує цілісність і узгодженість даних. Використання ORM Django дозволило ефективно поєднати об'єктно-орієнтовану модель застосунку з реляційною структурою бази даних.

Окрему увагу приділено проєктуванню алгоритмів основних бізнес-процесів системи, зокрема створенню та обробці бронювань, а також управлінню їх життєвим циклом. Використання UML-діаграм дозволило формалізувати логіку роботи системи та підвищити її зрозумілість.

У результаті виконаного проєктування було сформовано повну концептуальну та логічну основу системи, яка забезпечує її подальшу реалізацію у вигляді стабільного та підтримуваного вебзастосунку.

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		52

РОЗДІЛ 3 ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ ІЗ СИСТЕМОЮ

3.1 Порядок встановлення та налаштування параметрів системи

Система маркетплейсу оренди житла реалізується на основі архітектурного підходу Modular Monolith (модульний моноліт), у межах якого всі функціональні компоненти працюють як єдиний програмний застосунок, але логічно поділяються на незалежні модулі. Такий підхід поєднує переваги монолітної архітектури та принципів модульного проєктування, забезпечуючи високу продуктивність, спрощене адміністрування та можливість подальшого масштабування програмного забезпечення.

На відміну від мікросервісної архітектури, де кожен функціональний компонент розгортається окремо, модульний моноліт дозволяє уникнути складнощів, пов'язаних із міжсервісною взаємодією, мережевими затримками та необхідністю підтримки великої кількості окремих сервісів. Усі модулі системи працюють у межах єдиного середовища виконання Django, що забезпечує централізоване управління конфігурацією, безпекою та бізнес-логікою.

Функціональність системи логічно розподілена між кількома основними модулями:

- модулем користувачів та автентифікації;
- модулем керування готелями;
- модулем керування номерами;
- модулем бронювань;
- модулем відгуків та рейтингів;
- модулем адміністрування та модерації контенту.

Кожен із зазначених модулів містить власні моделі даних, представлення, шаблони та бізнес-логіку, що забезпечує низький рівень зв'язності між компонентами та спрощує подальший супровід програмного забезпечення. Незважаючи на логічне розділення, усі модулі використовують

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		53

спільну реляційну базу даних MySQL, що забезпечує цілісність даних та спрощує реалізацію складних бізнес-процесів.

У межах архітектури програмний застосунок розгортається на сервері додатків, який виконує обробку HTTP-запитів від клієнтів, реалізує бізнес-логіку та здійснює взаємодію із сервером бази даних. Для зберігання статичних та медіафайлів використовується файлове сховище сервера. До таких файлів належать фотографії готелів і номерів, зображення профілів користувачів, а також інші мультимедійні матеріали, що завантажуються під час роботи системи.

Взаємодія користувачів із системою здійснюється через вебінтерфейс за допомогою мережі Інтернет. Користувачі можуть працювати із системою з персональних комп'ютерів, ноутбуків, планшетів або мобільних пристроїв без необхідності встановлення додаткового програмного забезпечення. Усі операції виконуються через веб-браузер, що значно спрощує процес доступу до функціональних можливостей системи.

Використання централізованої архітектури забезпечує низку переваг під час експлуатації системи. Зокрема, адміністратори можуть виконувати оновлення програмного забезпечення без необхідності координувати роботу кількох незалежних сервісів. Крім того, спрощується процес резервного копіювання, моніторингу та відновлення працездатності системи у випадку виникнення збоїв. Впровадження нових функціональних можливостей також потребує менших витрат часу порівняно з розподіленими архітектурами, оскільки всі компоненти розгортаються як єдина програмна система.

Для кращого розуміння фізичного розміщення компонентів та взаємодії між ними було побудовано діаграму розгортання системи (рис. 3.1).

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		54

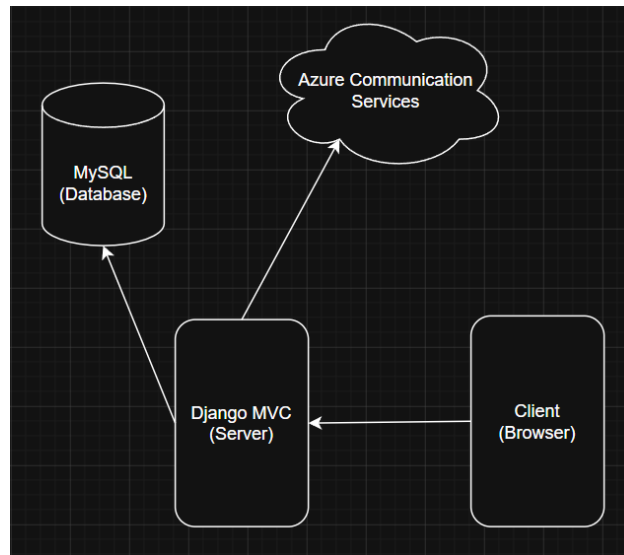


Рис. 3.1. - Діаграма розгортання системи маркетплейсу оренди житла

На діаграмі розгортання представлено три основні вузли системи: сервер застосунку Django, сервер бази даних MySQL та клієнтські пристрої користувачів. Клієнтські пристрої взаємодіють із сервером застосунку через мережу Інтернет за протоколом HTTP або HTTPS. Сервер застосунку обробляє отримані запити, виконує необхідні бізнес-операції та звертається до сервера бази даних для отримання або збереження інформації.

Сервер бази даних MySQL відповідає за централізоване зберігання інформації про користувачів, ролі доступу, готелі, номери, бронювання, відгуки, рейтинги, запити на модерацію та інші сутності предметної області. Використання реляційної системи керування базами даних дозволяє забезпечити високу цілісність даних, підтримку транзакцій та ефективне виконання складних запитів.

Для стабільної роботи серверної частини системи необхідно використовувати серверне обладнання (Табл. 3.1). Зазначені параметри є мінімально рекомендованими для забезпечення коректної роботи системи при невеликій кількості одночасних користувачів. У разі збільшення навантаження параметри серверного обладнання можуть бути масштабовані шляхом збільшення кількості процесорних ядер, обсягу оперативної пам'яті та дискового простору.

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		55

Таблиця 3.1

Мінімальні характеристики

Сервіс	Вимоги
Сервер застосунку	CPU: 2-4 ядра, RAM: 4-8 GB, SSD: 20-40 GB
Сервер бази даних	CPU: 2-6 ядер, RAM: 4-8 GB, SSD: 20+ GB

Після підготовки апаратного середовища необхідно виконати встановлення та початкове налаштування програмного забезпечення. Послідовність основних етапів встановлення наведена в Табл. 3.2:

Таблиця 3.2

Послідовність дій для встановлення та запуску розробленої системи

Етап	Компонент	Опис	Команда
1	Підготовка середовища	Встановлення Python, MySQL та необхідних залежностей проєкту	Встановити Python 3.12+, MySQL
2	Встановлення залежностей	Встановлення бібліотек, необхідних для роботи застосунку	<code>pip install -r requirements.txt</code>
3	Підготовка бази даних	Створення структури таблиць відповідно до моделей Django	<code>python manage.py migrate</code>
4	Створення першого адміністратора	Створення першого облікового запису адміністратора системи	<code>python manage.py create_initial_admin</code>
5	Заповнення тестовими даними	Генерація демонстраційних користувачів, готелів та номерів для тестування	<code>python manage.py seed</code>
6	Запуск системи	Запуск вебзастосунку та відкриття доступу до інтерфейсу користувача	<code>python app.py</code>

Після запуску система стає доступною через веб-браузер. Для взаємодії із системою користувач може використовувати будь-який сучасний веб-браузер, зокрема Google Chrome, Mozilla Firefox, Microsoft Edge, Safari та інші браузери з підтримкою HTML5, CSS3 та JavaScript.

3.2 Структура інтерфейсу платформи

Для початкового налаштування системи реалізовано спеціальну команду `create_initial_admin`, яка автоматично створює першого адміністратора платформи. Під час виконання команди система генерує випадковий пароль, створює обліковий запис адміністратора з необхідними правами доступу та надсилає дані для входу на електронну пошту (Рис. 3.2), вказану в налаштуваннях застосунку.

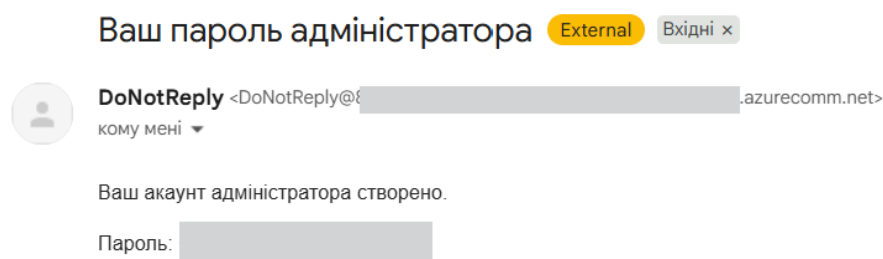


Рис. 3.2. - Лист із даними для входу першого адміністратора системи

Після створення облікового запису користувач отримує можливість авторизації в системі. Форма входу є спільною для всіх ролей платформи: адміністратора, власника готелю та клієнта. Для авторизації необхідно ввести адресу електронної пошти та пароль (Рис. 3.3), після чого система виконує перевірку облікових даних і надає доступ до відповідного функціоналу залежно від ролі користувача.

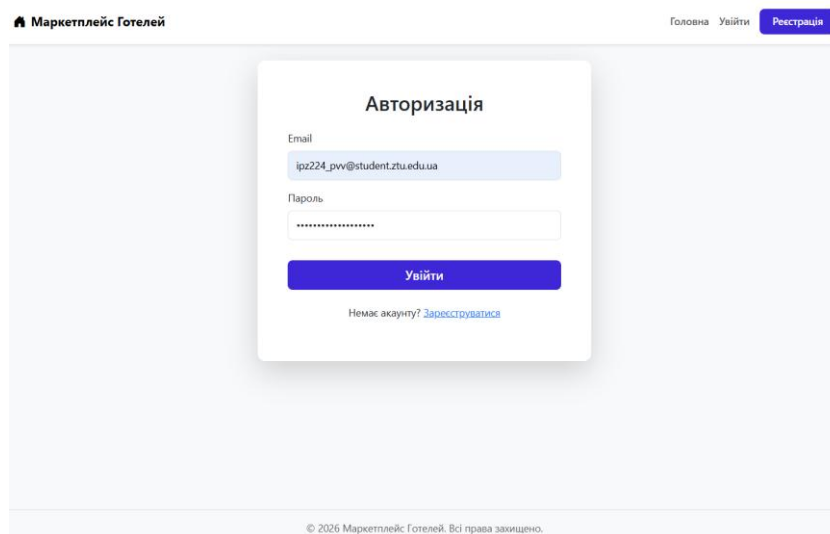


Рис. 3.3. - Форма авторизації користувача в системі

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		57

Головна сторінка платформи (Рис. 3.4) є центральним елементом взаємодії користувача із системою та призначена для пошуку й перегляду доступних варіантів проживання. У верхній частині сторінки розміщено короткий опис сервісу, який інформує користувача про можливості платформи. Основну частину інтерфейсу поділено на дві функціональні області. У лівій частині розташована панель фільтрації, що дозволяє виконувати пошук житла за різними критеріями: назвою, містом, областю, готелем, типом номера, кількістю місць, площею, вартістю проживання, рейтингом користувачів та зірковістю готелю. Також реалізовано механізми сортування результатів за ціною, рейтингом та назвою. У правій частині сторінки відображаються картки доступних номерів, які містять основну інформацію про житло: фотографію, назву, місцезнаходження, рейтинг та вартість проживання. Для підвищення зручності перегляду великої кількості результатів реалізовано механізм динамічного завантаження додаткових записів без перезавантаження сторінки.

