

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:

«Онлайн-платформа: курортні готелі»

Виконав студент групи ІПЗ-20-2
МЕЛЬНИЧЕНКО Надія Миколаївна

Керівник роботи:
ФАНТ Микола Олександрович

Рецензент:
МОРОЗОВ Андрій Васильович

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: **МЕЛЬНИЧЕНКО Надія Миколаївна**

Керівник роботи: **ФАНТ Микола Олександрович**

Тема роботи: **«Онлайн-платформа: курортні готелі»**,
затверджена Наказом закладу вищої освіти від «23» лютого 2024 р., №74/с
Вихідні дані для роботи: **процес бронювання номерів курортних готелів, який використовує сучасні веб-технології, зокрема React для фронтенду та Symfony для бекенду, з інтеграцією API платформи для ефективного управління даними та користувачами.**

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

Розділ	Консультант	Завдання видав	Завдання прийняв
1	Микола ФАНТ	27.02.2024р.	27.02.2024р.
2	Микола ФАНТ	27.02.2024р.	27.02.2024р.
3	Микола ФАНТ	27.02.2024р.	27.02.2024р.

РЕФЕРАТ

Випускна робота бакалавра складається з онлайн-платформи для бронювання номерів курортних готелів та пояснювальної записки. Пояснювальна записка до випускної роботи містить 72 сторінки, 55 ілюстрацій, 2 таблиці, 3 додатка та 15 використаних джерел.

Метою роботи є розробка веб-платформи для бронювання курортних готелів засобами Symfony та React, яка дозволяє користувачам шукати готелі за різними критеріями, бронювати номери, залишати відгуки та отримувати бонуси за бронювання, а також надає можливість власникам готелів додавати свої об'єкти на платформу.

У роботі визначено основні завдання для розробки системи, проаналізовано аналогічні платформи та обрано засоби реалізації після визначення основного функціоналу та вимог до системи. Проведено розробку структури бази даних, обрано технології для розробки платформи та розроблено концепцію програмного продукту. Програмний продукт був повністю створений та протестований.

КЛЮЧОВІ СЛОВА: ВЕБ-ПЛАТФОРМА, КУРОРТНІ ГОТЕЛІ, БРОНЮВАННЯ, ВІДГУКИ, БОНУСИ, SYMFONY, REACT, API, НОМЕРИ.

					ІПЗ.КР.Б-121-24-ПЗ				
Зм.	Арк.	№ докум.	Підпис	Дата					
Розроб.		Мельниченко Н.М.			Пояснювальна записка Онлайн-платформа: курортні готелі		Літ.	Арк.	Аркушів
Керівник		Фант М.О.						3	72
Рецензент		Морозов А.В.					Житомирська політехніка, група ІПЗ-20-2		
Зав. каф.		Вакалюк Т.А.							

ABSTRACT

The bachelor's thesis consists of an online platform for booking resort hotel rooms and an explanatory note. The explanatory note for the bachelor's thesis contains 72 pages, 55 illustrations, 2 tables, 3 appendices, and 15 references.

The aim of the work is to develop a web platform for booking resort hotels using Symfony and React, which allows users to search for hotels by various criteria, book rooms, leave reviews, and receive booking bonuses, as well as enable hotel owners to add their properties to the platform.

The work defines the main tasks for system development, analyzes similar platforms, and selects the implementation tools after determining the main functionality and system requirements. The database structure has been developed, the technologies for platform development have been chosen, and the concept of the software product has been created. The software product was fully developed and tested.

KEYWORDS: WEB PLATFORM, RESORT HOTELS, BOOKING, REVIEWS, BONUSES, SYMFONY, REACT, API, ROOMS.

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		4

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ	6
ВСТУП.....	7
РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ РЕАЛІЗАЦІЇ ОНЛАЙН-ПЛАТФОРМИ: КУРОРТНІ ГОТЕЛІ	9
1.1 Постановка задачі.....	9
1.2 Аналіз аналогів програмного продукту	10
1.3 Вибір архітектури онлайн-платформи для курортних готелів.....	18
1.4 Обґрунтування вибору інструментальних засобів	20
Висновки до першого розділу	21
РОЗДІЛ 2. ПРОЕКТУВАННЯ ОНЛАЙН-ПЛАТФОРМИ БРОНЮВАННЯ ГОТЕЛІВ.....	23
2.1 Вимоги до системи.....	23
2.1.1 Визначення варіантів використання та структури системи	23
2.1.2 Функціональні та нефункціональні вимоги	25
2.2 Розробка бази даних.....	31
2.3 Проектування та реалізація алгоритмів роботи платформи	34
2.4 Реалізація функціоналу платформи	38
Висновки до другого розділу	43
РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З ОНЛАЙН- ПЛАТФОРМОЮ ДЛЯ БРОНЮВАННЯ ГОТЕЛІВ	45
3.1 Користувацький інтерфейс	45
3.2 Інтерфейс для власника готелю та адміністратора платформи.....	53
3.3 REST API.....	57
3.4 Особливості бронювання номеру	59
Висновки до третього розділу	61
ВИСНОВКИ	64
Література.....	65
ДОДАТКИ.....	67

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

API (Application Programming Interface) – Інтерфейс програмування додатків, який дозволяє взаємодію між різними програмними компонентами.

HTTP (HyperText Transfer Protocol) – Протокол передачі гіпертексту, який використовується для обміну даними у всесвітній мережі (Інтернет).

JSON (JavaScript Object Notation) – Формат обміну даними, який використовується для зберігання та передачі структурованої інформації

REST (Representational State Transfer) – Архітектурний стиль веб-сервісів, що використовує HTTP запити для доступу до ресурсів.

CRUD (Create, Read, Update, Delete) – Набір основних операцій для роботи з базою даних.

JWT (JSON Web Token) – Стандарт відкритого JSON веб-токену, що використовується для безпечної передачі інформації між сторонами як об'єкт JSON.

SQL (Structured Query Language) – Мова програмування, призначена для управління даними в реляційних базах даних.

ORM (Object-Relational Mapping) – Технологія програмування, яка використовується для перетворення даних між несумісними системами типів об'єктів у мовах програмування.

DOM (Document Object Model) – Об'єктна модель документа, яка представляє структуру веб-сторінки у вигляді дерева.

MVC (Model-View-Controller) – Архітектурний шаблон, що розділяє додаток на три основні логічні компоненти: модель, подання та контролер.

SPA (Single Page Application) – Односторінковий додаток, де всі дії користувача обробляються на одній веб-сторінці без перезавантаження.

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						6
Змн.	Арк.	№ докум.	Підпис	Дата		

ВСТУП

В сучасному світі висока активність людей та потреба у мобільності роблять важливим завданням створення зручних та ефективних інструментів для планування та бронювання подорожей. Туристичний бізнес та індустрія гостинності є одними із сфер, що особливо активно розвиваються, і зростаюча популярність курортних готелів створює попит на ефективні засоби їх бронювання.

Актуальність даної теми зумовлена швидкими змінами в споживчих підходах та технологічними інноваціями. У сучасному світі користувачі все частіше шукають можливість бронювати готельні номери безпосередньо, обходячи традиційних тур-операторів. Це не лише прискорює та спрощує процес планування подорожей, але й дозволяє ефективніше контролювати витрати та вибирати оптимальні варіанти.

Метою кваліфікаційної роботи є проектування та розробка веб-платформи для бронювання курортних готелів, яка забезпечить користувачам можливість зручно знаходити і бронювати номери, залишати відгуки та отримувати бонуси за бронювання. Власники готелів зможуть додавати свої об'єкти, керувати інформацією про них та аналізувати статистику бронювань.

Встановлена мета обумовлює наступні **завдання**:

- провести аналіз аналогічних систем бронювання та їх функціональних можливостей;
- визначити архітектуру та структуру системи;
- обґрунтувати вибір засобів реалізації системи;
- спроектувати функціонал та алгоритми роботи системи;
- реалізувати програмний комплекс веб-платформи для бронювання курортних готелів.

Об'єкт дослідження є процес бронювання номерів у курортних готелях та взаємодія користувачів з онлайн-системами бронювання. Створення

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		7

платформи спрямоване на покращення цього процесу та забезпечення зручних і ефективних інструментів для користувачів та власників готелів.

Предмет дослідження є використання веб-орієнтованих технологій для створення веб-платформи, функціональні можливості платформи для бронювання курортних готелів. Платформа включатиме такі функції, як пошук і фільтрація готелів за різними критеріями, онлайн-бронювання номерів, система відгуків та рейтингу, накопичувальні бонуси для користувачів, а також інструменти для управління готелями та бронюваннями для власників.

Платформа розроблена з використанням сучасних технологій фреймворку Symfony та React, з використанням API Platform та базою даних MySQL.

У процесі розробки платформи використано такі методи дослідження: аналіз вимог до функціоналу системи, проектування архітектури та інтерфейсу платформи, розробка та тестування програмного забезпечення. Крім того, застосовано методи аналізу та порівняння аналогічних систем для вибору оптимальних рішень.

Практичне значення отриманих результатів полягає у можливості використання розробленої платформи у реальній діяльності курортних готелів. Платформа надає сучасні інструменти для управління бронюваннями, що покращує взаємодію з клієнтами і підвищує ефективність бізнесу.

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						8
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ РЕАЛІЗАЦІЇ ОНЛАЙН-ПЛАТФОРМИ: КУРОРТНІ ГОТЕЛІ

1.1 Постановка задачі

У сучасному світі платформи для бронювання готелів стають ключовими інструментами взаємодії з клієнтами та забезпеченням їхнього комфортного перебування на курорті.

Метою цього проекту є створення онлайн-платформи, яка надасть можливість бронювати готельні номери безпосередньо, обходячи традиційних тур-операторів. Важливим аспектом розробки є полегшення обміну інформацією між клієнтом та готелем, що не лише прискорює та спрощує процес планування подорожей, але й дозволяє ефективніше контролювати витрати та вибирати оптимальні варіанти.

Для досягнення цієї мети необхідно пройти кілька основних етапів:

1) Планування та аналіз:

- Визначення вимог: розробка докладного опису функціоналу та визначення вимог для платформи
- Обрання технологій: визначення інструментів для роботи, таких як API Platform, Symfony і React, а також обрання структури бази даних.
- Створення схеми бази даних: проектування бази даних та визначення зв'язків між таблицями.

2) Налаштування робочого середовища:

- Інтеграція Symfony та API Platform
- Запуск та налаштування React

3) Розробка Backend:

- Визначення сутностей, створення контролерів та налаштування точок входу API Platform
- Валідація та обробка запитів

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						9
Змн.	Арк.	№ докум.	Підпис	Дата		

- Розробка сервісів для взаємодії з базою даних, включаючи CRUD-операції.

4) Розробка Frontend: розробка компонентів та інтерфейсу користувача для взаємодії з API Platform.

5) Тестування та оптимізація продукту.

По завершенні виконання усіх етапів розробки, буде створена платформа для надання користувачам можливостей бронювання готелів, обходячи тур-операторів та зменшуючи витрати.

1.2 Аналіз аналогів програмного продукту

В умовах постійного технологічного розвитку і конкурентного середовища важливим етапом у розробці онлайн-платформи для курортних готелів є проведення аналізу аналогів. Це дозволяє виявити конкурентні переваги, а також інноваційні можливості для поліпшення розроблюваного програмного продукту.

Один із найвідоміших світових сервісів бронювання готелів є Booking.com [4] (рис. 1.1).

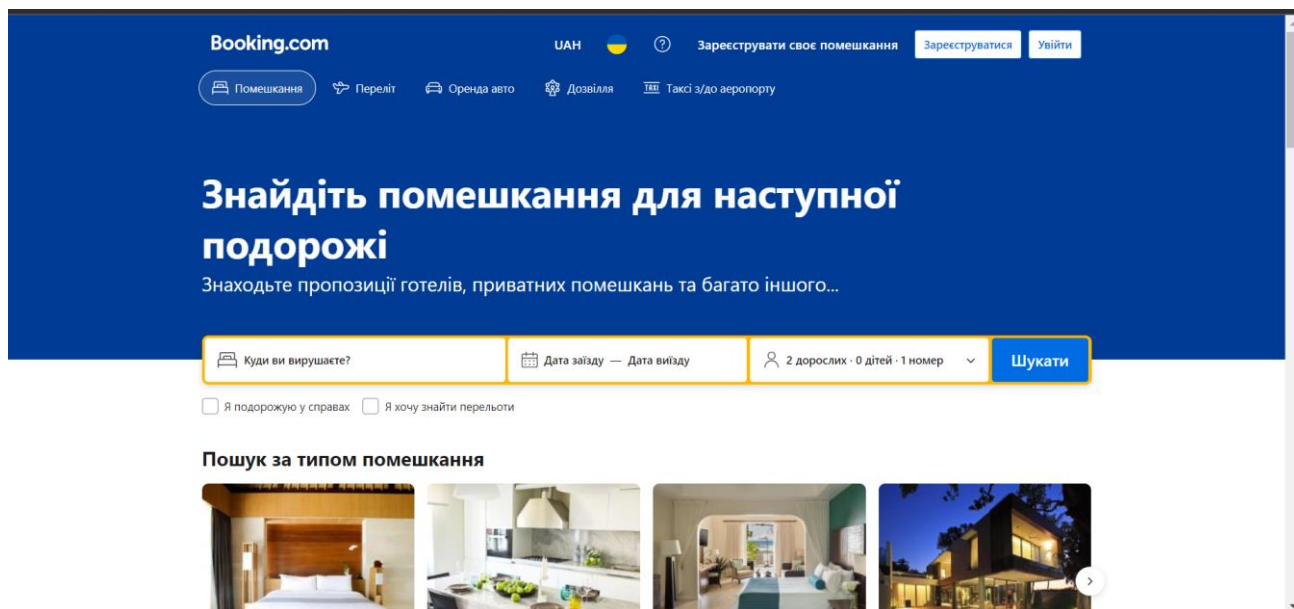


Рис. 1.1 Основна сторінка Booking.com

Дана платформа не має чіткого обмеження в розташуванні, тобто користувач може обрати необхідне житло майже в будь-якій точці світу. А

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						10
Змн.	Арк.	№ докум.	Підпис	Дата		

також обране місце проживання може бути як і орендоване приміщення/квартира, так і готель (рис. 1.2).

Пошук за типом помешкання



Рис. 1.2 Вибір типу помешкання

На сторінці помешкання розташована візуалізація місця проживання, карта розташування та опис. За наявності публікуються коментарі, виставляється оцінка, що вагомо впливає на вибір користувача (рис 1.3).

