

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:

**«Система управління бронюванням готелів та
житла для подорожей Україною»**

Виконав студент групи ІПЗ-20-3
КОЧУРА Віталій Вікторович

Керівник роботи:
ЧИЖМОТРЯ Олексій Володимирович

Рецензент:
БАКАЛЮК Тетяна Анатоліївна

Житомир – 2024

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

«26» лютого 2024 р.

ЗАВДАННЯ
на кваліфікаційну роботу

Здобувач вищої освіти: **КОЧУРА Віталій Вікторович**

Керівник роботи: **ЧИЖМОТРЕ Олексій Володимирович**

Тема роботи: **«Система управління бронюванням готелів та житла для подорожей Україною»**,

затверджена Наказом закладу вищої освіти від **«23» лютого 2024 р., №74/с**

Вихідні дані для роботи: **Об'єктом дослідження** є процес бронювання житла та готелів, який включає в себе всі етапи взаємодії користувача з системою.

Предметом дослідження є методи та засоби проектування і розробки системи управління бронюваннями.

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

Розділ	Консультант	Завдання видав	Завдання прийняв
1	Чижмотря О.В.	27.02.2024р.	27.02.2024р.
2	Чижмотря О.В.	27.02.2024р.	27.02.2024р.
3	Чижмотря О.В.	27.02.2024р.	27.02.2024р.

РЕФЕРАТ

Випускна кваліфікаційна робота складається з вебдодатка системи управління бронюванням готелів та житла для подорожей Україною та пояснювальної записки. Пояснювальна записка до випускної роботи містить 80 сторінок, 61 ілюстрацію, 21 таблицю, 2 додатки та 21 літературне джерело. Загальний обсяг роботи – 88 сторінок.

Метою роботи є проектування та створення системи управління бронюванням готелів та житла для подорожей Україною, яка дозволить користувачам швидко та зручно знаходити місця для проживання, враховуючи їх індивідуальні потреби.

У роботі було визначено основні вимоги до розроблюваного додатку, розглянуті й проаналізовані вже існуючі аналоги, обрана підходяща архітектура та засоби, розроблена база даних, спроектовано основні алгоритми роботи системи й реалізовано її. Для створення застосовувалися наступні інструменти: мова програмування PHP, її фреймворк Symfony, Api Platform, бази даних MySQL та MongoDB, Docker, вебсервер Nginx та JavaScript-бібліотека React.

КЛЮЧОВІ СЛОВА: БРОНЮВАННЯ, МІСЦЯ ДЛЯ ПРОЖИВАННЯ, ЖИТЛО, КІМНАТИ, ПОДОРОЖІ, ВЛАСНИК, КЛІЄНТ, АДМІНІСТРАТОР.

					ІПЗ.КР.Б – 121 – 24 – ПЗ			
Змн.	Арк.	№ докум.	Підпис	Дата				
Розроб.		Кочура В. В.			Система управління бронюванням готелів та житла для подорожей Україною	Літ.	Арк.	Аркуші
Керівник		Чижмотря О. В.					3	88
						Житомирська політехніка група ІПЗ-20-3		
Рецензент		Вакалюк Т. А.						
Зав. каф.		Вакалюк Т. А.						

ABSTRACT

The final qualification work consists of a web application of a hotel and accommodation booking management system for traveling in Ukraine and an explanatory note. The explanatory note to the graduation work contains 80 pages, 61 illustrations, 21 tables, 2 appendices, and 21 references. The total volume of the work is 88 pages.

The aim of the work is to design and create a hotel and accommodation booking management system for traveling in Ukraine, which will allow users to quickly and conveniently find places to stay, taking into account their individual needs.

In this work, we defined the main requirements for the application under development, reviewed and analyzed existing analogues, selected the appropriate architecture and tools, developed a database, designed the main algorithms of the system, and implemented it. The following tools were used for the development: PHP programming language, its Symfony framework, Api Platform, MySQL and MongoDB databases, Docker, Nginx web server, and the React JavaScript library.

KEYWORDS: BOOKING, PLACES TO STAY, ACCOMMODATION, ROOMS, TRAVEL, OWNER, CLIENT, ADMINISTRATOR.

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижевотря О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		4

ЗМІСТ

РЕФЕРАТ	3
ABSTRACT	4
ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ	7
ВСТУП.....	8
РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ РЕАЛІЗАЦІЇ СИСТЕМИ УПРАВЛІННЯ БРОНЮВАННЯМ ГОТЕЛІВ ТА ЖИТЛА ДЛЯ ПОДОРОЖЕЙ УКРАЇНОЮ ..	10
1.1. Опис проекту та постановка задачі.....	10
1.2. Аналіз аналогів програмного продукту.....	12
1.3. Вибір архітектури веб-орієнтованої системи бронювання	16
1.4. Обґрунтування вибору засобів реалізації та вимоги до апаратного забезпечення	22
Висновки до першого розділу	24
РОЗДІЛ 2. ПРОЕКТУВАННЯ СИСТЕМИ УПРАВЛІННЯ БРОНЮВАННЯМ ГОТЕЛІВ ТА ЖИТЛА ДЛЯ ПОДОРОЖЕЙ УКРАЇНОЮ.....	25
2.1. Визначення варіантів використання та об'єктно-орієнтованої структури системи.....	25
2.2. Розробка бази даних системи	30
2.3. Проектування та реалізація алгоритмів роботи системи.....	40
2.4. Реалізація системи управління бронюванням готелів та житла для подорожей Україною	44
Висновки до другого розділу.....	47
РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З СИСТЕМОЮ УПРАВЛІННЯ БРОНЮВАННЯМ ГОТЕЛІВ ТА ЖИТЛА ДЛЯ ПОДОРОЖЕЙ УКРАЇНОЮ	48

3.1. Порядок встановлення та налаштування параметрів системи.....	48
3.2. Структура інтерфейсу системи управління бронюванням готелів та житла для подорожей Україною	49
3.3. Тестування роботи програмного комплексу.....	73
Висновки до третього розділу	76
ВИСНОВКИ.....	77
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	79
ДОДАТКИ.....	82

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмотря О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		6

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

API (або Application Programming Interface) – це набір інструкцій, які дозволяють різним програмам спілкуватись між собою та обмінюватись даними.

БД – база даних.

СУБД – система управління базами.

HTTP – це протокол, в якому описано правила передачі даних. Він допомагає браузеру завантажувати вебсторінки, а серверу отримати інформацію, яку користувач ввів на сайті.

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмотря О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		7

ВСТУП

Актуальність теми. Сьогоднішнє життя неможливо уявити без подорожей у яких запорукою комфортного відпочинку або ділової поїздки є попереднє бронювання житла. На жаль, сучасні реалії нашої країни тимчасово змістили подорожі на другий план. Але кожен заслуговує на відпочинок, має право тимчасово змінити місце проживання та відволіктись від повсякденних турбот. В той же час, багато людей подорожують не тільки задля того, щоб відпочити. Це можуть бути різні причини: робота, навчання, ділова зустріч і т.д. Постає питання – як забронювати житло?

Вирішенням такої проблеми є система, що дозволить здійснювати бронювання житла, яке відповідатиме запитам клієнта, а також розміщувати власникам свої місця для проживання. Вона дозволить користувачам швидко знаходити потрібне житло, порівнювати ціни, переглядати відгуки, полегшуючи і пришвидшуючи пошук.

Така платформа не лише економить час, але й надає можливість вибору з широкого спектра варіантів, що дозволяє знайти житло, яке максимально відповідає вимогам користувача.

Метою кваліфікаційної роботи є проектування та розробка системи управління бронюванням готелів та житла для подорожей Україною, яка надасть можливість легко й швидко знаходити підходящі варіанти для проживання відповідно до індивідуальних потреб користувачів.

Щоб досягти поставленої мети, необхідно виконати такі **завдання**:

1. Аналіз вимог та порівняння існуючих аналогів.
2. Вибір архітектури системи та технологій для реалізації.
3. Проектування структури баз даних й основних алгоритмів роботи.
4. Створення та впровадження програмного забезпечення.
5. Перевірка роботи кінцевого продукту, усунення виявлених недоліків та оптимізація.

Об'єктом дослідження є процес бронювання житла та готелів, який включає в себе всі етапи взаємодії користувача з системою.

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижомотря О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		8

Предметом дослідження є методи та засоби проектування і розробки системи управління бронюваннями.

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмотря О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		9

РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ РЕАЛІЗАЦІЇ СИСТЕМИ УПРАВЛІННЯ БРОНЮВАННЯМ ГОТЕЛІВ ТА ЖИТЛА ДЛЯ ПОДОРОЖЕЙ УКРАЇНОЮ

1.1. Опис проекту та постановка задачі

У сучасному світі люди подорожують щодня, маючи при цьому різні цілі: починаючи з відпочинку, закінчуючи діловими поїздками. Тому даний проект є доволі актуальним, корисним та важливим у повсякденному житті. Мета його створення – це розробка вебплатформи, що дозволить швидко та легко знайти потрібне місце проживання, враховуючи побажання та вимоги до нього, й врешті-решт зробити бронювання.

Система буде включати різні ролі, кожна з яких матиме свій функціонал і братиме участь в різних процесах. У межах цієї системи клієнти зможуть проводити будь-які дії пов'язані з житлом (перегляд, бронювання, залишення відгуку і т.д.), власники зможуть розміщувати нові місця для проживання, додавати до них кімнати й стежити за бронюваннями.

Загалом, кінцевий продукт повинен спростити весь процес пошуку й бронювання місць проживання та надати необхідні інструменти для ведення бізнесу власникам таких місць, що дозволить покращити обслуговування та оптимізувати управління.

Під час проектування та розробки слід виконати ряд задач.

1) Аналіз вимог та функціональних можливостей системи:

- аналіз потреб та очікувань різних типів користувачів;
- визначення основних функціональних та нефункціональних вимог до системи;
- аналіз вже існуючих продуктів, їх порівняння та розгляд особливостей.

2) Вибір засобів та технологій для розробки:

- вибір мов програмування, фреймворків та баз даних, які використовуватимуться для створення додатку;

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмоторя О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		10

- аналіз переваг і недоліків різних технологічних рішень та обрання найбільш підходящих;
- вибір сторонніх сервісів, платформ або ж API, що інтегруватимуться в систему.

3) Розробка архітектури системи:

- визначення загальної структури системи, яка містить клієнтську і серверну частини, та взаємодії її компонентів;
- розробка допоміжних діаграм (діаграма варіантів використання, діаграма компонентів та інші).

4) Реалізація користувацького інтерфейсу:

- розробка привабливого, зручного та інтуїтивно-зрозумілого інтерфейсу для користувачів системи;
- забезпечення адаптивності задля можливості використовувати продукт на будь-яких пристроях.

5) Реалізація функціональних можливостей:

- розробка реєстрації та логіну, включаючи систему ролей;
- реалізація основних можливостей, що надаватиме система: перегляд місць проживання, як усіх, так і за типами, їх пошук, додавання, створення, редагування, видалення, здійснення бронювань та інші можливості для різних типів користувачів;
- реалізація функціоналу для фільтрації та сортування даних за різними критеріями.

6) Забезпечення безпеки та конфіденційності:

- захист даних від несанкціонованого доступу.

7) Інтеграція сторонніх сервісів:

- підключення систем авторизації, для забезпечення швидкого та безпечного входу користувачів;
- використання геолокаційних сервісів;
- використання платіжних систем;
- засоби для відправки та отримання електронних листів.

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмоторя О. В.				11
Змн.	Арк.	№ докум.	Підпис	Дата		

8) Тестування та оптимізація:

- проведення тестування всіх можливостей системи на різних етапах розробки;
- виявлення недоліків та їх покращення й оптимізація продукту загалом.

По завершенню розробки, на виході буде отримано досконалу та продуману систему, яка надасть можливість зручно, швидко й безперешкодно бронювати місця проживання. Окрім того, забезпечить функціоналом та інструментами, котрі допоможуть власникам покращити процес обслуговування та збільшити власний прибуток.

1.2. Аналіз аналогів програмного продукту

Оскільки тематика розробки є доволі поширеною, існує достатня кількість аналогів на ринку. Для того, щоб сформувати остаточне бачення, варто проаналізувати схожі продукти, визначивши їх переваги та недоліки. Порівняємо декілька з них.

1) Hotels24.ua

Це безкоштовний сервіс бронювання житла в Україні. Візуально сайт виглядає просто та застаріло, проте на ньому розміщена досить велика кількість готелів різних міст й він є популярним (рис. 1.1).

До плюсів можна віднести: широкий вибір готелів; зручний пошук, присутні фільтрація та сортування; можливість переглянути, де знаходиться житло на карті; є система відгуків.

Серед недоліків виявлено наступне: відсутність мати свій акаунт в системі (тільки власникам готелів), що обмежує користувача в можливостях; суб'єктивно дизайн та оформлення – не найкращі, виглядають застаріло та непривабливо. Відсутня можливість входу через Google.

<https://hotels24.ua/>

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмоторя О. В.				12
Змн.	Арк.	№ докум.	Підпис	Дата		

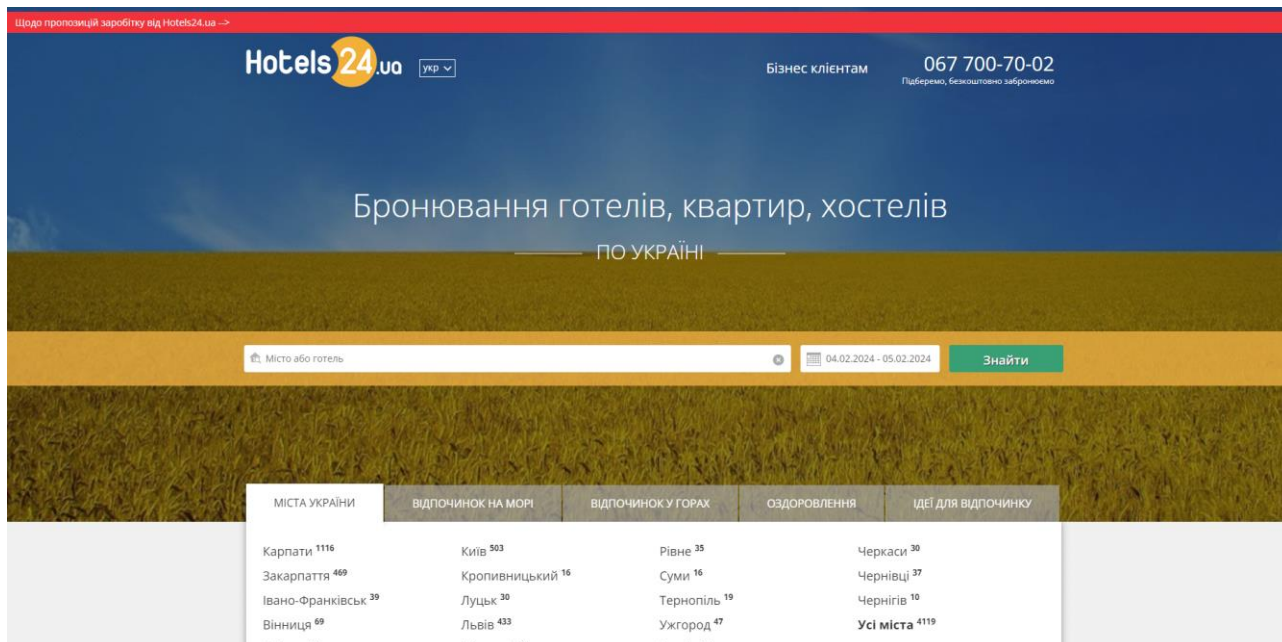


Рисунок 1.1 – Інтерфейс Hotels24.ua

2) GoHotels

Це також сервіс безкоштовного онлайн-бронювання місць проживання в Україні. Вебсайт має також дещо застарілий дизайн, проте більш інтерактивний та функціональний (рис. 1.2).

Основними перевагами було виділено: наявність особистого кабінету (порівняно з попереднім аналогом); сторінка анонсів та подій, де розміщуються цікаві статті про місця нашої країни; інтерактивна карта.

До мінусів можна віднести те, що було зазначено вище – суб'єктивно, дизайн застарілий. Також вагомим недоліком є відсутність адаптивного дизайну. Відсутня можливість входу через Google.

<https://gohotels.com.ua/>

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижемотря О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		13



Рисунок 1.2 – Інтерфейс GoHotels

3) HOTELS-OF-UKRAINE

Це вебсайт з вибору житла, котрий, порівняно з попередніми аналогами, має приємний дизайн, а головне адаптивний. Має надзвичайно велику кількість місць, що можна забронювати. В деталях готелів також використаний цікавий сервіс, що дозволяє переглянути визначні місця та ресторани поруч на інтерактивній карті (рис. 1.3).

Серед недоліків можна виділити відсутність авторизації на сайті, відповідно і можливості співпраці теж (створення облікового запису власника місця проживання).

<https://www.hotels-of-ukraine.com/>

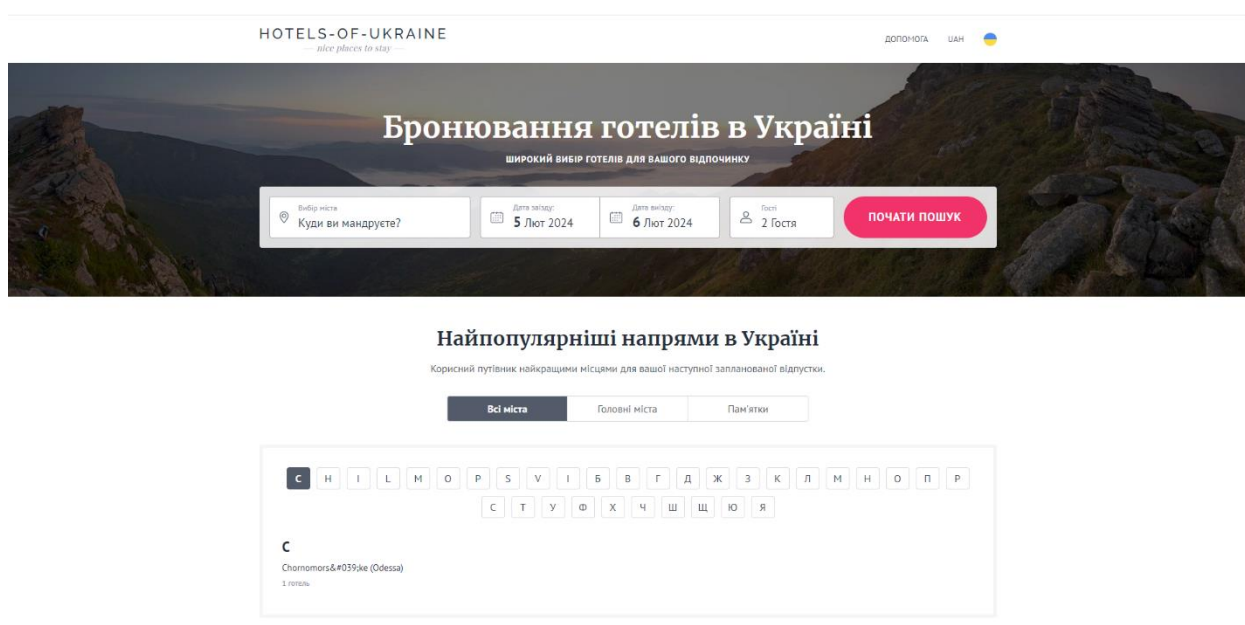


Рисунок 1.3 – Інтерфейс HOTELS-OF-UKRAINE

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмоторя О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		14

Після проведення аналізу існуючих аналогів, було виділено основні критерії та створено зведену таблицю їх порівняння (табл. 1.1).

Таблиця 1.1

Порівняльна таблиця аналогів

Найменування	Hotels24.ua	GoHotels	HOTELS-OF-UKRAINE
Привабливий та зручний інтерфейс (суб'єктивно)	-	+-	+
Пошук, фільтрація, сортування	+	+	+
Галерея зображень	+	+	+
Система відгуків	+	+	+
Карта з місцезнаходженням житла	+	+	+
Особистий кабінет (обліковий запис)	-	+	-
Можливість співпраці з власниками місць проживання	+	+	-
Можливість входу через Google	-	-	-
Адаптивний дизайн	+-	-	+

Кожен з обраних ресурсів має свої переваги та недоліки, які обов'язково будуть враховані при розробці власного додатка. Деякі з ідей будуть використані або вдосконалені в нашій системі.

Ознайомившись з аналогами було остаточно сформовано бачення розробки та підхід до неї, врахувавши вимоги клієнта. Також за допомогою аналізу визначено ключові аспекти, завдяки яким в майбутньому наш продукт зможе конкурувати на ринку.

1.3. Вибір архітектури веб-орієнтованої системи бронювання

Надважливим кроком при проектуванні будь-якого програмного продукту є вибір відповідної архітектури, що визначить продуктивність, масштабованість та зручність під час розробки. Від її якості залежить, наскільки просто проходитимуть зміни в системі, доповнення та підтримка.

Під час вибору архітектури, слід враховувати певні фактори.

Насамперед варто визначити чіткі вимоги до функціональності системи, проаналізувати, які дії користувачі будуть здійснювати та які типи даних будуть оброблятися й зберігатися системою при цьому.

Після цього можливим буде вибудовування логіки взаємодії між різними компонентами проекту. Компонент є архітектурною одиницею, що відповідає за функціональність програмного забезпечення та/або зберігання даних. Він забезпечує доступ до своїх сервісів через зовнішній інтерфейс та має чітко визначену залежність від контексту в якому його застосовують.

Задля забезпечення швидкодії та ефективності програмного забезпечення, варто враховувати можливу кількість користувачів та обсяг даних, котрий буде оброблятися. При цьому, обов'язково потрібно враховувати, що система потенційно може масштабуватися в майбутньому.

Також важливим пунктом є забезпечення надійного захисту даних, використовуючи продумані механізми аутентифікації та авторизації, шифрування даних, захист від несанкціонованого доступу та інші заходи безпеки, що гарантуватимуть конфіденційність та цілісність інформації.

Серед найпопулярніших архітектурних рішень розглянемо декілька:

- **мікросервісна архітектура** – це підхід до розробки програмного забезпечення, при якому додаток розбивається на невеликі та незалежні сервіси. Кожен такий сервіс відповідає за конкретну бізнес-функцію та може бути розроблений і впроваджений незалежно від інших [30].

Основними перевагами такого архітектурного рішення є:

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижомотря О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		16

- модульність, котра полегшує зміну та масштабування різних модулів системи;
- безвідмовність забезпечується тим, що у випадку відмови одного сервісу, інші продовжують працювати. Тобто неполадки у мікросервісі не повинні зламати весь додаток;
- кожен сервіс може бути масштабований окремо;
- простота виявлення та виправлення помилок у невеликих сервісах.

Серед недоліків, можна виділити наступне:

- розробка розподілених систем може бути надто складною;
 - мікросервіси, будучи автономними, при взаємодії через мережу, можуть призводити до додаткових затримок та зниження продуктивності системи, що варто враховувати при виборі архітектури;
 - захист даних у такій архітектурі набагато складніший, порівняно з іншими.
- **монолітна архітектура** – це традиційний підхід до розробки програмного забезпечення, де весь додаток розробляється як єдина, самостійна система. У монолітній архітектурі кожен компонент та пов'язані з ним компоненти є необхідними для виконання або компіляції коду та для правильної роботи програмного забезпечення [30].

Серед переваг такого підходу варто виділити:

- простота розробки, розгортання та тестування, адже все знаходиться в одному місці;
- процес відлагодження є також простим. Легко відстежити потік запиту та знайти проблему;
- дані зазвичай централізовані в одній або декількох доступних базах даних;

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмотря О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		17

- при такому підході менше наскрізної функціональності (cross-cutting concerns). Оскільки всі відповідні дані знаходяться в одному місці, легше відбувається логування, кешування та моніторинг продуктивності.

Також існує декілька недоліків:

- розростання функціоналу проекту може призвести до проблем з масштабуванням;
 - всі компоненти при такому підході сильно взаємозалежні, тому щоразу ризиковано впроваджувати нові функції або змінювати систему;
 - оновлення застарілих (legacy) технологій може стати справжнім викликом, оскільки міграція усього додатку на нову версію може бути складною, або навіть неможливою;
 - з часом код ставатиме все складнішим для розуміння.
- **безсерверна архітектура** (serverless) – це спосіб розробки та запуску додатків, який мінімізує потребу в управлінні ресурсами. Такий підхід надає можливість виконавцям запускати додатки без управління фізичними серверами. Це дозволяє хмарним провайдерам виконувати код, розподіляючи ресурси та масштабуючи інфраструктуру в залежності від потреб, які виникають [24].

Перевагами такого підходу є:

- завдяки такій архітектурі, розробники сфокусовані на впровадженні бізнес-логіки, що пришвидшує процес створення програмного продукту;
- автоматичне масштабування ресурсів в залежності від навантаження та потреб;
- оплата буде лише за фактичне використання, при цьому немає необхідності вкладати кошти в придбання та обслуговування серверного обладнання;

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмотря О. В.				18
Змн.	Арк.	№ докум.	Підпис	Дата		

- завдяки автоматизації, розгортання систем, при такому підході є доволі швидким.

До недоліків варто віднести наступне:

- відсутність контролю та прямого доступу до серверів і їх налаштування;
- процес відлагодження може бути складнішим порівняно з іншими підходами;
- постійна залежність від постачальника послуг, при цьому перемкнутися на іншого буде доволі проблематично;
- питання безпеки є дуже важливим при такому підході, його потрібно ретельно дослідити та знайти необхідні рішення.