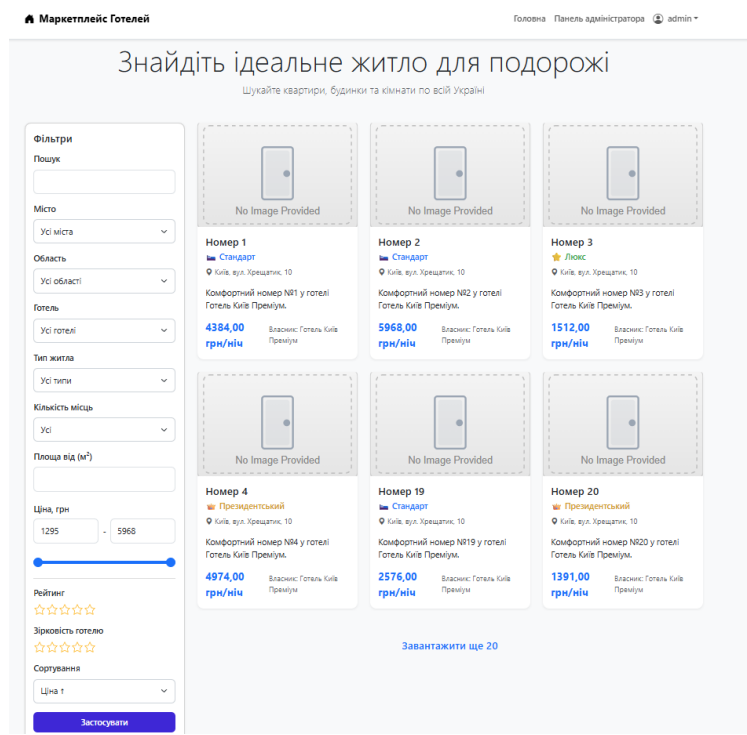


Рис. 3.4. - Головна сторінка маркетплейсу оренди житла з системою пошуку та фільтрації номерів

У випадку переходу на сторінку конкретного готелю (Рис. 3.5) користувачу додатково відображається детальна інформація про обраний заклад, включаючи опис, контактні дані, адресу, офіційну категорію та середню оцінку відвідувачів.

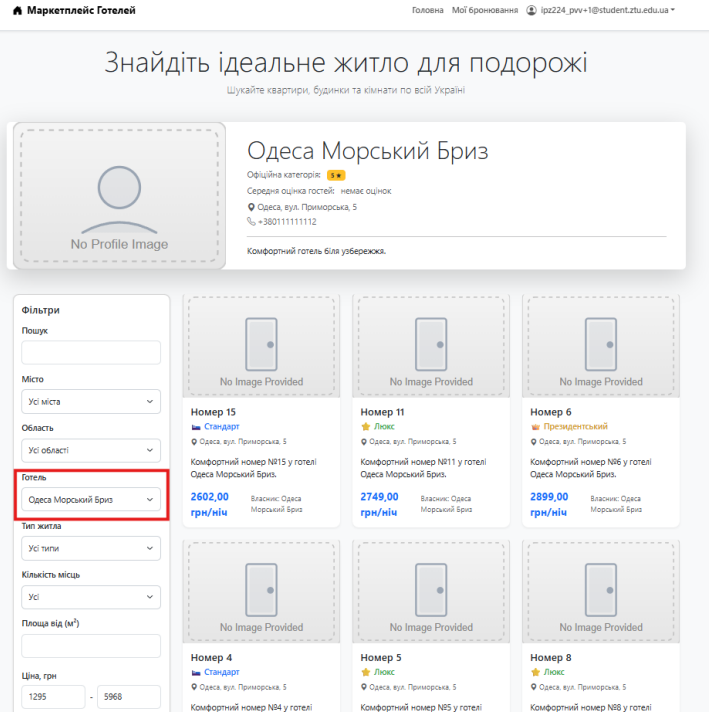


Рис. 3.5. - Сторінка готелю

Сторінка деталей житла надає повну інформацію про обраний об’єкт: назву, фото, опис, адресу, площу, кількість місць та середній рейтинг. Також відображається інформація про готель (Рис. 3.6). Праворуч розміщено блок бронювання з відображенням вартості проживання за ніч і формою вибору дат заїзду та виїзду, при цьому недоступні дати автоматично блокуються.

Маркетплейс Готелей

Головна

Міє бронювання

ipz224_pvv+1@student.ttu.edu.ua

4974,00 грн / ніч

Забронювати житло

Дата заїзду

Дата виїзду

Забронювати

No Image Provided

Номер 4

Київ, вул. Хрещатик, 10

Готель активний

Комфортний номер №4 у готелі Готель Київ Преміум.

100 м²

3 місяць(я)

Власник: Готель Київ Преміум

Відгуки

Для цього житла ще немає відгуків. Будьте першим!

Рис. 3.6. - Сторінка деталей житла, бронювання та відгуки

Після успішного створення бронювання система автоматично надсилає e-mail користувачу (Рис. 3.7) і перенаправляє користувача на сторінку “Історія моїх бронювань” (Рис. 3.8), де відображається повний список його бронювань. На цій сторінці користувач може переглядати деталі кожного бронювання: обране житло, готель, дати заїзду та виїзду, загальну вартість і поточний статус (очікує підтвердження, підтверджено, скасовано, завершено тощо). Також доступні інструменти пошуку, сортування та пагінації для зручного перегляду великої кількості записів. Для активних бронювань передбачена можливість скасування, а після завершення проживання користувач може перейти до сторінки житла та залишити відгук.

Ваше бронювання створено

External

Вхідні x

DoNotReply <DoNotReply@azurecomm.net>

кому ipz224_pvv+1

Ваше бронювання Створено!

Готель: Готель Київ Преміум

Номер: Номер 4

Дати: 2026-06-02 - 2026-06-05

Очікуйте на лист від власника готелю щодо підтвердження або відхилення вашого бронювання. Зазвичай це займає до 2 годин

Дякуємо, що обрали нас!

Рис. 3.7. - Email-повідомлення про створення бронювання

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				60
Змн.	Арк.	№ докум.	Підпис	Дата		

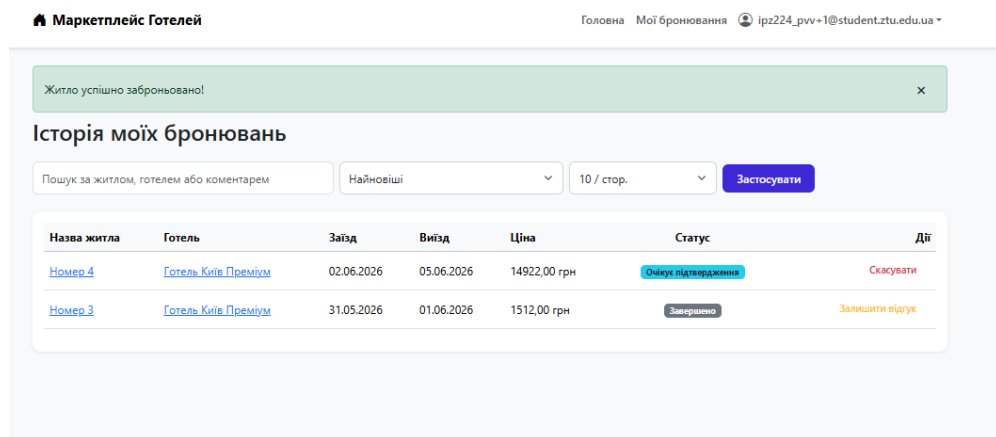


Рис. 3.8. - Сторінка історії бронювань

На сторінці “Керування бронюваннями” (Рис. 3.9) власник готелю переглядає всі заявки на бронювання, отримані від клієнтів, разом із контактними даними користувачів, інформацією про обране житло, датами проживання та поточним статусом.

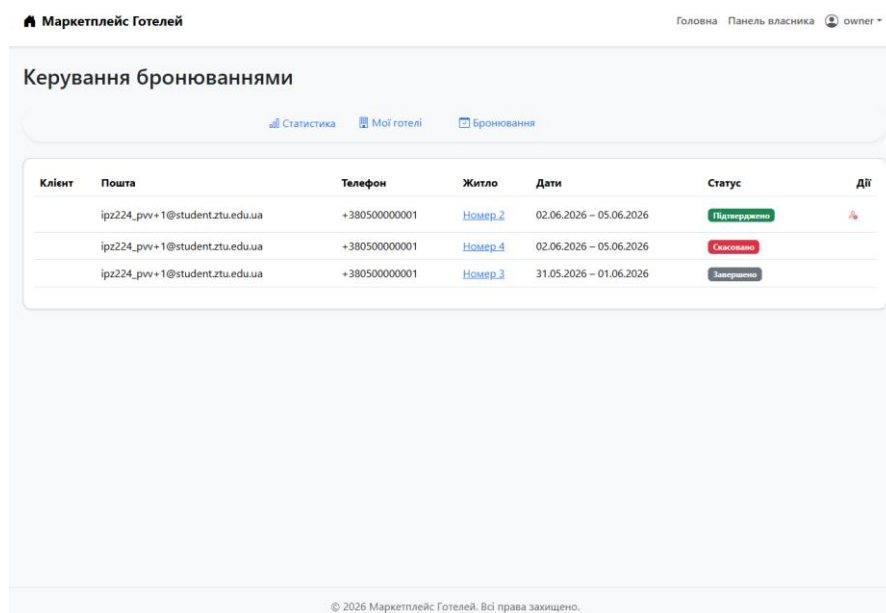


Рис. 3.9. - Сторінка “Керування бронюваннями”

Для бронювань зі статусом «створено» доступні дії підтвердження або скасування. Після підтвердження бронювання його статус змінюється на “підтверджено”, і система автоматично надсилає користувачу email-повідомлення про успішне підтвердження (Рис. 3.10).

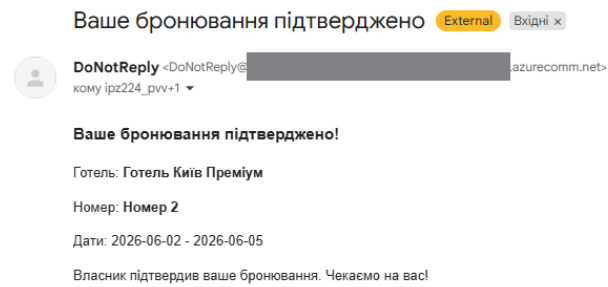


Рис. 3.10. - Email-повідомлення про успішне підтвердження бронювання

У випадку скасування бронювання система змінює статус на “скасовано” та надсилає користувачу відповідне email-повідомлення (Рис. 3.11).