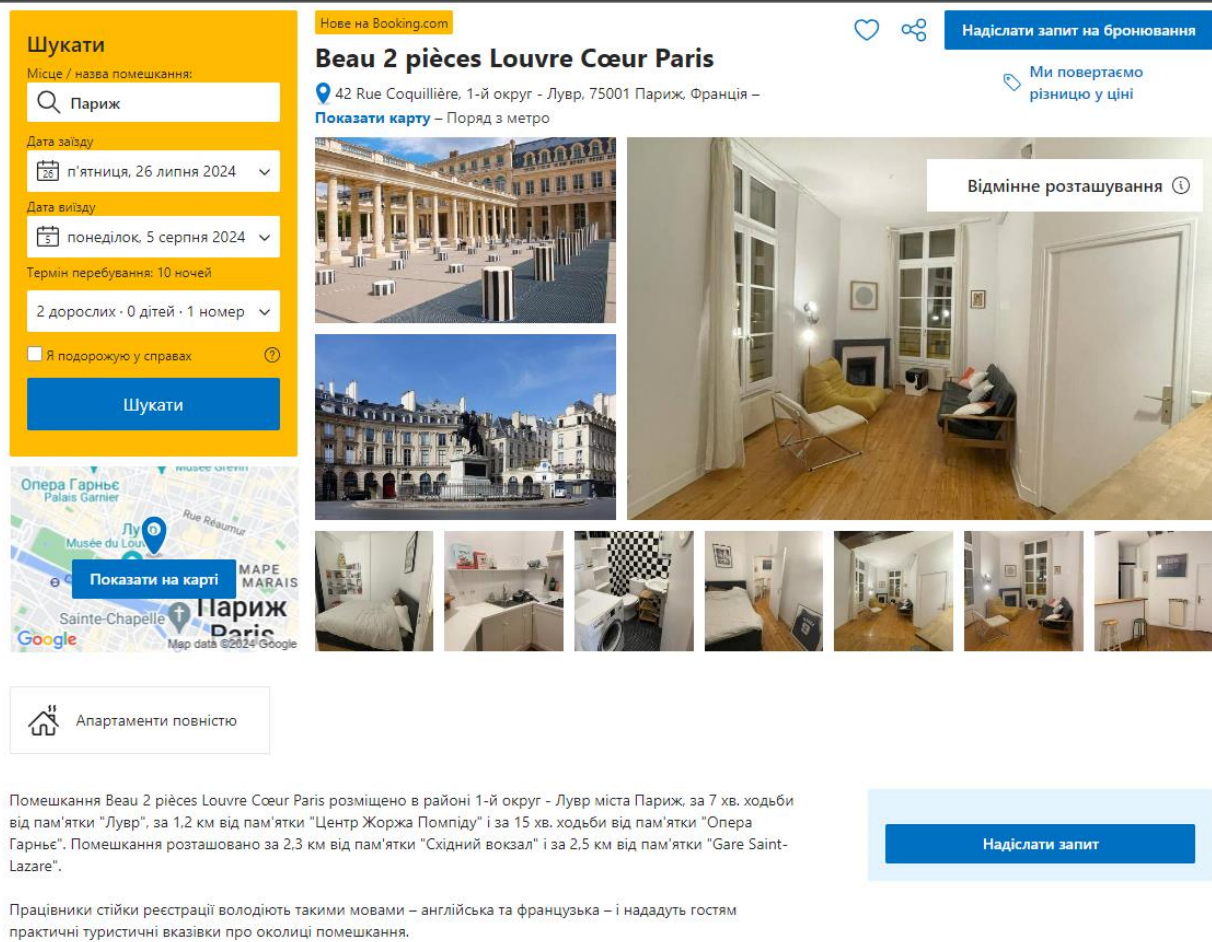


Рис. 1.3 Сторінка помешкання

Інтерфейс додатку є досить зручним, пошук необхідних готелів відбувається інтуїтивно зрозуміло (рис. 1.4). Система відгуків дозволяє

користувачам отримати об'єктивну інформацію про готель від інших мандрівників. Сервіс пропонує величезну кількість готелів та різних видів помешкань у всьому світі.

Серед недоліків для користувачів такої платформи з боку власників є висока конкуренція, а саме завдяки великій популярності платформи, готелі повинні конкурувати за увагу та відгуки з великою кількістю альтернативних варіантів, враховуючи послуги, надані орендодавцями. Також знижена взаємодія з готелями, а саме комунікація між користувачами та готелями може бути обмеженою, а зворотній зв'язок не завжди можливий.

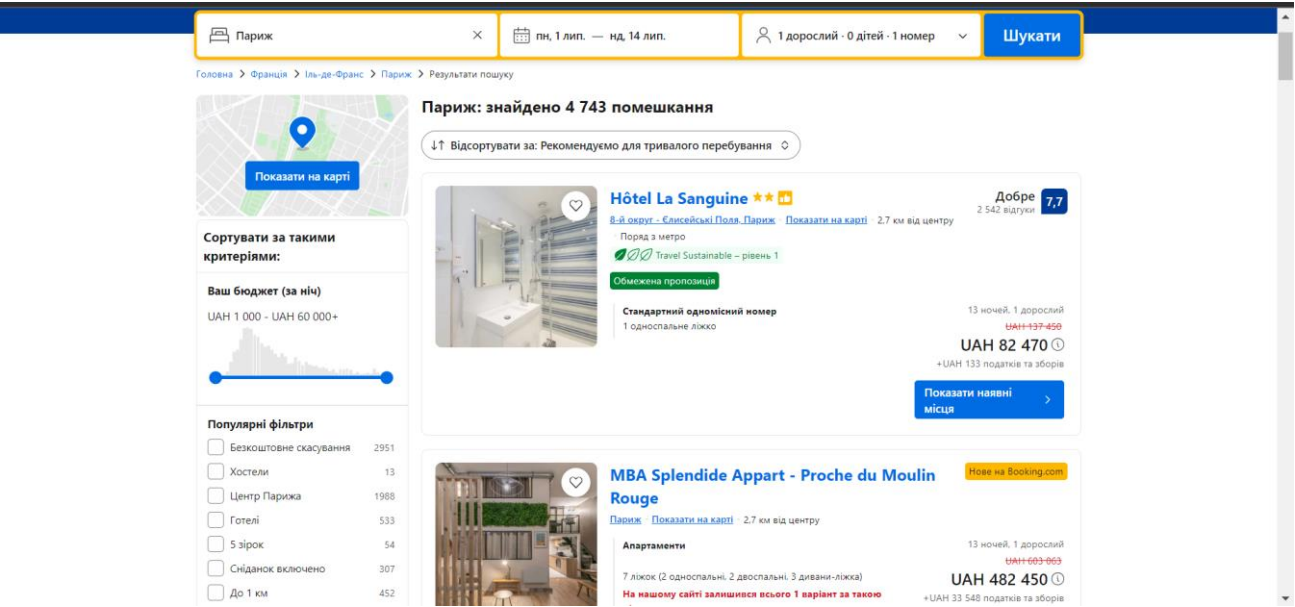


Рис. 1.4 Пошук готелів за критерієм на Booking.com

Інший доволі популярний сервіс для бронювання готелів та подорожей – це Expedia.com [10] (рис. 1.5).

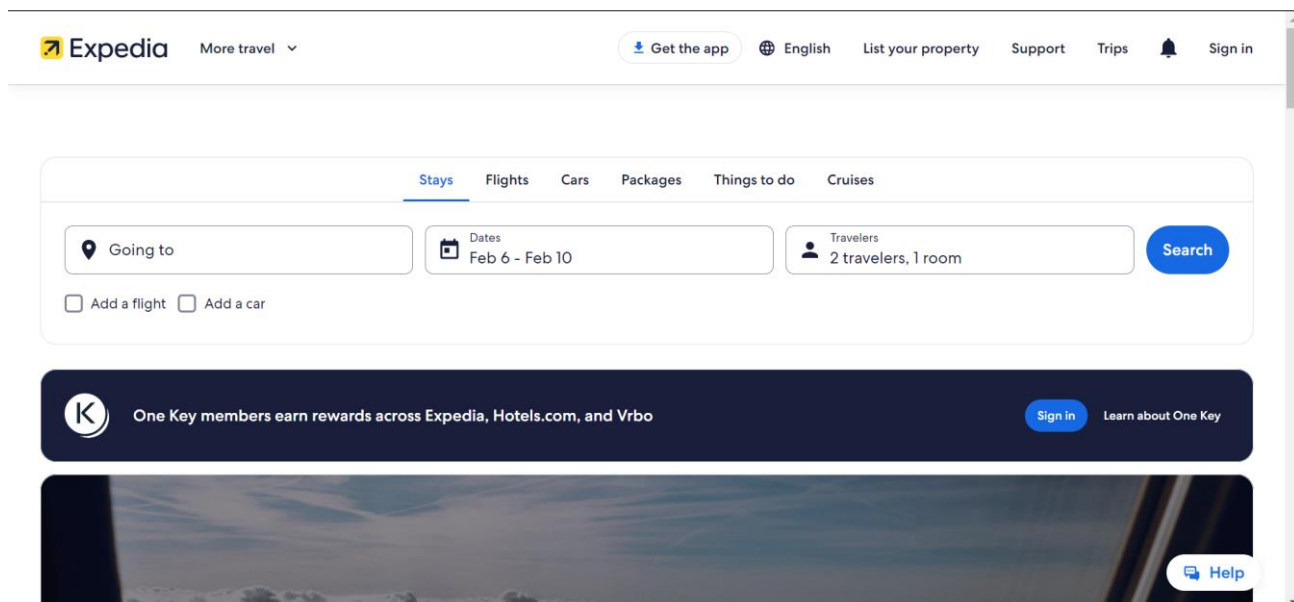


Рис. 1.5 Основна сторінка Expedia.com

Ця платформа також пропонує широкий вибір послуг, таких як резервації готелів, пакетні пропозиції та інші туристичні послуги, що робить його високофункціональним сервісом для подорожей. Наявність мобільного додатка дозволяє користувачам бронювати та керувати своїми подорожами з мобільних пристроїв.

Проте платформа зорієнтована на користувачів лише певних країн, пропонуючи список місця, відповідно список мов, лише які являються офіційними мовами країни (рис 1.6). Відповідно валюта обирається самостійно, без можливості зміни, що є досить незручним для користувача.

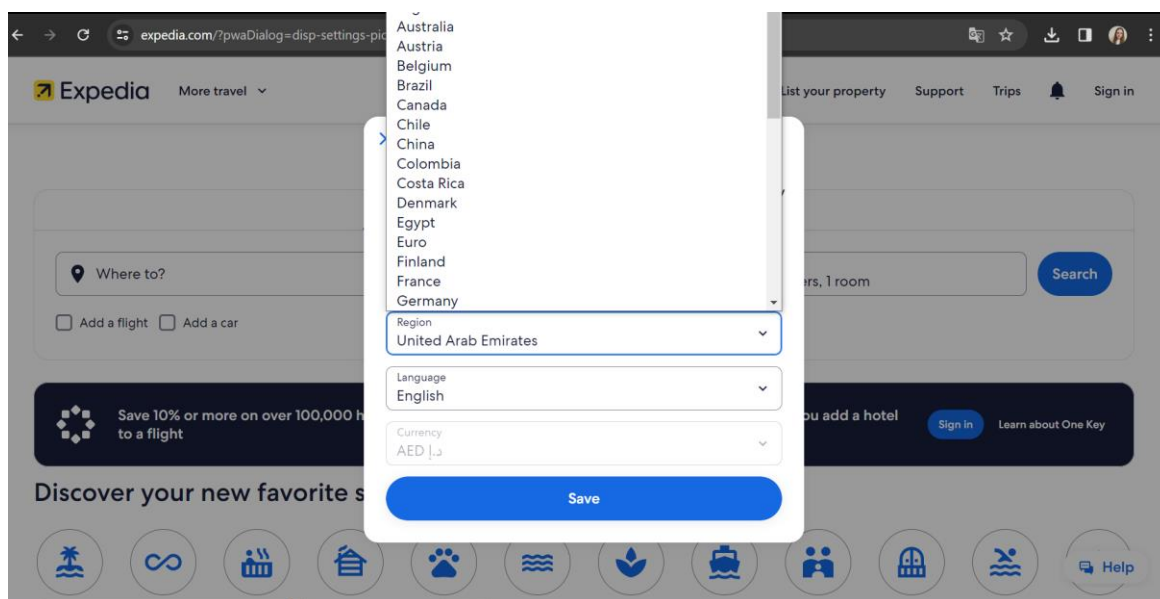


Рис. 1.6 Вибір мови платформи

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		13

Expedia вважається однією з провідних онлайн-платформ для подорожей, де важливо враховувати індивідуальні потреби та вимоги користувача під час вибору сервісу для планування подорожей. Проте серед основних недоліків є обмежена інформація про готелі, порівняно з іншими серверами.

Наприклад на сторінці певного готелю (рис 1.7) загальний детальний опис відсутній повністю. Перехід на сторінку номеру неможливий, а користувач може лише відразу забронювати обраний варіант. (рис 1.8)