Дослідивши різні архітектурні рішення, порівнявши їх та визначивши переваги й недоліки – найбільш підходящим виявилась саме монолітна архітектура. При цьому буде використано такі архітектурні патерни, як MVC та Client-Server.

MVC – це патерн, що дозволяє розбити додаток на логічні компоненти і ефективно керувати даними, їх виведенням та логікою системи, при цьому полегшуючи процес розробки, тестування та підтримки (рис. 1.4) [25].

Модель відповідає за обробку даних і бізнес-логіку. Це блок, який отримує дані від Контролера, потім на основі цих даних робить необхідні перетворення, або вибирає дані з БД, або змінює їх, а потім передає результат своєї роботи тому ж Контролеру [28].

Представлення відповідає за відображення даних користувачу. Демонструє інформацію з Моделі в зручному вигляді для користувача. Задача Представлення – зчитати дії або інформацію, введену користувачем, і передати її в конкретний метод або функцію Контролера.

Контролер, в свою чергу, відповідає за отримання даних від користувача, їх обробку, нормалізацію, а також валідацію і передачу цих даних у потрібну Модель. Окрім цього, він приймає дані від Моделі, потім обирає потрібне

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижомотря О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		19

Представлення, передаючи в нього ці дані. Коротко, можна сказати, що він слугує посередником між Моделлю та Представленням.

На сьогоднішній день, патерн MVC використовується для багатьох сучасних вебдодатків, оскільки він дозволяє застосунку бути масштабованим, підтримуваним і легко розширюваним.

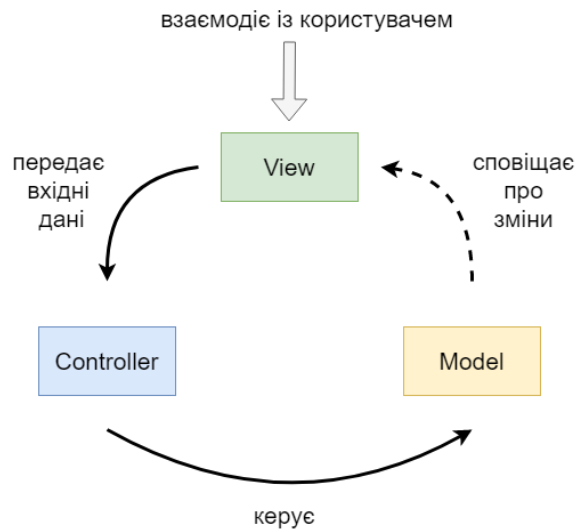


Рисунок 1.4 – Взаємодія між компонентами патерну MVC

Говорячи про клієнт-серверну архітектуру (Client-Server), хочеться зазначити, що майже всі вебсайти та інтернет-сервіси побудовані на ній. Як видно із назви, існує клієнтський процес, що вимагає певних ресурсів, а також серверний процес, що ці ресурси надає (рис. 1.5) [26], [27].

Така архітектура складається з двох компонентів:

1. Клієнт – це додаток або пристрій, який звертається до сервера, відправляючи запити, для отримання даних або виконання певних операцій.
2. Сервер, котрий приймає та обробляє запити від клієнта, після чого відправляє результат. Він забезпечує доступ до бази даних, файлової системи й алгоритмів обробки даних.

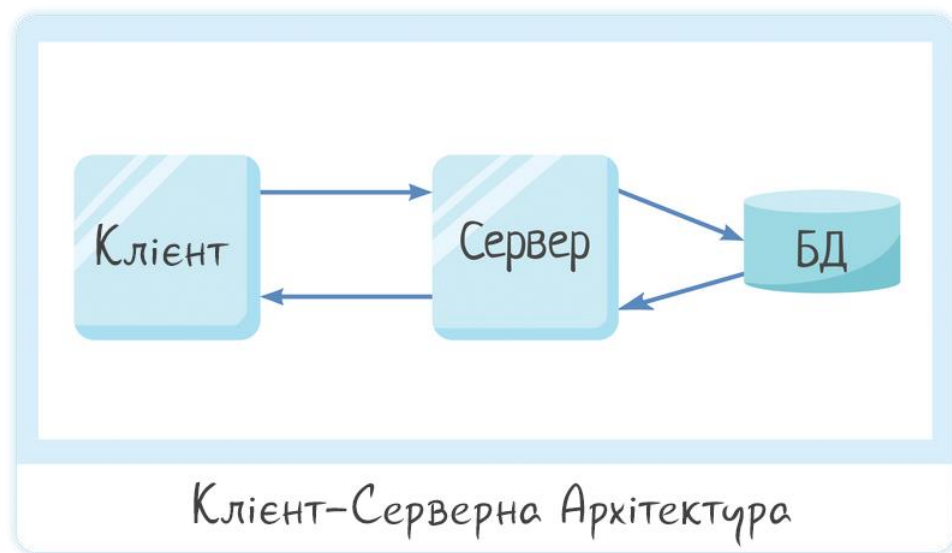


Рисунок 1.5 – Клієнт-серверна архітектура

Зв'язок між сервером та клієнтом здійснюватиметься через HTTP-запити, з використанням RESTful API [14].

RESTful API – це архітектура інтерфейсу прикладних програм (API), що використовує HTTP-запити для доступу до ресурсів та їхнього використання. Такими запитами є GET, PUT, POST і DELETE. Вони дають змогу, відповідно, читати, змінювати, створювати й видаляти ресурси. При роботі з RESTful API потрібно дотримуватись наступних принципів:

- кожен ресурс повинен мати свій унікальний URL, за допомогою якого його можна ідентифікувати;
- клієнтська частина відокремлена від серверної, їх можна розвивати незалежно одну від одної, що якраз і реалізує клієнт-серверна архітектура. При цьому клієнт та сервер не повинні зберігати жодної інформації про стан один одного;
- JSON або XML – формати, які використовуються при обміні даними;
- дані у відповіді на запит мають бути позначені як кешовані або некешовані;
- усі запити повинні бути незалежними один від одного.

Повністю продумавши архітектуру майбутньої системи можна з впевненістю сказати, що вона буде досконалою, простою в плані розробки, наскільки це можливо, та масштабованою.

1.4. Обґрунтування вибору засобів реалізації та вимоги до апаратного забезпечення

Правильний вибір інструментів для реалізації додатку є одним із важливих етапів, на початку розробки, оскільки це вплине на його успішність. До того ж, від цього залежатиме ефективність роботи над проектом та його подальший розвиток.

Для розробки серверної частини було обрано широковідому мову програмування PHP. Вона досить добре себе зарекомендувала та має широку спільноту розробників, що говорить про її популярність та робить легкою у вивченні й використанні [20].

PHP має досить велику кількість фреймворків, що значно спрощують та прискорюють розробку, одним із таких є Symfony. Його модульна система компонентів є досить гнучкою, і дозволяє виконувати стандартні завдання без написання довгого коду. При цьому він забезпечує високий рівень безпеки програмного забезпечення та його стабільність. Крім того, фреймворк пропонує широкий вибір готового функціоналу, компонентів та бібліотек, що може суттєво мінімізувати витрату часу при реалізації проекту [22], [23].

Важливо зазначити, що Symfony дозволяє розробляти програмне забезпечення з використанням архітектурного патерну MVC, що був описаний у попередньому розділі.

Для оптимізації обробки запитів було обрано фреймворк API Platform, котрий забезпечує повноцінний набір інструментів для ефективної та швидкої роботи з API згідно до сучасних стандартів [15].

У проекті використовуватиметься система управління базами даних MySQL. Вона є зручною в роботі, простою в розумінні, досить популярною, тому з легкістю можна знайти потрібну документацію. MySQL відома у всьому

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмоторя О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		22

світі як найбезпечніша та найнадійніша система керування базами даних, яка використовується в різних популярних вебдодатках [18].

Окрім того, також буде задіяна база даних MongoDB. Вона є оптимальним вибором для зберігання тимчасових або ж громіздких даних. MongoDB відома своєю гнучкістю та масштабованістю, що дає можливість ефективно обробляти великі обсяги інформації і швидко здійснювати операції запису та читання. Це робить її ідеальним доповненням до основної реляційної бази даних MySQL у нашій системі [1].

Як вебсервер, було обрано Nginx. Він відзначається високою продуктивністю, можливістю ефективно обробляти велику кількість одночасних підключень та надійністю.

Для тестування електронної пошти в локальному середовищі розробки, використовуватиметься інструмент Mailhog. Він перехоплює всі вихідні листи та зберігає їх у власному інтерфейсі, що дозволяє легко переглядати та аналізувати їх без зайвих труднощів. Завдяки ньому реалізація функціоналу, пов'язаного з відправкою через електронну пошту, значно спроститься [9].

Щоб реалізувати клієнтську частину було вибрано найпопулярнішу JavaScript-бібліотеку – React. Він використовується для швидкого та ефективного створення інтерактивних користувацьких інтерфейсів і вебдодатків із застосуванням значно меншої кількості коду, ніж під час використання звичайного JavaScript. React пропонує розробникам розділити призначені для користувача інтерфейси на окремі компоненти, які можна використовувати повторно [21].

Для розгортання додатку використовуватиметься Docker, котрий дозволяє представляти компоненти системи у вигляді контейнерів, легко управляти ними та масштабувати [16]. Однією з ключових переваг є ізоляція цих компонентів та їх залежностей, що гарантує стабільну роботу системи. Його використання значно підвищить ефективність та швидкість розробки продукту [6].

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижомотря О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		23

Даний набір технологій забезпечить зручну та легку розробку й при його правильному використанні буде отримано довершену систему, котра задовольнить поставлені вимоги та завдання.

Висновки до першого розділу

Роблячи висновки з першого розділу, слід зазначити, що було визначено основні вимоги до майбутньої системи, дотримуючись яких, під час розробки, на виході буде отримано досконалий продукт, що повністю задовільнить потреби користувача.

Проаналізовано аналоги системи, а саме такі сервіси як: Hotels24.ua, GoHotels та HOTELS-OF-UKRAINE. В результаті порівняння було визначено їх функціональні особливості, сильні сторони та недоліки, які слід врахувати при розробці. Після чого сформувалось остаточне бачення кінцевого результату та обрано оптимальний підхід, враховуючи вимоги сучасного клієнта.

Порівнюючи різні архітектурні підходи, було обрано ті, що найбільш задовольняють потреби й допоможуть при розробці. Вибрані рішення дозволять ефективно побудувати довершений та стабільний додаток, з можливістю розширення у майбутньому.

Зрештою обґрунтовано вибір інструментів, що використовуватимуться при розробці. Серед них: мова програмування PHP, на якій буде написана серверна частина, фреймворк Symfony, що значно спростить це написання, фреймворк API Platform, котрий оптимізує процес обробки запитів, вебсервер Nginx та JavaScript-бібліотека React, що буде використана для клієнтської частини додатка.

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижомотря О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		24

РОЗДІЛ 2. ПРОЕКТУВАННЯ СИСТЕМИ УПРАВЛІННЯ БРОНЮВАННЯМ ГОТЕЛІВ ТА ЖИТЛА ДЛЯ ПОДОРОЖЕЙ УКРАЇНОЮ

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

Зробивши аналіз напрямків використання інформаційних технологій для реалізації нашої системи, ми сформувавши певне бачення структури та функціональності додатку.

Далі потрібно провести детальний аналіз вимог до майбутньої системи, котрий дозволить визначити основні можливості та функції, що матиме вихідний продукт. Такий аналіз буде охоплювати як функціональні вимоги, які описують конкретні операції та завдання, що повинні виконуватися системою, так і нефункціональні, котрі визначають її атрибути, такі як надійність, продуктивність, захист даних та зручність використання.

Окрім цього, буде побудована діаграма використання, котра допоможе візуалізувати основні сценарії взаємодії користувачів різних типів із системою.

Усі вимоги формуються відповідно до потреб сучасних користувачів подібних систем, що дозволить створити продукт, який максимально відповідатиме їх очікуванням.

Бізнес вимоги:

1. Основні цілі: проект створюється з наступною метою – надання можливості людям, що хочуть подорожувати нашою країною, швидко та просто знайти й забронювати місце, де вони зможу зупинитись на певний час. Також, ще однією ціллю є автоматизація операцій для власників місць проживання. Система призначена для покращення доступності та точності інформації, а також для підвищення ефективності управління бронюваннями та роботи готельного бізнесу загалом, що сприятиме задоволеності клієнтів та зростанню прибутку власників.

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижомотря О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		25

2. Представлення проекту: проект буде реалізовано у формі веб-платформи, яка надасть користувачам зручний доступ до інформації по відповідній темі.
3. Гнучкість та масштабованість: платформа повинна бути розроблена з урахуванням можливості легкого масштабування та розширення функціоналу у майбутньому, щоб відповідати зростаючим потребам бізнесу та користувачів.
4. Інтеграція з платіжними системами: для забезпечення можливості оплати бронювань, у продукті має бути інтегрований платіжний сервіс.
5. Доступність: додаток має бути доступним та зручним для користування на будь-яких пристроях.

Функціональні вимоги:

1. Авторизація та реєстрація у системі: система повинна надавати можливість користувачам авторизуватися за допомогою облікових записів, включаючи Google. При цьому можна буде обрати роль Клієнта, або ж Власника місця проживання.
2. Збереження та керування інформацією: система повинна забезпечувати збереження інформації про користувачів, готелі, кімнати, бронювання та інші важливі дані й надавати можливість керувати ними.
3. Пошук та бронювання: система має надавати можливість користувачам шукати доступне житло та готелі в різних регіонах України за різними параметрами, й обирати в них бажану кімнату за типом, датою заїзду/виїзду, кількістю гостей, тощо.
4. Інформація про готелі та житло: система має давати детальну інформацію про місця проживання, включаючи фото як самого житла, так і його кімнат, опис, мапу з його місцем розташування, важливу інформацію та ціни.
5. Оплата: система повинна забезпечувати зручний і безпечний процес оплати бронювань.

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижомотря О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		26

6. Відгуки та оцінювання: користувачі повинні мати можливість залишати відгуки та оцінювати своє проживання відповідно.

Нефункціональні вимоги:

1. Сприйняття:

- час, необхідний для навчання звичайних користувачів – 20-30 хвилин. Для досвідчених користувачів цей час може бути скорочений до 10 хвилин;
- інтерфейс має бути зрозумілим та з відповідними підказками для полегшення навігації та користування.

2. Надійність:

- доступність – час, що витрачається на обслуговування системи не повинен перевищувати 5% від загального часу роботи;
- середній час безвідмовної роботи системи повинен становити 30 робочих днів;
- максимальна норма помилок та дефектів в роботі системи – 1 помилка на десять тисяч рядків коду.

3. Продуктивність:

- система повинна підтримувати мінімум 50 одночасно працюючих користувачів, пов'язаних із загальною базою даних.

4. Можливість експлуатації:

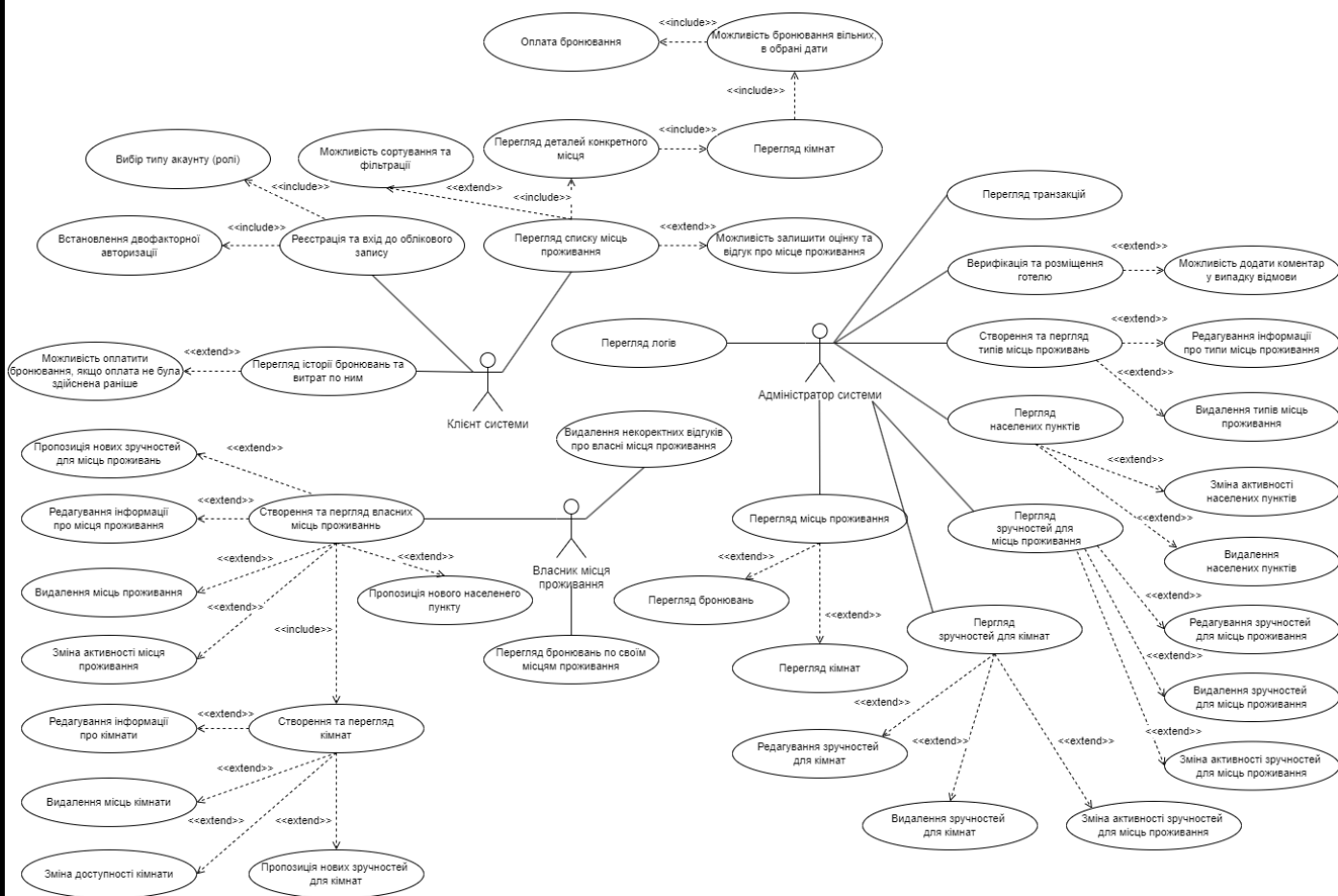
- система повинна мати можливість масштабування, іншими словами збільшення потужностей та продуктивності системи зі зростанням кількості користувачів без негативного впливу на її роботу;
- оновлення повинно проводитися у час, коли навантаження на платформу є найменшим, з тимчасовим призупиненням роботи сайту під час вивантаження.

Часові характеристики реакції системи:

- час відгуку для типових задач має не перебільшувати 5 секунд, для складних – не більше 10 секунд;

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмоторя О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		27

- Після визначення ряду вимог до системи, наступним кроком буде побудова діаграми варіантів використання (рис. 2.1). Ця діаграма допоможе візуалізувати основні сценарії взаємодії різних типів користувачів з додатком. Побудувавши її, ми матимемо чітке уявлення про те, які функції система має надати для задоволення всіх потреб [29].



Клієнт системи:

- реєстрація та вхід до облікового запису, разом із встановленням двуфакторної авторизації та вибором ролі;

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмотря О. В.				28
Змн.	Арк.	№ докум.	Підпис	Дата		

- перегляд місць проживань (застосувавши потрібні фільтри), разом з переглядом списку кімнат у ньому, з подальшим їх бронюванням в обрані дати та оплата цього бронювання;
- перегляд історії бронювань та витрат по ним, з можливістю оплатити бронювання протягом 20 хвилин, у випадку, якщо оплата не відбулась раніше.

Власник місць проживання:

- створення своїх місць проживання, разом з пропозиціями (на погодження адміністратора) нових зручностей, населених пунктів та редагування, видалення й зміна активності;
- додавання кімнат, разом з пропозиціями (на погодження адміністратора) нових зручностей, зміна доступності кімнат та редагування й видалення;
- перегляд бронювань по своїм місцям проживання;
- видалення некоректних відгуків про власні місця проживання.

Адміністратор системи:

- верифікація та розміщення готелів, з можливістю додати коментар у випадку відмови;
- перегляд усіх місць проживання, разом із переглядом їх кімнат та бронювань;
- створення та перегляд типів місць проживання, їх редагування та видалення;
- перегляд населених пунктів, з можливістю зміни їх активності, редагування та видалення;
- перегляд зручностей для місць проживання, з можливістю їх затвердження, редагування та видалення;
- перегляд зручностей для кімнат, з можливістю їх затвердження, редагування та видалення;
- перегляд логів;
- перегляд транзакцій.

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижомотря О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		29

2.2. Розробка бази даних системи

Даний етап є надважливим при розробці будь-якої системи, оскільки база даних визначає структуру та організацію інформації, яка буде зберігатися та оброблятися програмним забезпеченням. Розробка досконалої бази даних забезпечить ефективність роботи продукту та швидкий доступ до інформації, що зберігається в ній.

Правильний підхід до проектування бази даних сприятиме її гнучкості та масштабованості. Таким чином, коректна побудова дозволить без зайвих труднощів вдосконалити та доповнити функціональність додатку за потреби.

Основною базою даних в системі є MySQL. Це досить потужна та надійна СУБД, яка використовується для зберігання та обробки великих обсягів інформації. Її гнучкість та розширюваність дозволять легко адаптувати систему до нових потреб у майбутньому.

Під час проектування було враховано всі вимоги та в результаті отримано досить продуману та структуровану базу даних (рис. 2.2).

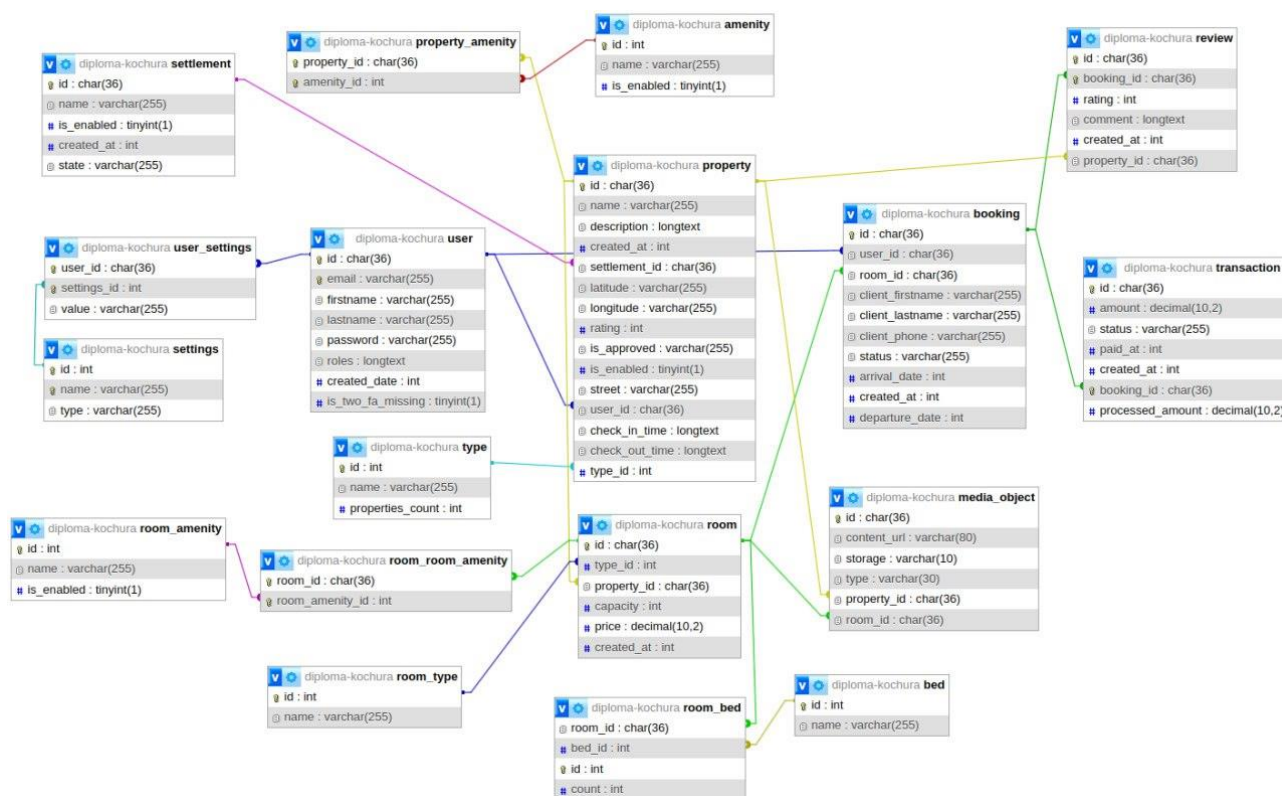


Рисунок 2.2 – Структура бази даних

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмоторя О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		30

Кожна сутність проектувалась окремо. Це гарантує належне функціонування всіх елементів системи та їх відповідність вимогам. Нижче наведено детальний опис кожної з таблиць.

Дані про користувачів системи зберігаються у таблиці «user» (табл. 2.1).