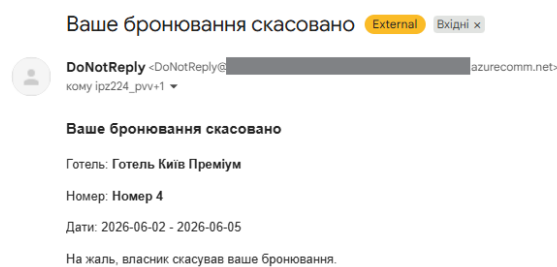


Рис. 3.11. - Email-повідомлення про скасування бронювання

Додатково в системі реалізовано механізм контролю неявок користувачів. Якщо до дати заїзду залишається менше або рівно 24 години, у власника готелю після підтвердження бронювання з’являється додаткова дія «Не прибув» (Рис. 3.12)

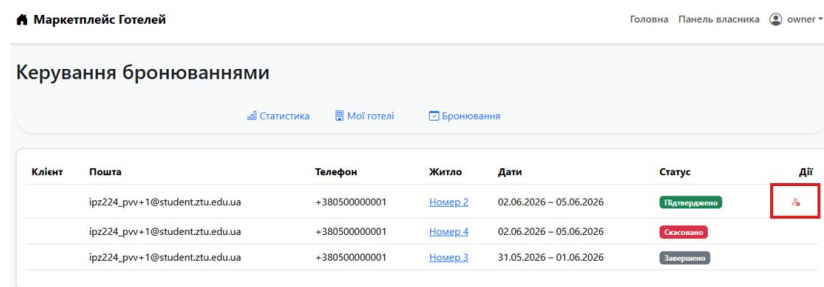


Рис. 3.12. - Можливість позначити бронювання окремим статусом коли користувач забронював, але не прибув

Коли користувач не прибув хоча б раз, йому показується системне попередження у хедері сторінки (Рис. 3.13).

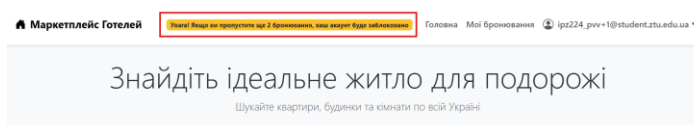


Рис. 3.13. - Системне попередження про блокування

Якщо користувач накопичує три попередження, його акаунт автоматично блокується і при авторизації показується сторінка блокування (Рис. 3.14), яке повідомляє користувача про блокування та необхідність звернення до адміністрації для відновлення доступу.

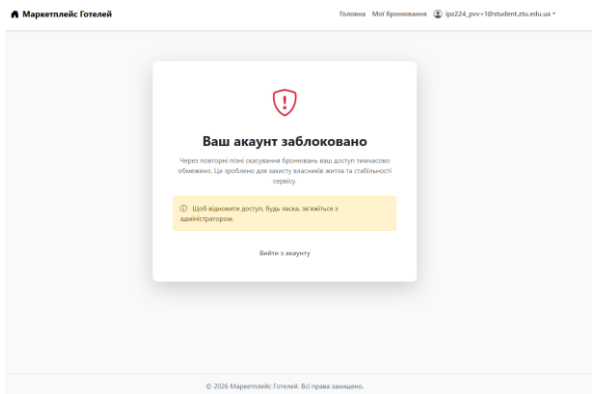


Рис. 3.14. - Сторінка блокування

Далі розглянемо панель керування власника. Перший розділ - статистика (Рис. 3.15), де відображаються ключові показники діяльності: дохід, кількість бронювань, їх статусний розподіл, а також топ житла за тривалістю оренди. Дані представлені у вигляді інтерактивних графіків та таблиць з можливістю фільтрації за готелем і періодом.

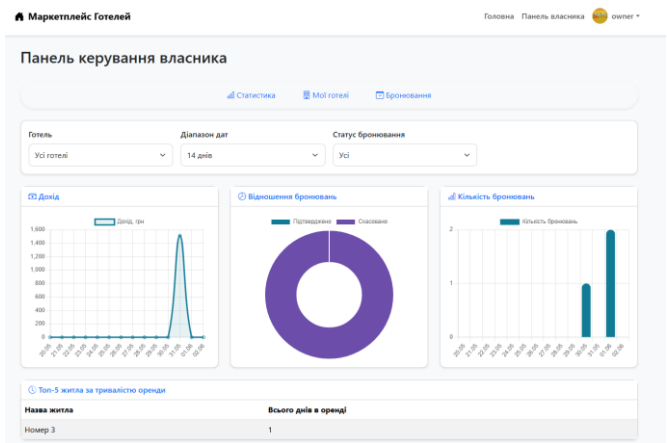


Рис. 3.15. - Панель статистики власника

У розділі “Мої готелі” (Рис. 3.16) власник може переглядати список своїх готелів, додавати нові, редагувати або видаляти існуючі. Також доступне керування станом активності готелю та перегляд базової інформації (місто, зірковість, статус активації).

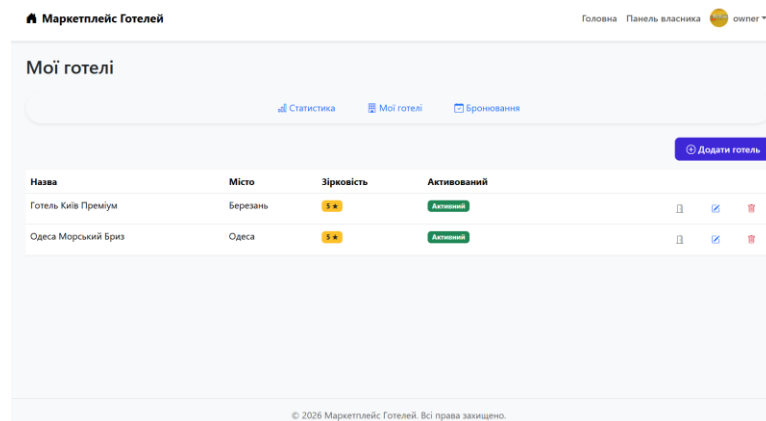


Рис. 3.16. - Панель статистики власника

Після натискання кнопки “керувати номерами” конкретного готелю власник переходить до керування житлом (Рис. 3.17). У цьому розділі він може створювати нові номери, редагувати існуючі та видаляти їх. Для кожного об’єкта відображається фото, назва, адреса та ціна за ніч.

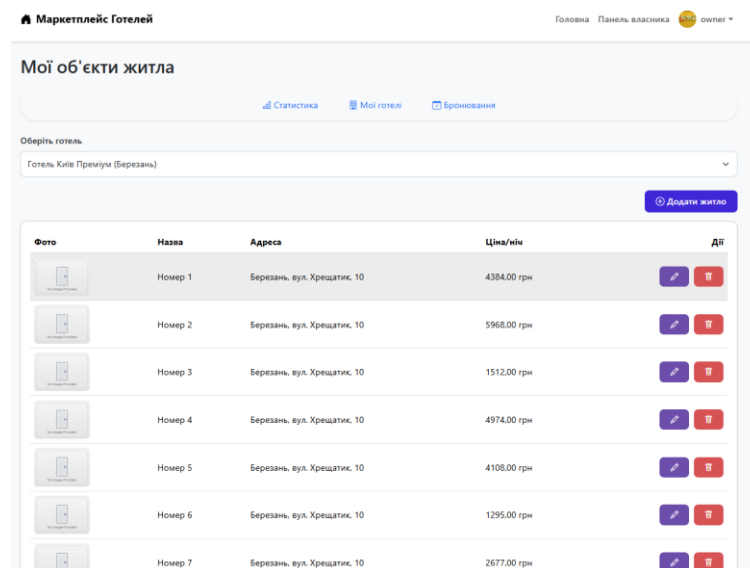


Рис. 3.17. - Керування номерами готелю

У формі редагування готелю (Рис. 3.18) власник може оновлювати інформацію про заклад, змінювати фото та основні параметри. Додатково доступні функції подачі запиту на верифікацію готелю та окремого запиту на присвоєння зірковості. Ці запити передаються на модерацію адміністрації.

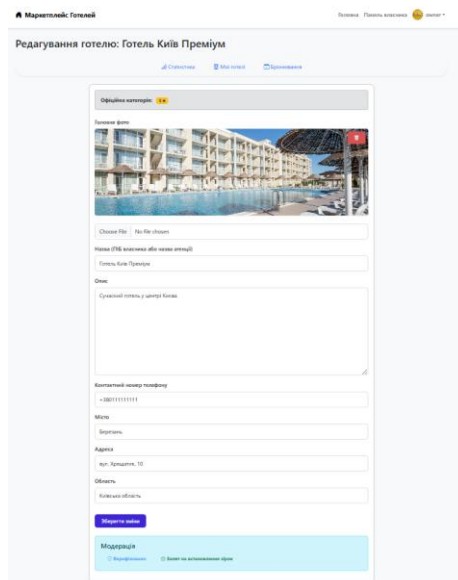


Рис. 3.18. - Сторінка редагування готелю з модераційними запитами

Панель адміністратора забезпечує централізоване управління всією платформою: користувачами, готелями, бронюваннями та модераційними запитами. Інтерфейс розділений на кілька ключових модулів.

Першим блоком є статистика системи (Рис. 3.19), де адміністратор отримує узагальнену аналітику: динаміку нових користувачів за останні 14 днів, співвідношення ролей у системі (клієнти, власники, адміністратори), кількість бронювань у часовому розрізі та рейтингові показники житла. Окремо відображаються топ-об'єкти за рейтингом та популярністю, що дозволяє оцінювати загальну активність платформи та приймати управлінські рішення.

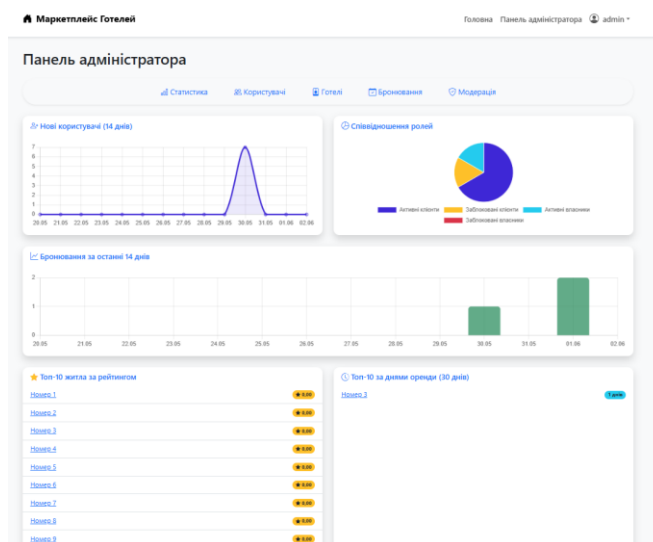


Рис. 3.19. - Сторінка статистики системи

Наступним розділом є керування користувачами (Рис. 3.20). Адміністратор може здійснювати пошук користувачів за email, іменем або телефоном, змінювати їх роль у системі та керувати статусом акаунта. Перед блокуванням або розблокуванням користувача система обов'язково запитує підтвердження дії через модальне вікно. Заблоковані користувачі втрачають доступ до функціоналу платформи до моменту повторного розблокування.