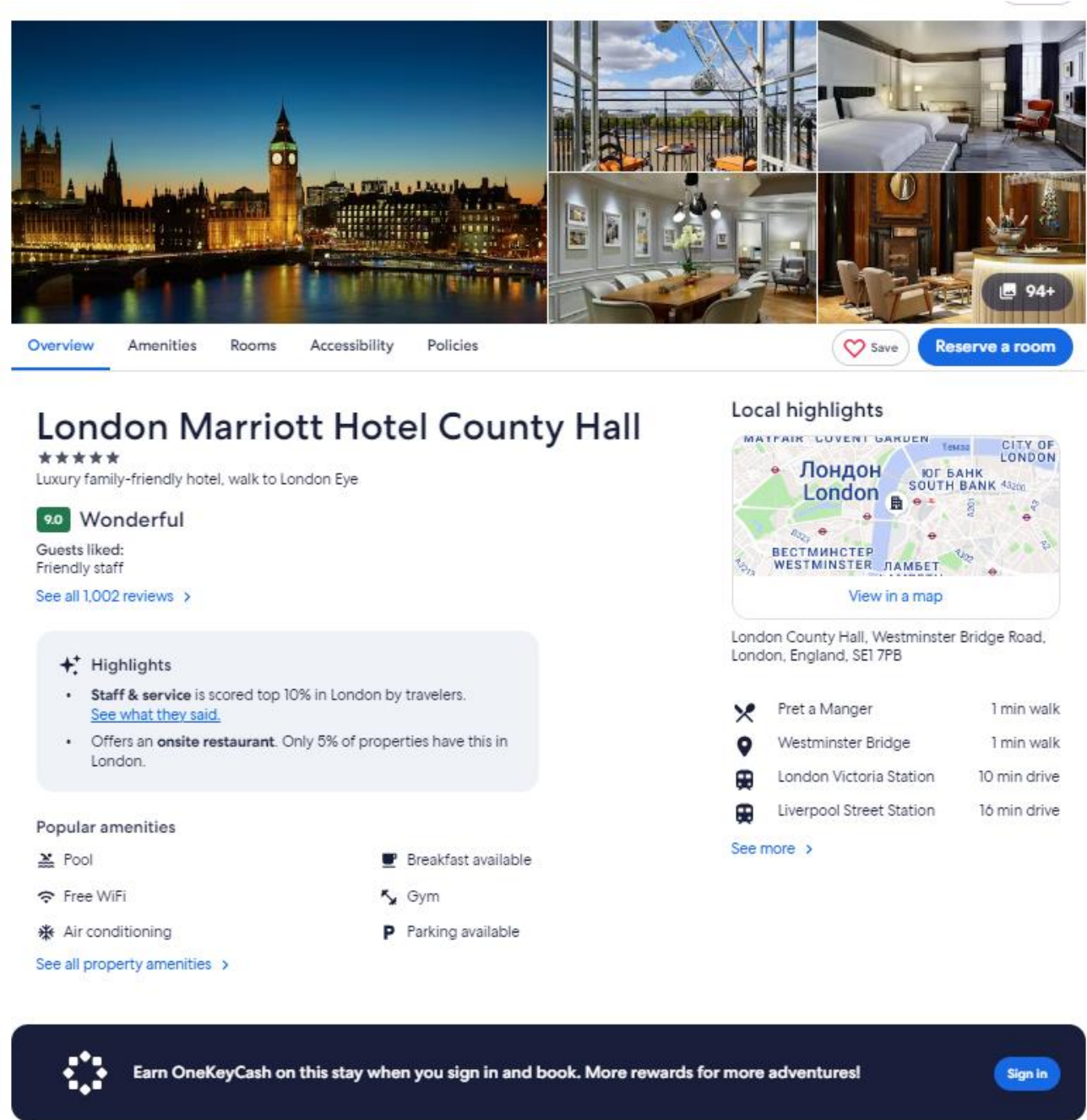


Рис. 1.7 Сторінка готелю

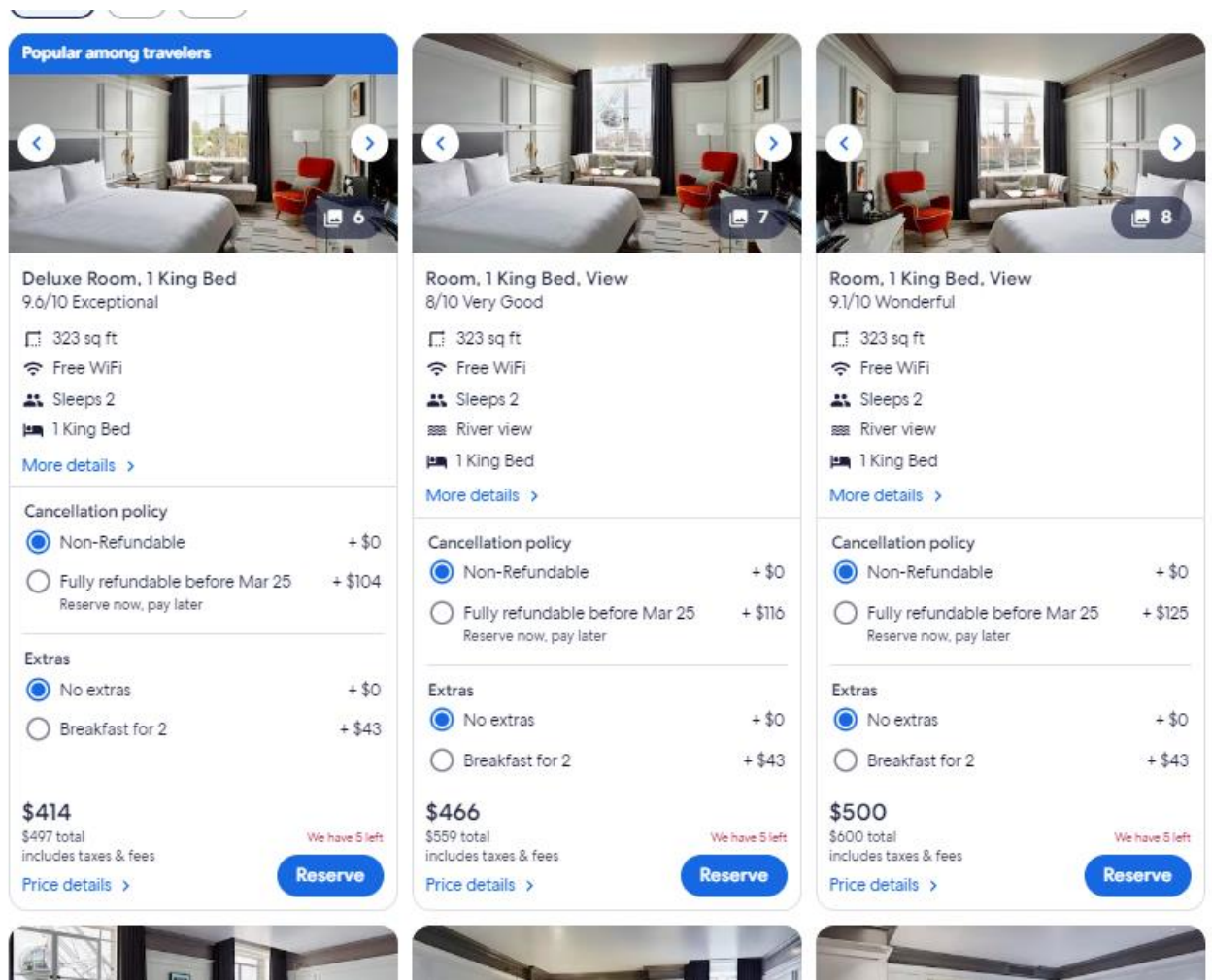


Рис. 1.8 Вибір номера

Іншим популярним аналогом є Airbnb.com [1] (рис. 1.9). Перевагою цього продукту є наявність API для розробників, що дозволяє створювати додатки та інтегрувати сервіс в інші платформи. А також платформа оснащена рядом фільтрів та опцій пошуку, які дозволяють користувачам точно визначити свої вимоги до помешкання, проте фільтрація має певне обмеження, а саме неможливість обрати конкретну країну для пошуку міста та готелів в ній відповідно (рис 1.10).

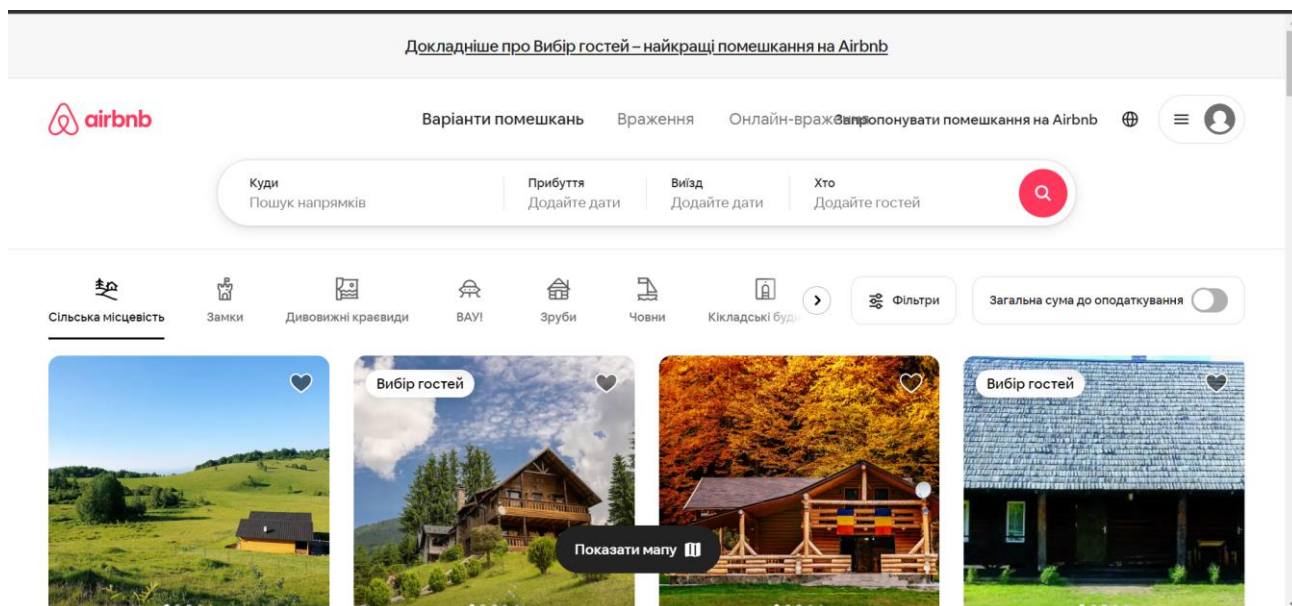


Рис. 1.9 Основна сторінка Airbnb.com

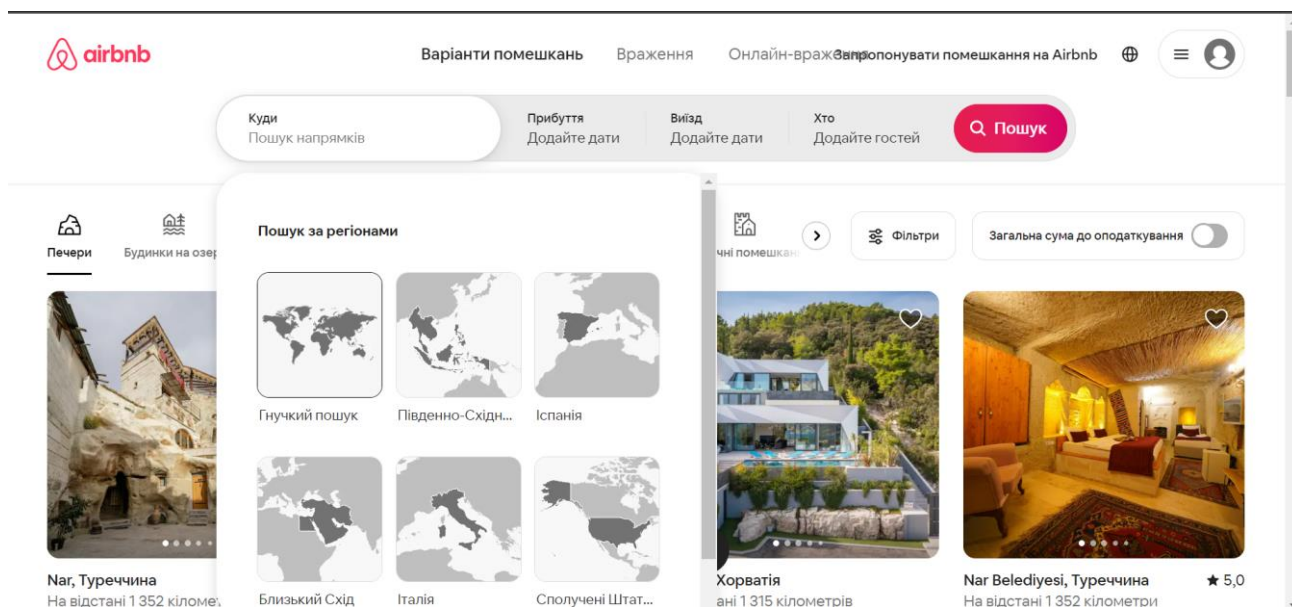


Рис. 1.10 Обмежений пошук на сторінці

Інтерфейс додатку не є обмеженим у виборі мови платформи та бажаної валюти, проте дизайн може бути дещо спаплюженим при перемиканні, наприклад, на українську (рис. 1.11)

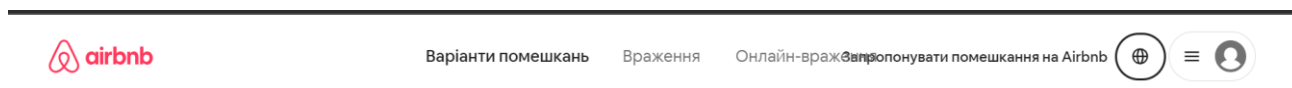


Рис. 1.11 Дефект в інтерфейсі додатку при зміні мови.

Серед недоліків також є обмежений функціонал для готелів. Платформа спеціалізується на житлових варіантах від приватних осіб, тому готелі можуть

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						16
Змн.	Арк.	№ докум.	Підпис	Дата		

відчувати обмеження у функціоналі для розміщення та просування своїх послуг.

Отже було складено таблиці порівняння аналогів:

Таблиця 1.1.

Порівняння веб-додатків аналогів

Характеристика	Booking	Expedia	Airbnb	Easystaychoice
Пошук по країні/місту	Зручний	Зручний	Відсутній (лише регіон)	Зручний
Перегляд готелів, номерів	Зручно	Задовільно	Задовільно	Зручно
Бронювання номерів	Незадовільне (бронювання може залишитися без відповіді)	Відмінно (моментальне)	Добре (лише після реєстрації)	Добре (якщо власник підтвердив)
Актуальність інформації	Відмінно	Відмінно	Добре	Відмінно
Адаптивність інтерфейсу	Адаптивний	Адаптивний	Адаптивний	Адаптивний
Пошук та фільтрація	Відмінно	Відмінно	Задовільно	Відмінно
Дизайн	Відмінно	Відмінно	Відмінно	Відмінно

На основі цього порівняння, виявлено декілька ключових аспектів, а саме:

- 1) Важливо забезпечити високу якість пошуку та фільтрації, аналогічну до Booking чи Expedia. Це допоможе користувачам швидко знаходити потрібні варіанти проживання.
- 2) Підтримка актуальності інформації про готелі та номери є важливою для забезпечення довіри користувачів. Важливо стежити за оновленнями та забезпечувати точні дані, як у Booking.
- 3) Забезпечення привабливого та зручного дизайну, а також адаптивності інтерфейсу для різних пристроїв допоможе залучити та утримати користувачів.

Більш розширене порівняння додатків наведено в додатках [Додаток А].

Аналізуючи таблиці, варто зауважити, що сервіси Booking та Expedia виконують поставлену задачу по бронюванню готелів. Airbnb майже не дає можливість бронювання готелів, лише оренда приміщень. Найбільш точним до поставленої мети – аналог Booking, враховуючи всі переваги та недоліки.

Загалом, аналіз цих аналогів дозволить визначити унікальні рішення та забезпечити покращення платформи для курортних готелів з урахуванням сучасних тенденцій та очікувань користувачів.

1.3 Вибір архітектури онлайн-платформи для курортних готелів

У світлі комплексності та об'єму функціоналу платформи, один з найважливіших виборів – це обрати архітектурний патерн, який краще відповідає вимогам проекту задля забезпечення найвищої продуктивності, гнучкості та масштабованості.

Вибір між монолітною архітектурою, архітектурою мікросервісів та безсерверною архітектурою [3] залежить від конкретних вимог проекту та бізнес-цілей. Кожен підхід має свої переваги та недоліки.

1. Monolithic Architecture

- Переваги: простота розробки та розгортання, взаємодія між компонентами простіша.
- Недоліки: складності при масштабуванні та розширенні, великий об'єм коду може призвести до важкості утримання та оновлення.

2. Microservices Architecture

- Переваги: легкість масштабування окремих сервісів, незалежність розробки та вдосконалення.
- Недоліки: складніше управління багатьма сервісами, збільшена складність в тестуванні та утриманні.

3. Serverless Architecture

- Переваги: автоматичне масштабування, оптимізація витрат ресурсів, відсутність необхідності управління серверами.

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						18
Змн.	Арк.	№ докум.	Підпис	Дата		

- Недоліки: може бути вартісніше в деяких сценаріях, обмеженість налаштувань та управління середовищем.

Розглядаючи різні підходи, було обрано Microservices Architecture, а саме SPA з мікросервісами, що є прикладом мікросервісної архітектури в поєднанні з фронтендом, який побудований як Single Page Application (SPA).

Вибір зумовлений рядом ключових факторів:

1. Розділення відповідальностей:

- **Symfony + API Platform** забезпечують серверну частину (бекенд), обробляють бізнес-логіку, взаємодію з базою даних та API.
- **React** відповідає за клієнтську частину (фронтенд), забезпечуючи динамічний і швидкий інтерфейс користувача.

2. Масштабованість:

- Використання React дозволяє створити швидкий і динамічний інтерфейс, який оновлюється без перезавантаження сторінки, покращуючи досвід користувача.

3. Реактивний інтерфейс:

- Мікросервісна архітектура дозволяє розбити додаток на незалежні сервіси, які можуть масштабуватися окремо.

4. Чітке розділення фронтенду і бекенду:

- Веб-платформа має чітко розділені шари: API як окремий сервіс, що обробляє запити та надає дані, і фронтенд-додаток, що споживає ці дані.

Отже, вибір обґрунтовано з точки зору потреб у масштабованості та гнучкості, необхідних для залучення нових користувачів та забезпечення ефективної роботи системи при зростанні її популярності.

Для розробки API та обробки запитів використовуватиметься фреймворк Symfony. Інтерфейс користувача буде розроблений за допомогою бібліотеки React.

Отож, Microservices Architecture (SPA з мікросервісами) спрощує шлях до створення добре організованого, легкого в управлінні та ефективного

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						19
Змн.	Арк.	№ докум.	Підпис	Дата		

середовища для розробки платформи. Такий підхід дозволить ефективно розробити продукт, задовільнити усі вимоги та досягти мети проекту.

1.4 Обґрунтування вибору інструментальних засобів

При створенні продукту важливу роль грає вибір інструментарію, задля забезпечення ефективності, зручності та надійності продукту.

Для серверної частини було обрано фреймворк Symfony [15] та API Platform [2] – набір інструментів для швидкої розробки API, що надає засоби для автоматичної генерації документації, обробки запитів та відповідей, а також підтримки різних форматів передачі даних.