Таблиця 2.1

Структура таблиці «user»

Назва поля	Тип	PK	FK	Опис
id	char(36)	+		Ідентифікатор
email	varchar(255)			Пошта користувача
firstname	varchar(255)			Ім'я користувача
lastname	varchar(255)			Прізвище користувача
password	varchar(255)			Пароль
roles	longtext			Роль
created_date	int			Дата створення облікового запису
is_two_fa_missing	tinyint(1)			Статус встановлення двофакторної автентифікації

Для збереження типів налаштувань, наприклад таких як google_auth_token, було створено таблицю «settings» (табл. 2.2).

Таблиця 2.2

Структура таблиці «settings»

Назва поля	Тип	PK	FK	Опис
id	int	+		Ідентифікатор
name	varchar(255)			Назва
type	varchar(255)			Тип

Щоб зберігати налаштування для користувачів, було створено відповідну таблицю «user_settings» (табл. 2.3).

Таблиця 2.3

Структура таблиці «user_settings»

Назва поля	Тип	PK	FK	Опис
1	2	3	4	5
user_id	char(36)		+	Ідентифікатор користувача, зв'язок з таблицею «user»

Продовження таблиці 2.3

1	2	3	4	5
settings_id	int	+		Ідентифікатор налаштування, зв'язок з таблицею «settings»
value	varchar(255)			Значення

Після створення таблиць пов'язаних з користувачем, було спроектовано одну з найголовніших таблиць у системі – «property», що описує сутність місць для проживання, що бронюватимуть користувачі (табл. 2.4).

Таблиця 2.4

Структура таблиці «property»

Назва поля	Тип	PK	FK	Опис
id	char(36)	+		Ідентифікатор
name	varchar(255)			Назва
description	longtext			Опис
created_at	int			Дата створення
settlement_id	char(36)		+	Ідентифікатор населеного пункту, зв'язок з таблицею «settlement»
latitude	varchar(255)			Широта (використовуватиметься для відображення на мапі)
longitude	varchar(255)			Довгота (використовуватиметься для відображення на мапі)
rating	int			Рейтинг місця проживання (на клієнтській частині відображатиметься у вигляді зірочок)
is_approved	varchar(255)			Статус верифікації місця проживання
is_enabled	tinyint(1)			Статус активності місця проживання
street	varchar(255)			Вулиця
user_id	char(36)		+	Ідентифікатор користувача, зв'язок з таблицею «user»
check_in_time	longtext			Час заїзду
check_out_time	longtext			Час виїзду
type_id	int		+	Ідентифікатор типу житла, зв'язок з таблицею «type»

Для зберігання типів житла створено окрему таблицю «type» (табл. 2.5).

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмотря О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		32

Таблиця 2.5

Структура таблиці «type»

Назва поля	Тип	PK	FK	Опис
id	int	+		Ідентифікатор
name	varchar(255)			Назва
properties_count	int			Кількість місць проживання закріплених за типом

Щоб зберігати населені пункти в яких знаходяться ті чи інші місця проживання, створено таблицю «settlement» (табл. 2.6).

Таблиця 2.6

Структура таблиці «settlement»

Назва поля	Тип	PK	FK	Опис
id	char(36)	+		Ідентифікатор
name	varchar(255)			Назва
is_enabled	tinyint(1)			Статус активності населеного пункту
created_at	int			Дата створення
state	varchar(255)			Область

Для зручностей, що надає житло, було створено таблицю «amenity» (табл. 2.7).

Таблиця 2.7

Структура таблиці «amenity»

Назва поля	Тип	PK	FK	Опис
id	int	+		Ідентифікатор
name	varchar(255)			Назва
is_enabled	tinyint(1)			Статус активності зручності

Для збереження зв'язків між таблицями «property» та «amenity» було додано таблицю «property_amenity» (табл. 2.8).

Таблиця 2.8

Структура таблиці «property_amenity»

Назва поля	Тип	РК	FK	Опис
property_id	char(36)		+	Ідентифікатор місця проживання, зв'язок з таблицею «property»
amenity_id	int		+	Ідентифікатор зручності, зв'язок з таблицею «amenity»

Далі було спроектовано, також одну з найважливіших таблиць системи, що описує сутність номерів, що містяться у місцях для проживання – «room» (табл. 2.9).

Таблиця 2.9

Структура таблиці «room»

Назва поля	Тип	РК	FK	Опис
id	char(36)	+		Ідентифікатор
type_id	int		+	Ідентифікатор типу кімнати, зв'язок з таблицею «type»
property_id	char(36)		+	Ідентифікатор місця проживання, зв'язок з таблицею «property»
capacity	int			Кількість гостей, що може зупинитись в номері
price	decimal(10, 2)			Ціна за ніч
created_at	int			Дата створення

Для зручностей, що пропонують кімнати, було створено відповідну таблицю «room_amenity» (табл. 2.10).

Таблиця 2.10

Структура таблиці «room_amenity»

Назва поля	Тип	РК	FK	Опис
id	int	+		Ідентифікатор
name	varchar(255)			Назва
is_enabled	tinyint(1)			Статус активності зручності

Для зв'язку між кімнатами та їх зручностями (таблиці «room» та «room_amenity») було створено таблицю «room_room_amenity» (табл. 2.11).

Таблиця 2.11

Структура таблиці «room_room_amenity»

Назва поля	Тип	PK	FK	Опис
room_id	char(36)		+	Ідентифікатор номеру, зв'язок з таблицею «room»
room_amenity_id	int		+	Ідентифікатор зручності кімнати, зв'язок з таблицею «room_amenity»

Для зберігання типів кімнат було створено окрему таблицю «room_type». Заповнюється вона завдяки відповідній команді `AddDefaultRoomTypes`. В цій команді виконується процес вставки найпоширеніших типів кімнат в таблицю (табл. 2.12).

Таблиця 2.12

Структура таблиці «room_type»

Назва поля	Тип	PK	FK	Опис
id	int	+		Ідентифікатор
name	varchar(255)			Назва

У кожній з кімнат є певна кількість ліжок, саме для них була створена окрема таблиця «bed». Її наповнення, як і наповнення попередньої таблиці відбувається завдяки команді, з назвою `AddDefaultBeds` (табл. 2.13).

Таблиця 2.13

Структура таблиці «bed»

Назва поля	Тип	PK	FK	Опис
id	int	+		Ідентифікатор
name	varchar(255)			Назва

Так як у кімнатах може бути декілька ліжок, задля цього до структури бази даних додана таблиця «room_bed» (табл. 2.14). При її створенні недостатньо прописати doctrine-анотації `ManyToMany` [5]. Варто додавати окрему Entity `RoomBed`, для того щоб мати можливість вказати додаткове поле `count` в таблиці, що пов'язує ліжка та кімнати [10].

Таблиця 2.14

Структура таблиці «room_bed»

Назва поля	Тип	РК	FK	Опис
room_id	char(36)		+	Ідентифікатор номеру, зв'язок з таблицею «room»
bed_id	int		+	Ідентифікатор ліжка, зв'язок з таблицею «bed»
id	int	+		Ідентифікатор
count	int			Кількість ліжок в кімнаті

Кожне місце проживання, як і кожна з кімнат, матимуть свої зображення. Для прикладу, за місцем проживання також будуть закріплені зображення документів про володіння, для верифікації. Саме для цього створена таблиця «media_object» (табл. 2.15).

Таблиця 2.15

Структура таблиці «media_object»

Назва поля	Тип	РК	FK	Опис
id	char(36)	+		Ідентифікатор
content_url	varchar(80)			Назва файлу
storage	varchar(10)			Директорія, де зберігається файл
type	varchar(30)			Тип
property_id	char(36)		+	Ідентифікатор місця проживання, зв'язок з таблицею «property»
room_id	char(36)		+	Ідентифікатор номеру, зв'язок з таблицею «room»

Також для однієї з досить важливої сутності – бронювання, було створено таблицю «booking» (табл. 2.16).

Таблиця 2.16

Структура таблиці «booking»

Назва поля	Тип	РК	FK	Опис
1	2	3	4	5
id	char(36)	+		Ідентифікатор

Продовження таблиці 2.16

1	2	3	4	5
user_id	char(36)		+	Ідентифікатор користувача, зв'язок з таблицею «user»
room_id	char(36)		+	Ідентифікатор номеру, зв'язок з таблицею «room»
client_firstname	varchar(255)			Ім'я на кого оформлено бронювання
client_lastname	varchar(255)			Прізвище на кого оформлено бронювання
client_phone	varchar(255)			Телефон на кого оформлено бронювання
status	varchar(255)			Статус бронювання
arrival_date	int			Дата заїзду
departure_date	int			Дата виїзду
created_date	int			Дата створення

При створенні бронювань, також створюються транзакції, для їх зберігання створено таблицю «transaction» (табл. 2.17).

Таблиця 2.17

Структура таблиці «transaction»

Назва поля	Тип	PK	FK	Опис
id	char(36)	+		Ідентифікатор
amount	decimal(10,2)			Сума транзакції
status	varchar(255)			Статус транзакції
paid_at	int			Дата оплати
created_at	int			Дата створення
booking_id	char(36)		+	Ідентифікатор бронювання, зв'язок з таблицею «booking»
property_id	char(36)		+	Ідентифікатор місця проживання, зв'язок з таблицею «property»

Також користувач має можливість залишити відгук про житло, після свого візиту. Для цього було створено таблицю «review» (табл. 2.18).

Структура таблиці «review»

Назва поля	Тип	PK	FK	Опис
id	char(36)	+		Ідентифікатор
booking_id	char(36)		+	Ідентифікатор бронювання, зв'язок з таблицею «booking»
rating	varchar(255)			Оцінка місця проживання
comment	longtext			Коментар до відгуку
created_at	int			Дата створення
booking_id	char(36)		+	Ідентифікатор бронювання, зв'язок з таблицею «booking»

Як було зазначено на початку розділу, MySQL – це основна база даних, у системі також використовуватиметься MongoDB. Використання цієї бази дозволить нам ефективно працювати з тимчасовими даними. Окрім цього, MongoDB використовуватиметься для зберігання логів [8]. Це дозволить уникнути перевантаження основної бази даних, покращить продуктивність системи та спростить обробку великих обсягів інформації.

База даних складається з двох таблиць, нижче буде описано кожну з них (рис. 2.3).



Рисунок 2.3 – Таблиці, що знаходяться в MongoDB

При реєстрації, до моменту коли користувач введе код, що прийшов йому на пошту, буде створюватись його тимчасова сутність та записуватиметься в MongoDB. Це потрібно для роботи с кодом, котрий слугуватиме для підтвердження створення облікового запису, він приходитиме користувачам на пошту. Завдяки запису в MongoDB ми зможемо з легкістю перевіряти

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмоторя О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		38

правильність коду, перестворювати його для користувача, або коли він покине форму, на цьому етапі, то зможе ввести ті ж самі дані про себе та той самий код, або для нього створиться новий. Для цього була створена таблиця «registrationdata» з приставкою у вигляді назви проекту «diploma-kochura_» (табл. 2.19).

Структура цієї таблиці наступна:

Таблиця 2.19

Структура таблиці «diploma-kochura_registrationdata»

Назва поля	Тип	Опис
_id	ObjectId	Ідентифікатор
id	string	Ідентифікатор тимчасової сутності користувача
email	string	Пошта
password	string	Пароль
firstname	string	Ім'я
lastname	string	Прізвище
code	string	Код, що має бути введений при реєстрації
datetime	int	Дата та час створення коду
createdAt	int	Дата створення
attempts	int	Кількість спроб на введення коду

Була створена таблиця «AppLogs» для зберігання логів додатку, котрі потрібні для відстеження подій та дій користувачів або системи (табл. 2.20).

Таблиця 2.20

Структура таблиці «AppLogs»

Назва поля	Тип	Опис
_id	ObjectId	Ідентифікатор
message	string	Повідомлення
level	string	Рівень важливості
level_name	string	Назва рівню важливості
channel	string	Канал (джерело події)
datetime	date_immutable	Дата створення
context	array	Контекст (додаткова інформація для логу), у нашому випадку туди записуватиметься пошта користувача та роль

Загалом було побудовано дві бази даних, використання яких дозволить ефективно управляти даними в системі та легко масштабуватись у майбутньому. Цей етап є одним із ключових при розробці будь-якої системи, оскільки продумана структура впливатиме на подальшу роботу всього додатку.

2.3. Проектування та реалізація алгоритмів роботи системи

Після формування структури бази даних, було розпочато процес проектування алгоритмів роботи системи, а в подальшому їх реалізацію. Цей етап є також важливим, оскільки правильно спроектовані алгоритми, дозволять побудувати коректну та правильну логіку роботи додатку загалом.

Для цього використовувались діаграми активності та послідовності. Вони слугують для візуалізації взаємодії між об'єктами та послідовності дій, що виконуються в системі [29].

На першій діаграмі активності зображено процес реєстрації користувача у системі (рис. 2.4).

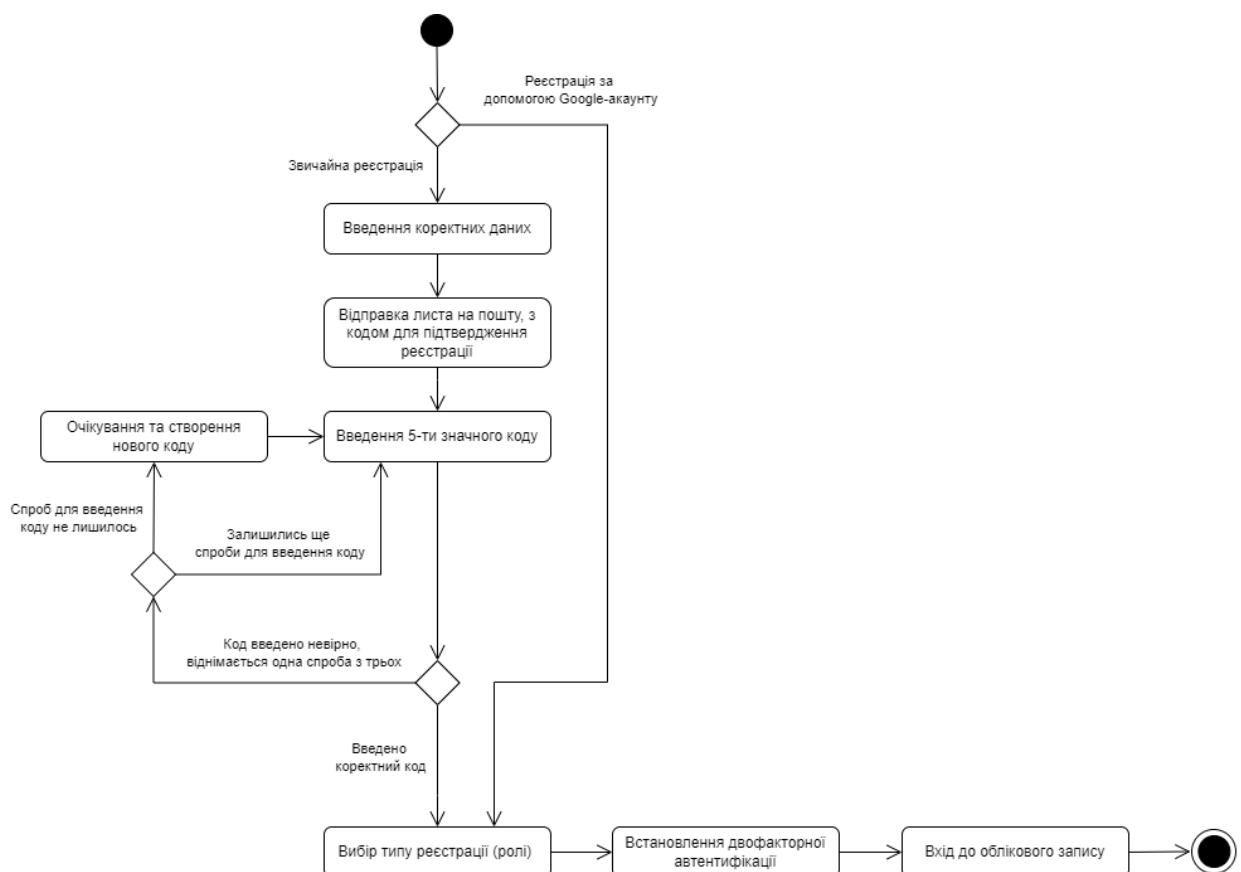


Рисунок 2.4 – Діаграма активності для процесу реєстрації

На початку користувачу буде представлена форма для введення даних про себе та встановлення паролю (у відповідному форматі). При цьому користувач матиме можливість зареєструватись за допомогою свого Google-акаунту та перейти одразу до етапу з вибором типу реєстрації.

У випадку звичайної реєстрації, після заповнення форми, користувачу придатиме лист на пошту, в якому буде зазначений 5-ти значний код, котрий йому потрібно ввести для підтвердження реєстрації на сайті. Задля безпеки та унеможливлення введення коду нескінченну кількість разів, користувачу надається лише три спроби. При кожній невдалій спробі надати неправильний код, їх кількість буде зменшуватись на одну. Врешті-решт, користувачу доведеться зачекати 60 секунд, щоб створити новий код.

Як зазначалось у попередньому підрозділі, MongoDB використовуватиметься задля збереження тимчасових даних, саме це і є той випадок, оскільки після введення коректного коду, сутність користувача записуватиметься в основну базу.

Після того, як сутність користувача фактично створена, він має можливість обрати тип реєстрації – індивідуальний (роль звичайного Клієнта) або ж бізнес (роль Власника місць для проживання).

Останнім кроком є встановлення двофакторної автентифікації, користувачу відображається QR-код, котрий йому потрібно відсканувати за допомогою телефону та додатку Google Authenticator. Ввівши його, він повністю завершує процес реєстрації.

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмотря О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		41

Наступна діаграма активності демонструє процес додавання нового житла власником (рис. 2.5).

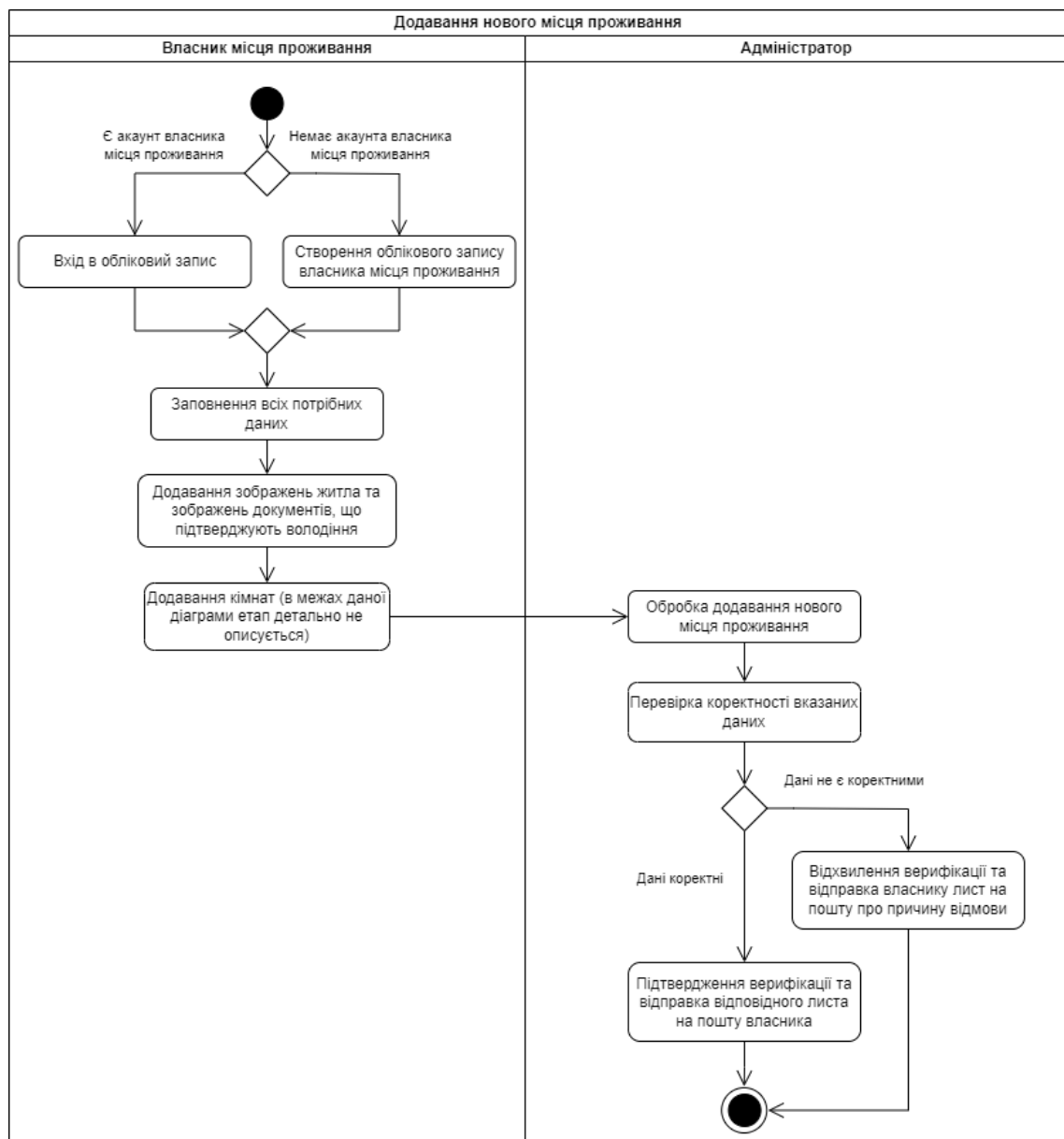


Рисунок 2.5 – Діаграма активності для процесу додавання нового місця проживання

Ключовим об'єктом у системі є місце проживання. Процес його створення починається з реєстрації облікового запису власника, у разі його відсутності. Після чого користувач зможе перейти до форми додавання нового місця проживання. На ній він матиме можливість заповнити всю необхідну інформацію, таку як назва, опис, зможе обрати тип житла, обрати, або додати власні зручності, встановити точку на карті, попередньо вказавши адресу, тощо.

Зробивши це, він додає зображення місця проживання та прикріплює документ, котрий засвідчує те, що саме цей користувач володіє даним житлом.

Після чого власник повинен додати кімнати до цього місця проживання, щоб його могли верифікувати. Цей процес детально не описується в межах даної діаграми.

Далі адміністратор опрацьовує інформацію пов’язану з новим місцем проживання. Переглядає коректність, правильність вказаних даних. Якщо на його погляд все в нормі, він верифікує дане житло. У випадку, якщо власник вказав якісь некоректні дані, додав фіктивні документи чи ще якимсь чином викликав сумніви у адміністратора – верифікація такого місця проживання відхиляється, з вказанням причини. В обох випадках власнику приходить лист на електронну пошту з результатом верифікації.

Для демонстрації процесу бронювання, було використано діаграму послідовності (рис. 2.6). Вона показує всі етапи взаємодії користувача із системою від початку до кінця цього процесу.

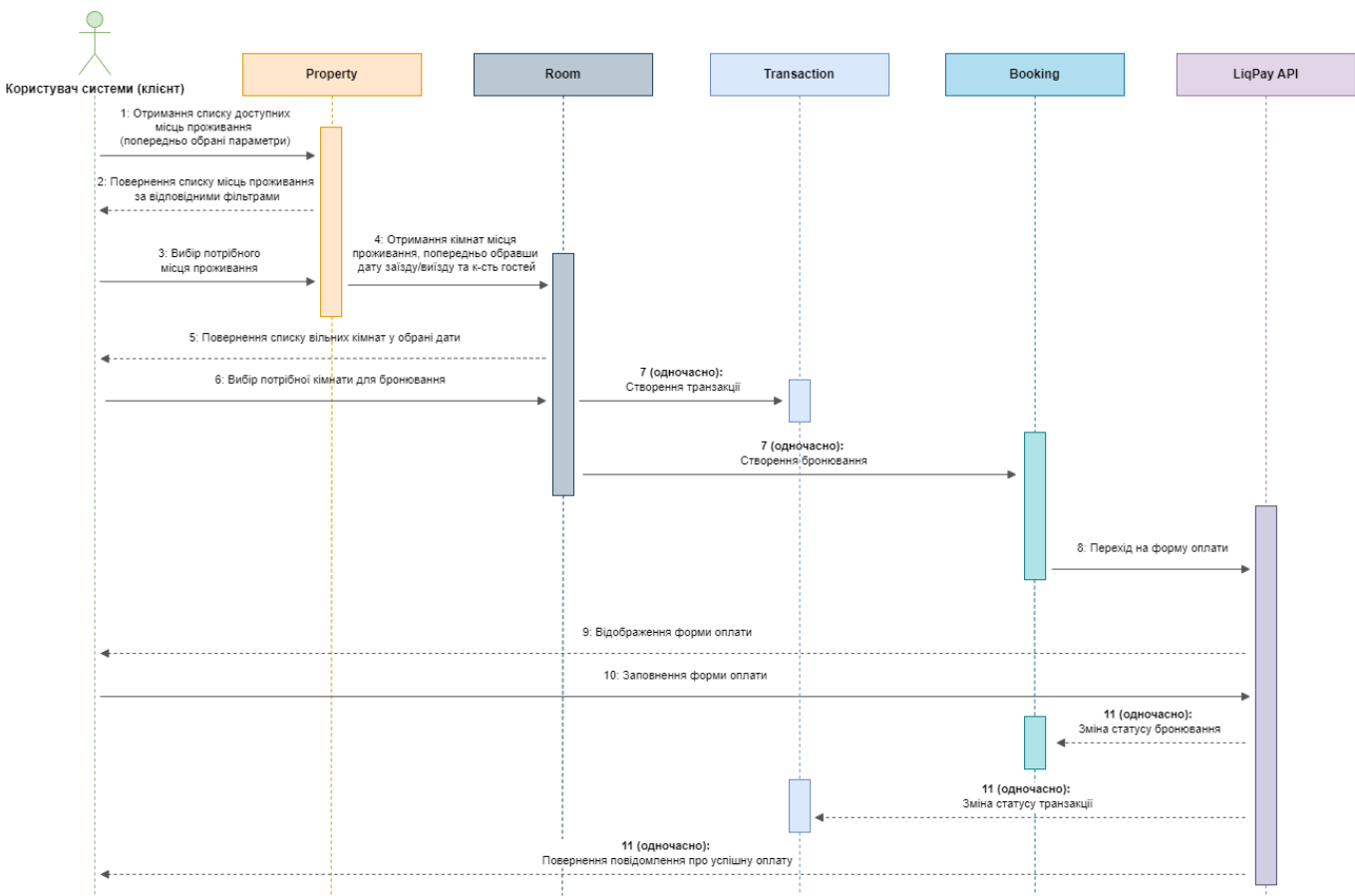


Рисунок 2.6 – Діаграма послідовності

З самого початку, користувач знаходить бажане місце проживання зі списку, попередньо обравши параметри для пошуку. Ознайомлюється з інформацією про нього. Після цього вказує дату заїзду та виїзду й кількість гостей, що буде проживати в кімнаті. З відфільтрованого списку, у якому відображаються саме вільні кімнати, клієнт обирає потрібну й переходить на форму з заповненням даних бронювання.