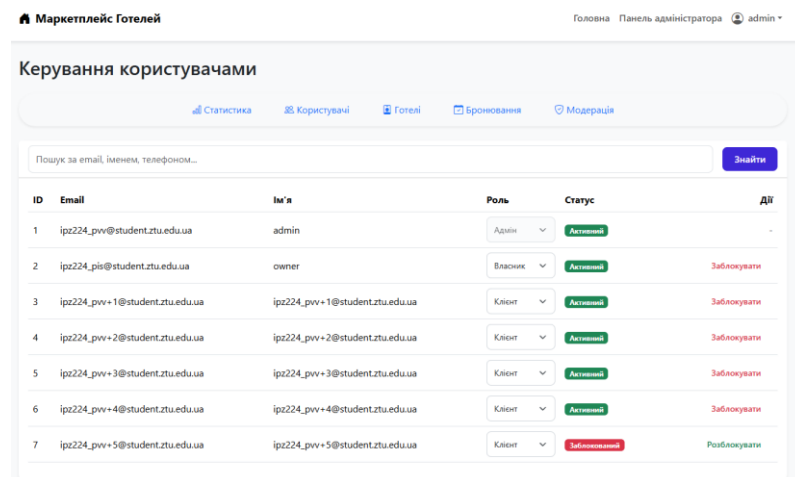


Рис. 3.20. - Сторінка керування користувачами

Розділ керування готелями (Рис. 3.21) дозволяє адміністратору переглядати всі зареєстровані готелі на платформі, включаючи інформацію про власника, контактні дані та статус активності. Адміністратор може переходити до модерації конкретного об'єкта у випадку, якщо готель ще не активований або очікує перевірки.

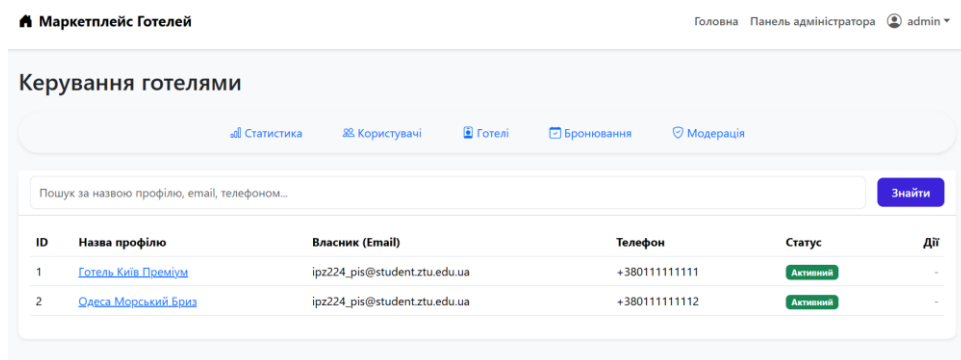


Рис. 3.21. - Сторінка керування готелями

Окремий модуль відповідає за відображення усіх бронювань (Рис. 3.22) на платформі. Тут адміністратор має можливість переглядати всі бронювання, фільтрувати їх за статусами та аналізувати взаємодію між

клієнтами, власниками та житлом. Це дозволяє контролювати життєвий цикл бронювання від створення до завершення або скасування.

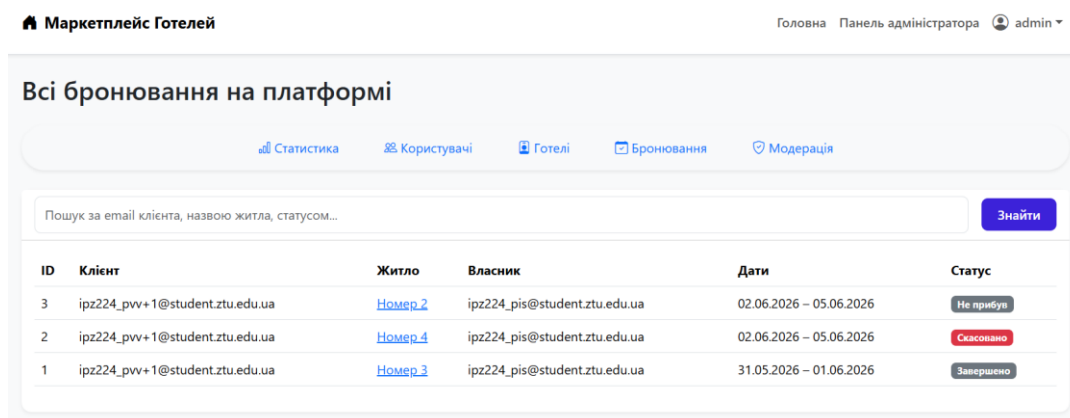


Рис. 3.22. - Сторінка відображення усіх бронювань

Найважливішим блоком є модерація (Рис. 3.23), де адміністратор опрацьовує запити від власників готелів. Сюди входять запити на верифікацію готелю та встановлення зірковості. Кожен запит має статус (очікує, підтверджено або відхилено), а також може містити додаткові дані, наприклад обґрунтування або сертифікати. Після розгляду запиту адміністратор приймає рішення - підтверджує або відхиляє його із можливістю додати коментар.

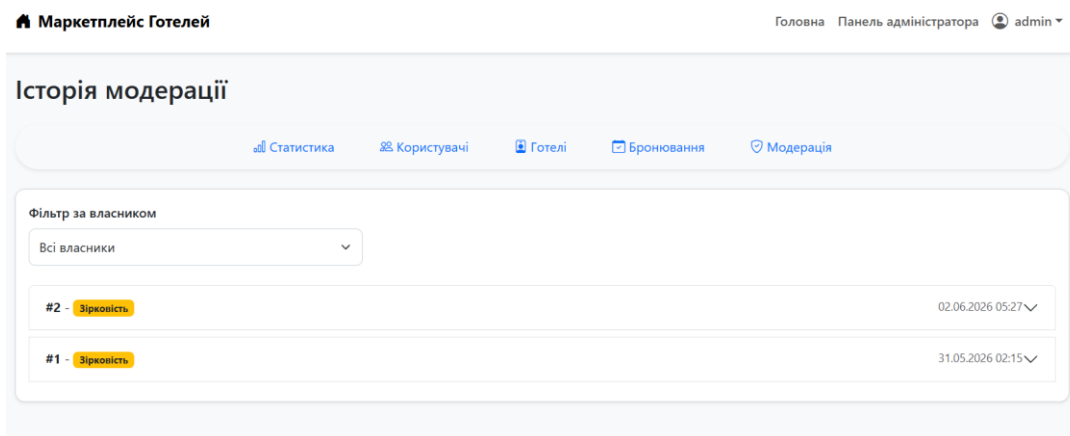


Рис. 3.23. - Сторінка модерація

Після того як адміністратор погоджує модераційний запит, система автоматично надсилає власнику готелю електронного листа з повідомленням про успішне підтвердження (Рис. 3.24). Це дозволяє власнику одразу дізнатися про зміну статусу його об'єкта та продовжити роботу з готелем.

Зірковість готелю відмовлено Вхідні x

кому мені ▼

06:45 (2 хвилини тому) ☆

Ваш запит на зірковість відмовлено!

Готель: Готель Київ Преміум

Зірковість: 5★

Коментар: погана якість сертифікату, виправте і ще раз надішліть запит

Рис. 3.24. - Електронний лист з повідомленням про успішне підтвердження

У випадку відхилення модераційного запиту власник також отримує автоматичний лист, у якому зазначається причина відмови та рекомендації щодо виправлення недоліків (Рис. 3.25). Це забезпечує прозорий процес модерації та дозволяє власникам покращувати свої об'єкти відповідно до вимог платформи.

Зірковість готелю підтверджено Вхідні x

кому мені ▼

06:45 (2 хвилини тому)

Ваш запит на зірковість підтверджено!

Готель: Готель Київ Преміум

Нова зірковість: 5★

Коментар: погоджено

Рис. 3.25. - Електронний лист з повідомленням про відмову в запиті

3.3 Тестування програмної системи

Тестування є одним із найважливіших етапів розробки програмного забезпечення, оскільки саме на цьому етапі здійснюється перевірка відповідності реалізованого функціоналу поставленим вимогам. Якісно проведене тестування дозволяє виявити помилки та недоліки системи до її введення в експлуатацію, забезпечити стабільність роботи програмного продукту та підвищити рівень задоволеності кінцевих користувачів. Для вебплатформи бронювання житла особливого значення набуває перевірка

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		68

бізнес-логіки, коректності обробки даних, надійності механізмів автентифікації та безпеки збереження інформації.

У процесі розробки вебплатформи було застосовано комплексний підхід до тестування, який поєднував ручне тестування користувацького інтерфейсу та перевірку серверної логіки шляхом виконання функціональних сценаріїв для різних ролей користувачів. Такий підхід дозволив оцінити систему не лише з технічної точки зору, але й з позиції кінцевого користувача. Основна увага приділялася перевірці коректності роботи ключових модулів системи, серед яких реєстрація та авторизація користувачів, пошук житла, бронювання номерів, керування готелями та кімнатами, система відгуків, механізми модерації та адміністративна панель.

Першим етапом тестування стала перевірка роботи користувацького інтерфейсу. Метою даного етапу було підтвердження правильності відображення сторінок, навігаційних елементів, форм введення даних та інших компонентів інтерфейсу. Особлива увага приділялася адаптивності сторінок та коректному відображенню контенту в різних браузерях. Також перевірялася працездатність навігаційного меню, посилань, кнопок та інших інтерактивних елементів.

Ручне тестування (Рис. 3.26) застосовувалося для перевірки взаємодії користувача із системою через вебінтерфейс. Під час тестування виконувалися типові сценарії роботи гостей, клієнтів, власників житла та адміністраторів. Для кожного сценарію перевірялися правильність переходів між сторінками, відображення інформації та коректність реакції системи на дії користувача.

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		69

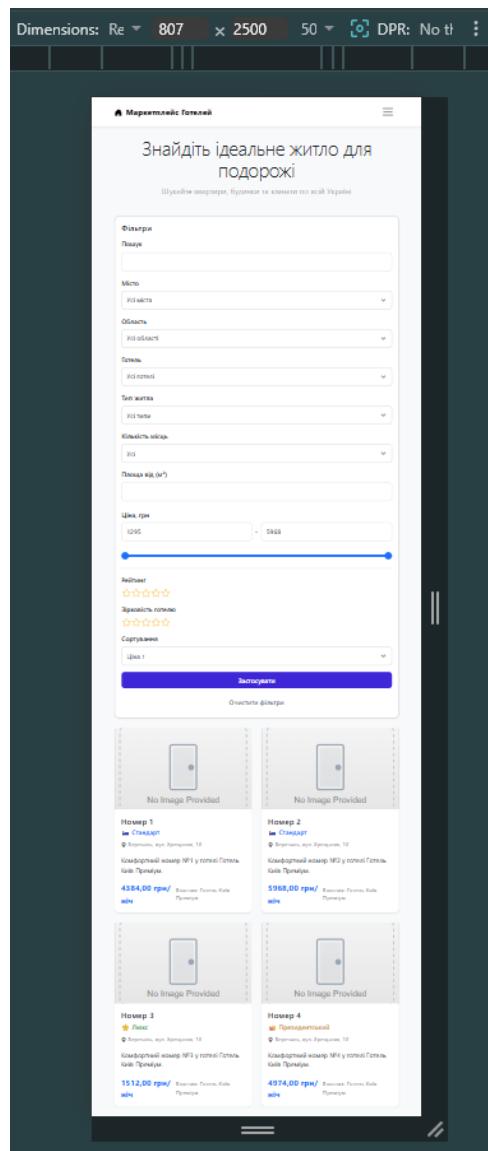


Рис. 3.26. - Ручне тестування головної сторінки системи бронювання житла

Окремий блок перевірок було присвячено модулю реєстрації та авторизації користувачів. Під час тестування перевірялося створення нових облікових записів із використанням унікальної електронної адреси, валідація обов'язкових полів форми та обробка помилкових даних. Також перевірялася робота механізму входу в систему, відновлення доступу до облікового запису та завершення користувацької сесії. Особливу увагу було приділено перевірці розмежування прав доступу між різними ролями користувачів.