В контексті обраної архітектури буде реалізовано **REST API**, а саме створення ендпоінтів для обробки CRUD операцій з даними (користувачі, готелі, бронювання, відгуки тощо). Для **аутентифікації та авторизації** буде **використано** JWT для захисту API. Бізнес-логіка також реалізована на серверній частині, та складається з обробки логіки бронювання, перевірка доступності номерів, управління бонусами та відгуками. Валідація вхідних даних проводиться із використанням валідаторів Symfony, що також пропонує структуру та набір компонентів для ефективної розробки та управління різними сервісами. Symfony використовує паттерн Model-View-Controller, що сприяє чіткому розділенню логіки, представлення та управління даними. Фреймворк має вбудований ORM, який дозволяє ефективно взаємодіяти з базою даних через об'єктно-орієнтований підхід, а вбудовані засоби безпеки, такі як компонент Security, допомагають захищати застосунок від потенційних загроз та атак.

React є потужною та популярною бібліотекою для розробки інтерфейсів користувача [12][13]. В контексті обраної архітектури, фронтенд складається із окремих компонентів, а саме UI-компонентів для відображення інтерфейсу користувача. Організовано маршрутизацію в залежності від ролей користувачів (гості, зареєстровані користувачі, власники готелів, адміністратори). Для

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						20
Змн.	Арк.	№ докум.	Підпис	Дата		

взаємодії з API використано бібліотеки (наприклад, Axios) для обміну даними з бекендом.

У сполученні з PHP для розробки застосунку було обрано використання бази даних MySQL, враховуючи усі переваги:

- 1) Сумісність із PHP – MySQL є однією з найпоширеніших реляційних баз даних і взаємодіє дуже добре з PHP
- 2) Відкритий код та загальнодоступність – база даних безкоштовна для використання, що робить її доступною для широкого кола розробників та підприємств.
- 3) Стабільність та надійність
- 4) Широкі можливості, а саме спектр функцій, включаючи транзакції, зовнішні ключі, індексацію та процедурні мови програмування.

Висновки до першого розділу

В даному розділі було проаналізовано проблему розробки онлайн-платформи для бронювання курортних готелів. Поставленою метою було створення платформи, яка спрощує бронювання готельних номерів для клієнтів, обходячи традиційних тур-операторів.

Аналізуючи аналогічні програмні продукти, було виявлено, що сервіси, такі як Booking та Expedia, успішно виконують поставлені завдання, проте, Airbnb, хоч і популярний, не забезпечує можливість бронювання готелів. Було проаналізовано переваги та недоліки аналогів для покращення власного продукту.

Була обрана архітектура SPA з мікросервісами, що поєднує переваги мікросервісної архітектури з швидкістю і зручністю SPA, що робить його оптимальним вибором для проєктів, таких як онлайн-платформа для бронювання номерів курортних готелів. Вибір інструментальних засобів також був обґрунтований: Symfony, вибраний за його вбудований ORM та API Platform, разом з React, використовуються для створення потужного та ефективного інтерфейсу користувача.

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						21
Змн.	Арк.	№ докум.	Підпис	Дата		

Таким чином, перший розділ забезпечив аналіз, обґрунтування та вибір необхідних напрямків для успішної реалізації онлайн-платформи для бронювання курортних готелів.

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						22
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 2. ПРОЕКТУВАННЯ ОНЛАЙН-ПЛАТФОРМИ БРОНЮВАННЯ ГОТЕЛІВ

2.1 Вимоги до системи

Для успішної розробки онлайн-платформи бронювання готелів необхідно визначити варіанти використання та структуру системи, а також функціональні та нефункціональні вимоги. Це забезпечить розуміння основних процесів, що відбуваються на платформі, і дозволить розробити оптимальну архітектуру для реалізації необхідного функціоналу.

2.1.1 Визначення варіантів використання та структури системи

Для забезпечення виконання необхідних функцій, було створено вимоги, яким повинна відповідати веб-платформа для розміщення даних курортних готелів. За допомогою цих вимог система працює безперебійно та відповідає всім необхідним стандартам і специфікаціям.

В підрозділі 1.2 було проаналізовано аналоги і на їх основі було виведено критичні функції системи, а саме:

- система бронювання готелю
- система пошуку готелю
- система авторизації користувача
- система лояльності (накопичувальні бонуси при бронюванні)

Відповідно до цього ми отримали наступні бізнес вимоги:

1. Монетизація через комісію: власники готелів сплачують певний відсоток від кожного успішного бронювання. Це основний спосіб заробітку платформи.

2. Програма лояльності: механізм накопичення бонусів, які користувачі можуть використовувати для зниження вартості наступних бронювань. Це стимулює повторні бронювання за допомогою платформи.

3. Аналітика та звітність: надання власникам готелів аналітичних даних про бронювання, що допоможе їм краще управляти своїм бізнесом.

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						23
Змн.	Арк.	№ докум.	Підпис	Дата		

4. Представлення проекту: платформа реалізована у вигляді інтерактивного веб-сайту, що міститиме інформацію про різні курортні готелі, їх послуги та доступність номерів.

Вимоги користувачів:

1. **Реєстрація та аутентифікація:** можливість створення та редагування облікового запису; можливість авторизації для доступу до персональних даних, бронювань та функцій.

2. **Бронювання номерів:** можливість користувачів здійснювати онлайн-бронювання номерів, обираючи дати перебування та переглядаючи доступні варіанти, відповідно здійснювати бронювання з можливістю використання бонусів та подальшого накопичення.

3. **Внесення та перегляд відгуків до готелю:** можливість додавати та переглядати відгуки до готелю, щоб мати краще уявлення про якість, комфорт та відповідність опису від реальних клієнтів. Це допомагає іншим користувачам при виборі місця для відпочинку.

4. **Пошук готелів/номерів:** можливість перегляду готелів відповідно до вподобань користувача, а саме: пошук за певним містом/країною, відповідно скорочення списку пропонуєваних номерів за допомогою фільтрації по типу кімнати та додаткової класифікації.

Вимоги персоналу веб-додатку (власники готелів):

1. **Керування інформацією про готель:** можливість внесення, видалення та оновлення інформації про готель, включаючи наявність номерів, ціни, та послуги.

2. **Керування бронюванням:** можливість підтвердити чи відхилити бронювання та завершити бронювання при заселенні до готелю користувачів.

Для більш детального розуміння взаємодії користувачів з системою, була розроблена діаграма варіантів використання (use-case diagram) (рис.2.1), яка відображає всі основні дії, які можуть виконувати користувачі (актори) системи.

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						24
Змн.	Арк.	№ докум.	Підпис	Дата		

3. **Перегляд готелів відповідно до побажань користувачів:** фільтрація готелів та перегляд сторінки певного з можливим подальшим переходом до списку номерів вказаного готелю.

4. **Бронювання кімнат (для авторизованих користувачів):** система здійснює перевірку доступності кімнати на обрану дату. Користувач може забронювати за наявності з можливістю накопичення та списування бонусів.

5. **Управління акаунтом:** система забезпечує можливість перегляду та редагування персональних даних, перегляд історії бронювань (для власників та користувачів), скасування замовлення у разі необхідності (для користувачів) та підтвердження/відхилення/завершення бронювання (для власників).

6. **Відгуки та рейтинги (для користувачів):** платформа дає можливість залишити відгуки та оцінити готелі після завершення бронювання.

7. **Управління готелем та номерами (для власників):** редагування інформації про готель та його кімнати.

8. **Запит на співпрацю (для власників та адміністратора):** система дає можливість надіслати запит на співпрацю з відповідними даними про готель, які адмін, при підтвердженні, вносить до бази даних, та відкриває можливість користуватися платформою з функціоналом для власника.

8. **Валідація місця розташування:** за допомогою стороннього API, а саме openstreetmap [11], система перевіряє чи власник вказав правильне та реальне місце розташування готелю, звіряючи це із глобальними даними і лише після цього, якщо обраного місце розташування ще не було, створюю розташування.

Нефункціональні вимоги:

1. Продуктивність та масштабованість:

– платформа повинна витримувати навантаження мінімум 20 одночасних користувачів без зниження продуктивності.

– час, необхідний для ознайомлення користувачів із платформою та її функціоналом – не більше 3 годин;

– час відповіді сервера повинен не повинен перевищувати 30 секунд при звичайному навантаженні.

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						26
Змн.	Арк.	№ докум.	Підпис	Дата		

- інтерфейс має бути зрозумілим та з відповідними підказками.
- час завантаження сторінок та сторінок результатів пошуку – не більше 40 секунд.

2. Надійність та безперервність роботи:

- система повинна працювати безперервно, з мінімальними простоями через технічне обслуговування. Час, що витрачається на обслуговування платформи, не повинен перевищувати 4% від загального часу роботи.

3. Безпека:

- шифрування чутливих даних, таких як паролі користувачів.
- регулярне оновлення системи для запобігання вразливостям.

4. Модульність та підтримка:

- система забезпечує легкість у внесенні змін та оновлень без зупинки роботи платформи.

- наявна підтримка платформи з регулярними оновленнями та виправленнями помилок.

- система повинна мати можливість збільшувати продуктивність зі збільшенням користувачів таким чином, щоб це не вплинуло негативно на її роботу.

4. Кросплатформеність:

- платформа повинна коректно працювати на різних пристроях і в різних браузерах (мобільні телефони, планшети, настільні ПК).

Таким чином, структура та функціонал платформи для бронювання готелів забезпечують її стабільну роботу та відповідність всім необхідним стандартам, що дозволяє їй успішно функціонувати та розвиватися.

У результаті аналізу функціональних вимог було створено діаграму сутностей класів (рис 2.2).

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						27
Змн.	Арк.	№ докум.	Підпис	Дата		

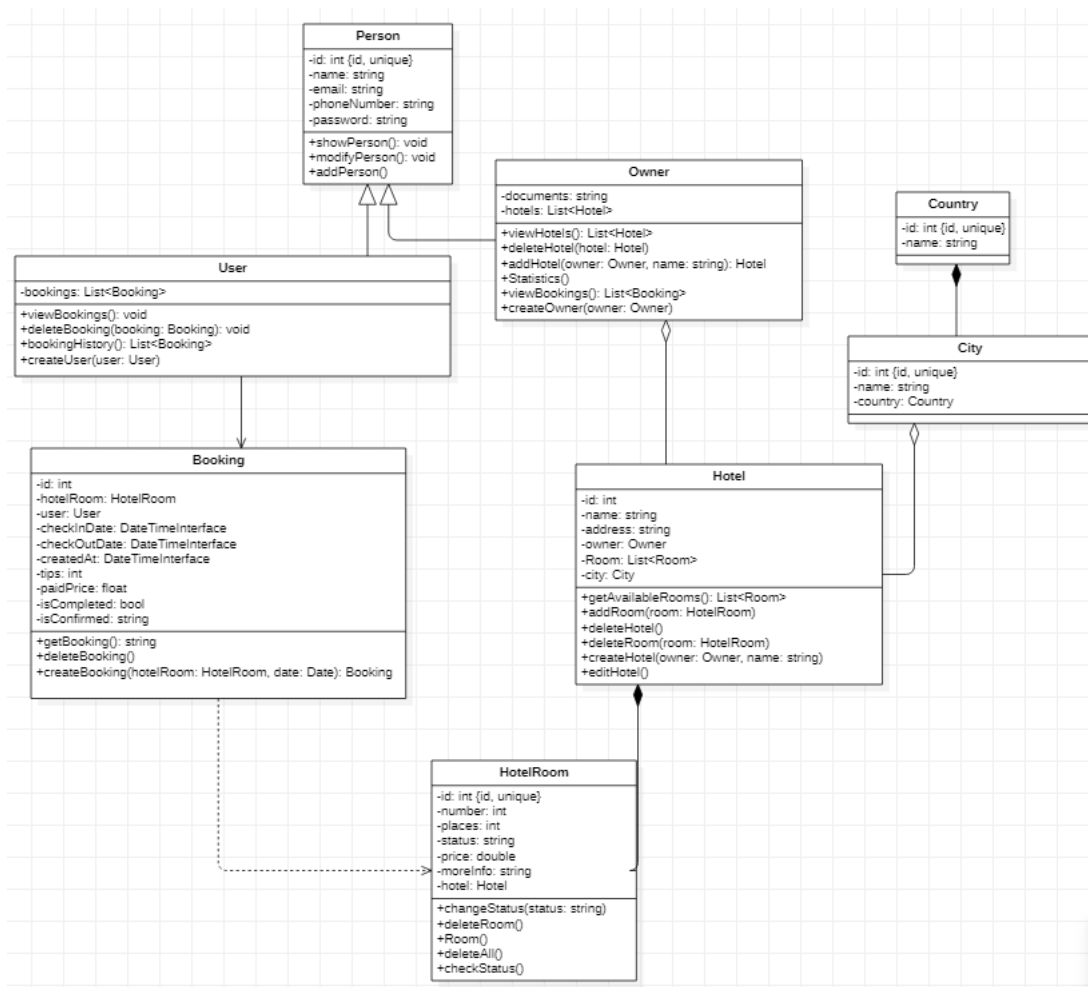


Рис.2.2 Діаграма сутностей класів

Діаграма сутностей представляє ключові класи та їх взаємозв'язки у системі веб-платформи для бронювання курортних готелів. Клас Person має методи для управління інформацією про особу. Клас User представляє звичайних користувачів платформи, які можуть здійснювати бронювання готелів. Методи включають перегляд та управління бронюваннями.

Клас Owner представляє власників готелів, які можуть керувати своїми об'єктами. Методи включають перегляд і управління готелями та бронюваннями.

Клас Booking представляє бронювання номерів у готелях. Методи включають отримання деталей бронювання, видалення та створення нового бронювання.

Класи Hotel та HotelRoom представляють готелі, що додаються власниками та номери в готелях відповідно. Їхні методи включають управління номерами та готелем, статусом і інформацією про номер.

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		28

Класи Country та City представляють локацію розташування.

Ця діаграма демонструє, як різні сутності взаємодіють одна з одною, забезпечуючи функціонування системи бронювання готелів. Класи користувачів (User і Owner) мають можливість управляти своїми бронюваннями та готелями, тоді як класи Booking, Hotel, і Room забезпечують основну функціональність бронювання і управління номерами.

На основі наведених даних для кращого розуміння архітектури системи та взаємодії між її компонентами було створено діаграмами розгортання (рис. 2.3) та компонентів (рис. 2.4), що відображають логічну структуру системи, показуючи взаємодію між різними компонентами системи.

На сервері використовуються дії для створення сутностей [8], контролери, сутності [7], EntityListener [6], розширення [9] для отримання готелів певного власника, репозиторії, та сервіси для валідації місця розташування готелю за допомогою стороннього API.

На клієнтській частині (фронтенді) використовуються компоненти для елементів сторінки, сторінки, маршрути (routes) для ролей, storage для зберігання даних, та утиліти (utils) для логіки обробки токенів, авторизації, логіну та отримання інформації про користувачів.