Впевнившись у деталях бронювання, користувач переходить на форму LiqPay, при цьому в системі створюється транзакція та бронювання зі статусами «Очікується оплата». Після введення платіжних даних на формі, клієнт оплачує бронювання, в цей час бронювання разом із транзакцією переходять у статус «Оплачено». Відповідно – кімната є заброньованою.

Отже, було побудовано різні типи діаграм, що детально відображають ті чи інші процеси, забезпечуючи їх чітке розуміння та дозволяючи передбачити різні сценарії.

2.4. Реалізація системи управління бронюванням готелів та житла для подорожей Україною

Визначивши варіанти використання, сформувавши вимоги, спроектувавши базу даних та розглянувши основні алгоритми роботи додатку, було перейдено до етапу його реалізації. У цьому розділі буде детально описано процес розробки найважливіших модулів системи.

Перш за все, варто приділити увагу одному з методів контролера PropertyController. Це POST-метод createProperty, що відповідає за додавання нового місця проживання (Додаток А).

Спочатку отримуються дані про житло з запиту та файли прикріплені під час створення. Одразу здійснюється перевірка завдяки сервісу CheckRequestDataService, що дозволяє переконатися чи прийшли усі необхідні дані з клієнта. Після цього отримується поточний користувач. У випадку, якщо такого не знайдено, або запит створений без попередньої аутентифікації, повернеться відповідна помилка.

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмотря О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		44

Використовуючи метод `isAddressExists` `servcy` `PropertyService`, відбувається перевірка чи справді обрана користувачем адреса житла існує. Для цього використовується окремо написаний сервіс `LocationService`, що дозволяє шукати локації за певними параметрами, у нашому випадку це місто, область та вулиця.

Після чого йде перевірка, чи обраний тип місць проживання справді існує і створюється об'єкт сутності `Property`. В середині цього методу також створюються нові зручності, якщо користувач такі вказав. Потім об'єкт валідується і у випадку, якщо якийсь з полів не є коректним, на фронтенд повертається об'єкт з помилками.

Останнім етапом при створенні є збереження файлів. Знову ж таки використовується `PropertyService` та один з його методів, у якому відбувається процес зберігання наданих фото житла та документів.

Далі дані поміщаються в базу даних, а власнику повертається повідомлення про успішне додавання нового місця проживання, проте для того щоб відбулась його верифікація, потрібно додати кімнати до нього. В логи записується відповідна подія.

Говорячи про процес бронювання, важливим етапом при його створенні є робота з зовнішнім сервісом `LiqPay`, котрий використовується для оплати [17]. Він застосовується в контролері `TransactionController` (Додаток Б).

У методі `createTransaction`, де відбувається створення сутностей `Transaction` та `Booking`, спочатку отримуються дані з запиту та знову ж таки з використанням `CheckRequestDataService` відбувається перевірка наявності потрібних. Після чого йдуть перевірки на існування юзера та обраної кімнати.

Створюється та валідується бронювання й створюється транзакція з використанням відповідних сервісів. Обидві сутності отримують статус «Очікується оплата». Після чого вони записуються в базу даних.

Далі йде безпосередня робота з `LiqPay` [2]. Генерується сигнатура з попередньо вказаними параметрами, що заповнюються відповідно до документації. Окрім цього використовується також `Webhook.site`. Це онлайн-

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмоторя О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		45

сервіс, який дозволяє тестувати та налагоджувати отримання вебхуків. У даному випадку, він використовується для отримання payment-callback від LiqPay, після чого дані перенаправляються на наш сайт для подальшої обробки.

У відповіді повертається сама сигнатура та її параметри, для використання на фронтенд-частині.

Після того як відбувається оплата, LiqPay повертає результат – payment-callback, як сказано вище, котрий оброблюється в методі paymentCallback.

На початку порівнюється сигнатура, що прийшла, з очікуваною, у випадку, якщо вони різняться повертається відповідне повідомлення. Далі знаходиться транзакція. Якщо статус оплати успішний, то він встановлюється для транзакції, також оброблена сума (з урахуванням комісії) й дата оплати. Бронювання переходить у статус «Оплачено». Користувачу надходить лист про успішне бронювання [7].

Якщо користувач не оплачує бронювання протягом 20 хвилин, воно разом зі створеною транзакцією будуть видалені завдяки cron-команді CronRemoveExpiredTransactions [3]:

```
protected function execute(InputInterface $input, OutputInterface $output): int
{
    $io = new SymfonyStyle($input, $output);

    $transactions = $this->entityManager->getRepository(Transaction::class)-
>getExpiredTransactions();

    $count = count($transactions);

    if($count === 0) {
        $io->success('No transactions are expired. Success at ' . date('d/m/Y h:i:s'));
        return Command::SUCCESS;
    }

    foreach ($transactions as $transaction) {
        /** @var Booking $booking */
        $booking = $transaction->getBooking();

        $this->entityManager->remove($booking);
        $this->entityManager->remove($transaction);
    }

    $this->entityManager->flush();

    $io->success('Success at ' . date('d/m/Y h:i:s'));

    return Command::SUCCESS;
}
```

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмотря О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		46

Ця команда виконується кожні 20 хвилин, для цього вона була додана до крон-файлу crontab.ini:

```
*/20 * * * * /usr/local/bin/php /var/www/diploma-kochura/api/bin/console cron:remove-  
transactions-and-bookings >> /cronfiles/logs/remove_transactions_and_bookings.log
```

Висновки до другого розділу

У цьому розділі, було розроблено та впроваджено функціональні складові системи управління бронюванням готелів та житла.

Перш за все, було детально розглянуто варіанти використання, для цього використовувалась також відповідна діаграма, наведено ряд бізнес-вимог, функціональних та нефункціональних вимог.

Особливу увагу було приділено розробці баз даних, котрі забезпечать ефективне зберігання та доступ до інформації. Було ретельно розглянуто та детально описано кожну з таблиць у контексті двох популярних систем управління базами даних – MySQL та MongoDB.

Після чого було спроектовано основні алгоритми роботи системи. Для цього використовувались діаграми активності та послідовності, що дозволили явно показати, як будуть влаштовані ті чи інші процеси.

Врешті-решт, було здійснено безпосередню реалізацію системи. При цьому було враховано всі спроектовані алгоритми та продуману логіку. Розглянуто ключові методи додатку, зокрема додавання нових місць для проживання, створення транзакцій, бронювань та взаємодія з різними сервісами.

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижевотря О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		47

РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З СИСТЕМОЮ УПРАВЛІННЯ БРОНЮВАННЯМ ГОТЕЛІВ ТА ЖИТЛА ДЛЯ ПОДОРОЖЕЙ УКРАЇНОЮ

3.1. Порядок встановлення та налаштування параметрів системи

Щоб працювати з додатком, потрібно мати систему, що відповідатиме вимогам. Також важливо дотримуватись певного порядку дій для уникнення можливих помилок та забезпечення стабільної роботи програмного забезпечення.

Бажані системні характеристики:

- для роботи серверу потрібно мати потужний процесор, з щонайменше 4 ядрами; мінімум 8 ГБ оперативної пам'яті;
- сучасна версія Linux (рекомендовано Ubuntu 22.04 LTS);
- стабільне підключення до мережі Інтернет.

Як розгорнути додаток:

- встановити Docker та Docker Compose;
- зклонувати актуальний програмний код з репозиторію;
- додати Google-ключі (OAuth) для можливості авторизації через нього, занести до конфігураційних файлів (.env) [11];
- створити акаунт на LiqPay, створити проект, отримати public та private ключі, занести до конфігураційних файлів (.env);
- впевнившись, що усі попередні дії виконані, слід виконати команду ./recreate_docker.sh, також при роботі на бекенді, варто час від часу виконувати команду ./clear_cache.sh, що дозволяє очищати кеш;
- для заповнення таблиць bed та room_type даними, слід виконати дві команди відповідно: AddDefaultBeds та AddDefaultRoomTypes.

Також було побудовано діаграму розгортання (рис. 3.1). Вона показує, як компоненти системи будуть розгорнуті на різних вузлах та допомагає зрозуміти, які компоненти системи взаємодіють один з одним та де вони будуть розміщені [4].

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмотря О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		48

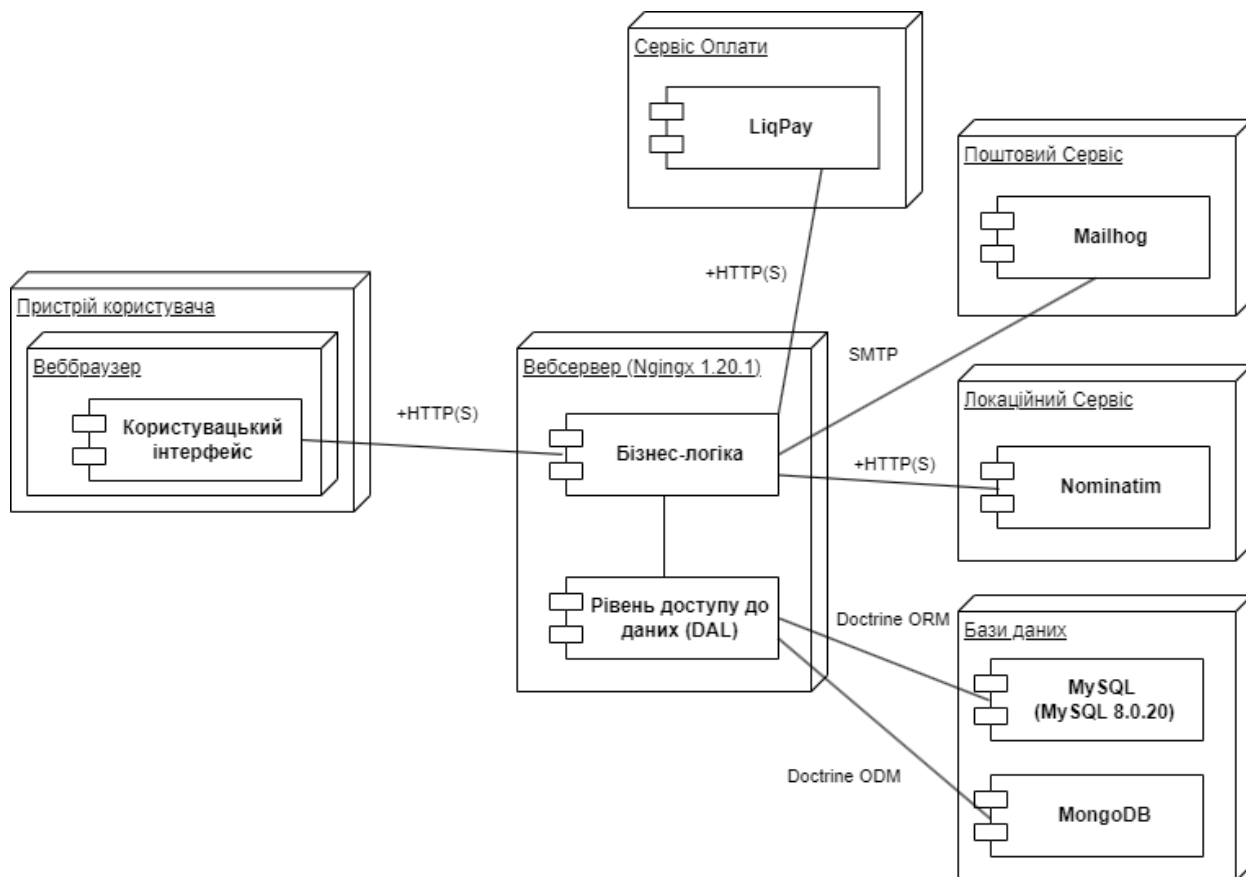


Рисунок 3.1 – Діаграма розгортання

З діаграми видно головні вузли та їхній зв'язок між собою. Вузол «Пристрій користувача», включає в себе вузол «Веббраузер», котрий містить компонент «Користувацький інтерфейс». «Пристрій користувача» пов'язаний з вузлом вебсервера, який в свою чергу пов'язаний з різними сервісами, що ми використовували протягом розробки, а саме: LiqPay, Mailhog, Nominatim [19]. А також з базами даних – MySQL та MongoDB відповідно.

3.2. Структура інтерфейсу системи управління бронюванням готелів та житла для подорожей Україною

Даний розділ демонструватиме інтерфейс додатку та принципи його роботи. Разом із скриншотами буде надаватись детальний опис.

Розпочавши роботу з вебсайтом, користувача зустрічає головна сторінка на якій розміщені всі доступні до бронювання місця для проживання (рис. 3.2). Є можливість вибрати кількість відображуваних місць на сторінці, показується загальна кількість по праву сторону, а також присутня можливість фільтрації. Нею можна скористатись, щоб знайти підходяще та шукане житло.

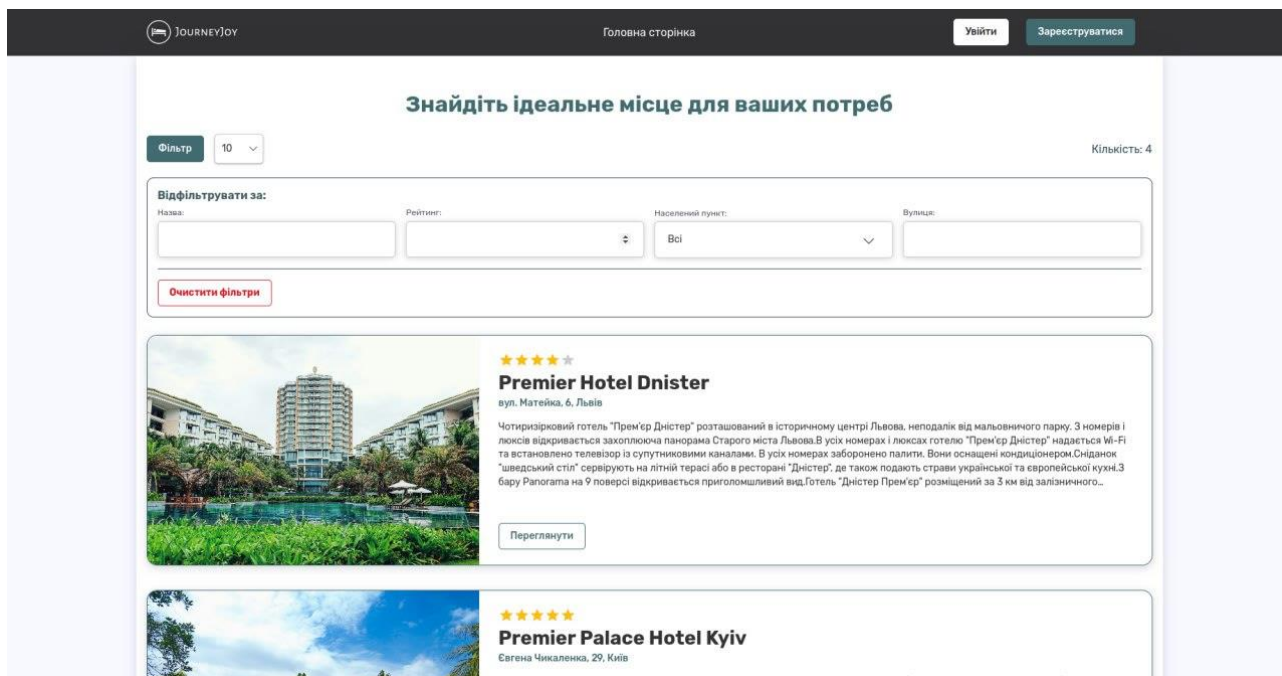


Рисунок 3.2 – Головна сторінка

Після вибору місця для проживання, користувач має можливість перейти до його деталей, просто натиснувши на нього. В деталях він може ознайомитись з галереєю зображень житла (рис. 3.3).

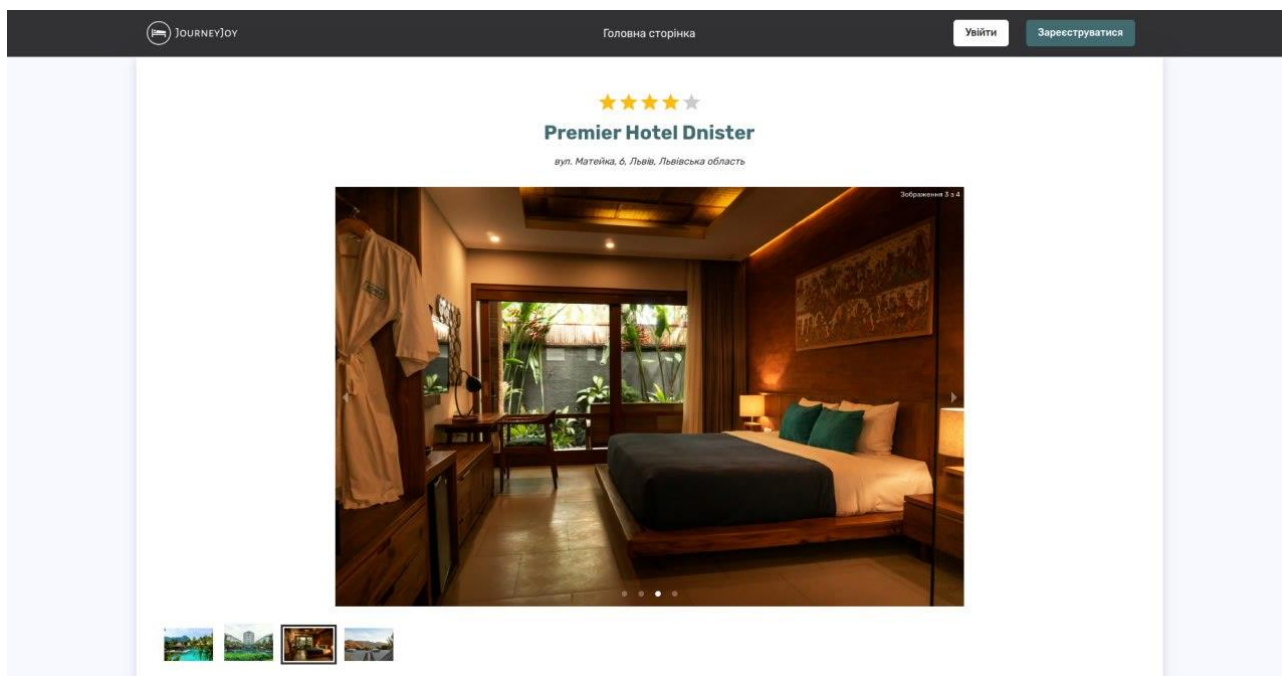


Рисунок 3.3 – Деталі місця проживання (галерея)

Після фото, йде секція з описом даного житла, в якій користувач може дізнатися більше інформації. Далі знаходиться перелік зручностей, що надає житло, це бути ресторан, басейн, тощо. Наступний блок містить в собі мапу, на якій позначено де саме знаходиться місце проживання. Окрім цього, її можна

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмотря О. В.				50
Змн.	Арк.	№ докум.	Підпис	Дата		

збільшувати, зменшувати, переглядати поруч цікаві місця і т.д (рис. 3.4). Для цього використовувалась бібліотека React Leaflet [12].

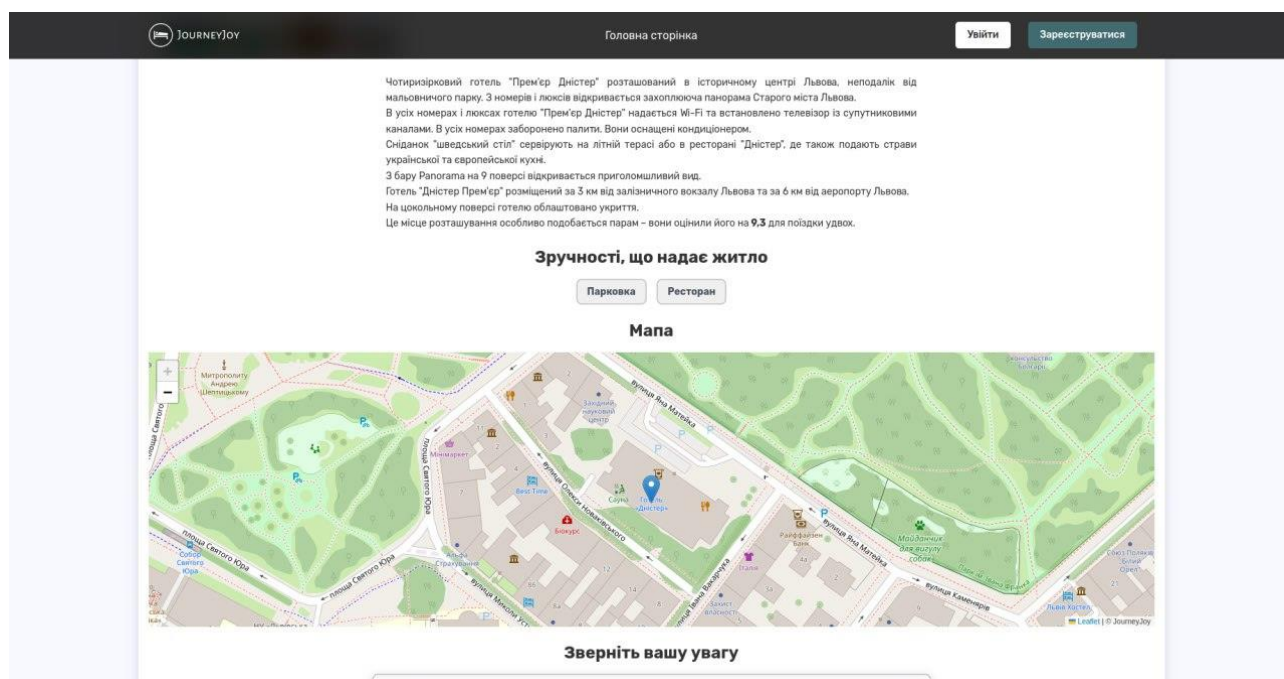


Рисунок 3.4 – Деталі місця проживання (опис, зручності, мапа)

За мапою слідує блок з важливою інформацією для того, хто хоче забронювати житло – це час заїзду та виїзду (рис. 3.5). Під ним знаходиться перелік кімнат даного місця проживання. Для того, щоб переглянути їх доступність, користувачу обов’язково потрібно вказати дату заїзду та дату виїзду й кількість гостей. Цей процес також буде згаданий нижче, при описі створення бронювань.

Також є можливість ближче розглянути усі зображення кімнати, просто натиснувши на головне з них (рис. 3.6).