Для ролі клієнта було протестовано весь цикл взаємодії із платформою. Насамперед перевірялася робота механізму пошуку житла за різними параметрами, включаючи місто, область, ціновий діапазон та інші характеристики. Було підтверджено коректне застосування фільтрів та

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				70
Змн.	Арк.	№ докум.	Підпис	Дата		

правильність відображення результатів пошуку. Крім того, перевірялася швидкість обробки запитів та відповідність відображених даних інформації, що зберігається в базі даних.

Особливе значення для системи має процес бронювання житла, тому його тестуванню було приділено найбільшу увагу. Для перевірки даного функціоналу використовувалися різні сценарії взаємодії користувача із системою, включаючи успішне створення бронювання, спробу бронювання зайнятого номера, введення некоректних дат та скасування вже існуючого бронювання.

Окремий етап тестування був присвячений перевірці функціоналу бронювання (Рис. 3.27). Здійснювалася перевірка правильності формування бронювань, блокування зайнятих дат, розрахунку вартості проживання та відображення історії бронювань користувача. Також перевірялися механізми підтвердження та скасування бронювання власником готелю, а також автоматичне надсилання електронних повідомлень про зміну статусу бронювання.

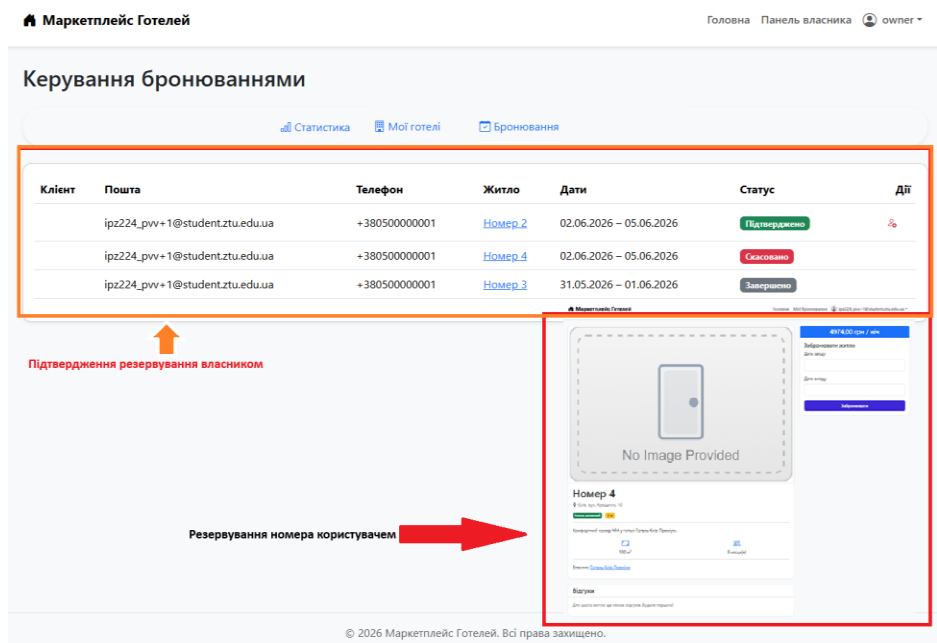


Рис. 3.27. - Тестування процесу створення та підтвердження бронювання

Під час перевірки було встановлено, що система коректно виконує перевірку доступності номера перед створенням бронювання. У випадку

виявлення перетину дат із вже існуючими бронюваннями користувач отримує відповідне повідомлення про неможливість виконання операції. Такий механізм дозволяє уникнути подвійного резервування одного й того самого номера та забезпечує цілісність даних.

Крім того, було перевірено правильність розрахунку загальної вартості проживання. Для цього використовувалися тестові сценарії з різною тривалістю проживання та різними типами номерів. У всіх випадках система виконувала обчислення коректно та відображала користувачу актуальну суму бронювання.

Для ролі власника житла було проведено тестування функціоналу керування готелями та номерами. Перевірялися операції створення, редагування та видалення готелів, додавання нових номерів, зміна характеристик житлових одиниць та оновлення фотографій. Також було протестовано механізм надсилання запитів на верифікацію об'єкта та присвоєння категорії зірковості.

Особлива увага приділялася перевірці коректності роботи панелі власника. Було підтверджено правильне відображення списку створених готелів, статистичних показників та інформації про отримані бронювання. Крім того, перевірялася можливість перегляду інформації про клієнтів та історію взаємодії з ними

Окремо було протестовано механізм керування статусами бронювань. Власник мав можливість підтверджувати або відхиляти отримані заявки, після чого система автоматично змінювала статус бронювання та надсилала відповідне повідомлення клієнту. Усі перевірені сценарії виконувалися коректно та відповідали визначеній бізнес-логіці системи.

Особливу увагу було приділено перевірці механізму контролю неявки гостей. Було протестовано сценарій, за якого за 24 години до початку бронювання власнику стає доступною функція «Не прибув». Після її підтвердження бронювання автоматично переводиться у відповідний статус, а користувач отримує попередження електронною поштою. Також

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		72

перевірено механізм автоматичного блокування акаунта після отримання трьох попереджень та відображення інформаційного повідомлення про блокування в інтерфейсі системи.

Для ролі адміністратора було проведено комплексне тестування адміністративної панелі. Основною метою даного етапу була перевірка можливостей централізованого керування користувачами, готелями, бронюваннями та модераційними запитами.

Для адміністратора було проведено тестування всіх модулів адміністративної панелі, включаючи статистику, керування користувачами, готелями, бронюваннями та модераційними запитами (Рис. 3.28). Перевірялася коректність зміни ролей користувачів, блокування та розблокування акаунтів, а також процеси підтвердження та відхилення запитів на верифікацію готелів і присвоєння зірковості.

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		73

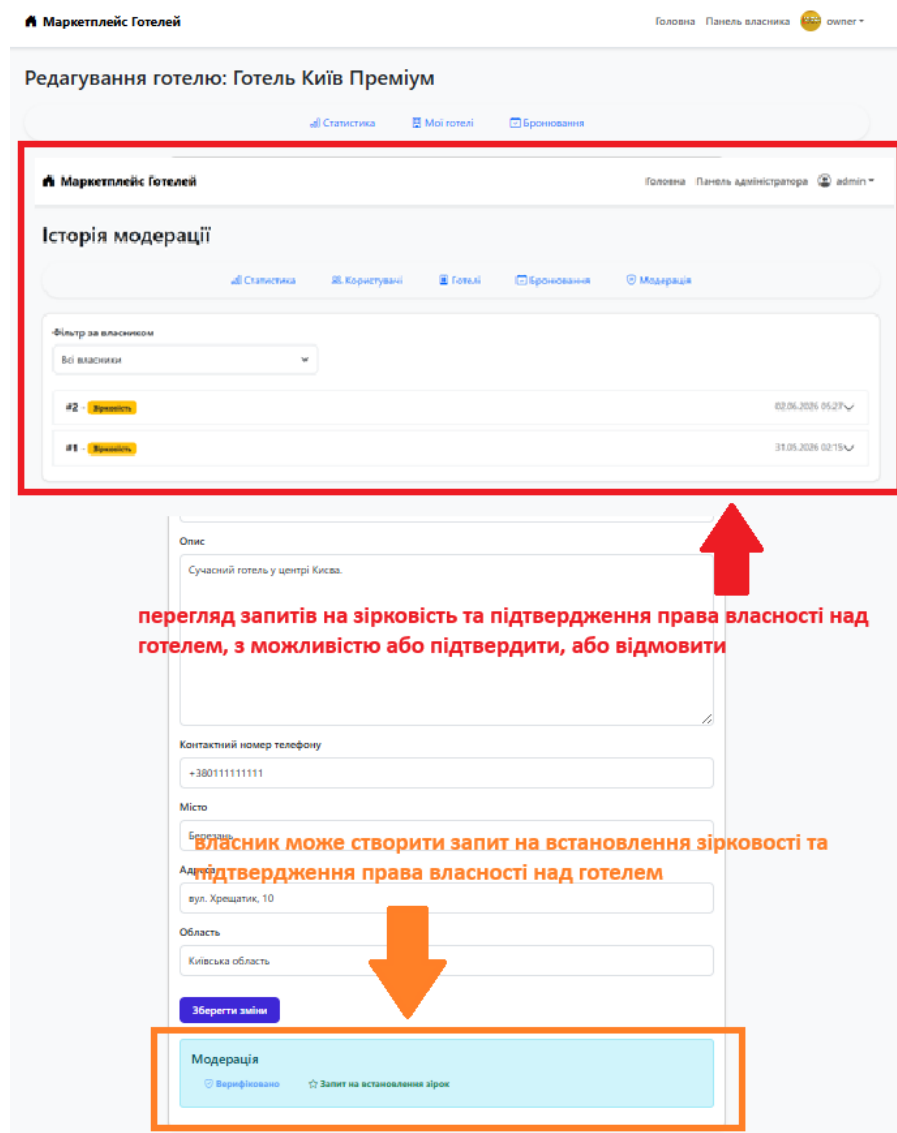


Рис. 3.28. - Тестування адміністративної панелі та процесу модерації

Під час перевірки модераційного функціоналу було підтверджено коректну роботу автоматичних сповіщень власників готелів. У випадку схвалення запиту система надсилає електронний лист із повідомленням про успішне проходження модерації, а при відхиленні — лист із причиною відмови та коментарем адміністратора. Це дозволяє забезпечити прозору взаємодію між адміністрацією платформи та власниками житла

Додатково було проведено тестування механізмів безпеки системи. Перевірялися обмеження доступу до сторінок відповідно до ролі користувача, захист від несанкціонованого доступу до адміністративного функціоналу та коректність завершення користувацьких сесій після виходу з

системи. Також перевірялася робота механізмів валідації даних та обробки помилок введення.

У ході тестування були перевірені сценарії роботи з помилковими даними. Користувачі навмисно вводили некоректні електронні адреси, порожні значення обов'язкових полів, неправильні дати бронювання та інші недопустимі значення. У всіх випадках система коректно виявляла помилки та відображала зрозумілі повідомлення для користувача без порушення роботи застосунку.

Також було проведено перевірку роботи системи після внесення змін до даних. Оновлення інформації про готелі, номери, профілі користувачів та бронювання коректно відображалося у всіх пов'язаних компонентах системи, що підтвердило правильність реалізації механізмів взаємодії між модулями.

Проведене тестування дозволило виявити та усунути більшість помилок на етапі розробки, а також підтвердити коректність реалізації основних бізнес-процесів системи. У результаті було забезпечено стабільну роботу всіх ключових модулів платформи, включаючи пошук житла, бронювання, систему відгуків, механізми модерації та адміністративне керування.

Отримані результати свідчать про те, що розроблена веб-платформа відповідає поставленим функціональним вимогам та готова до експлуатації в реальних умовах використання. У подальшому доцільним є впровадження автоматизованого тестування, зокрема модульних та інтеграційних тестів, що дозволить ще більше підвищити якість програмного забезпечення та спростити процес його супроводу й розвитку.