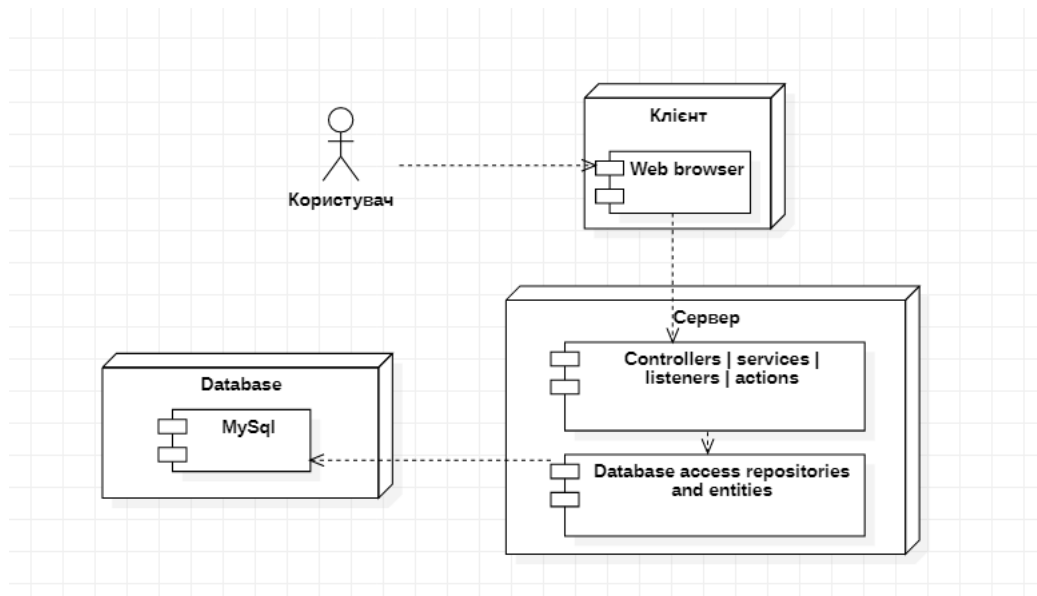


Рис.2.3 Діаграма розгортання платформи

Діаграма розгортання показує, як програмні компоненти розподіляються на фізичні машини або вузли. В даному випадку, серверна частина включає в себе такі елементи:

- **Controllers:** обробляють HTTP-запити і викликають відповідні сервіси.
- **Services:** логіка додатку, включаючи валідацію місця розташування готелю.
- **Repositories:** доступ до бази даних.
- **Entities:** моделі даних.
- **EntityListener:** прослуховувачі подій для сутностей.
- **Extensions:** додаткові модулі для специфічних функцій, таких як отримання готелів певного власника.

Клієнтська частина (пристрої) включає в себе **Web Browser**, де працює фронтендова частина додатку, що включає компоненти, сторінки, маршрути, сховища даних та утиліти.

Діаграма компонентів деталізує логічну структуру системи:

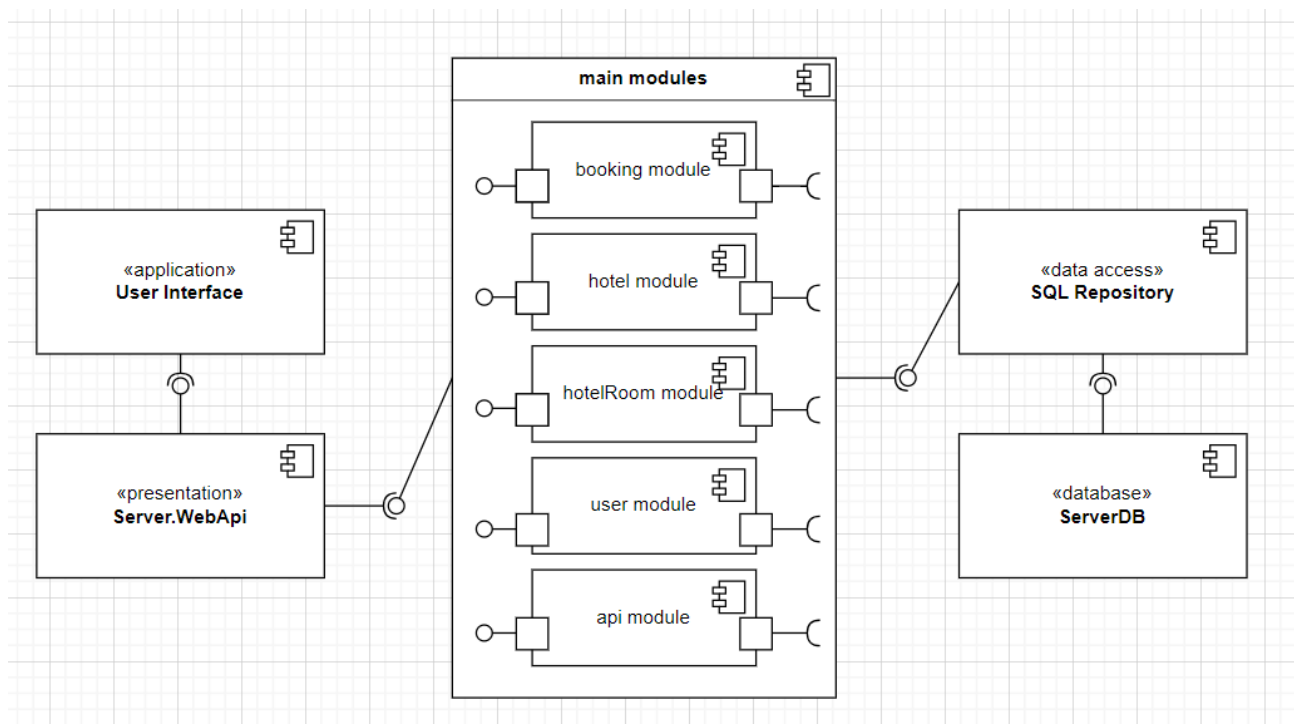


Рис.2.4 Діаграма компонентів

Основні елементи діаграми:

- Інтерфейс користувача – це компоненти, які надають графічний інтерфейс для користувачів платформи. Цей модуль складається з таких частин:

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		30

- **Components:** елементи UI, які формують інтерфейс користувача.
- **Pages:** основні сторінки додатку, такі як головна сторінка, сторінка пошуку, сторінка бронювання. На цьому етапі формуються сторінки відповідно до ролей користувача.
- **Routes:** маршрути для навігації користувача по сторінках додатку.
- **Storage:** налаштування сховища для локального зберігання даних.
- **Utils:** допоміжні функції для обробки токенів, авторизації, логіну та отримання інформації про користувачів
- **Server.WebApi** – це компоненти, які надають API для взаємодії з іншими модулями та компонентами. Використовується протокол HTTP для обміну запитами та відповідями.
- **Main Modules** – це компоненти, які реалізують основну логіку системи бронювання готелю. Елементи модуля можуть взаємодіяти один з одним та з іншими компонентами за допомогою API.
- **SQLRepository** – це компонент, який надає доступ до даних, які зберігаються в базі даних. Він використовує мову SQL для виконання запитів та команд до бази даних
- **ServerDB** – це компонент, який представляє базу даних, яка зберігає всю інформацію про готелі, кімнати, бронювання, користувачів та власників.

Отже ця діаграма компонентів класів показує, як різні модулі програмного забезпечення для системи бронювання готелю взаємодіють один з одним та з іншими компонентами, такими як інтерфейс користувача, API, репозиторій даних та база даних.

2.2 Розробка бази даних

Для розробки бази даних було обрано MySQL, зокрема phpMyAdmin як інструмент для керування базою даних. Вибір MySQL обґрунтований кількома ключовими причинами:

1. **Широке використання та підтримка:** MySQL є однією з найпопулярніших систем керування базами даних (СКБД) у світі. Його активно

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						31
Змн.	Арк.	№ докум.	Підпис	Дата		

використовують у багатьох великих проектах, що забезпечує надійність та стабільність.

2. Продуктивність та масштабованість: MySQL забезпечує високу продуктивність та може легко масштабуватися для обслуговування великих обсягів даних, що є критичним для багатьох веб-додатків.

3. Інтеграція з PHP: MySQL чудово інтегрується з PHP, що є однією з основних мов програмування для веб-розробки. phpMyAdmin, як інструмент для адміністрування MySQL, надає зручний веб-інтерфейс для керування базою даних, що спрощує процес налаштування та управління.

4. Безпека: MySQL пропонує розширені можливості управління користувачами та правами доступу, що забезпечує високий рівень безпеки даних.

Для створення бази даних було розроблено діаграму класів, кожний клас якої відповідає за окремий аспект системи управління готелями. Діаграма класів (рис. 2.5) демонструє зв'язки між основними сутностями в системі.

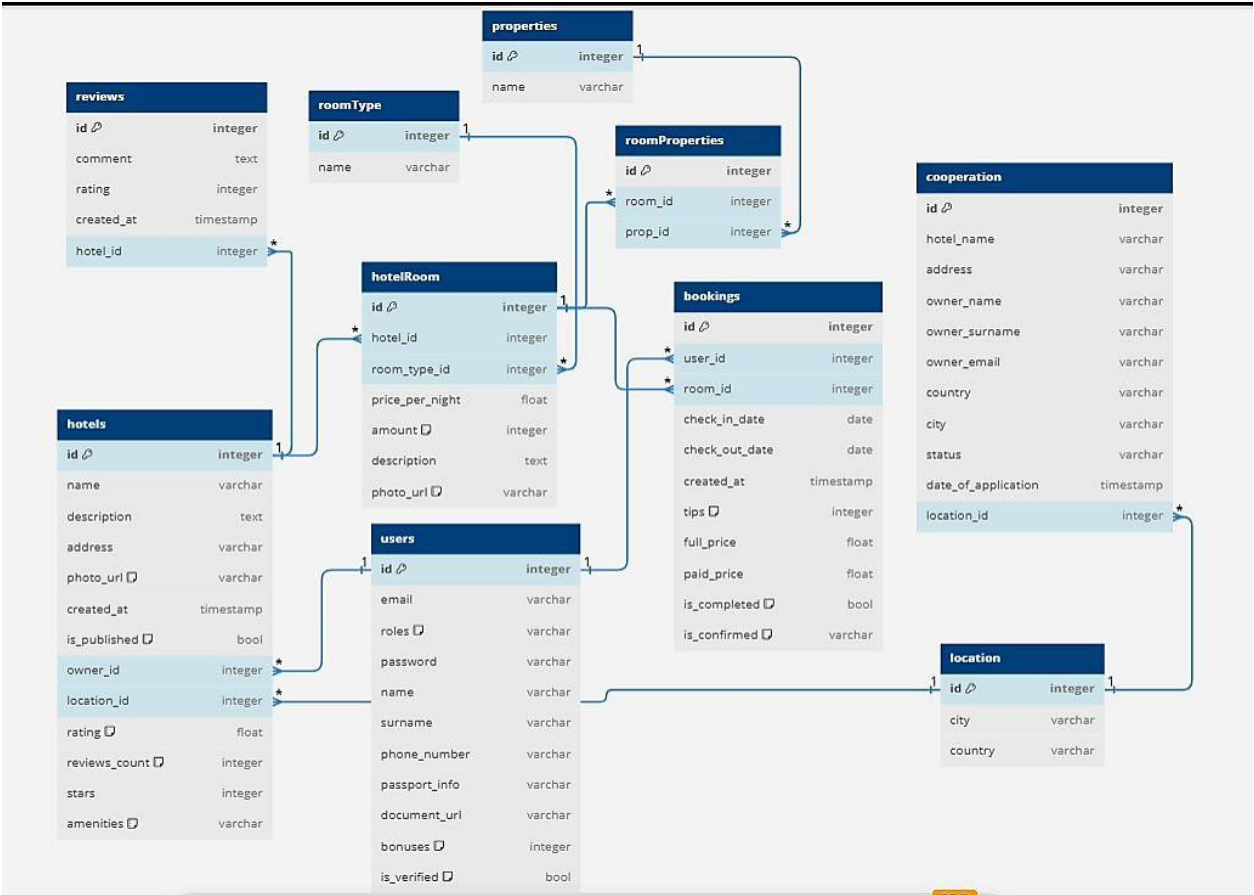


Рис.2.5 Діаграма класів системи для бронювання готелів

З діаграми видно, що усі таблиці пов'язані між собою зв'язками один до багатьох, а таблиці HotelRoom та Properties пов'язані як багато до багатьох за допомогою допоміжної таблиці, що їх зв'язує.

Таблиця users зберігає інформацію про користувачів платформи. Користувачі можуть бути як звичайними користувачами, так і власниками готелів. Вона пов'язана з таблицею bookings через user_id, що означає, що користувачі можуть робити бронювання.

Таблиця roomType містить типи номерів у готелях, наприклад, "одномісний", "двомісний" тощо. Вона пов'язана з таблицею hotelRoom через room_type_id, що дозволяє визначити тип кожного номеру в готелі.

В reviews зберігаються відгуки користувачів про готелі. Вона пов'язана з таблицею hotels через hotel_id, що дозволяє кожному відгуку бути прив'язаним до певного готелю.

Таблиця properties містить властивості номерів, такі як "кондиціонер", "балкон" тощо. Вона пов'язана з таблицею roomProperties через prop_id, що дозволяє визначити, які властивості має кожен номер у готелі.

Location містить інформацію про місцезнаходження, включаючи міста і країни. Вона пов'язана з таблицею hotels через location_id, що дозволяє вказати місцезнаходження кожного готелю. Таблиця також пов'язана з таблицею cooperation через location_id, що використовується для подання заявок на співпрацю.

В hotelRoom зберігаються номери готелів, включаючи їх типи, ціни, кількість та іншу інформацію. Вона пов'язана з таблицею hotels через hotel_id і з таблицею roomType через room_type_id. Також ця таблиця пов'язана з таблицею bookings через room_id, що дозволяє відстежувати бронювання кожного номера.

Hotels зберігає інформацію про готелі, включаючи назву, опис, адресу, фотографії, рейтинг та інші деталі. Вона пов'язана з таблицею users через owner_id, що дозволяє визначити власника кожного готелю. Таблиця також пов'язана з таблицею location через location_id.

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						33
Змн.	Арк.	№ докум.	Підпис	Дата		

Cooperation містить заявки на співпрацю від власників готелів, які хочуть додати свої готелі до платформи. Вона пов'язана з таблицею location через location_id, що дозволяє вказати місцезнаходження готелю у заявці.

Bookings зберігає інформацію про бронювання номерів, включаючи дати заїзду та виїзду, ціни, статуси тощо. Вона пов'язана з таблицями users та hotelRoom через user_id і room_id відповідно, що дозволяє відстежувати, які користувачі забронювали які номери.

RoomProperties містить зв'язки між номерами в готелях і їх властивостями. Вона пов'язана з таблицями hotelRoom та properties через room_id і prop_id відповідно, що дозволяє визначити, які властивості має кожен номер.

2.3 Проектування та реалізація алгоритмів роботи платформи

Проектування та реалізація алгоритмів роботи платформи є ключовим етапом у створенні ефективної та надійної системи управління готелями. Особлива увага буде приділена процесу бронювання, який є критично важливим для системи бронювання готелів.

Функціональність системи розділена на окремі сервіси, які можуть бути розгорнуті та масштабовані незалежно один від одного. Це забезпечує гнучкість, підвищує стійкість системи та полегшує її супровід і оновлення.

Основні сервіси, що використовуються в платформі:

1. **Сервіс управління користувачами:** відповідає за реєстрацію, аутентифікацію та управління профілями користувачів.
2. **Сервіс управління готелями:** обробляє додавання, редагування та видалення інформації про готелі.
3. **Сервіс управління номерами:** відповідає за управління інформацією про номери в готелях.
4. **Сервіс бронювання:** обробляє запити на бронювання номерів, перевіряє їх доступність на певну дату.

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						34
Змн.	Арк.	№ докум.	Підпис	Дата		

5. **Сервіс управління відгуками:** відповідає за додавання, редагування та видалення відгуків користувачів.

Система управління готелями складається з кількох ключових модулів, кожен з яких виконує певні функції. До основних модулів належать:

- Процес бронювання: бронювання номерів, перевірка доступності номеру на вказані дати, підтвердження бронювання з боку власника.
- Управління готелями: додавання, редагування, видалення інформації про готель, керування операціями CRUD [5] для роботи з номерами готелю.
- Управління користувачами: реєстрація, авторизація, управління профілями. З боку адміністратора – можливість надавати відповідні ролі користувачам.