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижомотря О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		51

Головна сторінка

Увійти
Зареєструватися

Зверніть вашу увагу

Час заїзду: 11:00

Час виїзду: 10:00

Кімнати

Дата заїзду: 15-06-2024

Дата виїзду: 22-06-2024

Кількість гостей: 3

Очистити

Тримісний номер

К-ть гостей, що можуть зупинитися: 3

Зручності:

Міні-бар

Холодильник

Сейф

Ліжка: 1 - Односпальне ліжко, 1 - Широке двоспальне (king size) ліжко

Ціна: 16800 грн

Забронувати

Відгуки

Рисунок 3.5 – Деталі місця проживання (час заїзду/виїзду, кімнати)

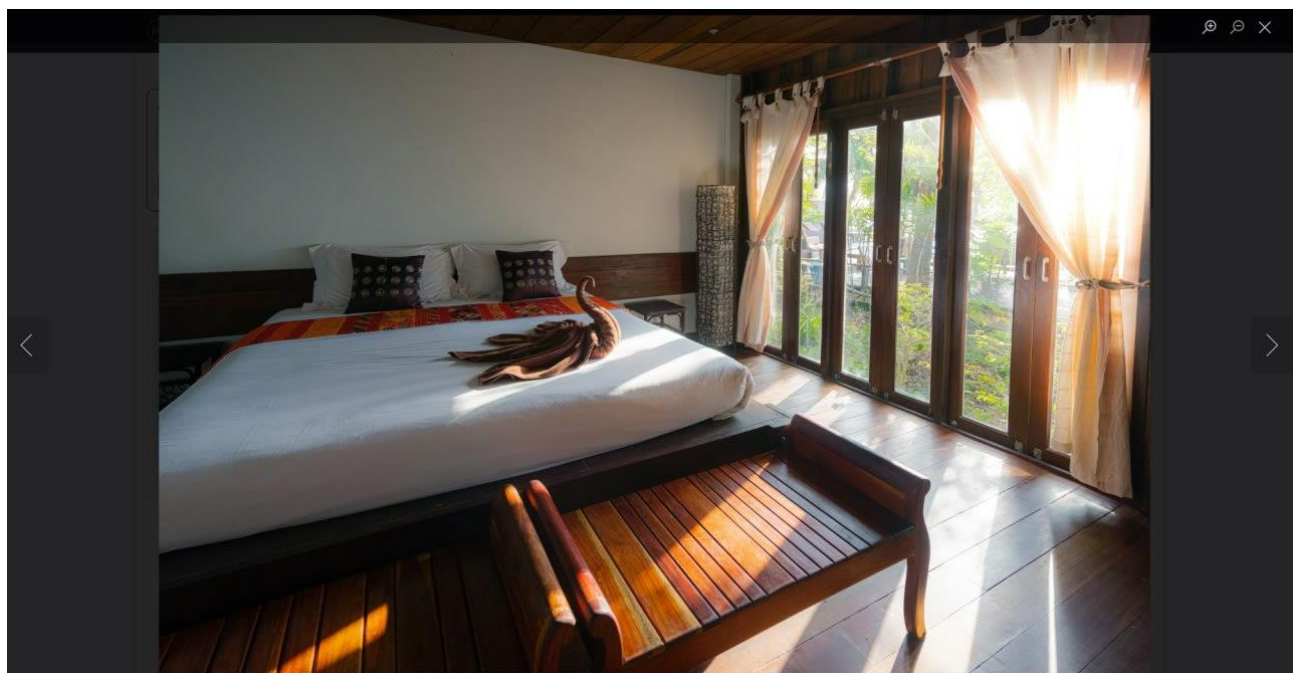


Рисунок 3.6 – Перегляд усіх зображень кімнати

Знизу деталей знаходяться відгуки. Можна ознайомитись з досвідом проживання попередніх клієнтів, прочитавши їх коментар та побачивши оцінку (рис. 3.7).

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмоторя О. В.				52
Змн.	Арк.	№ докум.	Підпис	Дата		

Відгуки

Визначте кількість зірок:

★★★★★

Залишити відгук:

Надіслати

Віталій

Прекрасний готель! Кімнати чисті та комфортні, персонал дуже доброзичливий і уважний до деталей. Локація зручна - близько до всіх основних пам'яток і транспортних вузлів. Сніданки смачні та різноманітні.

★★★★★

12.06.2024 02:21

Рисунок 3.7 – Деталі місця проживання (відгуки)

На сторінці реєстрації розміщена форма з полями імені, прізвища, пошти, а також паролю з валідатором. Також можна скористатись можливістю реєстрації через свій Google-акаунт (рис. 3.8).

Створити обліковий запис

Введіть ваші дані в поля нижче

Ім'я:

Прізвище:

E-mail:

Пароль:

8-50 символів | Велика буква | Маленька буква | Цифра | Символ

☐ Я погоджуюся з Умовами користування та Політикою конфіденційності

Реєстрація

або використовуйте соціальні мережі

Продовжити через Google

Рисунок 3.8 – Форма реєстрації

Після введення коректних даних, користувач переходить на етап введення 5-ти значного коду, з листа, що прийшов йому на вказану, в попередній формі, пошту. Код дійсний протягом 10 хвилин, можна відправити знову новий код за потреби, або ж якщо закінчились 3 спроби введення (рис. 3.9 – 3.10).

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижомотря О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		53

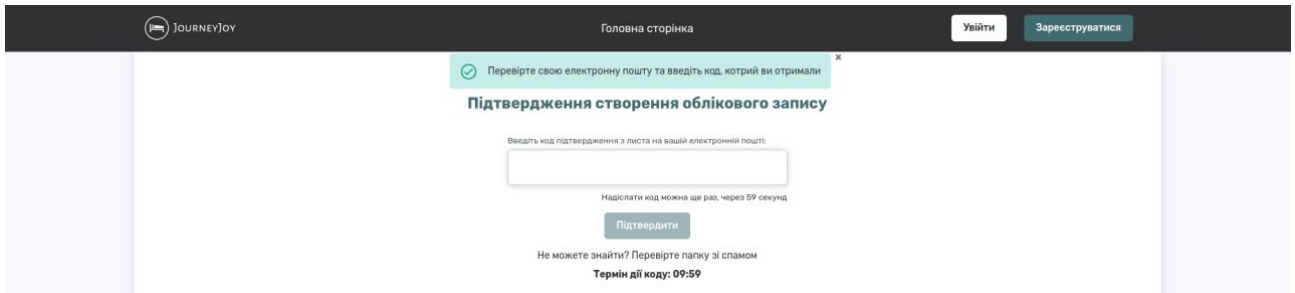


Рисунок 3.9 – Форма для введення 5-ти значного коду

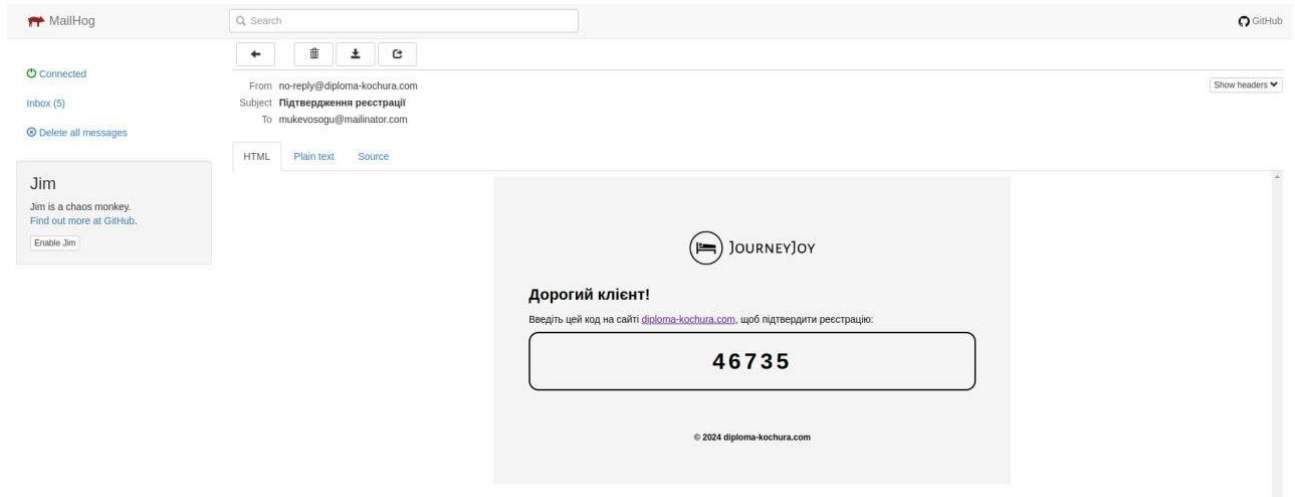


Рисунок 3.10 – Лист з кодом, що приходить користувачу на пошту

Після успішного введення, користувач обирає бажану роль. У випадку, якщо він хоче просто оформлювати бронювання, залишати відгуки і т.д, він обирає «Індивідуальний» тип реєстрації. Якщо ж це власник житла, котрий хоче розмістити його на вебсайті, то підходящий тип – «Бізнес» (рис. 3.11).

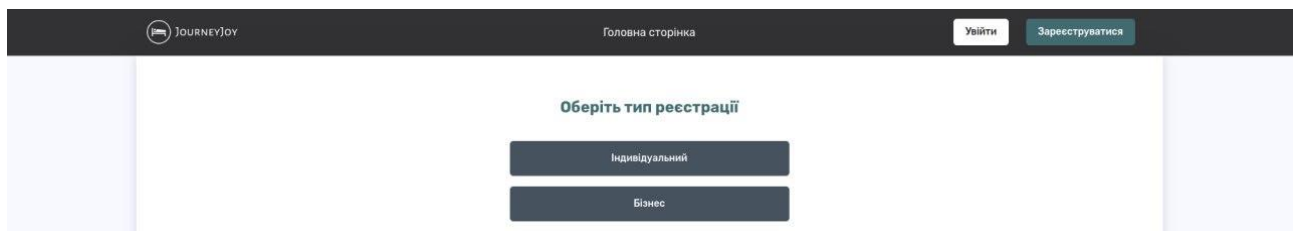


Рисунок 3.11 – Вибір типу реєстрації

Останнім кроком при реєстрації, є встановлення двуфакторної авторизації, шляхом сканування QR-коду на телефоні, в додатку Google Authenticator. Ввівши код, відбувається успішний вхід в акаунт (рис. 3.12).

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмотря О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		54

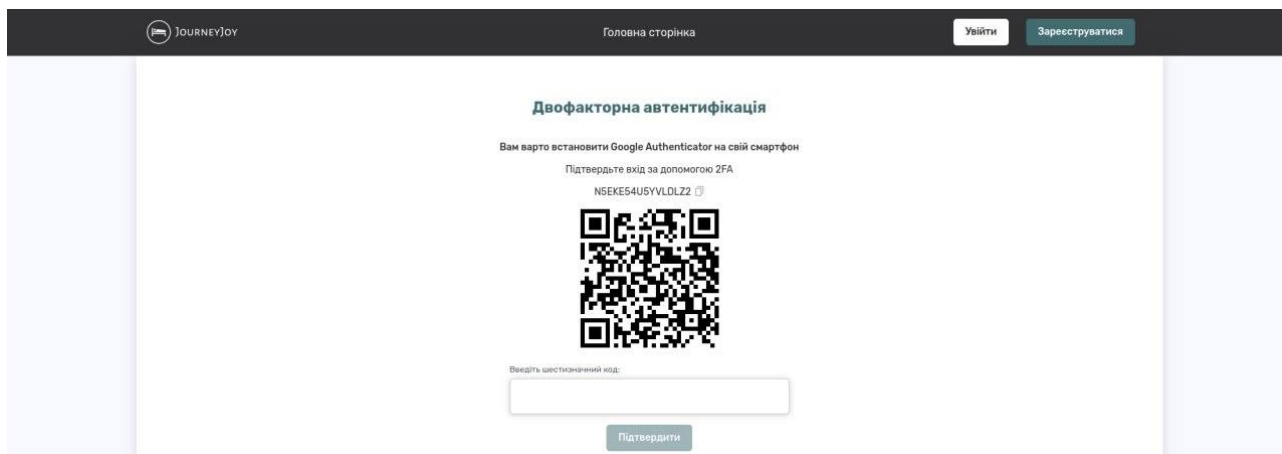


Рисунок 3.12 – Сканування QR-коду та введення коду двофакторної авторизації

Як сказано вище, зареєструватись у системі, а в подальшому входити в акаунт, можна використовуючи Google (рис. 3.13).

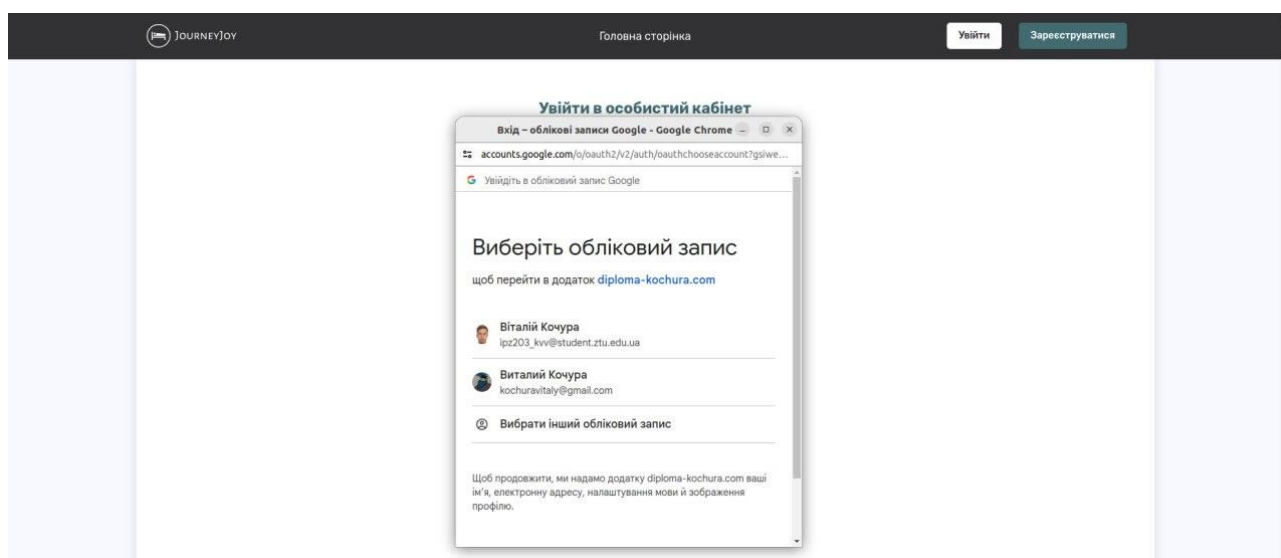


Рисунок 3.13 – Вхід з використанням Google-акаунту

Увійшовши з-під ролі Клієнта, є можливість переглянути свої замовлення, попередньо відфільтрувавши їх за багатьма параметрами. Також наявна можливість вибору кількості відображуваних записів та поруч їх загальна кількість. Такий блок буде присутній на багатьох сторінках, що будуть описані в подальшому (рис. 3.14).

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмотря О. В.				55
Змн.	Арк.	№ докум.	Підпис	Дата		

Головна сторінка

Бронювання

Фільтр 10

Кількість: 2

Відфільтрувати за:

ID:

Назва житла:

Назва кімнати:

Статус: Bcl

Дата заїзду:

Дата заїзду від:

Дата заїзду до:

Дата виїзду:

Дата виїзду від:

Дата виїзду до:

Дата створення:

Дата від:

Дата до:

Очистити фільтри

ID	Назва житла	Назва кімнати	Прізвище та ім'я	Статус	Дата заїзду	Дата виїзду	Дата створення
32993556-e734-42a1-9738-0d9084b42549	Optima Collection Парк Готель Івано-Франківськ	Чотиримісний номер	Кочура Віталій	Оплатено	28.06.2024	06.07.2024	12.06.2024 02:16
3dda27bc-7a11-43dc-bf6e-c09944162538	Premier Hotel Drister	Номер-студію	Кочура Віталій	Оплатено	17.06.2024	22.06.2024	12.06.2024 02:04

Рисунок 3.14 – Список бронювань

Перейшовши у деталі бронювання, Клієнт має можливість детальніше ознайомитись зі своїми бронюваннями (рис. 3.15).

Головна сторінка

Деталі бронювання

ID	32993556-e734-42a1-9738-0d9084b42549
Назва готелю	Optima Collection Парк Готель Івано-Франківськ
Назва кімнати	Чотиримісний номер
Сума бронювання	6400 грн
Прізвище та ім'я	Кочура Віталій
Статус	Оплатено
Дата заїзду	28.06.2024
Дата виїзду	06.07.2024
Дата створення	12.06.2024 02:16

Рисунок 3.15 – Деталі бронювання

Після того як Клієнт обирає бажану кімнату, вказавши попередньо дату заїзду та дату виїзду й кількість гостей, натиснувши «Забронювати», він переходить на форму бронювання кімнати. На ній користувач вводить ім'я, прізвище та телефон того, хто заїжджатиме, а також має можливість ознайомитись знизу з деталями свого бронювання. (рис. 3.16)

Рисунок 3.16 – Форма бронювання кімнати

Після успішного заповнення форми, у користувача відкривається вкладка з оплатою, з використанням сервісу LiqPay. На ній одразу зазначена сума платежу та залишається тільки ввести коректні платіжні дані (рис. 3.17).

Рисунок 3.17 – LiqPay-форма оплати

У разі успішної оплати, користувачу виводиться відповідне повідомлення. В цей час вже повернувся payment-callback та транзакція разом з бронюванням набули потрібних статусів (рис. 3.18).

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмотря О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		57

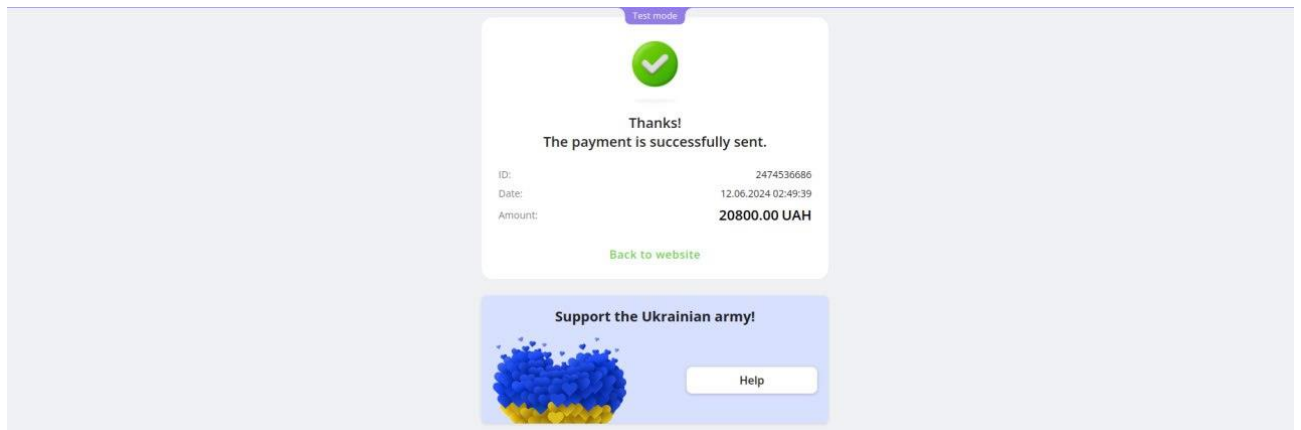


Рисунок 3.18 – Повідомлення про успішність оплати

Може статися випадок, коли користувач по якимось причинам перервав процес оплати. Це передбачено системою, у такому разі він має можливість протягом 20 хвилин олатити бронювання, зайшовши у його деталі та натиснувши кнопку «Оплатити» (рис. 3.18).

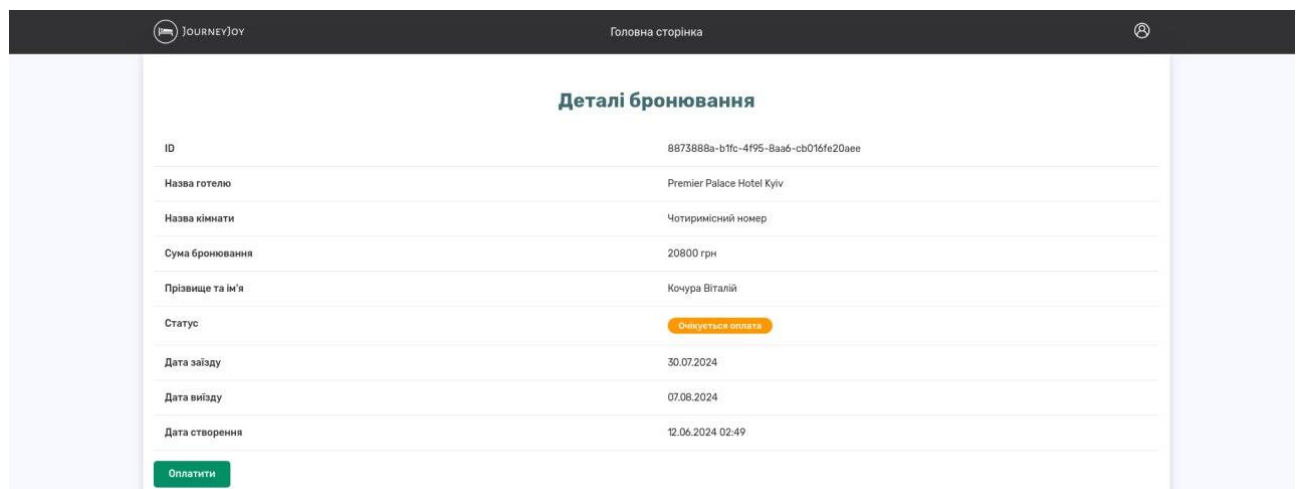


Рисунок 3.19 – Можливість оплати протягом 20 хвилин в деталях бронювання

Після успішного бронювання, Клієнту надходить лист на його пошту, в якому зазначаються деталі бронювання (рис. 3.20).

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмоторя О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		58

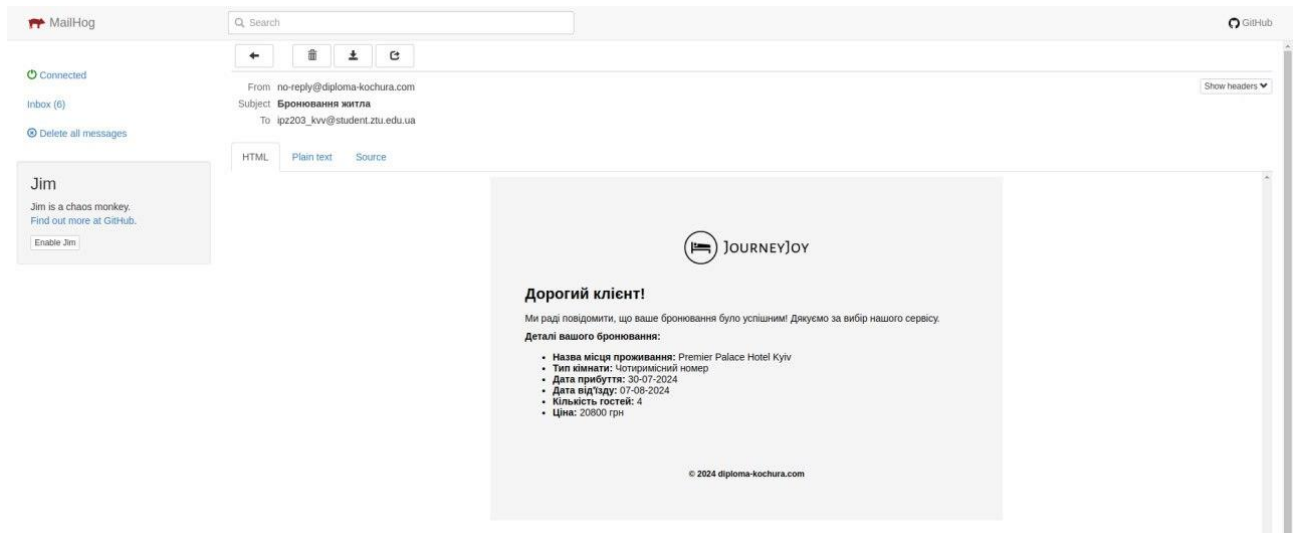


Рисунок 3.20 – Лист про успішне бронювання разом з його деталями

Якщо Клієнту хочеться поділитися своїм досвідом проживання, він може залишити відгук під житлом, де бронював кімнату раніше. При відправці відгуку він має ввести ідентифікатор бронювання (рис. 3.21 – 3.22).

Рисунок 3.21 – Форма для відгуків

Рисунок 3.22 – Модальне вікно у якому варто ввести ID бронювання

Далі ввійшовши у систему з-під ролі Власника, демонструються усі місця проживання котрими він володіє. Можна з легкістю знайти потрібне за допомогою фільтрації (рис. 3.23).

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмоторя О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		59

Головна сторінка

Кількість: 3

Місця для проживання

Створити нове місце

Фільтр 10

Відфільтрувати за:

Назва:

Опис:

Тип житла:

Вулиця:

Населений пункт:

Активність:

Верифікація:

Дата створення:

Дата від:

Дата до:

Очистити фільтри

Назва	Опис	Тип житла	Вулиця	Населений пункт	Активність	Верифікація	Дата створення
Optima Collection Житомир	Готель "Optima Collection Житомир" розташований в історичному та діловому центрі...	Готелі	Замковий Майдан, 5/8	Житомир	Вимкнено	Очікує верифікації	12.06.2024 01:47
Optima Collection Дніпро	Сучасний готель зі східними елементами декору та безкоштовним Wi-Fi розташований у центр...	Готелі	Шевченко, 53а	Дніпро	Увімкнено	Верифіковане	12.06.2024 01:32
Optima Collection Панк	У цьому готелі є						

При потребі додати нове житло, Власник натискає на кнопку «Створити нове місце» та переходить на форму, де має заповнити дані. Спочатку йому варто вказати назву, опис завдяки react-quill відформатувавши його попередньо. Обирається тип місця проживання та вказується його рейтинг. Для додавання зручностей до свого житла можна обрати вже існуючі у системі, або ж створити свої шляхом вводу, проте вони не будуть активні до тих пір, поки не будуть переглянуті Адміністратором або житло не буде верифіковане (рис. 3.24). Також є можливість вказати час заїду та час виїзду.

JOURNEYJOY

Головна сторінка

Додати місце

Назва:

Готель для демонстрації

Опис:

Normal : B I U G " A [] | = | ☞ 📷 ⌂ ✎
Опис готелю для демонстрації. Опис готелю для демонстрації. Опис готелю для демонстрації. Опис готелю для демонстрації.