Висновки до третього розділу

У третьому розділі було розглянуто процес встановлення, налаштування та використання розробленої веб-платформи для бронювання житла. Описано архітектуру розгортання системи, вимоги до програмного й апаратного забезпечення, а також послідовність встановлення та запуску

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		75

застосунку на базі фреймворку Django і системи управління базами даних MySQL.

Детально проаналізовано структуру користувацького інтерфейсу для всіх ролей системи: клієнта, власника готелю та адміністратора. Продемонстровано основні сценарії роботи із платформою, включаючи пошук і фільтрацію житла, створення та керування бронюваннями, роботу з відгуками, управління готелями та номерами, а також виконання адміністративних функцій. Окрему увагу приділено реалізованим механізмам електронних сповіщень, модерації готелів, контролю неявок користувачів та автоматичного блокування облікових записів у разі систематичного порушення правил користування платформою.

Також було проведено комплексне тестування розробленої системи, яке охоплювало перевірку функціональних можливостей для всіх категорій користувачів. У процесі тестування підтверджено коректність роботи механізмів авторизації, бронювання, керування готелями та номерами, модерації, надсилання електронних повідомлень і функціонування адміністративної панелі. Отримані результати засвідчили стабільну роботу системи, відповідність реалізованого функціоналу поставленим вимогам та готовність платформи до практичного використання і подальшого розвитку.

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		76

ВИСНОВКИ

У результаті виконання кваліфікаційної роботи було розроблено та реалізовано веб-платформу для бронювання житла, яка забезпечує взаємодію між клієнтами, власниками готелів та адміністраторами в межах єдиного інформаційного середовища. Під час виконання роботи було проведено аналіз існуючих сервісів бронювання житла, визначено їх переваги та недоліки, а також сформовано функціональні та нефункціональні вимоги до системи.

Для реалізації програмного продукту було обрано модульну монолітну архітектуру, що забезпечує простоту розробки, централізоване управління даними та можливість подальшого масштабування. Серверну частину реалізовано з використанням фреймворку Django, для зберігання даних використано СУБД MySQL, а клієнтський інтерфейс побудовано за допомогою HTML, CSS та JavaScript.

У процесі розробки реалізовано основні функціональні можливості системи: реєстрацію та авторизацію користувачів, пошук і фільтрацію житла, бронювання номерів, систему відгуків і рейтингів, особисті кабінети користувачів, панель керування власника готелю, адміністративну панель, механізм модерації готелів та систему електронних сповіщень. Додатково впроваджено механізм контролю неявки гостей та автоматичного блокування користувачів у разі систематичних порушень правил платформи.

Проведене тестування підтвердило коректність роботи реалізованого функціоналу, стабільність основних бізнес-процесів та відповідність системи поставленим вимогам. Отримані результати свідчать про готовність розробленого програмного продукту до практичного використання та подальшого розвитку шляхом розширення функціональних можливостей і вдосконалення механізмів автоматизованого тестування.

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		77

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Django documentation. Режим доступу:
<https://docs.djangoproject.com/en/6.0/intro/tutorial01/>
2. Python documentation. Режим доступу:
<https://docs.python.org/3/tutorial/classes.html/>
3. MySQL 8.0 reference manual. Режим доступу:
<https://dev.mysql.com/doc/refman/8.0/en/innodb-storage-engine.html/>
4. Bootstrap documentation. Режим доступу:
<https://getbootstrap.com/docs/5.3/getting-started/introduction/>
5. Chart.js documentation. Режим доступу:
<https://www.chartjs.org/docs/latest/getting-started/>
6. HTML: hypertext markup language. Режим доступу:
<https://developer.mozilla.org/en-US/docs/Web/HTML/Element>
7. CSS: cascading style sheets. Режим доступу:
https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Grid_Layout
8. JavaScript guide. Режим доступу: <https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide>
9. Authentication in django. Режим доступу:
<https://docs.djangoproject.com/en/stable/topics/auth/default/>
10. Django ORM documentation. Режим доступу:
<https://docs.djangoproject.com/en/stable/topics/db/queries/>
11. Working with forms. Режим доступу:
<https://docs.djangoproject.com/en/stable/topics/forms/modelforms/>
12. Class-based views. Режим доступу:
<https://docs.djangoproject.com/en/stable/topics/class-based-views/generic-display/>
13. Sending email with django. Режим доступу:
<https://docs.djangoproject.com/en/stable/topics/email/>
14. Pillow documentation. Режим доступу:
<https://pillow.readthedocs.io/en/stable/handbook/tutorial.html>

15. Nominatim API documentation. Режим доступу:
<https://nominatim.org/release-docs/latest/api/Search/>
16. Hotel management system. Режим доступу:
<https://www.geeksforgeeks.org/system-design-for-hotel-management-system/>
17. Fowler M. Patterns of Enterprise Application Architecture. Режим доступу:
<https://martinfowler.com/eaCatalog/domainModel.html>
18. HTTP overview. Режим доступу: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview>
19. OWASP top 10 web application security risks. Режим доступу:
<https://owasp.org/www-project-top-ten/>
20. What is HTTPS?. Режим доступу:
<https://www.cloudflare.com/learning/ssl/what-is-https/>
21. Git documentation. Режим доступу: <https://git-scm.com/docs/git-branch>
22. PyCharm documentation. Режим доступу:
<https://www.jetbrains.com/help/pycharm/django-support.html>
23. Bootstrap icons. Режим доступу:
<https://icons.getbootstrap.com/icons/building/>
24. Web accessibility initiative (WAI). Режим доступу:
<https://www.w3.org/WAI/standards-guidelines/wcag/>
25. Sommerville I. Software Engineering. 10th ed. Boston: Pearson Education, 2016. 816 p.
26. Gamma J., Helm R., Johnson R., Vlissides J. Design Patterns: Elements of Reusable Object-Oriented Software. Boston: Addison-Wesley, 1994. 395 p.
27. Django deployment checklist. Режим доступу:
<https://docs.djangoproject.com/en/stable/howto/deployment/checklist/>
28. Django REST framework documentation. Режим доступу:
<https://www.django-rest-framework.org/api-guide/serializers/>
29. Web security guidelines. Режим доступу: https://developer.mozilla.org/en-US/docs/Web/Security/Same-origin_policy

30.OWASP web security testing guide. Режим доступу:

https://owasp.org/www-project-web-security-testing-guide/stable/4-Web_Application_Security_Testing/

		Парова В.В.			ІПЗ.КР.Б-121-26-ІЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		80

ДОДАТКИ

		Парова В.В.			ІПЗ.КР.Б-121-26-ІЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		81

Лістинг вихідного коду

```

def cancellation_warning(request):
    if request.user.is_authenticated and hasattr(request.user, "late_cancellations"):
        late = request.user.late_cancellations
        remaining = max(0, 3 - late)
        return {
            "late_cancellations": late,
            "show_cancellations_warning": late != 0,
            "remaining_cancellations": remaining,
            "user_is_blocked": not getattr(request.user, "is_active", False),
        }
    return {}

import os
from pathlib import Path

BASE_DIR = Path(__file__).resolve().parent.parent

SECRET_KEY = 'django-insecure-your-secret-key-here'

DEBUG = True

ALLOWED_HOSTS = []

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'widget_tweaks',
    'booking',
]

MIDDLEWARE = [
    'django.middleware.security.SecurityMiddleware',

```

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		82

```

'django.contrib.sessions.middleware.SessionMiddleware',
'django.middleware.common.CommonMiddleware',
'django.middleware.csrf.CsrfViewMiddleware',
'django.contrib.auth.middleware.AuthenticationMiddleware',
'django.contrib.messages.middleware.MessageMiddleware',
'django.middleware.clickjacking.XFrameOptionsMiddleware',
'booking.middlewares.blocked_user_middleware.BlockedUserMiddleware'
]

```

```

ROOT_URLCONF = 'hotel_booking_project.urls'

```

```

TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [os.path.join(BASE_DIR, 'booking/templates')],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',
                'booking.context_processors.cancellation_warning'
            ],
        },
    },
]

```

```

WSGI_APPLICATION = 'hotel_booking_project.wsgi.application'

```

```

# Налаштування бази даних (MySQL, як у вимогах)
# DATABASES = {
#     'default': {
#         'ENGINE': 'django.db.backends.mysql',
#         'NAME': 'hotel_db',
#         'USER': 'root',
#         'PASSWORD': 'your_password',
#         'HOST': '127.0.0.1',
#         'PORT': '3306',
#     }
# }

```

		Парова В.В.			ІПЗ.КР.Б-121-26-ІЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		83

```
#}
```

```
# Налаштування бази даних SQL Lite для розробки
```

```
DATABASES = {
```

```
    'default': {
```

```
        'ENGINE': 'django.db.backends.sqlite3',
```

```
        'NAME': BASE_DIR / 'db.sqlite3',
```

```
    }
```

```
}
```

```
AUTH_PASSWORD_VALIDATORS = [
```

```
    {'NAME': 'django.contrib.auth.password_validation.UserAttributeSimilarityValidator'},
```

```
    {'NAME': 'django.contrib.auth.password_validation.MinimumLengthValidator'},
```

```
    {'NAME': 'django.contrib.auth.password_validation.CommonPasswordValidator'},
```

```
    {'NAME': 'django.contrib.auth.password_validation.NumericPasswordValidator'},
```

```
]
```

```
LANGUAGE_CODE = 'uk-ua'
```

```
TIME_ZONE = 'Europe/Kiev'
```

```
USE_I18N = True
```

```
USE_TZ = True
```

```
STATIC_URL = '/static/'
```

```
STATICFILES_DIRS = [os.path.join(BASE_DIR, 'static')]
```

```
MEDIA_URL = '/media/'
```

```
MEDIA_ROOT = os.path.join(BASE_DIR, 'media')
```

```
AUTH_USER_MODEL = 'booking.CustomUser'
```

```
AUTHENTICATION_BACKENDS = [
```

```
    'booking.backends.EmailBackend',
```

```
    'django.contrib.auth.backends.ModelBackend',
```

```
]
```

```
AZURE_EMAIL_CONNECTION_STRING = ""
```

```
AZURE_EMAIL_SENDER = ""
```

```
INITIAL_ADMIN_EMAIL = ""
```

```
INITIAL_ADMIN_PHONE_NUMBER = ""
```

		Парова В.В.			ІПЗ.КР.Б-121-26-ІЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		84

```
DEFAULT_AUTO_FIELD = 'django.db.models.BigAutoField'
```

```
LOGIN_REDIRECT_URL = '/'
```

```
LOGOUT_REDIRECT_URL = '/'
```

```
LOGIN_URL = 'login'
```

```
import os
```

```
from django.core.wsgi import get_wsgi_application
```

```
os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'hotel_booking_project.settings')
```

```
application = get_wsgi_application()
```

```
import os
```

```
from django.core.asgi import get_asgi_application
```

```
os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'hotel_booking_project.settings')
```

```
application = get_asgi_application()
```

```
from datetime import timedelta, date, datetime
```

```
def is_late_cancellation(booking):
```

```
    now = date.today()
```

```
    return now > (booking.check_in_date - timedelta(hours=24))
```

```
def register_late_cancellation(user):
```

```
    today = datetime.today().date()
```

```
    history = user.last_cancellation_dates
```

```
    history.append(today.isoformat())
```

```
    four_months_ago = today - timedelta(days=120)
```

```
    history = [d for d in history if datetime.fromisoformat(d).date() >= four_months_ago]
```

```
    user.last_cancellation_dates = history
```

```
    user.late_cancellations = len(history)
```