Кожен з цих модулів взаємодіє з базою даних для збереження та отримання необхідної інформації.

Для ілюстрації роботи системи було використано діаграму активності, яка використовується для моделювання динамічного характеру системи, представляючи взаємодії між об'єктами та процесами у часі, які включають у себе послідовність активностей, рішення та обмін повідомленнями між об'єктами. Вона надає візуальне представлення дій, які виконуються в системі та порядку їх виконання.

«Діаграма основних дій» (рис.2.6) відображає послідовність дій користувача, що досягає кінцевого призначення за бізнес вимогою платформи.

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						35
Змн.	Арк.	№ докум.	Підпис	Дата		

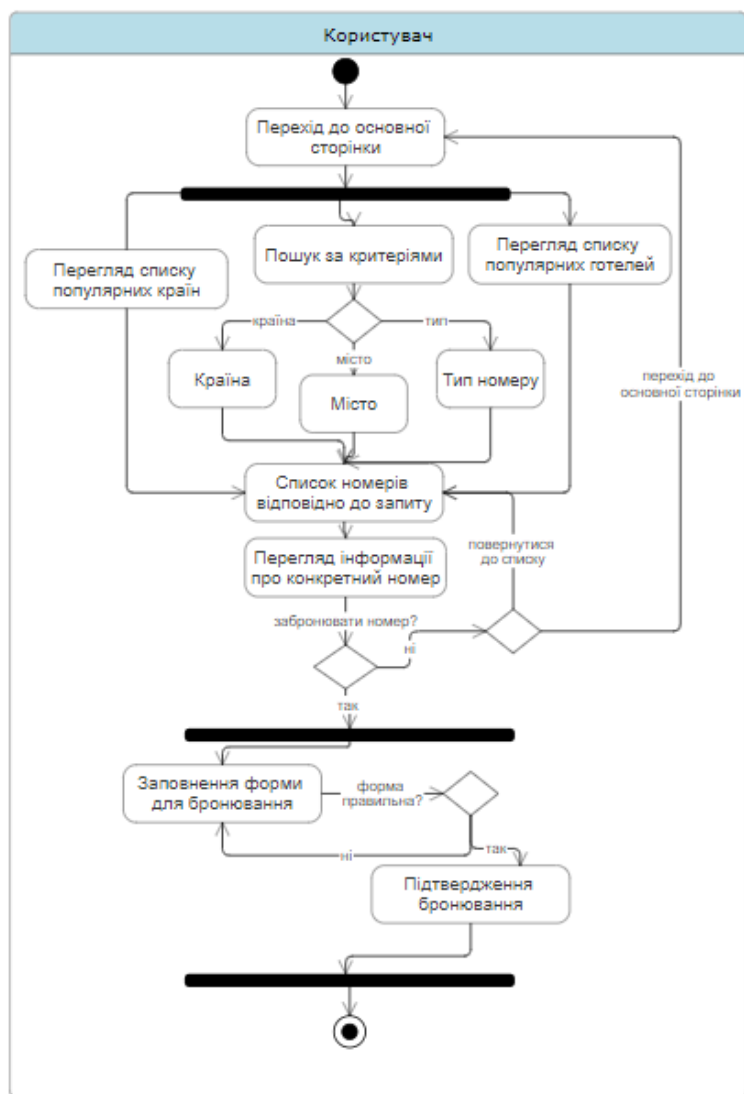


Рис.2.6 Діаграма дій системи для процесу обрання номеру для бронювання

Отже розглянемо шлях користувача від початку роботи з платформою до бізнес цілі – бронювання номеру. Спочатку користувач починає взаємодію з системою, обираючи головну сторінку, де виконує пошук за критерієм. Розгалуження об'єднує 3 потоки управління (пошук за країною, містом та типом номеру). Якщо користувач не обирає пошук, тоді йому запропоновано переглянути список топ країн чи готелів. Відповідно не залежно від вибору, користувач потрапляє на сторінку списку номерів.

Серед запропонованого списку, користувач обирає конкретний номер та переходить до перегляду інформації. Якщо користувач не хоче забронювати саме цей номер, він повернеться або до попереднього списку, або до основної

сторінки (залежно від вибору користувача). Якщо номер підходить користувачеві, фіксується вибір номеру, відкривається форма для заповнення та відправлення запиту на бронювання, якщо користувач заповнив все вірно.

Наступна «діаграма дій для процесу бронювання обраного номеру» (рис.2.7) відображає послідовність дій, враховуючи взаємодію між користувачем, системою та власником готелю.

В даній діаграмі використано розподіл на зону відповідальності: користувач, система, власник.

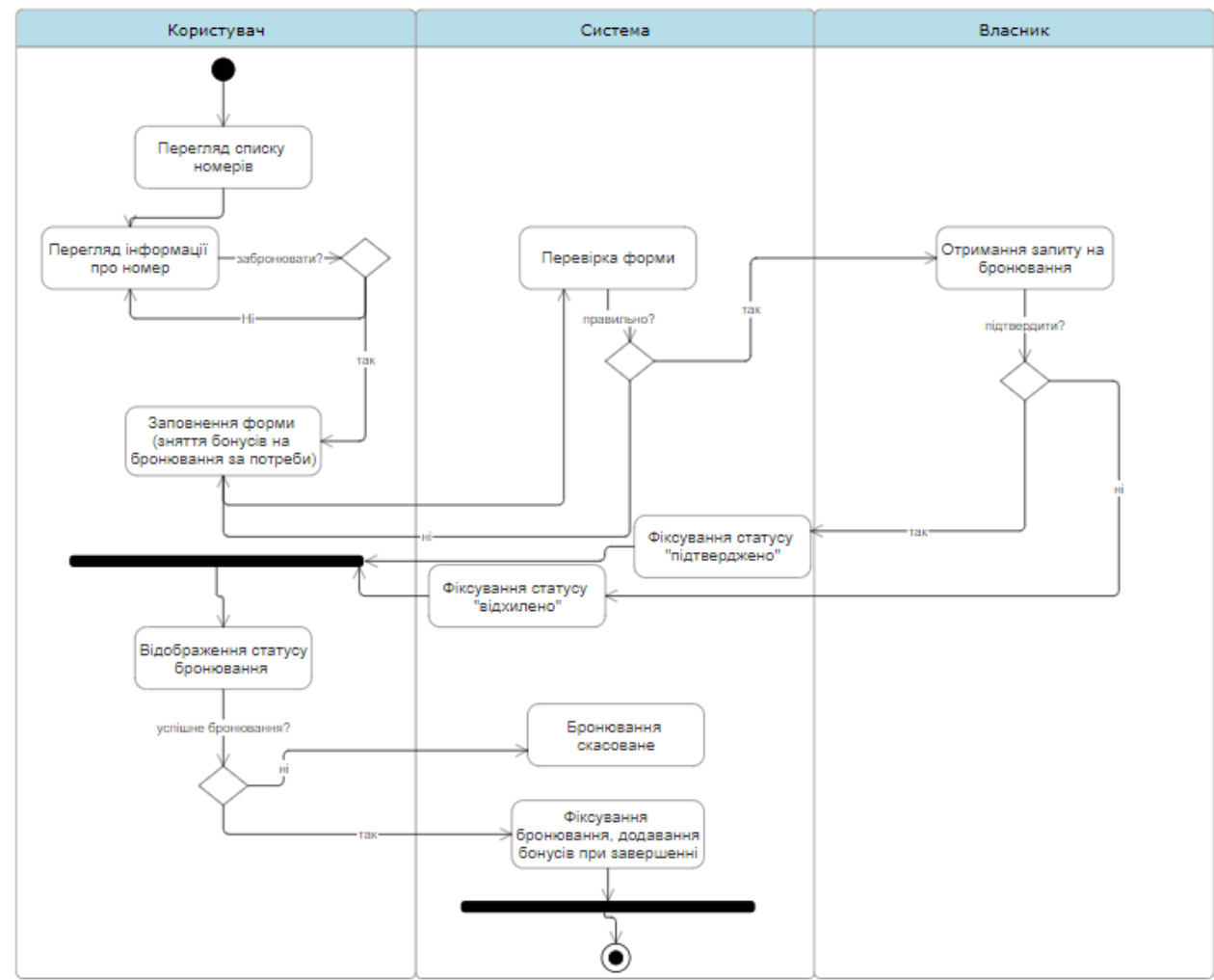


Рис.2.7 Діаграма дій для процесу бронювання обраного номеру

Розглянемо процес бронювання номеру. При переході на сторінку номеру, користувач може обрати чи забронювати номер. Якщо так, відбувається заповнення форми, яку перевіряє на помилки система і відповідно відправляє власнику, після правильного внесення даних. Власник може

підтвердити бронювання, чи відхилити. Відповідно користувач побачить змінений статус свого замовлення і при успішному, бронювання буде закріплено за ним.

2.4 Реалізація функціоналу платформи

Додаток для бронювання курортних готелів складається з фронтенд і бекенд частин, реалізованих за допомогою React та Symfony відповідно.

Основні компоненти системи включають фронтенд на React для взаємодії з користувачами, бекенд на Symfony, що забезпечує роботу API, база даних для збереження інформації про готелі, номери, користувачів і бронювання, JWT для авторизації та аутентифікації користувачів.

Основні компоненти фронтенд частини додатку включають **App.js** – головний компонент, що відповідає за рендеринг всіх інших компонентів.

Для створення стану `authenticated`, який вказує, чи автентифікований користувач, використовується хук `useState`, `useLocation` і `useNavigate` з React Router для визначення поточного місцезнаходження та навігації.

Фрагмент коду:

```
function App () {  
  const [authenticated, setAuthenticated] =  
    useState(localStorage.getItem("token"));  
  const location = useLocation();  
  const navigate = useNavigate();
```

Рис.2.8 Використання хуків

Функція `authRouteRender` визначає, які маршрути відображатимуться залежно від статусу автентифікації користувача та його ролі.

Якщо користувач не автентифікований, відображаються загальні маршрути.

Якщо користувач автентифікований, відображаються маршрути, що відповідають його ролі (клієнт, власник готелю або адміністратор).

Фрагмент коду:

```
const authRouteRender = () => {  
  if (!authenticated) {  
    return (  
      routes.map((route, index) => (  
        <Route key={index} path={route.path} element={route.element} />  
      ))
```

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						38
Змн.	Арк.	№ докум.	Підпис	Дата		

```

    );
  } else {
    const userInfo = getUserInfo();

    if (userInfo) {
      //if user logged in
      const userRoles = userInfo.roles;

      //USER
      if (userRoles.includes(roles.CLIENT)) {
        return (
          userRoutesConcat.map((route, index) => (
            <Route key={index} path={route.path} element={route.element} />
          )))
      }
      //OWNER
      else if (userRoles.includes(roles.OWNER)) {
        return (
          ownerRoutesConcat.map((route, index) => (
            <Route key={index} path={route.path} element={route.element} />
          )))
      }
      // ADMIN
      else if (userRoles.includes(roles.ADMIN)) {
        return (
          adminRoutesConcat.map((route, index) => (
            <Route key={index} path={route.path} element={route.element} />
          )))
      }
    }
  }
};

```

Рис.2.9 Розподіл роутів за ролями

Функція `handleOnIdle` відповідає за обробку події неактивності користувача та виконання відповідних дій, таких як видалення токена та переадресація на головну сторінку.

Фрагмент коду:

```

const handleOnIdle = () => {
  EventBus.on("logout", (data) => {

    localStorage.removeItem("clientId");
    localStorage.removeItem("token");

    setAuthenticated(false);
    navigate("/");
  });
};

```

Рис.2.10 `handleOnIdle`

Хуки `useMemo` та `useEffect` використовуються для запуску та завершення прогресу завантаження сторінки.

`UseEffect` також викликає функцію `handleOnIdle` для налаштування обробника подій неактивності.

Фрагмент коду:

```
useMemo(() => {  
  nprogress.start();  
}, [location]);  
  
useEffect(() => {  
  nprogress.done();  
}, [location]);  
  
useEffect(() => {  
  handleOnIdle();  
}, []);
```

Рис.2.11 Контроль прогресу сторінки

Відповідно основний компонент рендерить інтерфейс та відповідає за весь життєвий цикл додатка, від автентифікації користувача до відображення відповідних сторінок залежно від його ролі.

AppContext.Provider передає стан автентифікації та інформацію про користувача всім дочірнім компонентам.

MenuAppBar, HelmetProvider, Suspense, і Footer забезпечують навігацію, управління метаданими, асинхронний рендеринг та відображення футера відповідно.

Фрагмент коду:

```
return (  
  <AppContext.Provider  
    value={{  
      authenticated,  
      setAuthenticated,  
      user: getUserInfo()  
    }}  
  >  
    <StyledEngineProvider injectFirst>  
      <MenuAppBar />  
    </StyledEngineProvider>  
    <HelmetProvider>  
      <div className="container" style={{ width: "100%" }}>  
        <Suspense fallback={<CircularProgress />}>  
          <Routes>  
            {authRouteRender()}  
            <Route path="*" element={<NotFoundPage />} />  
          </Routes>  
        </Suspense>  
      </div>  
    </HelmetProvider>  
    <Footer />  
  </AppContext.Provider>  
  
export default App;
```

Рис.2.12 Return App.js

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						40
Змн.	Арк.	№ докум.	Підпис	Дата		

Файли для управління доступами.

Файли `rbac-consts.js` та `rbac-rules.js` використовуються для визначення ролей і правил доступу в додатку. Вони допомагають забезпечити, що різні типи користувачів мають доступ до відповідних функцій додатку.

Лістинг коду `rbac-rules.js`:

```
import { toolbar, bookHotelRoom, hotels, cooperations, hotelRooms } from
"./rbac-consts";

const rules = {
  ROLE_ADMIN: {
    static: [
      toolbar.ADMIN,
      cooperations.ADMIN
    ],
    dynamic: {}
  },

  ROLE_USER: {
    static: [
      toolbar.USER,
      bookHotelRoom.USER,
      cooperations.USER,
      hotelRooms.USER
    ],
    dynamic: {}
  },

  ROLE_OWNER: {
    static: [
      toolbar.OWNER,
      hotels.OWNER,
      bookHotelRoom.OWNER,
      cooperations.OWNER,
      hotelRooms.OWNER
    ],
    dynamic: {}
  },
};

export default rules;
```

Рис.2.13 Зміст `rbac-rules.js`

Для збереження стану авторизації використовуються JWT токени, які зберігаються в локальному сховищі браузера.

Використання кукі для зберігання токенів може забезпечити додаткову безпеку завдяки опціям `HttpOnly` та `Secure`. Однак, кукі можуть бути обмежені в розмірі та складнішими у використанні через потребу в налаштуванні та передачі з кожним HTTP-запитом.

`SessionStorage` зберігає дані лише в межах однієї сесії браузера. Це означає, що дані будуть втрачені після закриття вкладки або браузера.

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						41
Змн.	Арк.	№ докум.	Підпис	Дата		

Використання `sessionStorage` може бути корисним для збереження тимчасових даних, які не потрібні після завершення сесії.

Було обрано зберігання токенів у `localStorage`, що має декілька переваг, особливо те, що токени в `localStorage` зберігаються навіть після закриття вкладки чи браузера. Це забезпечує зручність для користувачів, оскільки їм не потрібно повторно вводити свої облікові дані щоразу, коли вони повертаються на платформу. Також робота з `localStorage` є досить простою та інтуїтивно зрозумілою, що полегшує реалізацію зберігання та отримання даних. Це особливо корисно при розробці масштабованих додатків, де важливо мати швидкий та легкий доступ до збережених даних. `LocalStorage` підтримується усіма сучасними браузерами, що забезпечує стабільну роботу додатку на різних платформах і пристроях.