Тип місця проживання:

Апартаменти

Вкажіть кількість зрок вашого життя (при бажанні):

★★★★★

Очистити рейтинг

Зручності (при потребі створіть нову, вибавив її при виборі):

Ресторан x Парковка x

Час заїзду

14:00

x

Час виїзду

12:00

x

Після цього Власнику варто ввести точну адресу свого житла задля того, щоб воно коректно відображалось на карті. Спочатку він обирає населений пункт з представлених, або ж вписує свій, якщо такий відсутній. Далі вказується область та вулиця, через кому з номером. У випадку, якщо було вказано все правильно, на карті з'явиться точка за вказаною локацією. Є випадки, коли точка вказана не точно, можна перемістити її в потрібне місце, в межах адреси.

Після відображення на карті, користувачу варто додати ряд зображень свого місця проживання (рис. 3.25).

Рисунок 3.25 – Форма створення місця проживання (адреса, мапа, зображення)

Останнім кроком при створенні нового місця проживання є додавання документів, котрі підтверджують, що Власник справді володіє даним житлом (рис. 3.26).

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмотря О. В.				61
Змн.	Арк.	№ докум.	Підпис	Дата		

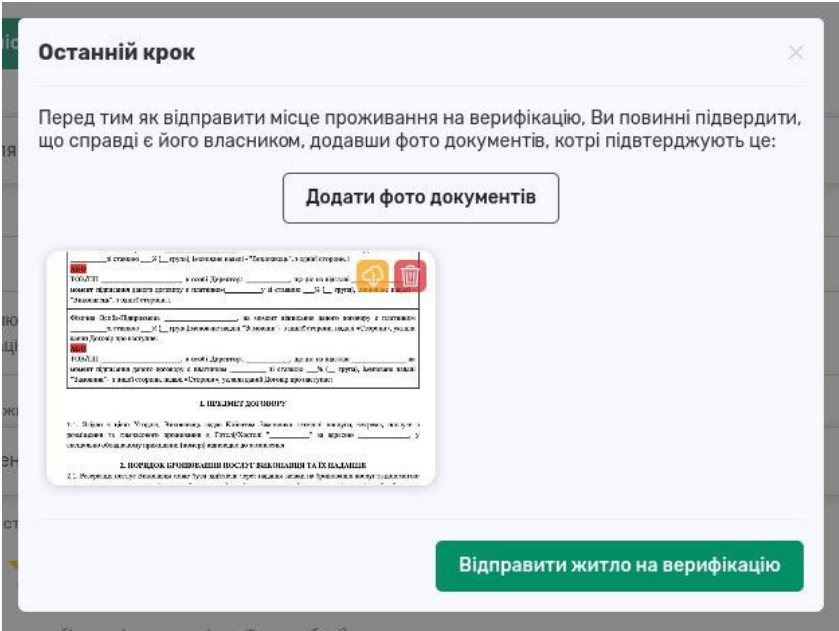


Рисунок 3.26 – Модальне вікно у якому додається фото документів

Перейшовши в деталі, Власник має змогу ознайомитись ближче з житлом, редагувати його, якщо воно не знаходиться в статусі «Не верифіковане» (рис. 3.27).

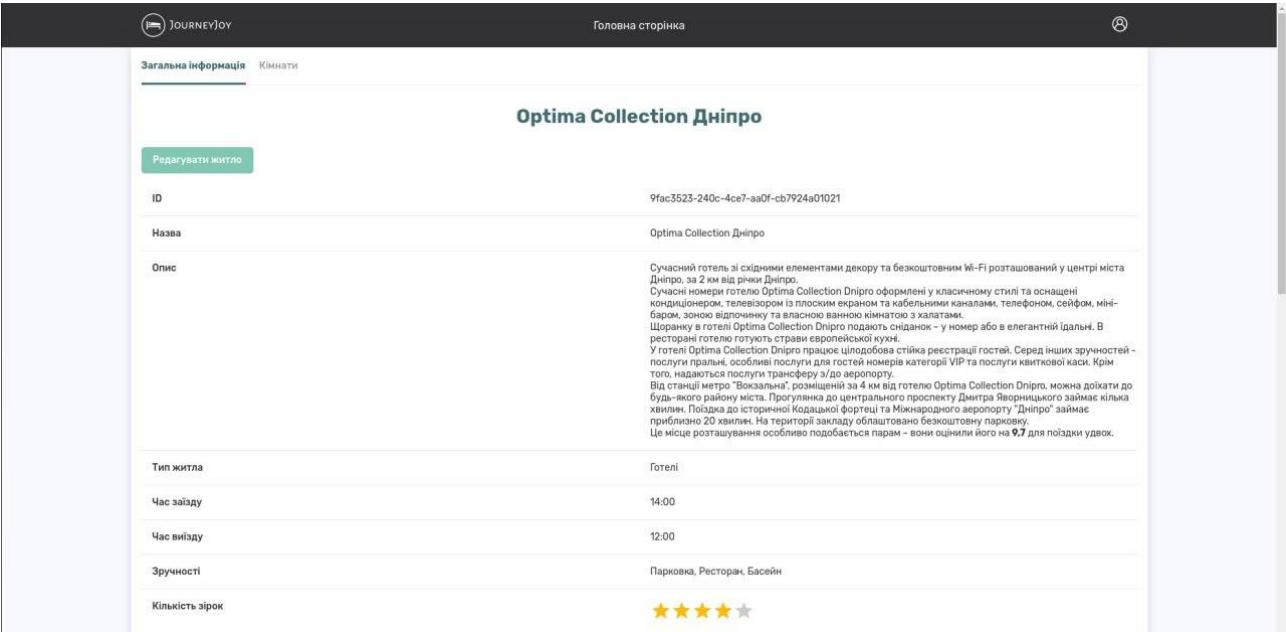


Рисунок 3.27 – Деталі місця для проживання

Власник має можливість змінити доступність місця проживання, вимкнувши його з певних причин (рис. 3.28), а також видалити взагалі (рис. 3.29).

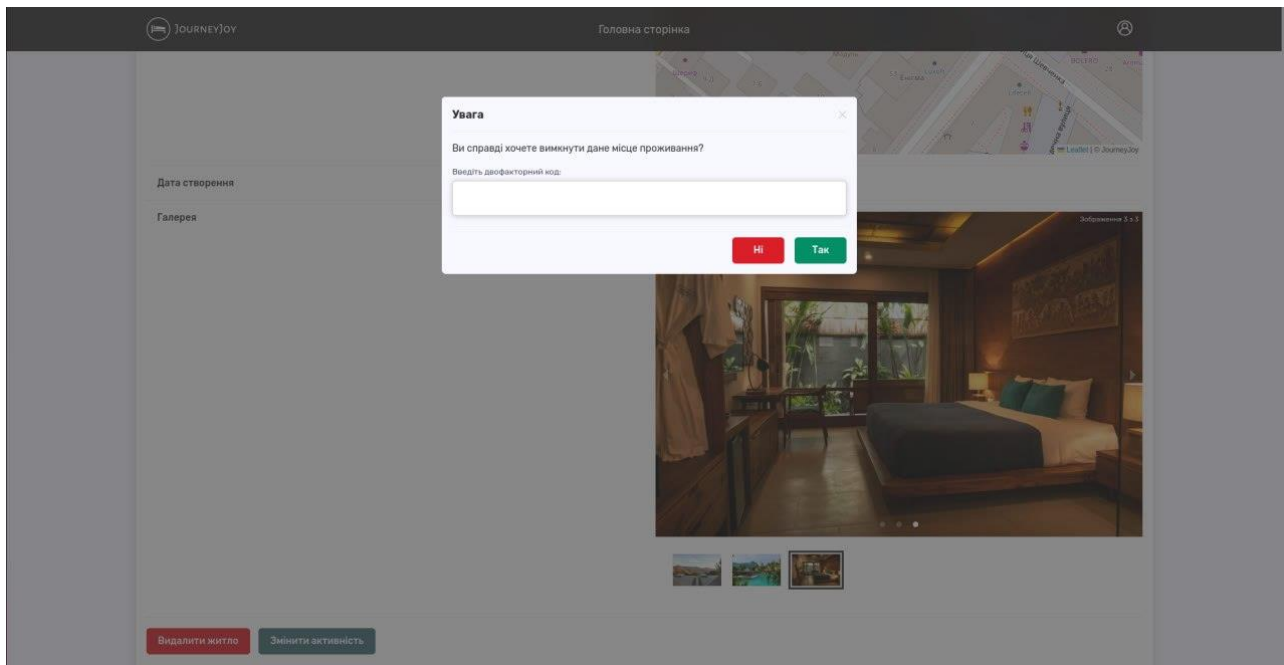


Рисунок 3.28 – Зміна доступності місця проживання

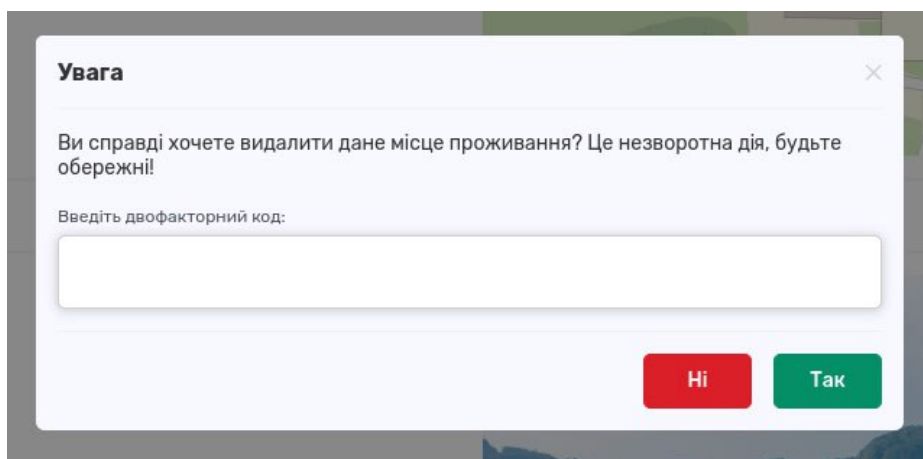


Рисунок 3.29 – Видалення місця для проживання

Обов'язково житло має мати свої кімнати. В деталях місця проживання переключившись на табу «Кімнати», Власник має можливість переглянути список кімнат (рис. 3.30).

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмотря О. В.				63
Змн.	Арк.	№ докум.	Підпис	Дата		

Головна сторінка

Загальна інформація

Кімнати

Додати нову кімнату

Фільтр

10

Кількість: 3

Відфільтрувати за:

Тип кімнати:

Всі

Кількість гостей:

1

Ціна за ніч:

Ціна за ніч від:

Ціна за ніч до:

Дата створення:

Дата від:

Дата до:

Очистити фільтри

Тип кімнати	Кількість гостей	Ціна за ніч	Дата створення
Люкс	3	1500 грн	12.06.2024 01:44
Однісний номер	1	600 грн	12.06.2024 01:43
Однісний номер	1	550 грн	12.06.2024 01:42

Рисунок 3.30 – Список кімнат

Також скориставшись кнопкою «Додати нову кімнату» йде перехід на відповідну форму. Завдяки ній можна створювати нові кімнати, обравши для них тип, місце проживання за яким закріплюється, кількість гостей, що можуть зупинитись й ціну. По аналогії до зручностей місць проживання, можна обрати вже створені, або ж запропонувати свої. Варто обрати кількість певних ліжок в кімнаті й додати зображення (рис. 3.31).

Головна сторінка

Додати кімнату

Тип кімнати:

Тримісний номер

Місце проживання:

Готель для демонстрації

К-ть гостей, що можуть зупинитись:

3

Ціна за ніч, грн:

800

Зручності (при потребі створіть нову, вибравши її при виборі):

Сейф x

Холодильник x

Кондиціонер x

Праска x

Телевізор x

Однісальне ліжко:

Двоспальне ліжко:

Широке двоспальне (king size) ліжко:

1

0

1

Додати фото до галереї

Прибрати всі фото

Рисунок 3.31 – Форма додавання нової кімнати

Дані про кімнату можна переглянути детальніше, або ж змінити їх у разі необхідності (рис. 3.32).

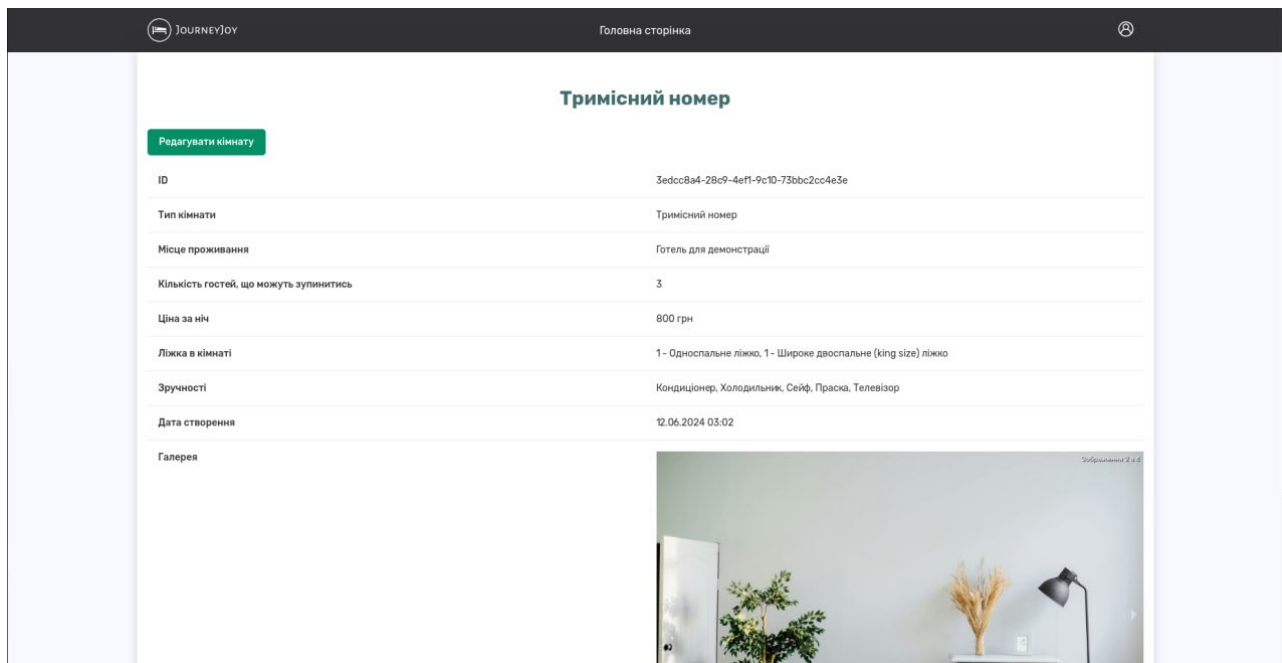


Рисунок 3.32 – Деталі кімнати

При роботі з бронюваннями, Власник повинен перевести їх в статус «Завершено», якщо Клієнт вже виїхав, для того щоб кімната, що була заброньована знову була доступна. Це можна зробити в деталях бронювання (рис. 3.33).

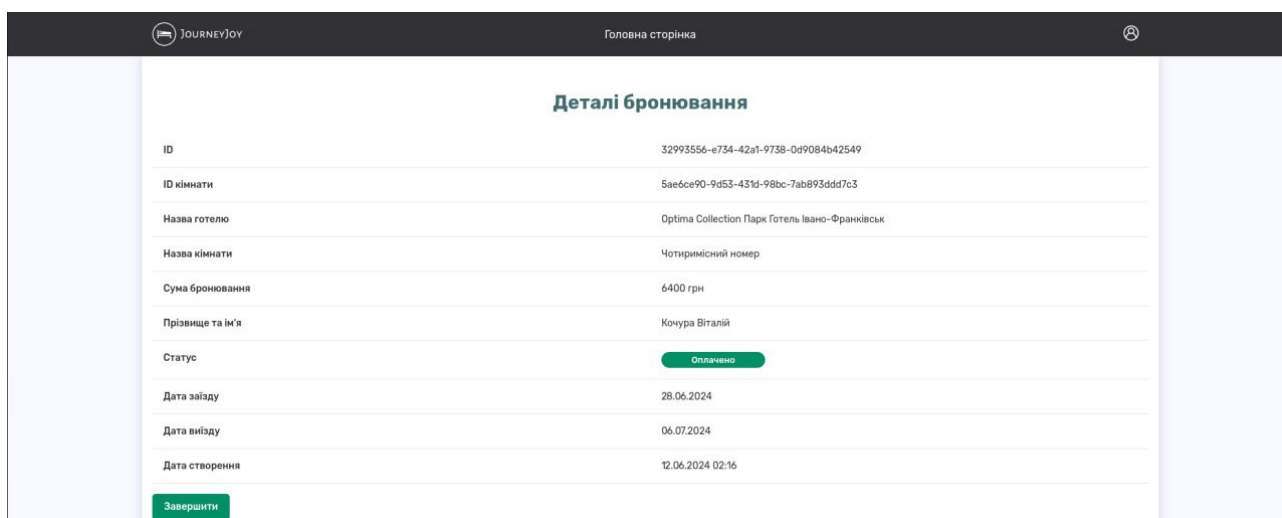


Рисунок 3.33 – Деталі бронювання, де Власник може завершити його

Якщо говорити про роль Адміна, то він працює з багатьма сутностями в системі. Розпочати варто з місць проживань. Має можливість переглядати їх за списком (рис. 3.34).

Головна сторінка

Місця для проживання

Фільтр

10

Кількість: 6

Відфільтрувати за:

Ім'я власника:

Прізвище власника:

Назва:

Тип житла:

Населений пункт:

Активність:

Верифікація:

Дата створення:

Дата від:

Дата до:

Очистити фільтри

Ім'я власника	Прізвище власника	Назва	Тип житла	Населений пункт	Активність	Верифікація	Дата створення
Sonia	Pittman	Готель для демонстрації	Апартаменти	Житомирська	Вимкнено	Почекує верифікації	12.06.2024 02:57
Kirk	Reid	Premier Hotel Onister	Готелі	Львів	Увімкнено	Верифіковане	12.06.2024 01:59
Kirk	Reid	Premier Palace Hotel Kyiv	Готелі	Київ	Увімкнено	Верифіковане	12.06.2024 01:55
Sonia	Pittman	Optima Collection Житомир	Готелі	Житомир	Вимкнено	Почекує верифікації	12.06.2024 01:47
Sonia	Pittman	Optima Collection Дніпро	Готелі	Дніпро	Увімкнено	Верифіковане	12.06.2024 01:32
Sonia	Pittman	Optima Collection Парк Готель Івано-	Готелі	Івано-Франківськ	Увімкнено	Верифіковане	12.06.2024 01:06

Рисунок 3.34 – Список усіх місць для проживання

У деталях він має можливість переглянути усю інформацію про готель, перед тим як верифікувати його. Також перейти по табам «Кімнати» та «Бронювання» і ознайомитись з їх переліками (рис. 3.35).

Головна сторінка

Загальна інформація

Кімнати

Бронювання

Готель для демонстрації

ID

24bcc6eb-1324-4310-af0a-da813f2af4b6

Власник

Sonia Pittman

Назва

Готель для демонстрації

Опис

Опис готелю для демонстрації. Опис готелю для демонстрації. Опис готелю для демонстрації. Опис готелю для демонстрації.

Тип житла

Апартаменти

Час заїзду

14:00

Час виїзду

12:00

Зручності

Парковка, Ресторан

Кількість зірок

★★★★★

Активність

Вимкнено

Верифікація

Почекує верифікації

Адреса

Чуднівська, 103, Житомирська, Житомир область

Мапа

Рисунок 3.35 – Деталі місця проживання

Адміністратор має можливість змінити активність, видалити житло, верифікувати його або ж не верифікувати (рис. 3.36 – 3.37).

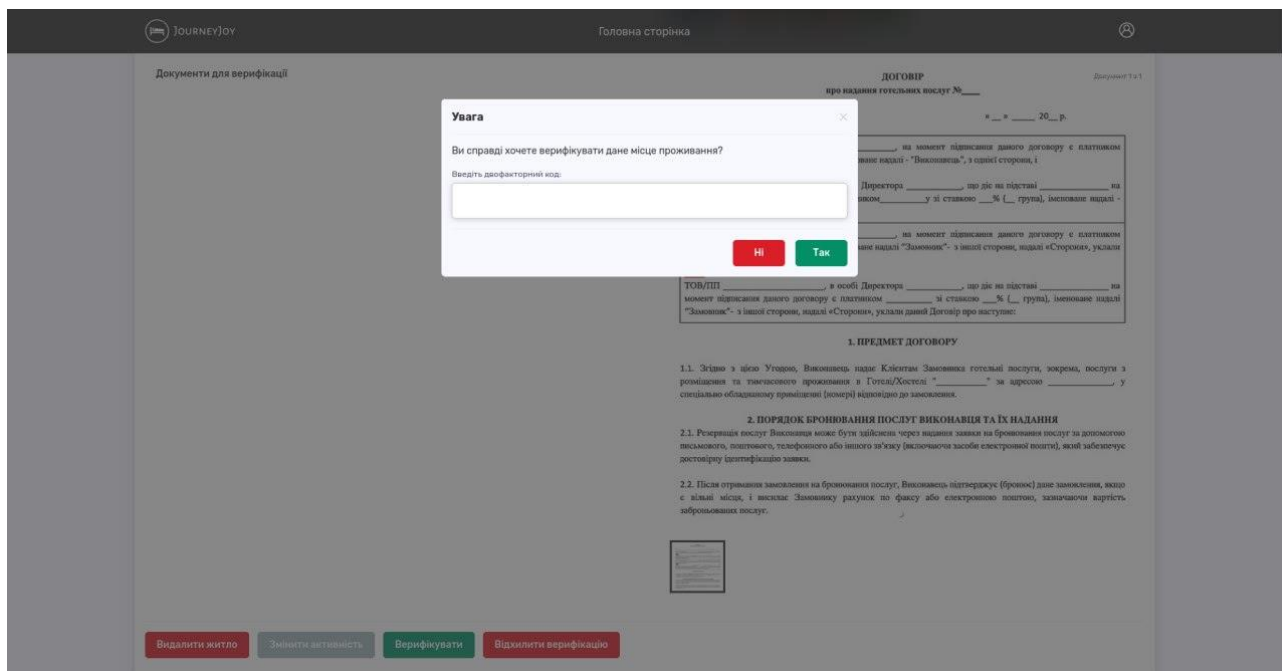


Рисунок 3.36 – Модальне вікно, з вводом коду двуфакторної авторизації для верифікації місця проживання

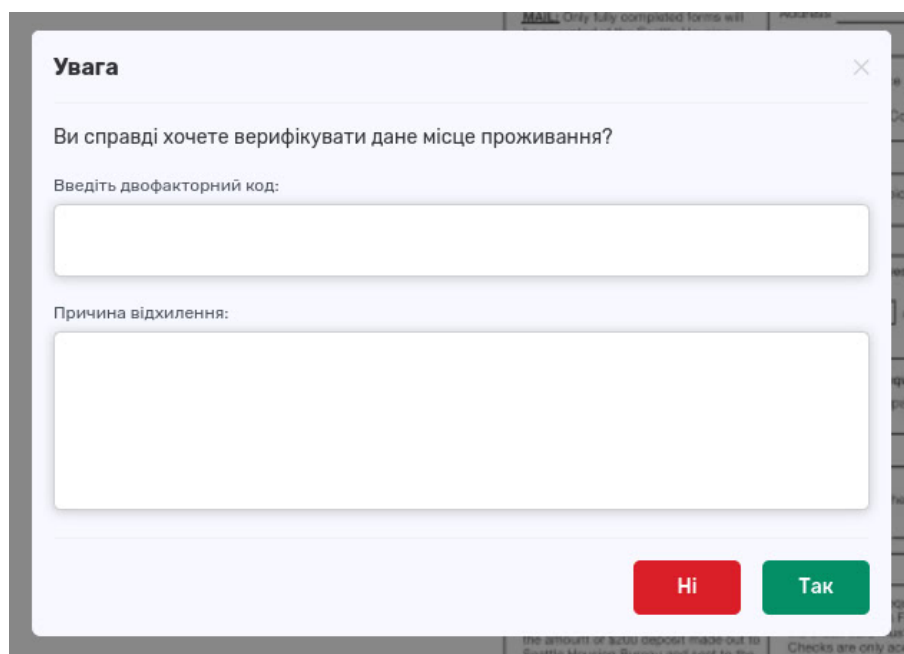


Рисунок 3.37 – Модальне вікно, з вводом коду двуфакторної авторизації для відхилення верифікації місця проживання, де можна вказати причину

При зміні статусу верифікації, Власнику приходиться лист на пошту, у якому зазначається або повідомлення про успішність, або причина відхилення (рис. 3.38).