		Парова В.В.			ІПЗ.КР.Б-121-26-ІЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		85

```

if user.late_cancellations == 1:
    user.save()
    return "warning"

if user.late_cancellations >= 3:
    user.is_active = False

user.save()
return "blocked" if not user.is_active else "ok"

```

```

from django.urls import path

```

```

from .views.auth_views import (
    CustomLoginView,
    register,
    custom_logout,
    change_password,
)

```

```

from .views.public_views import (
    home,
    hotel_detail,
    room_detail,
    property_ratings_json,
    load_more_properties,
    blocked_info
)

```

```

from .views.client_views import (
    profile,
    edit_profile,
    my_reviews,
    my_bookings,
    book_room,
    cancel_booking,
    add_rating,
    delete_rating,
)

```

		Парова В.В.			ІПЗ.КР.Б-121-26-ІЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		86

```

from .views.owner_views import (
    owner_dashboard,
    manage_hotels,
    manage_rooms,
    add_room,
    edit_room,
    delete_room,
    manage_bookings,
    confirm_booking,
    reject_booking,
    complete_booking,
    add_hotel,
    edit_hotel,
    request_hotel_verification,
    request_hotel_stars,
    delete_hotel,
    mark_no_show
)

```

```

from .views.admin_views import (
    admin_dashboard,
    admin_manage_users,
    admin_toggle_user_active,
    admin_change_user_role,
    admin_manage_landlords,
    admin_all_bookings,
)

```

```

from .views.moderation_views import (
    moderation_queue,
    approve_hotel,
    reject_hotel,
    approve_stars,
    reject_stars,
)

```

```
urlpatterns = [
```

```
# =====
```

```
# PUBLIC PAGES
```

		Парова В.В.			ІПЗ.КР.Б-121-26-ІЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		87

```

# =====
path("", home, name='home'),
path('landlord/<int:hotel_id>/', hotel_detail, name='hotel_detail'),
path('property/<int:room_id>/', room_detail, name='room_detail'),
path('property/<int:room_id>/ratings/', property_ratings_json, name='property_ratings_json'),
path("load-more/", load_more_properties, name="load_more_properties"),

# =====
# AUTH & PROFILE
# =====

path('register/', register, name='register'),
path('login/', CustomLoginView.as_view(), name='login'),
path('logout/', custom_logout, name='logout'),

path('profile/', profile, name='profile'),
path('profile/edit/', edit_profile, name='edit_profile'),
path('profile/change_password/', change_password, name='change_password'),

# =====
# CLIENT FUNCTIONALITY
# =====

path('my_reviews/', my_reviews, name='my_reviews'),
path('my-bookings/', my_bookings, name='my_bookings'),

path('book/<int:room_id>/', book_room, name='book_room'),
path('cancel-booking/<int:booking_id>/', cancel_booking, name='cancel_booking'),

path('property/<int:room_id>/rate/', add_rating, name='add_rating'),
path('rating/<int:rating_id>/delete/', delete_rating, name='delete_rating'),

# =====
# OWNER FUNCTIONALITY
# =====

# Dashboard
path('owner/dashboard/', owner_dashboard, name='owner_dashboard'),

# Hotels
path('owner/hotels/', manage_hotels, name='manage_hotels'),
path('owner/hotels/add/', add_hotel, name='add_hotel'),

```

		Парова В.В.			ІПЗ.КР.Б-121-26-ІЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		88

```

    path('owner/hotels/<int:hotel_id>/edit/', edit_hotel, name='edit_hotel'),
        path("owner/hotels/<int:hotel_id>/request-verification/", request_hotel_verification,
name="request_hotel_verification"),
    path("owner/hotels/<int:hotel_id>/request-stars/", request_hotel_stars, name="request_hotel_stars"),
    path('owner/hotel/<int:hotel_id>/delete/', delete_hotel, name='delete_hotel'),

# Rooms
path('owner/properties/', manage_rooms, name='manage_rooms'),
path('owner/properties/add/', add_room, name='add_room'),
path('owner/properties/edit/<int:room_id>/', edit_room, name='edit_room'),
path('owner/properties/delete/<int:room_id>/', delete_room, name='delete_room'),

# Bookings
path('owner/bookings/', manage_bookings, name='manage_bookings'),
path('owner/booking/confirm/<int:booking_id>/', confirm_booking, name='confirm_booking'),
path('owner/booking/reject/<int:booking_id>/', reject_booking, name='reject_booking'),
path('owner/booking/complete/<int:booking_id>/', complete_booking, name='complete_booking'),
path('owner/bookings/<int:booking_id>/no-show/', mark_no_show, name='mark_no_show'),

# =====
#  ADMIN FUNCTIONALITY
# =====

path('platform-admin/dashboard/', admin_dashboard, name='admin_dashboard'),

path('platform-admin/users/', admin_manage_users, name='admin_manage_users'),
    path('platform-admin/user/toggle-active/<int:user_id>/', admin_toggle_user_active,
name='admin_toggle_user_active'),
    path('platform-admin/user/change-role/<int:user_id>/', admin_change_user_role,
name='admin_change_user_role'),

path('platform-admin/landlord-profiles/', admin_manage_landlords, name='admin_manage_landlords'),

path('platform-admin/bookings/', admin_all_bookings, name='admin_all_bookings'),

# =====
#  MODERATION PANEL
# =====

path('platform-admin/moderation/', moderation_queue, name='moderation_queue'),
path('platform-admin/moderation/hotel/approve/<int:request_id>/', approve_hotel, name='approve_hotel'),
path('platform-admin/moderation/hotel/reject/<int:request_id>/', reject_hotel, name='reject_hotel'),

```

		Парова В.В.			ІПЗ.КР.Б-121-26-ІЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		89

```

    path('platform-admin/moderation/stars/approve/<int:request_id>/', approve_stars, name='approve_stars'),
    path('platform-admin/moderation/stars/reject/<int:request_id>/', reject_stars, name='reject_stars'),

    path("blocked/", blocked_info, name="blocked_info")
]

def is_client(user): return user.is_authenticated and user.role == 'client'

def is_owner(user): return user.is_authenticated and user.role == 'owner'

def is_admin(user): return user.is_authenticated and user.role == 'admin'

from django.contrib.auth.models import AbstractUser
from django.db import models
from django.conf import settings
from django.db.models import Avg
from .constants import UKRAINE_CITIES_BY_OBLAST

class CustomUser(AbstractUser):
    ROLE_CHOICES = (
        ('client', 'Клієнт'),
        ('owner', 'Власник'),
        ('admin', 'Адміністратор'),
    )
    email = models.EmailField(unique=True, verbose_name="Email")
    role = models.CharField(max_length=10, choices=ROLE_CHOICES, default='client',
        verbose_name="Роль")
    phone_number = models.CharField(max_length=20, unique=True, verbose_name="Номер телефону")
    avatar = models.ImageField(upload_to='avatars/', null=True, blank=True, verbose_name="Фото профі-
        лю")
    late_cancellations = models.PositiveIntegerField(default=0)
    last_cancellation_dates = models.JSONField(default=list)

    USERNAME_FIELD = 'email'
    REQUIRED_FIELDS = ['username']

    def __str__(self):
        return self.email

class Hotel(models.Model):

```

		Парова В.В.			ІПЗ.КР.Б-121-26-ІЗ	Арк.
		Єфремов Ю.М.				90
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        owner = models.ForeignKey(settings.AUTH_USER_MODEL, on_delete=models.CASCADE,
related_name='landlord_profiles',
        verbose_name="Власник")
name = models.CharField(max_length=255, verbose_name="Назва (ПІБ власника або назва агенції)")
description = models.TextField(verbose_name="Опис")
main_photo = models.ImageField(upload_to='landlord_photos/', verbose_name="Фото профі-
лю/логотипу", blank=True,
        null=True)
city = models.CharField(max_length=100, verbose_name = "Місто")
address = models.CharField(max_length=255, verbose_name = "Адреса")
phone_number = models.CharField(max_length=20, verbose_name="Контактний номер телефону")
is_active = models.BooleanField(default=True, verbose_name="Активний")
stars = models.PositiveSmallIntegerField(default=0)
certificate_link = models.URLField(null=True, blank=True)
stars_justification_link = models.URLField(null=True, blank=True)
is_approved = models.BooleanField(default=False)
created_at = models.DateTimeField(auto_now_add=True)
oblast = models.CharField(
    max_length=100,
    choices=[(oblast, oblast) for oblast in UKRAINE_CITIES_BY_OBLAST.keys()],
    verbose_name="Область"
)

def __str__(self):
    return self.name

class Room(models.Model):
    ROOM_GRADES = (
        ('standard', 'Стандарт'),
        ('lux', 'Люкс'),
        ('president', 'Президентський'),
    )

    hotel = models.ForeignKey(Hotel, on_delete=models.CASCADE, related_name='properties',
        verbose_name="Профіль власника")
name = models.CharField(max_length=255, verbose_name="Назва житла")
price_per_night = models.DecimalField(max_digits=10, decimal_places=2, verbose_name="Ціна за ніч")
description = models.TextField(blank=True, null=True, verbose_name="Опис житла")
main_photo = models.ImageField(upload_to='property_photos/', verbose_name="Головне фото",
blank=True, null=True)

```

		Парова В.В.			ПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				91
Змн.	Арк.	№ докум.	Підпис	Дата		

```

number_of_rooms = models.PositiveIntegerField(default=1, verbose_name="Кількість місць")
square_meters = models.PositiveIntegerField(verbose_name="Площа (м²)")
grade = models.CharField(max_length=20, choices=ROOM_GRADES, default='standard',
verbose_name="Тип номера")

@property
def average_rating(self):
    return self.ratings.all().aggregate(avg_score=Avg('score'))['avg_score'] or 0.0

def __str__(self):
    return f"{self.name} ({self.hotel.city})"

class Booking(models.Model):
    STATUS_CHOICES = (
        ('created', 'Створене'),
        ('confirmed', 'Підтверджене'),
        ('cancelled', 'Скасоване'),
        ('completed', 'Завершене'),
    )
    user = models.ForeignKey(settings.AUTH_USER_MODEL, on_delete=models.CASCADE,
related_name='bookings',
        verbose_name="Клієнт")
    room = models.ForeignKey(Room, on_delete=models.CASCADE, related_name='bookings',
verbose_name="Житло")
    check_in_date = models.DateField(verbose_name="Дата заїзду")
    check_out_date = models.DateField(verbose_name="Дата виїзду")
    total_price = models.DecimalField(max_digits=10, decimal_places=2, verbose_name="Загальна вар-
мість")
    status = models.CharField(max_length=20, choices=STATUS_CHOICES, default='created',
verbose_name="Статус")
    created_at = models.DateTimeField(auto_now_add=True)

    def __str__(self):
        return f"Бронювання {self.id} для {self.user.email}"

class Rating(models.Model):
    user = models.ForeignKey(settings.AUTH_USER_MODEL, on_delete=models.CASCADE,
related_name='ratings',
        verbose_name="Автор відгуку")

```

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		92

```

        room = models.ForeignKey(Room, on_delete=models.CASCADE, related_name='ratings', verbose_name="Житло")
        score = models.PositiveIntegerField(choices=[(i, i) for i in range(1, 6)], verbose_name="Оцінка")
        comment = models.TextField(max_length=500, blank=True, null=True, verbose_name="Коментар")
        created_at = models.DateTimeField(auto_now_add=True)

class Meta:
    unique_together = ('user', 'room')

    def __str__(self):
        return f"Рейтинг {self.score} від {self.user.email} для {self.room.name}"

class ModerationRequest(models.Model):
    REQUEST_TYPES = (
        ('hotel_stars', 'Запит на встановлення зірок готелю'),
        ('hotel_verification', 'Верифікація готелю'),
    )