Приклад запису токена:

Лістинг `storage.js`

```
const TOKEN = "token";

const storageGetItem = (key) => {
  return localStorage.getItem(key);
};

const storageSetItem = (key, value) => {
  if (key === TOKEN) {
    sessionStorage.clear();
  }
  return localStorage.setItem(key, value);
};

const storageKeyExists = (key) => {
  return storageGetItem(key) !== "";
};

export { TOKEN, storageKeyExists, storageGetItem, storageSetItem };
```

Рис.2.14 Збереження токена

Отже, зберігання токенів у `localStorage` є оптимальним вибором, оскільки це забезпечує безперервність авторизації для користувачів, навіть якщо вони закривають браузер або перезавантажують комп'ютер

Приклад коду для збереження та отримання токена:

Лістинг `loginRequest.js`

```
const loginRequest = (authData, onSuccess = null, onError = null, onFinally = null) => {
  axios.post(`${ApiUrl}/api/login-check`, authData).then(response => {
```

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						42
Змн.	Арк.	№ докум.	Підпис	Дата		

```

if (response.status === responseStatus.HTTP_OK && response.data.token) {
    storageSetItem(TOKEN, response.data.token);
    if (onSuccess) {
        onSuccess();
    }
}
}).catch(error => {
    if (onError) {
        onError(error.response.data.message);
    }
}).finally(() => {
    if (onFinally) {
        onFinally();
    }
});
};

```

Рис.2.15 Логін із збереженням токена

Що стосується збереження облікових даних користувача, таких як username, то було вирішено зберігати їх у sessionStorage. Це зумовлено тим, що ці дані є більш чутливими і відповідно потрібно обмежити їх зберігання лише в межах однієї сесії браузера. Збереження username у sessionStorage дозволяє забезпечити додатковий рівень безпеки, оскільки ці дані будуть автоматично видалені після закриття вкладки або браузера, зменшуючи ризик їх компрометації.

Таким чином, комбінація використання localStorage для збереження токенів та sessionStorage для збереження username забезпечує оптимальний баланс між зручністю для користувачів та безпекою даних.

Висновки до другого розділу

У розділі 2 «Проектування онлайн-платформи бронювання готелів» було ретельно розглянуто всі аспекти, необхідні для успішного проектування та реалізації системи бронювання готелів. Описано варіанти використання системи, розроблено структуру бази даних, розроблено та реалізовано алгоритми роботи платформи, а також реалізовано функціонал платформи.

Починаючи з опису бізнес-вимог і вимог користувачів, було визначено основні потреби і очікування від платформи бронювання готелів. На основі цього були сформульовані функціональні і нефункціональні вимоги, які визначали обсяг та характеристики системи.

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						43
Змн.	Арк.	№ докум.	Підпис	Дата		

Схема варіантів використання платформи надає загальне уявлення про те, як користувачі будуть взаємодіяти з системою та які операції вони можуть виконувати. Це допомагає зрозуміти основні потреби користувачів та визначити необхідний функціонал для забезпечення їхніх потреб.

Діаграми сутностей класів, розгортання платформи та класів надають детальний огляд структури системи та взаємозв'язків між її компонентами. Вони допомагають зрозуміти, як система буде організована і які процеси будуть в ній відбуватися.

Опис сервісів платформи вказує на основні функціональні складові системи та їхні взаємозв'язки. Це допомагає уникнути непорозумінь і забезпечує зручну архітектуру для подальшої розробки.

Діаграми дій системи про обрання номеру та про бронювання дозволяють відстежувати послідовність подій та процесів, які відбуваються під час користування платформою. Це важливо для забезпечення ефективної роботи системи та задоволення потреб користувачів.

Отже, в цілому, розділ 2 надає повний і глибокий огляд проектування та реалізації платформи бронювання готелів, що дозволяє зрозуміти всі аспекти і складові цієї системи.

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						44
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З ОНЛАЙН-ПЛАТФОРМОЮ ДЛЯ БРОНЮВАННЯ ГОТЕЛІВ

3.1 Користувацький інтерфейс

Онлайн-платформа для бронювання готелів має інтуїтивно зрозумілий та зручний користувацький інтерфейс, який дозволяє користувачам швидко та легко знайти та забронювати потрібний готель. Зайти на сторінку easystaychoice користувачі можуть як з комп'ютера, так і з мобільного телефону чи інших пристроїв. Платформа є повністю адаптивною для стандартних пристроїв.

Головна сторінка платформи є стартовою точкою для користувачів. Вона містить пошукову форму, популярні готелі за оцінкою користувачів, спеціальні пропозиції та популярні напрямки.

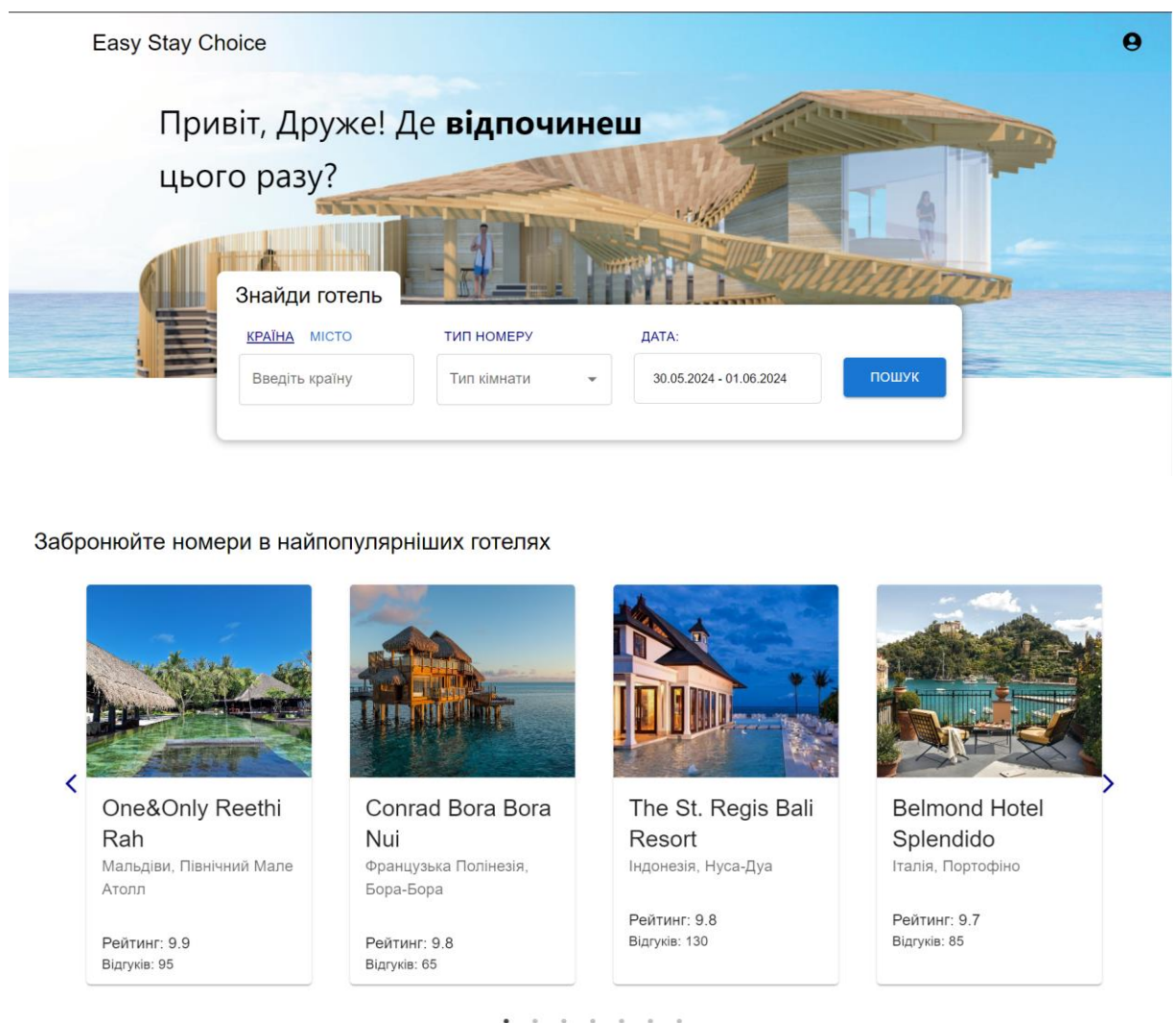


Рис.3.1 Головна сторінка платформи

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						45
Змн.	Арк.	№ докум.	Підпис	Дата		

Пошукова форма (рис.3.2) дозволяє користувачам ввести основні параметри пошуку, такі як місце призначення, дати проживання та тип бажаного номеру. Після натискання кнопки «Пошук» користувач перенаправляється на сторінку результатів пошуку (рис.3.3). Дана форма доступна на будь-якій сторінці для зручності.

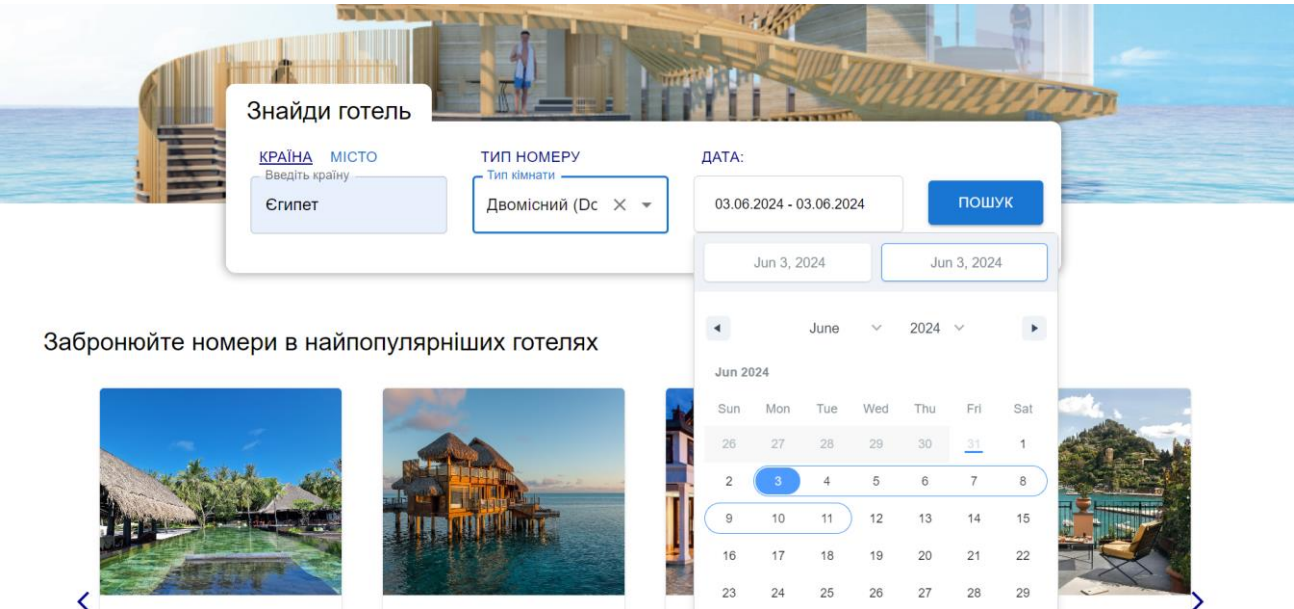


Рис.3.2 Пошукова форма

На сторінці результатів пошуку відображаються всі номери, що відповідають критеріям пошуку. Користувачі можуть фільтрувати результати за різними параметрами, такими як ціна, рейтинг, відстань до центру міста тощо.

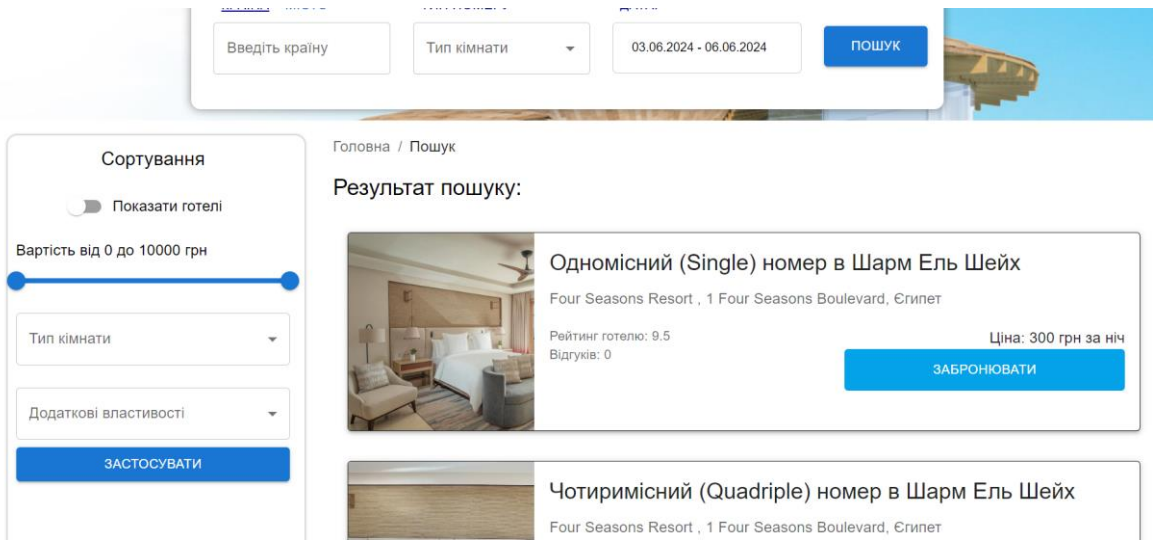


Рис.3.3 Сторінка результату пошуку за фільтрами для номерів

Окрім цього, користувач може обрати показ усіх готелів, щоб дізнатися більш детальну інформацію, чи обрати номер лише в конкретному готелі (рис.3.4).

КРАЇНА МІСТО
Введіть країну
Єгипет

ТИП НОМЕРУ
Тип кімнати

ДАТА:
03.06.2024 - 06.06.2024

ПОШУК

Сортування
Показати готелі

Вартість від 0 до 10000 грн

Тип кімнати

Додаткові властивості

ЗАСТОСУВАТИ

Головна / Пошук

Результат пошуку:

Four Seasons Resort
Шарм Ель Шейх, Єгипет
1 Four Seasons Boulevard
Відгуків: 0
Оцінка: 9.5

ПЕРЕГЛЯНУТИ ГОТЕЛЬ

Hilton Sharm Waterfalls Resort
Шарм Ель Шейх, Єгипет
Оцінка: 8.7

Рис.3.4 Сторінка результату пошуку за фільтрами для номерів

Сторінка готелю (рис. 3.5) надає детальну інформацію про вибраний готель, включаючи фото, опис, список зручностей, відгуки гостей. Користувач може перейти за посиланням назад до сторінки результату пошуку, проте окремо від попередніх критеріїв пошуку, буде додано ще й критерій по готелю. Надалі користувач може обрати номер та здійснити бронювання.

КРАЇНА МІСТО
Введіть країну
Єгипет

ТИП НОМЕРУ
Тип кімнати

ДАТА:
03.06.2024 - 06.06.2024

ПОШУК

Сортування
Показати готелі

Вартість від 0 до 10000 грн

Тип кімнати

Додаткові властивості

ЗАСТОСУВАТИ

Головна / Пошук

Результат пошуку:

Four Seasons Resort
Шарм Ель Шейх, Єгипет
1 Four Seasons Boulevard
Відгуків: 0
Оцінка: 9.5

ПЕРЕГЛЯНУТИ ГОТЕЛЬ

Hilton Sharm Waterfalls Resort
Шарм Ель Шейх, Єгипет
Оцінка: 8.7

Рис.3.5 Сторінка результату пошуку за фільтрами для готелів

Сторінка готелю (рис.3.6) надає детальну інформацію про вибраний готель, включаючи фото, опис, список зручностей, відгуки гостей. Користувач може перейти за посиланням назад до сторінки результату пошуку, проте окремо від попередніх критеріїв пошуку, буде додано ще й критерій по готелю. Надалі користувач може обрати номер та здійснити бронювання.