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижомотря О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		67

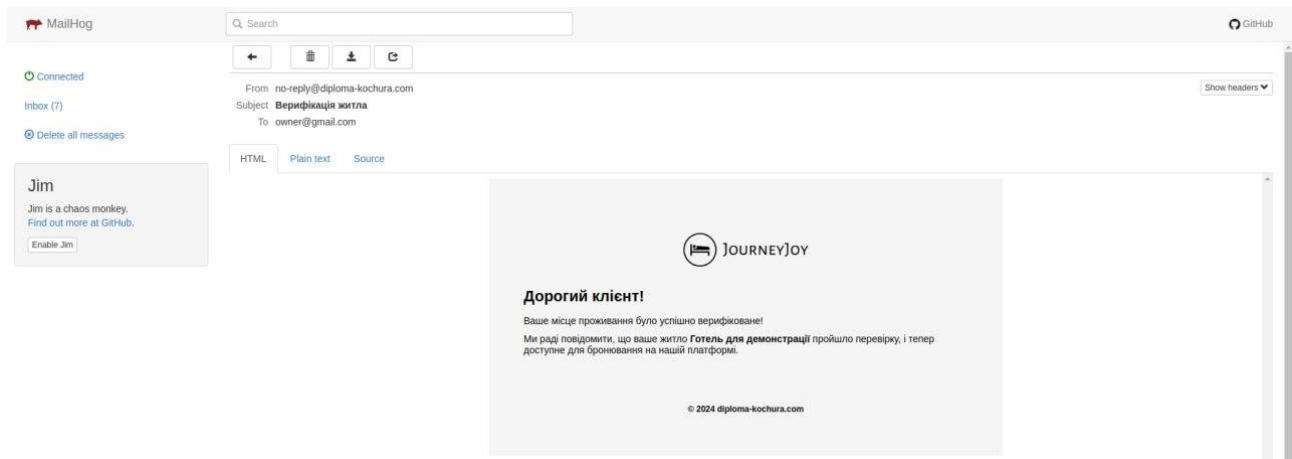


Рисунок 3.38 – Лист про успішну верифікацію місця проживання

На сторінці типів місць для проживання, Адміністратор має можливість їх відфільтрувати, відсортувати, переглянути кількість закріплених місць за кожним з них. А також відредагувати, видалити або ж створити новий (рис. 3.39 – 3.40).

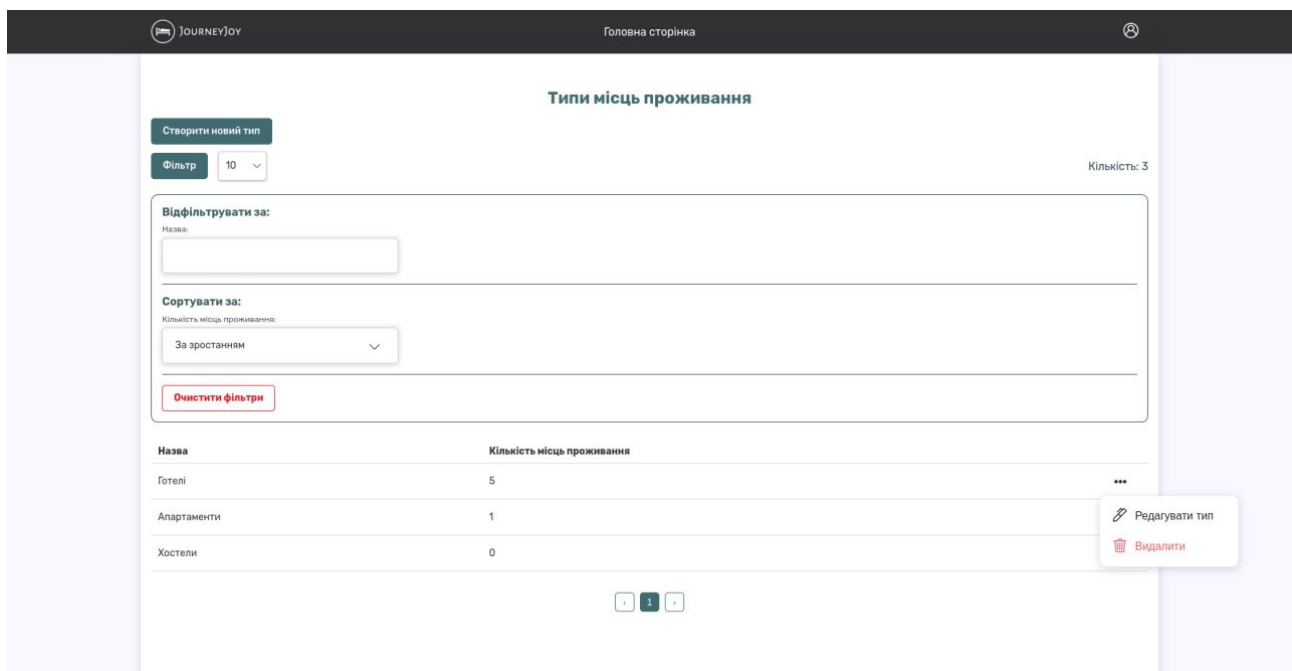


Рисунок 3.39 – Список типів місць проживання

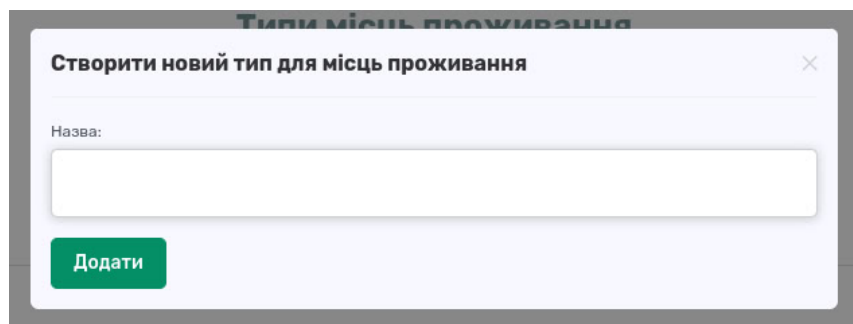


Рисунок 3.40 – Форма створення нового типу

Говорячи про зручності житла та зручності кімнат, хочеться зазначити, що Адміністратор може переглядати їх всі, змінити для них активність, тобто зробити доступними для вибору Власникам (рис. 3.41 – 3.42).

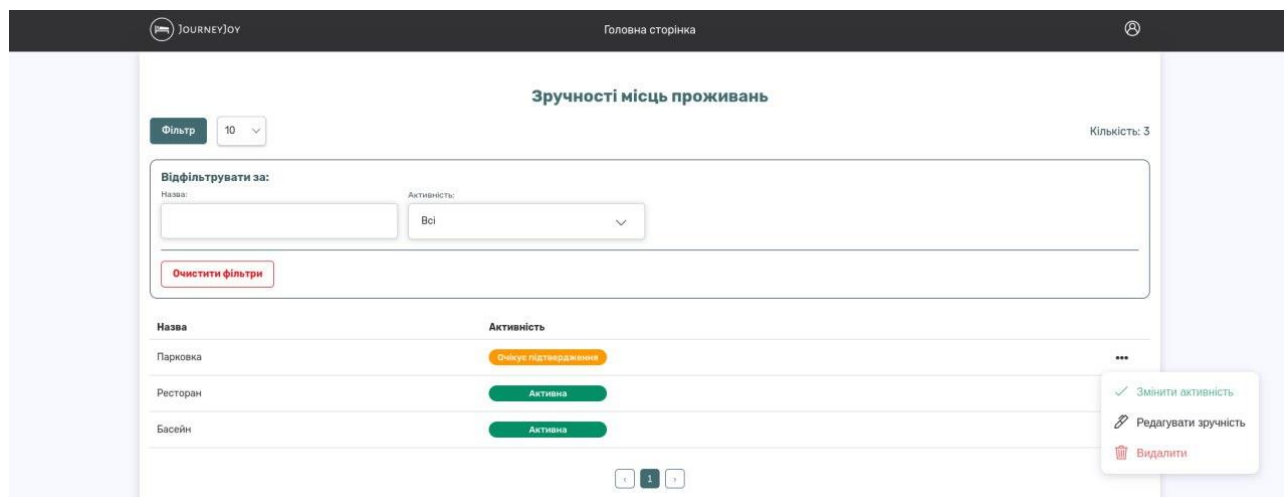


Рисунок 3.41 – Список зручностей місць проживань

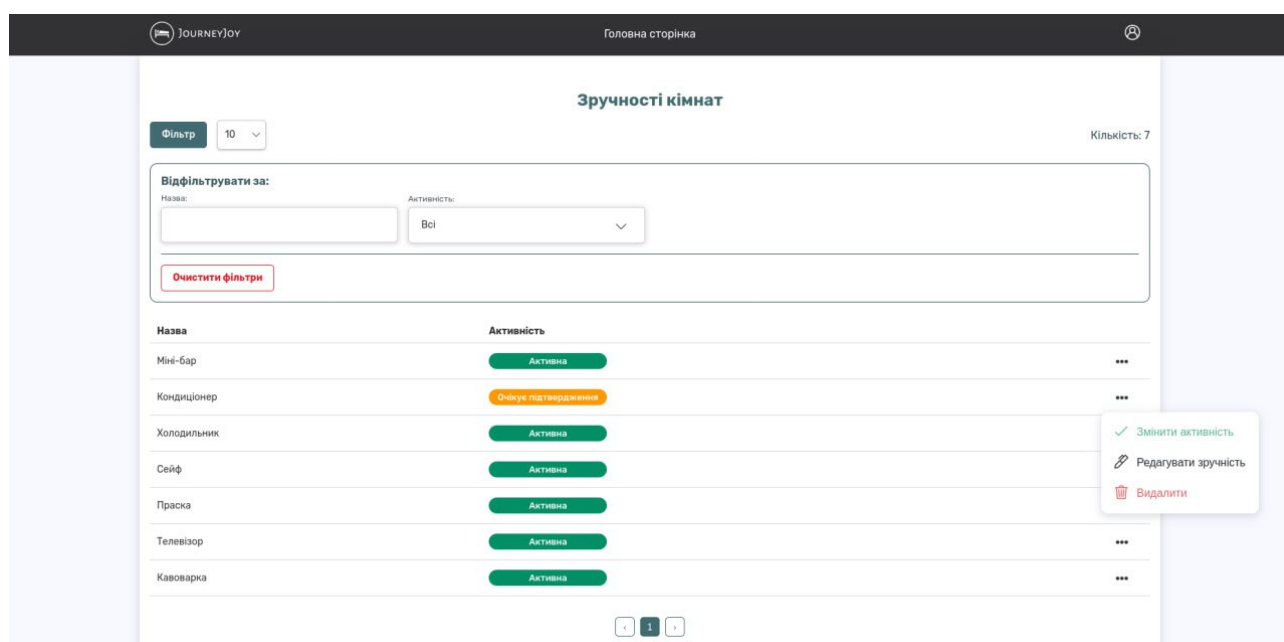


Рисунок 3.42 – Список зручностей кімнат

Також Адміністратор має можливість редагувати населені пункти, видаляти їх або ж вимикати. Це може бути потрібним по багатьом причинам, наприклад було вирішено перестати працювати з котримсь із них, відповідно місця проживання закріплені за ним, не будуть виводитись Клієнтам (рис. 3.43).

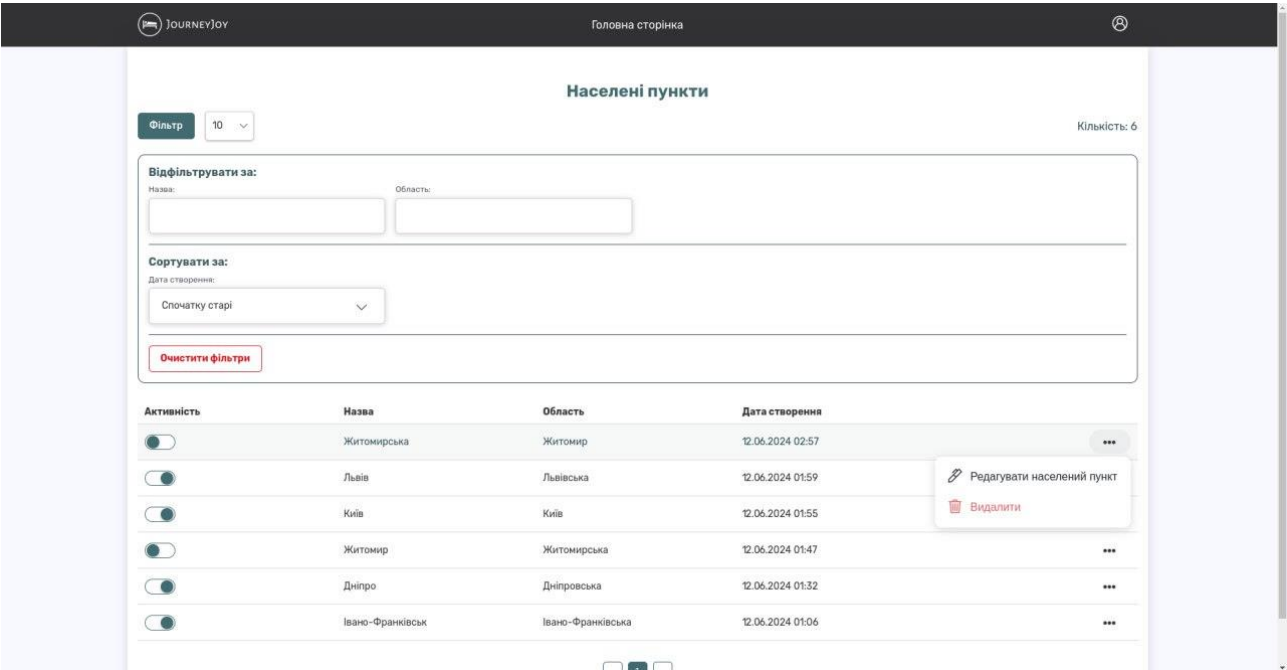


Рисунок 3.43 – Список населених пунктів

Адміністратор також проглядає всі транзакції, що були створені під час бронювань (рис. 3.44).

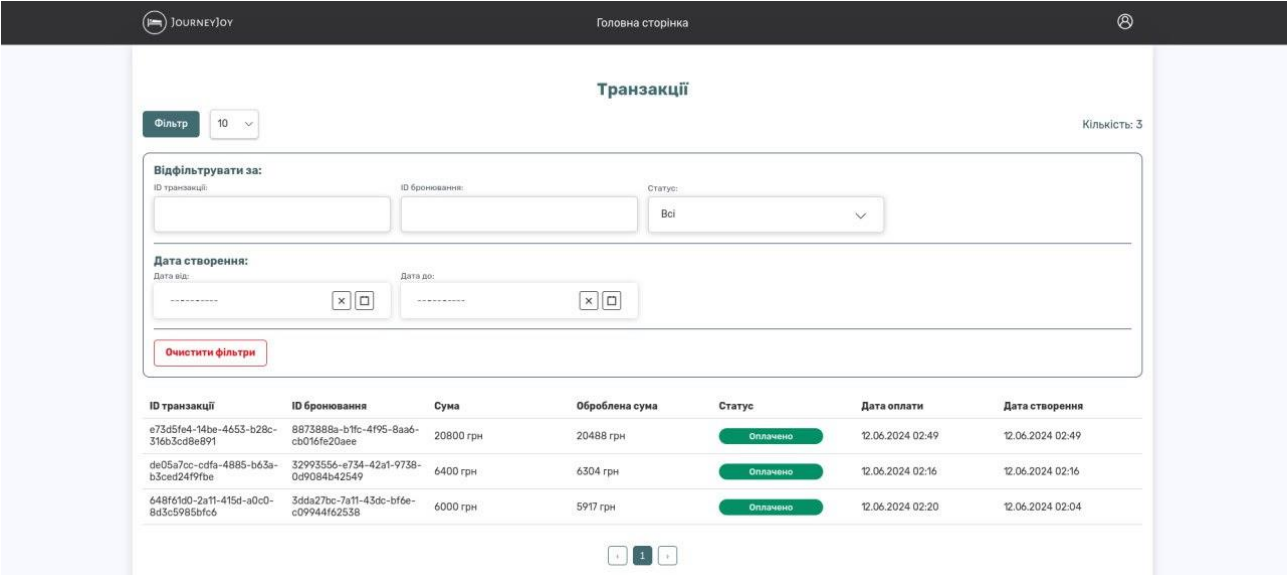


Рисунок 3.44 – Список транзакцій

Що є важливим, у нього також є можливість переглядати логи системи, з можливістю їх фільтрування (рис. 3.45).

LOGO

Головна сторінка

LOGO

Лог

Фільтр

10

Кількість: 25

Відфільтрувати за:

Користувач:

Повідомлення:

Роль:

Рівень:

Дата створення:

Дата від:

Дата до:

Очистити фільтри

Користувач	Рівень	Повідомлення	Дата та час
owner@gmail.com (Власник)	Інформація	Було додано нову кімнату до місця проживання: Готель для демонстрації	12.06.2024, 03:02:19
owner@gmail.com (Власник)	Інформація	Було додано нове місце проживання: Готель для демонстрації	12.06.2024, 02:57:22
ipz203_kviv@student.zbu.edu.ua (Клієнт)	Інформація	Була створена транзакція: e73d5fe4-14be-4653-b28c-316a3cd8e691	12.06.2024, 02:49:03
system (Система)	Інформація	Зареєструвався новий користувач: mukevosogu@mailinator.com	12.06.2024, 02:33:48
ipz203_kviv@student.zbu.edu.ua (Клієнт)	Інформація	Була створена транзакція: de05a7cc-cdfa-4885-b63a-b3ced24f9bbe	12.06.2024, 02:16:02
ipz203_kviv@student.zbu.edu.ua (Клієнт)	Інформація	Була створена транзакція: 648f61d0-2a11-415d-a0c0-8d3c5985bfc6	12.06.2024, 02:04:15
secondowner@gmail.com (Власник)	Інформація	Було додано нову кімнату до місця проживання: Premier Hotel Onister	12.06.2024, 02:01:40

Рисунок 3.45 – Список логів

Варто зазначити, що додаток є повністю адаптивним, кожна сторінка з легкістю може бути переглянута на мобільних пристроях. Це забезпечує користувачам зручний доступ до всіх функцій і сервісів додатку, незалежно від типу або розміру їхнього екрану, що значно покращує загальний користувацький досвід (рис. 3.46 – 3.48).

LOGO

LOGO

LOGO

Знайдіть ідеальне місце для ваших потреб

Фільтр

10

Кількість: 5

★★★★★

Готель для демонстрації

Чуднівська, 103, Житомирська

Опис готелю для демонстрації. Опис готелю для демонстрації. Опис готелю для демонстрації. Опис готелю для демонстрації.

Переглянути

LOGO

LOGO

LOGO

Знайдіть ідеальне місце для ваших потреб

Фільтр

10

Кількість: 5

Відфільтрувати за:

Назва:

Рейтинг:

Населений пункт:

Всі

Вулиця:

Очистити фільтри

Рисунок 3.46 – Адаптивність додатку

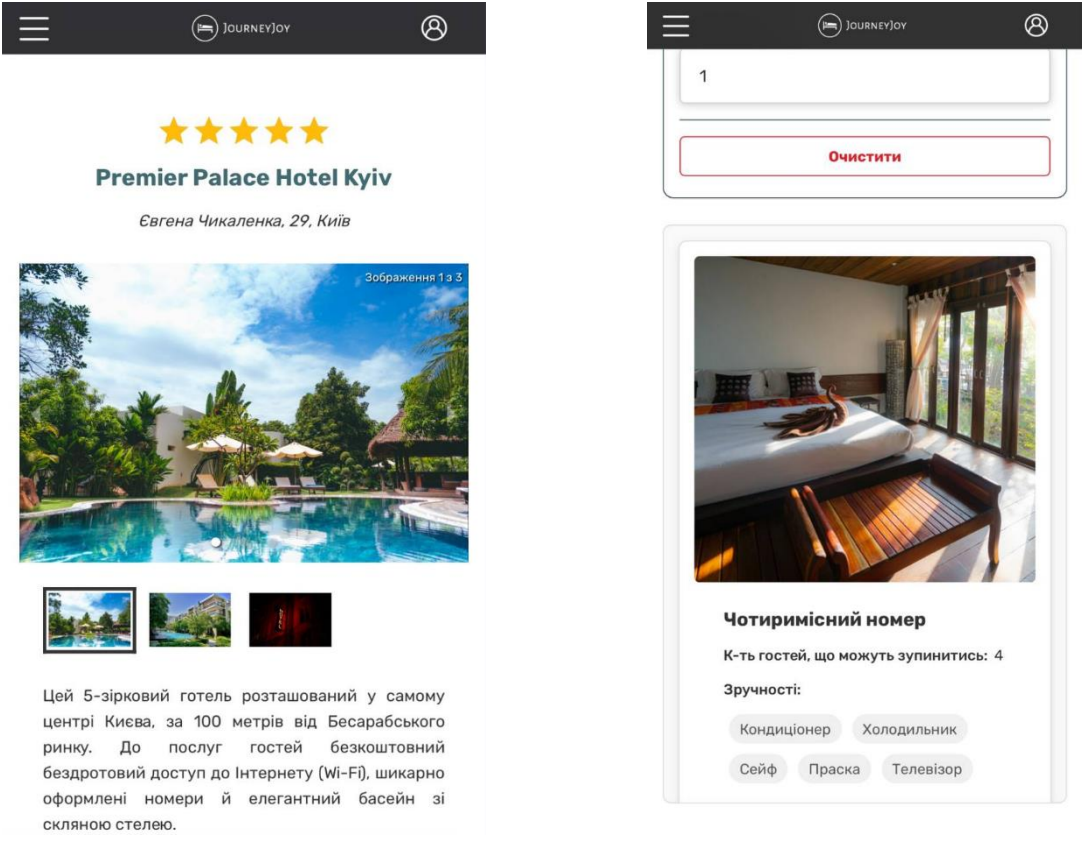


Рисунок 3.47 – Адаптивність додатку

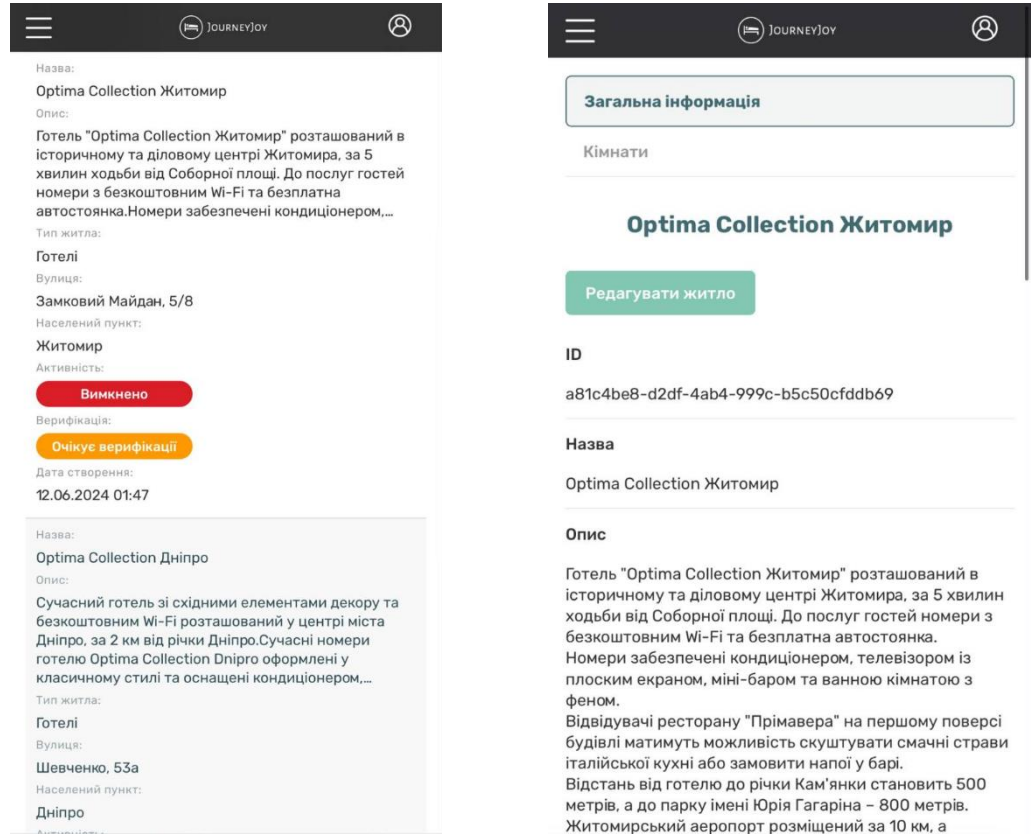


Рисунок 3.48 – Адаптивність додатку

3.3. Тестування роботи програмного комплексу

Протягом усієї розробки програмного продукту проводилось ретельне тестування тих чи інших модулів, для забезпечення надійності та стабільності додатку. Особливу увагу було приділено обробці різноманітних помилок та запобіганню небажаних або повторних дій користувачів, що сприяло підвищенню загальної безпеки та правильності роботи системи.

Обробки помилок були реалізовані, як на серверній, так і на клієнтській частинах, для забезпечення комплексного підходу до виявлення та вирішення проблем.

Прикладом обробки таких помилок є наступні ситуації: якщо при виборі дат заїзду та виїзду, користувач спробує вибрати дату заїзду більшою, ніж дату виїзду; якщо при відправці коментаря буде введено невалідний ідентифікатор; якщо при логіні буде введено некоректний Google Auth код, тощо (рис. 3.49).