    STATUS_CHOICES = (
        ('pending', 'Очікує розгляду'),
        ('approved', 'Схвалено'),
        ('rejected', 'Відхилено'),
    )

    request_type = models.CharField(max_length=50, choices=REQUEST_TYPES)
    created_by = models.ForeignKey(
        settings.AUTH_USER_MODEL,
        on_delete=models.CASCADE,
        related_name='moderation_requests'
    )

    object_id = models.PositiveIntegerField()

    payload = models.JSONField(null=True, blank=True)

    status = models.CharField(max_length=20, choices=STATUS_CHOICES, default='pending')
    admin_comment = models.TextField(null=True, blank=True)

    created_at = models.DateTimeField(auto_now_add=True)
    reviewed_at = models.DateTimeField(null=True, blank=True)

```

		Парова В.В.			ІПЗ.КР.Б-121-26-ІЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		93

```

def __str__(self):
    return f"{self.get_request_type_display()} ({self.status})"

import re
from django import forms
from django.contrib.auth.forms import UserCreationForm, PasswordChangeForm
from .models import CustomUser, Hotel, Room, Rating

class HotelVerificationRequestForm(forms.Form):
    certificate_link = forms.URLField(
        required=True,
        label="Посилання на документ, що підтверджує право власності"
    )

class HotelStarsRequestForm(forms.Form):
    stars = forms.IntegerField(
        min_value=1,
        max_value=5,
        label="Бажана кількість зірок"
    )
    justification_link = forms.URLField(
        required=True,
        label="Посилання на документ, що підтверджує зірковість"
    )

class CustomUserCreationForm(UserCreationForm):
    ROLE_CHOICES = [
        ('client', 'Шукач житла'),
        ('owner', 'Власник житла'),
    ]

    role = forms.ChoiceField(
        choices=ROLE_CHOICES,
        widget=forms.RadioSelect(attrs={'class': 'btn-check-group'}),
        label="Виберіть роль"
    )

class Meta(UserCreationForm.Meta):
    model = CustomUser

```

		Парова В.В.			ІПЗ.КР.Б-121-26-ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		94

```
fields = ('email', 'first_name', 'last_name', 'phone_number', 'role')
```

```
def __init__(self, *args, **kwargs):
```

```
    super().__init__(*args, **kwargs)
```

```
    for field_name, field in self.fields.items():
```

```
        if field.widget.__class__.__name__ != "RadioSelect":
```

```
            field.widget.attrs['class'] = 'form-control'
```

```
def clean_phone_number(self):
```

```
    phone_number = self.cleaned_data.get('phone_number')
```

```
    if not re.match(r'^\+380\d{9}$', phone_number):
```

```
        raise forms.ValidationError("Неправильний формат номера. Введіть у форматі +380xxxxxxx.")
```

```
    return phone_number
```

```
class UserProfileEditForm(forms.ModelForm):
```

```
    class Meta:
```

```
        model = CustomUser
```

```
        fields = ('first_name', 'last_name', 'phone_number', 'avatar')
```

```
        widgets = {
```

```
            'avatar': forms.ClearableFileInput(attrs={'class': 'form-control'}),
```

```
        }
```

```
def __init__(self, *args, **kwargs):
```

```
    super().__init__(*args, **kwargs)
```

```
    for field_name, field in self.fields.items():
```

```
        if field_name != 'avatar':
```

```
            field.widget.attrs['class'] = 'form-control'
```

```
class OwnerProfileForm(forms.ModelForm):
```

```
    class Meta:
```

```
        model = CustomUser
```

```
        fields = ["avatar", "username", "phone_number"]
```

```
class HotelForm(forms.ModelForm):
```

```
    class Meta:
```

```
        model = Hotel
```

```
        fields = [
```

```
            'name', 'description', 'main_photo', 'phone_number',
```

```
            'city', 'address', 'oblast'
```

```
        ]
```

		Парова В.В.			ІПЗ.КР.Б-121-26-ІПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		95

```

widgets = {
    'main_photo': forms.ClearableFileInput(attrs={'class': 'form-control'}),
    'city': forms.Select(attrs={'class': 'form-control', 'id': 'id_city'}),
    'oblast': forms.Select(attrs={'class': 'form-control', 'id': 'id_oblast'}),
}

def __init__(self, *args, **kwargs):
    super().__init__(*args, **kwargs)
    for name, field in self.fields.items():
        if name not in ['main_photo', 'city', 'oblast']:
            field.widget.attrs['class'] = 'form-control'

class HotelCreateForm(forms.ModelForm):
    class Meta:
        model = Hotel
        fields = [
            'name', 'description', 'main_photo', 'phone_number',
            'city', 'address', 'oblast'
        ]
        widgets = {
            'name': forms.TextInput(attrs={'class': 'form-control'}),
            'description': forms.Textarea(attrs={'class': 'form-control'}),
            'main_photo': forms.ClearableFileInput(attrs={'class': 'form-control'}),
            'phone_number': forms.TextInput(attrs={'class': 'form-control'}),
            'city': forms.Select(attrs={'class': 'form-control', 'id': 'id_city'}),
            'address': forms.TextInput(attrs={'class': 'form-control'}),
            'oblast': forms.Select(attrs={'class': 'form-control', 'id': 'id_oblast'}),
        }

class RoomForm(forms.ModelForm):
    class Meta:
        model = Room
        fields = [
            'name',
            'price_per_night',
            'description',
            'main_photo',
            'number_of_rooms',
            'square_meters',
            'grade',

```

```

]
widgets = {
    'name': forms.TextInput(attrs={'class': 'custom-input'}),
    'price_per_night': forms.NumberInput(attrs={'class': 'custom-input'}),
    'description': forms.Textarea(attrs={'class': 'custom-textarea'}),
    'main_photo': forms.ClearableFileInput(attrs={'class': 'custom-file-input'}),
    'number_of_rooms': forms.NumberInput(attrs={'class': 'custom-input'}),
    'square_meters': forms.NumberInput(attrs={'class': 'custom-input'}),
    'grade': forms.Select(attrs={'class': 'custom-input'}),
}

```

```

class RatingForm(forms.ModelForm):
    class Meta:
        model = Rating
        fields = ['score', 'comment']
        widgets = {
            'score': forms.Select(
                choices=[(' ', ' '), (1, '1'), (2, '2'), (3, '3'), (4, '4'), (5, '5')],
                attrs={'class': 'form-select star-rating'}
            ),
            'comment': forms.Textarea(attrs={'class': 'form-control', 'rows': 4}),
        }

```

```

class UserPasswordChangeForm(PasswordChangeForm):
    old_password = forms.CharField(
        label="Старий пароль",
        widget=forms.PasswordInput(attrs={'class': 'form-control'}),
    )
    new_password1 = forms.CharField(
        label="Новий пароль",
        widget=forms.PasswordInput(attrs={'class': 'form-control'}),
    )
    new_password2 = forms.CharField(
        label="Підтвердження нового пароля",
        widget=forms.PasswordInput(attrs={'class': 'form-control'}),
    )

```

```

def cancellation_warning(request):
    if request.user.is_authenticated and hasattr(request.user, "late_cancellations"):

```

		Парова В.В.			ІПЗ.КР.Б-121-26-ІЗ	Арк.
		Єфремов Ю.М.				97
Змн.	Арк.	№ докум.	Підпис	Дата		

```

late = request.user.late_cancellations
remaining = max(0, 3 - late)
return {
    "late_cancellations": late,
    "show_cancellations_warning": late != 0,
    "remaining_cancellations": remaining,
    "user_is_blocked": not getattr(request.user, "is_active", False),
}
return {}

```

```

from django.contrib.auth.backends import ModelBackend
from django.contrib.auth import get_user_model

```

```

UserModel = get_user_model()

```

```

class EmailBackend(ModelBackend):

```

```

    """

```

```

    Кастомний бекенд для автентифікації користувачів по email.

```

```

    """

```

```

def authenticate(self, request, username=None, password=None, **kwargs):

```

```

    try:

```

```

        user = UserModel.objects.get(email__iexact=username)

```

```

    except UserModel.DoesNotExist:

```

```

        return None

```

```

    if user.check_password(password) and self.user_can_authenticate(user):

```

```

        return user

```

```

    return None

```

```

def get_user(self, user_id):

```

```

    try:

```

```

        return UserModel.objects.get(pk=user_id)

```

```

    except UserModel.DoesNotExist:

```

```

        return None

```

```

from django.apps import AppConfig

```

```

class BookingConfig(AppConfig):

```

```

    default_auto_field = 'django.db.models.BigAutoField'

```

		Парова В.В.			ІПЗ.КР.Б-121-26-ІЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		98

```
name = 'booking'
```

```
from django.contrib import admin
```

```
from .models import CustomUser, Hotel, Room, Booking, Rating, ModerationRequest
```

```
admin.site.register(CustomUser)
```

```
admin.site.register(Hotel)
```

```
admin.site.register(Room)
```

```
admin.site.register(Booking)
```

```
admin.site.register(Rating)
```

```
admin.site.register(ModerationRequest)
```

```
from apscheduler.schedulers.background import BackgroundScheduler
```

```
from django.utils import timezone
```

```
from booking.services.email_service import EmailService
```

```
from booking.models import Booking
```

```
from datetime import timedelta
```

```
def send_checkin_reminders():
```

```
    tomorrow = timezone.localtime() + timedelta(days=1)
```

```
    bookings = Booking.objects.filter(
```

```
        check_in_date=tomorrow,
```

```
        status="confirmed"
```

```
    ).select_related("user", "room")
```

```
    for b in bookings:
```

```
        EmailService().send_email(
```

```
            to=b.user.email,
```

```
            subject="Нагадування про заїзд",
```

```
            html_body=f"""
```

```
                <h3>Нагадування про заїзд</h3>
```

```
                <p>Ви заїжджаєте завтра у житло: <b>{b.room.name}</b></p>
```

```
                <p>Дата заїзду: {b.check_in_date}</p>
```

```
                <p>Гарної подорожі!</p>
```

```
            """
```

```
        )
```

```
    print(f"Sent {bookings.count()} check-in reminders for {tomorrow}")
```

		Парова В.В.			ІПЗ.КР.Б-121-26-ІЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		99

```

def update_bookings_status():
    today = timezone.localdate()
    ALLOWED_STATUSES = ['confirmed']

    updated = (
        Booking.objects
        .filter(
            check_out_date__lte=today,
            status__in=ALLOWED_STATUSES,
        )
        .update(status='completed')
    )

    print(f"☑ Updated {updated} bookings to 'completed' ({today})")

def start_scheduler():
    scheduler = BackgroundScheduler(timezone=timezone.get_current_timezone())
    scheduler.add_job(update_bookings_status, 'cron', hour=0, minute=0)
    scheduler.add_job(send_checkin_reminders, 'cron', hour=0, minute=5)

    update_bookings_status()
    send_checkin_reminders()

    scheduler.start()

```

		Парова В.В.			ІПЗ.КР.Б-121-26-ІЗ	Арк.
		Єфремов Ю.М.				100
Змн.	Арк.	№ докум.	Підпис	Дата		