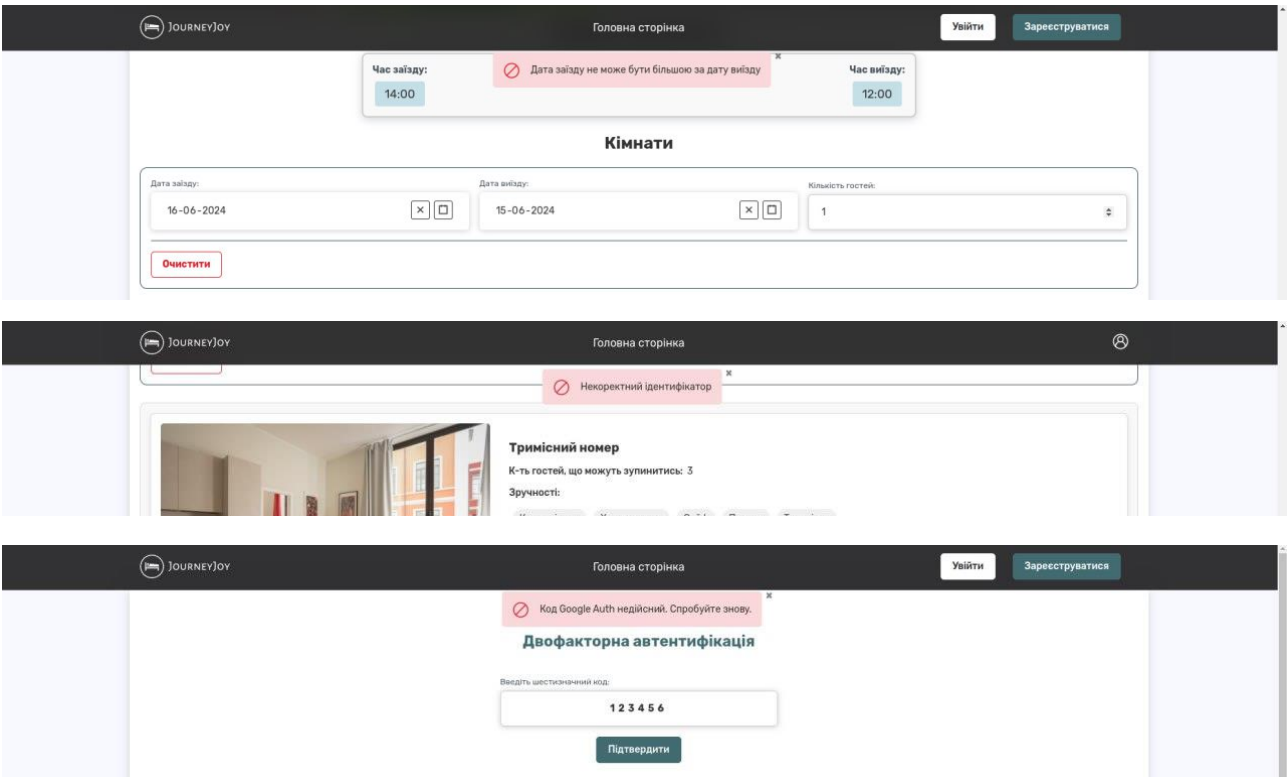


Рис. 3.49 – Приклади обробки різних помилок

Також для того, щоб користувачі не заклікували кнопки та фільтри, було додано лоадери для них. Таким чином, користувачі бачать, що виконується процес обробки їхніх дій. Це значно покращує зручність користування системою та запобігає виникненню непередбачених ситуацій (рис. 3.50).

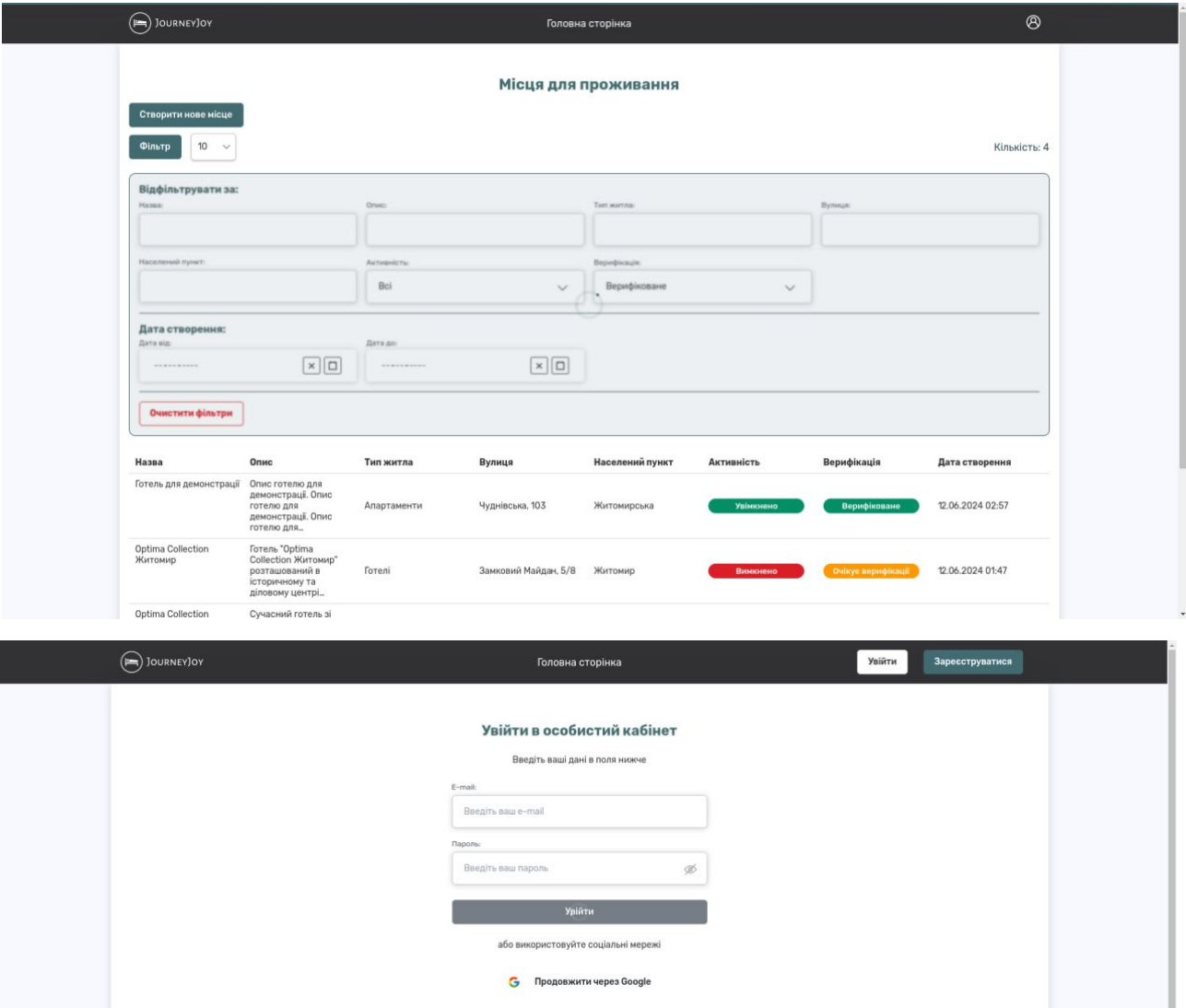


Рис. 3.50 – Приклади лоадерів на фільтрах та кнопках

Задля запобігання введення користувачами невірних значень, або ж взагалі пустих, було додано валідації на такі форми та їх поля. Це забезпечило впевненість у тому, що будуть введені коректні дані (рис. 3.51).

Додати місце

Назва:

Значення не може бути пустим

Опис:

Normal : B I U B " A X E E G B I

Значення не може бути пустим

Тип місця проживання:

Значення не може бути пустим

Вкажіть кількість зрок вадого житла (при бажанні):

★ ★ ★ ★ ★

Зручності (при потребі створіть нову, вибравши її при виборі):

Значення не може бути пустим

Час заїзду

Значення не може бути пустим

Час виїзду

Значення не може бути пустим

Область (назва):

Створити новий тип для місць проживання

Назва:

Test Type #12345

Поле назви має містити виключно українські літери

Додати

Рисунок 3.51 – Приклади валідації полів різних форм

У випадках коли дія недоступна для користувача з якихось причин, блокуються відповідні кнопки. Яскравий приклад – житло в статусі «Очікує верифікації», змінювати його активність або ж редагувати дані в цей час неможливо, відповідно ці кнопки є заблокованими (рис. 3.52).

Optima Collection Житомир

Редагувати житло

ID: a81c4be8-d2df-4ab4-999c-b5c50cfd6b69

Видалити житло

Змінити активність

Рисунок 3.52 – Приклади блокування певних дій для користувача

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижомотря О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		75

Висновки до третього розділу

Підводячи підсумки по третьому розділу, можна зазначити, що в межах нього було детально розглянуто різні важливі аспекти.

Насамперед було чітко визначено вимоги для роботи з системою. Це необхідно для того, щоб користувачі та адміністратори мали чітке розуміння, які ресурси та умови необхідні для коректного функціонування додатку. Після чого покроково розписано яким чином його розгорнути – це мінімізує ризик помилок під час встановлення. Також було побудовано діаграму розгортання.

Потім детально описано структуру інтерфейсу системи управління бронюванням готелів та житла. Надано відповідні зображення з поясненнями до кожного з них. Було отримано зручний та інтуїтивно-зрозумілий інтерфейс, який також є адаптивним і забезпечує коректну роботу на різних пристроях та розмірах екранів.

У кінці було розглянуто яким чином відбувалось тестування та обробка помилок. Продемонстровано результати цієї обробки у різних її проявах: алерти, лоадери, валідація, блокування дій, тощо.

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмотря О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		76

ВИСНОВКИ

На сьогоднішній день, коли інформаційні технології відіграють ключову роль у всіх сферах життя, автоматизація процесів бронювання готелів та житла набуває особливої актуальності.

У ході написання кваліфікаційної роботи було, перш за все, проаналізовано напрямки використання інформаційних технологій для реалізації бажаної системи. Описано мету її створення та визначено перелік задач, що слід було виконати при проектуванні та розробці. Детально розглянуто вже існуючі продукти на ринку, зібрано перелік їх переваг та недоліків, що враховувались при проектуванні нашої системи. Сформовано бачення кінцевого продукту, з урахуванням потреб сучасного клієнта.

Одним із найважливіших початкових етапів був вибір підходящої архітектури, що дозволить продукту бути масштабованим у майбутньому та зручним під час розробки. Розглянуто декілька архітектурних рішень та вибрано найбільш підходяще, що задовільнило наші потреби. Також обрано підходящий стек технологій, котрий використовувався для реалізації кінцевого додатку.

Під час написання другого розділу, було визначено варіанти використання і разом з тим ряд вимог. Після чого відбулася розробка баз даних, що відповідали визначеним варіантам і з урахуванням вимог поставлених перед системою. Далі, з використанням відповідних діаграм, було описано основні алгоритми роботи системи, наведено пояснення до реалізації ключових функціональних рішень.

Завершальний розділ класифікаційної роботи містить в собі вимоги для користування системою та покрокову інструкції з її розгортанням. В ньому наведено, разом із скриншотами, структуру інтерфейсу додатку з поясненнями. Це дозволило не лише показати зовнішній вигляд продукту, але й продемонструвати його можливості та переваги для кінцевих користувачів. Окрім цього, роз'яснено яким чином проводилось тестування та як

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмотря О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		77

оброблювались ті, чи інші можливі проблеми, котрі могли б виникати при роботі з системою.

Врешті-решт, після обробки великої кількості інформації та багатоденної праці мета кваліфікаційної роботи була досягнута, завдання – виконане. Ми отримали довершений продукт, що відповідає початково поставленим вимогам й очікуванням користувачів. Впровадження такої системи управління бронюванням готелів та житла для подорожей Україною, дозволить забезпечити зручний процес бронювання, підвищить рівень обслуговування клієнтів і спростить процес управління місцями для проживання.

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмотря О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		78

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Build PHP Symfony Apps with MongoDB Atlas [Електронний ресурс] - Режим доступу до ресурсу: <https://mongodb-developer.github.io/symfony-mongodb-rental-workshop/>.
2. Create a payment button using LiqPay [Електронний ресурс] - Режим доступу до ресурсу: <https://medium.com/@testjokerqwerty/create-a-payment-button-using-liqpay-aadcfee99c06>.
3. Cron Jobs: повна інструкція для початківців з прикладами [Електронний ресурс] - Режим доступу до ресурсу: <https://highload.today/uk/cron-jobs/>.
4. Deployment Diagram Tutorial [Електронний ресурс] - Режим доступу до ресурсу: <https://www.lucidchart.com/pages/uml-deployment-diagram>.
5. Doctrine: PHP Open Source Project [Електронний ресурс] - Режим доступу до ресурсу: <https://www.doctrine-project.org/>.
6. How to Get Started with Docker Compose and Symfony [Електронний ресурс] - Режим доступу до ресурсу: <https://www.twilio.com/en-us/blog/get-started-docker-symfony>.
7. How to Send Emails in Symfony using SMTP or API Methods: A Step-by-Step Tutorial [Електронний ресурс] - Режим доступу до ресурсу: <https://mailtrap.io/blog/send-emails-in-symfony/>.
8. Logging events to database in Symfony [Електронний ресурс] - Режим доступу до ресурсу: <https://nehalist.io/logging-events-to-database-in-symfony/>.
9. MailHog Tutorial [Електронний ресурс] - Режим доступу до ресурсу: <https://mailtrap.io/blog/mailhog-explained/>.
10. ManyToMany... with Extra Fields on the Join Table? [Електронний ресурс] - Режим доступу до ресурсу: <https://symfonycasts.com/screencast/doctrine-relations/complex-many-to-many>.
11. OAuth 2.0 аутентифікація через Google: як реалізувати вхід через Google на сайті [Електронний ресурс] - Режим доступу до ресурсу:

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижевотря О. В.				79
Змн.	Арк.	№ докум.	Підпис	Дата		

<https://blog.ithillel.ua/articles/oauth-2-0-autentifikaciya-cerez-google-yak-realizuvati-vxid-cerez-google-na-saiti>.

12. React components for Leaflet maps [Електронний ресурс] - Режим доступу до ресурсу: <https://react-leaflet.js.org/>.
13. Архітектура програмного забезпечення: все що треба знати [Електронний ресурс] – Режим доступу до ресурсу: <https://wezom.com.ua/ua/blog/arhitektura-programmnogo-obespecheniya>.
14. Вступ до REST API — RESTful вебсервіси [Електронний ресурс] - Режим доступу до ресурсу: <https://robotdreams.cc/uk/blog/466-vstup-do-rest-api-restful-vebservisi>.
15. Документація Api Platform [Електронний ресурс] - Режим доступу до ресурсу: <https://api-platform.com/docs/distribution/>.
16. Документація Docker [Електронний ресурс] - Режим доступу до ресурсу: <https://docs.docker.com/>.
17. Документація LiqPay [Електронний ресурс] - Режим доступу до ресурсу: <https://www.liqpay.ua/doc>.
18. Документація MySQL [Електронний ресурс] - Режим доступу до ресурсу: <https://dev.mysql.com/doc/>.
19. Документація Nominatim [Електронний ресурс] - Режим доступу до ресурсу: <https://nominatim.org/release-docs/latest/>.
20. Документація PHP [Електронний ресурс] - Режим доступу до ресурсу: <https://www.php.net/docs.php>.
21. Документація React [Електронний ресурс] - Режим доступу до ресурсу: <https://react.dev/>.
22. Документація Symfony [Електронний ресурс] - Режим доступу до ресурсу: <https://symfony.com/doc/current/index.html>.
23. Курс з Symfony [Електронний ресурс] - Режим доступу до ресурсу: <https://spacelab.ua/course/symfony/>.

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижевотря О. В.				80
Змн.	Арк.	№ докум.	Підпис	Дата		

24. Огляд Serverless архітектури та її переваги [Електронний ресурс] - Режим доступу до ресурсу: <https://elartu.tntu.edu.ua/bitstream/lib/41424/2/110-111.pdf>.
25. Побудова динамічних вебзастосунків за допомогою MVC [Електронний ресурс] - Режим доступу до ресурсу: <https://blog.ithillel.ua/articles/building-dynamic-web-applications-using-mvc>.
26. Розуміння Клієнт-Серверної Архітектури на прикладах [Електронний ресурс] - Режим доступу до ресурсу: <https://dou.ua/forums/topic/44636/>.
27. Сучасні клієнт-серверні технології та їх застосування при вивченні систем управління базами даних [Електронний ресурс] - Режим доступу до ресурсу: https://fi.npu.edu.ua/files/Zbirnik_KOSN/12/29.pdf.
28. Шаблон проектування MVC [Електронний ресурс] - Режим доступу до ресурсу: <http://yoip.com.ua/shablon-proektuvannya-mvc/>.
29. Як будувати UML-діаграми [Електронний ресурс] - Режим доступу до ресурсу: <https://dou.ua/forums/topic/40575/>.
30. Як архітектура проекту впливає роботу РМ-а та навіщо вона йому потрібна [Електронний ресурс] - Режим доступу до ресурсу: <https://iampm.club/ua/blog/yak-arhitektura-projektu-vplivaje-robotu-pm-a-ta-navishho-vona-jomu-potribna/>.

ДОДАТКИ

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижевотря О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		82

```

/**
 * @param Request $request
 * @return JsonResponse
 * @throws Exception|GuzzleException|ExceptionInterface
 */
#[Route(path: '/property/create', name: 'property_create', methods:
[Request::METHOD_POST])]
public function createProperty(Request $request): JsonResponse
{
    $content = $request->request->all();

    $files = $request->files->all();

    CheckRequestDataService::check($content, self::CREATE_PROPERTY_REQUIRED_FIELDS);

    /** @var User $user */
    $user = $this->security->getUser();

    if (!$user) {
        throw new RuntimeException('Не вдається знайти користувача',
Response::HTTP_BAD_REQUEST);
    }

    $settlement = $this->propertyService->isAddressExists($content['settlement'],
$content['state'], $content['street']);

    $type = $this->entityManager->getRepository(Type::class)->findOneBy(['name' =>
$content['type']]);

    if (!$type) {
        throw new RuntimeException('Такого типу місць проживання не існує');
    }

    $property = $this->propertyService->createPropertyObject($content, $type,
$settlement, $user);

    $errors = $this->validator->validate($property);

    $errorsString = [];

    if (count($errors) > 0) {
        foreach ($errors as $error) {
            $errorsString += [$error->getPropertyPath() => $error->getMessage()];
        }
    }

    if (count($errorsString) > 0) {
        throw new RuntimeException(json_encode($errorsString, JSON_UNESCAPED_UNICODE),
Response::HTTP_UNPROCESSABLE_ENTITY);
    }

    try {
        $this->propertyService->saveImages($property, $files);
    } catch (Throwable $throwable) {
        throw new RuntimeException('Під час збереження файлу сталася помилка',
Response::HTTP_BAD_REQUEST);
    }

    $this->entityManager->persist($property);
    $this->entityManager->flush();

```

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмоторя О. В.				83
Змн.	Арк.	№ докум.	Підпис	Дата		

```

$this->logger->info('Було додане нове місце проживання: ' . $property->getName());

return new JsonResponse(
    $this->normalizer->normalize([
        'property' => $property,
        'success' => 'Місце проживання успішно створено, для того, щоб верифікація
відбулась - необхідно додати кімнати до нього'
    ], null, ['groups' => 'get:item:property']), Response::HTTP_CREATED);
}

```

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмотря О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		84

```

<?php

namespace App\Controller\Transaction;

use App\Entity\Booking\Booking;
use App\Entity\Room\Room;
use App\Entity\Transaction\Transaction;
use App\Entity\User;
use App\Services\Booking\BookingService;
use App\Services\Request\CheckRequestDataService;
use App\Services\Transaction\TransactionService;
use Doctrine\ORM\EntityManagerInterface;
use Exception;
use LiqPay;
use Psr\Log\LoggerInterface;
use RuntimeException;
use Symfony\Bundle\SecurityBundle\Security;
use Symfony\Component\HttpFoundation\JsonResponse;
use Symfony\Component\HttpFoundation\Request;
use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\Routing\Attribute\Route;
use Symfony\Component\Validator\Validator\ValidatorInterface;

#[Route(path: '/api')]
class TransactionController
{
    private const WEBHOOK_URL = 'https://webhook.site/06dca134-2539-48ea-851c-
bf157474cd13';

    private const CREATE_TRANSACTION_AND_BOOKING_REQUIRED_FIELDS = [
        'room',
        'clientFirstname',
        'clientLastname',
        'clientPhone',
        'arrivalDate',
        'departureDate',
        'amount'
    ];

    public function __construct(
        private readonly EntityManagerInterface $entityManager,
        private readonly TransactionService $transactionService,
        private readonly BookingService $bookingService,
        private readonly ValidatorInterface $validator,
        private readonly Security $security,
        private readonly LoggerInterface $logger
    ) {}

    /**
     * @param Request $request
     * @return JsonResponse
     * @throws Exception
     */
    #[Route(path: '/transaction/create', name: 'transaction_create', methods:
[Request::METHOD_POST])]
    public function createTransaction(Request $request): JsonResponse
    {
        $content = json_decode($request->getContent(), true);

        CheckRequestDataService::check($content,

```

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмоторя О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		85

```

self::CREATE_TRANSACTION_AND_BOOKING_REQUIRED_FIELDS);

    /** @var User $user */
    $user = $this->security->getUser();

    if (!$user) {
        throw new RuntimeException('Не вдається знайти користувача',
Response::HTTP_BAD_REQUEST);
    }

    $room = $this->entityManager->getRepository(Room::class)->findOneBy(['id' =>
$content['room']]);

    if (!$room) {
        throw new RuntimeException('Такої кімнати не існує');
    }

    $booking = $this->bookingService->creatBookingObject($content, $room, $user);

    $errors = $this->validator->validate($booking);

    $errorsString = [];

    if (count($errors) > 0) {
        foreach ($errors as $error) {
            $errorsString += [$error->getPropertyPath() => $error->getMessage()];
        }
    }

    if (count($errorsString) > 0) {
        throw new RuntimeException(json_encode($errorsString,
JSON_UNESCAPED_UNICODE), Response::HTTP_UNPROCESSABLE_ENTITY);
    }

    $transaction = $this->transactionService->createTransactionObject($content,
$booking);

    $this->entityManager->flush();

    $liqPay = new LiqPay($_ENV['LIQPAY_PUBLIC_KEY'], $_ENV['LIQPAY_PRIVATE_KEY']);

    $params = [
        'action'      => 'pay',
        'amount'      => $transaction->getAmount(),
        'currency'    => 'UAH',
        'description' => 'Оплата бронювання',
        'order_id'    => $transaction->getId(),
        'version'     => '3',
        'public_key'  => $_ENV['LIQPAY_PUBLIC_KEY'],
        'result_url'  => $_ENV['HOST'] . '/',
        'server_url'  => self::WEBHOOK_URL
    ];

    $signature = $liqPay->cnb_signature($params);

    $this->logger->info('Була створена транзакція: ' . $transaction->getId());

    return new JsonResponse([
        'signature' => $signature,
        'data'      => base64_encode(json_encode($params))
    ], Response::HTTP_OK);
}

```

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмоторя О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		86

```

/**
 * @param Request $request
 * @return JsonResponse
 * @throws Exception
 */
#[Route(path: '/transaction/booking', name: 'transaction_booking', methods:
[Request::METHOD_POST])]
public function bookingTransaction(Request $request): JsonResponse
{
    $content = json_decode($request->getContent(), true);

    if (!isset($content['bookingId'])) {
        throw new RuntimeException('Бронювання відсутнє',
Response::HTTP_BAD_REQUEST);
    }

    $booking = $this->entityManager->getRepository(Booking::class)->findOneBy(['id'
=> $content['bookingId']]);

    /** @var User $user */
    $user = $this->security->getUser();

    if (!$user) {
        throw new RuntimeException('Не вдається знайти користувача',
Response::HTTP_BAD_REQUEST);
    }

    $liqPay = new LiqPay($_ENV['LIQPAY_PUBLIC_KEY'], $_ENV['LIQPAY_PRIVATE_KEY']);

    $params = [
        'action'      => 'pay',
        'amount'      => $booking->getTransaction()->getAmount(),
        'currency'    => 'UAH',
        'description' => 'Оплата бронювання',
        'order_id'    => $booking->getTransaction()->getId(),
        'version'     => '3',
        'public_key'  => $_ENV['LIQPAY_PUBLIC_KEY'],
        'result_url'  => $_ENV['HOST'] . '/',
        'server_url'  => self::WEBHOOK_URL
    ];

    $signature = $liqPay->cnb_signature($params);

    return new JsonResponse([
        'signature' => $signature,
        'data'      => base64_encode(json_encode($params))
    ], Response::HTTP_OK);
}

/**
 * @param Request $request
 * @return Response
 */
#[Route(path: '/payment-callback', name: 'payment_callback', methods:
[Request::METHOD_POST])]
public function paymentCallback(Request $request): Response
{
    parse_str($request->getContent(), $content);

    $signatureFromRequest = $content['signature'];
    $data = json_decode(base64_decode($content['data']));

```

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмоторя О. В.				87
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        $expectedSignature = base64_encode(sha1($_ENV['LIQPAY_PRIVATE_KEY'] .
$content['data'] . $_ENV['LIQPAY_PRIVATE_KEY'], 1));

        if ($signatureFromRequest !== $expectedSignature) {
            return new Response('Некоректна сигнатура', Response::HTTP_OK);
        }

        $transaction = $this->entityManager->getRepository(Transaction::class)-
>findOneBy(['id' => $data->order_id]);

        if ($data->status === 'success') {
            $transaction->setStatus(Transaction::PAID_STATUS_PAYMENT);
            $transaction->setProcessedAmount($data->amount - $data-
>receiver_commission);
            $transaction->setPaidAt(time());

            $booking = $transaction->getBooking();
            $booking->setStatus(Booking::PAID_STATUS);

            $this->bookingService->sendSuccessfulBookingMessage($booking);
        }

        $this->entityManager->flush();

        return new Response('Payment callback успішно відпрацював', Response::HTTP_OK);
    }
}

```

		Кочура В. В.			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		Чижмотря О. В.				
Змн.	Арк.	№ докум.	Підпис	Дата		88