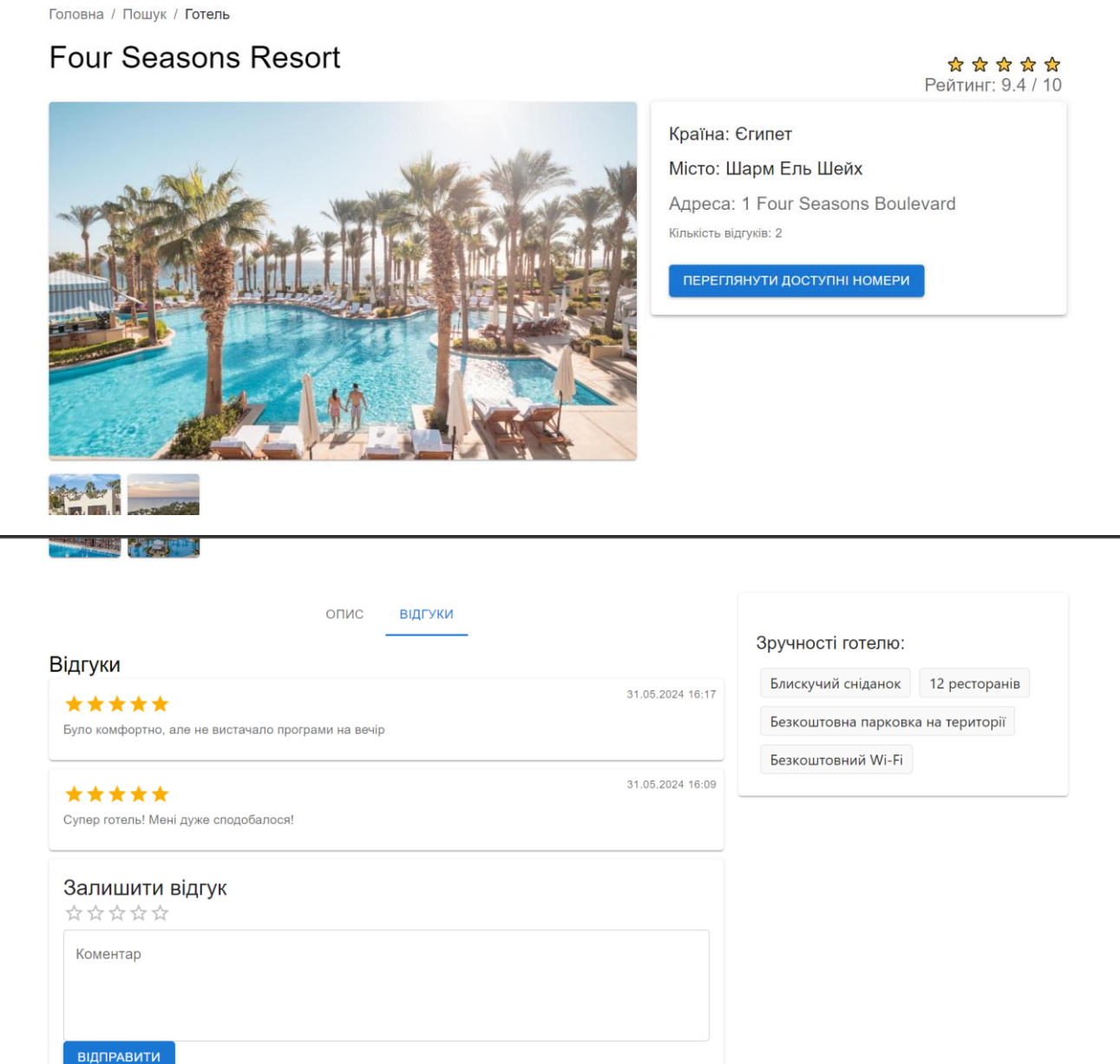


Рис.3.6 Сторінка готелю

Сторінка номеру (рис.3.7) надає детальну інформацію про вибраний номер, включаючи фото, опис, список зручностей, додаткову класифікацію, відгуки гостей.

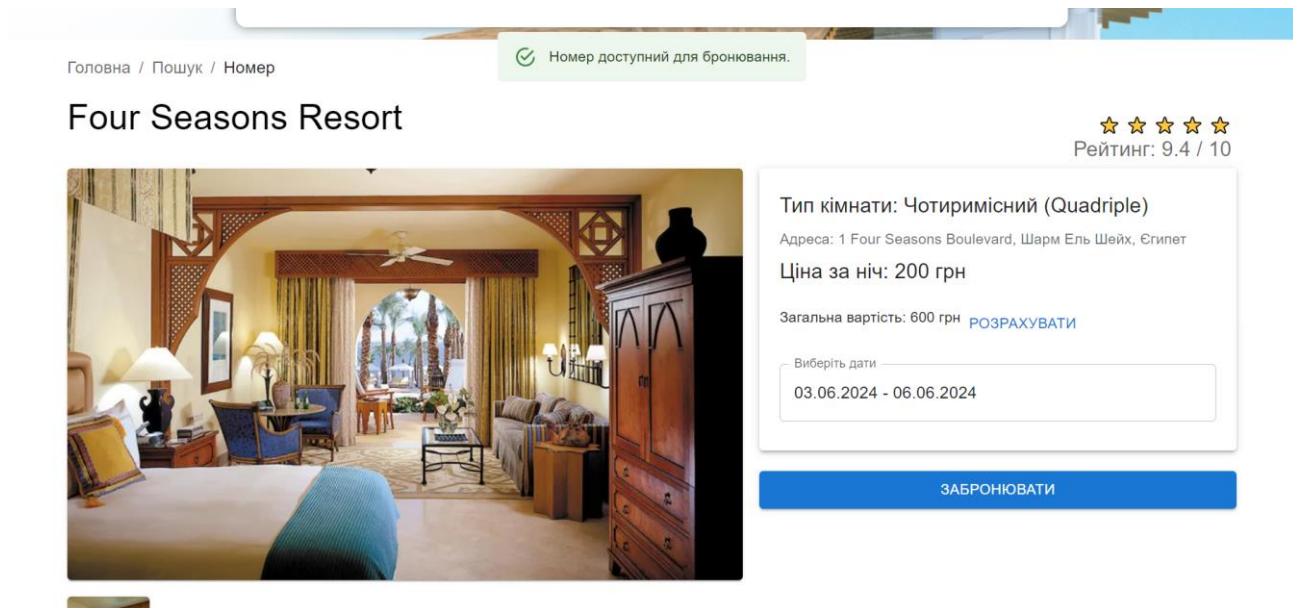


Рис.3.7 Сторінка номеру

В залежності від дати йде перевірка чи доступний цей номер на вказаний період. Якщо перевірка пройдена, користувач може забронювати номер та скористатися «програмою лояльності», тобто використати бонуси, за наявності.

Користувач може перейти за посиланням назад до сторінки результату пошуку, проте окремо від попередніх критеріїв пошуку, буде додано ще й критерій по готелю. Надалі користувач може обрати номер та здійснити бронювання.

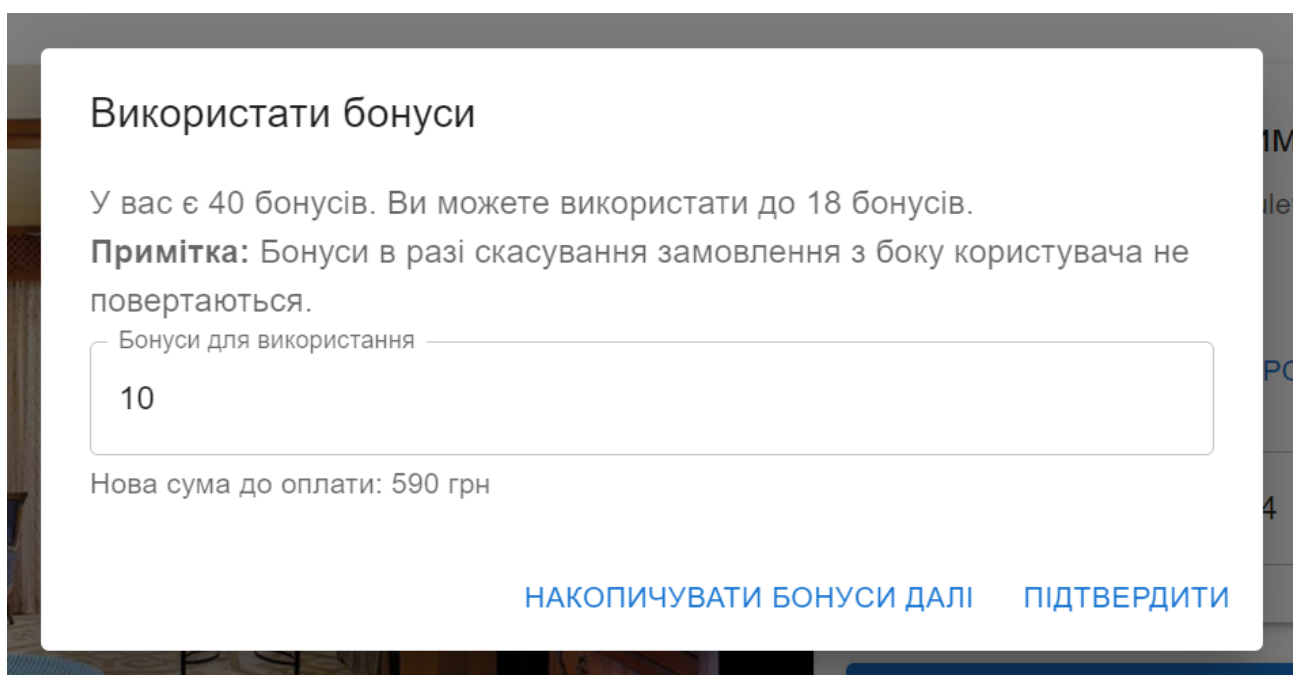


Рис.3.8 Етап бронювання номеру

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						49
Змн.	Арк.	№ докум.	Підпис	Дата		

У шапці сторінки можна побачити кнопку входу до особистого кабінету (рис.3.9).

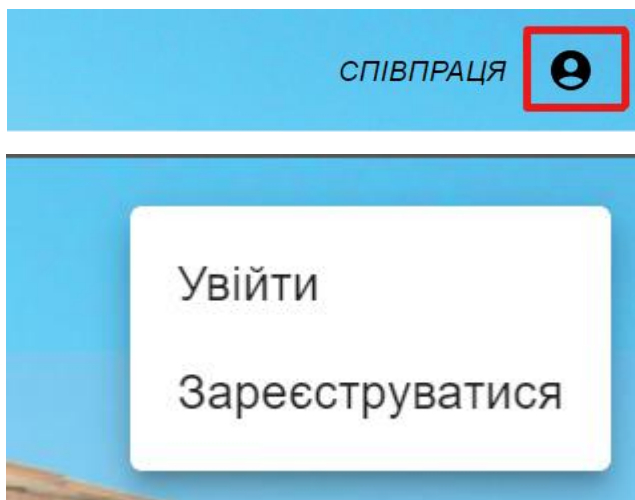


Рис.3.9 Вхід до особистого кабінету

Відповідно користувач потрапляє на сторінку входу (рис.3.10) чи реєстрації, якщо користувач вперше на платформі (рис.3.11).

Easy Stay Choice 9

Увійти!

E-mail *

Пароль 👁

УВІЙТИ

[Немає профілю? Зареєструватися](#)

Про нас
 Ми платформа для бронювання готелів, яка допомагає вам знайти найкращі пропозиції по всьому світу.

Співпраця
 Ви власник готелю? [Увійдіть в свій акаунт користувача](#) та зареєструйте свій готель на нашій платформі, отримуйте більше бронювань.

Клієнтська підтримка
[Часті питання](#)
[Контакти](#)
[Політика конфіденційності](#)
[Умови використання](#)

Дізнавайся першим!
 Підпишіться на нашу розсилку та отримуйте останні новини і спеціальні пропозиції.

Рис.3.10 Сторінка входу

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						50
Змн.	Арк.	№ докум.	Підпис	Дата		

Створити профіль

Ім'я *

Прізвище *

E-mail *



+380

Пароль



ЗАРЕЄСТРУВАТИСЯ

[Вже є профіль? Увійти](#)

Рис.3.11 Сторінка реєстрації

Якщо користувач намагався здійснити бронювання номеру, без входу в профіль, його автоматично перекине на сторінку входу, при цьому його попередній шлях буде збережено, а отже по завершенні входу, перенаправлено знову на сторінку бронювання.

Особистий кабінет користувача (рис.3.12) дозволяє переглядати та керувати всіма бронюваннями. Користувачі можуть змінювати особисті дані, переглядати історію бронювань, їхні статуси, а також скасувати наявні бронювання (рис.3.13), якщо вони ще не були завершені (рис.3.14).

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						51
Змн.	Арк.	№ докум.	Підпис	Дата		

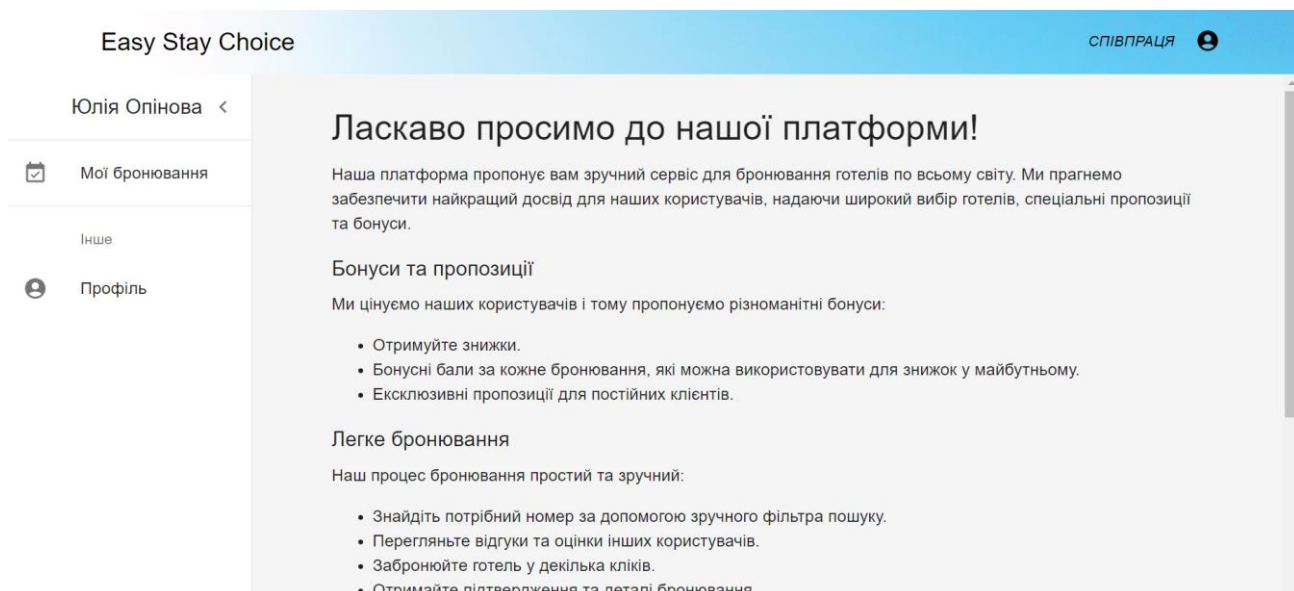


Рис.3.12 Особистий кабінет звичайного користувача

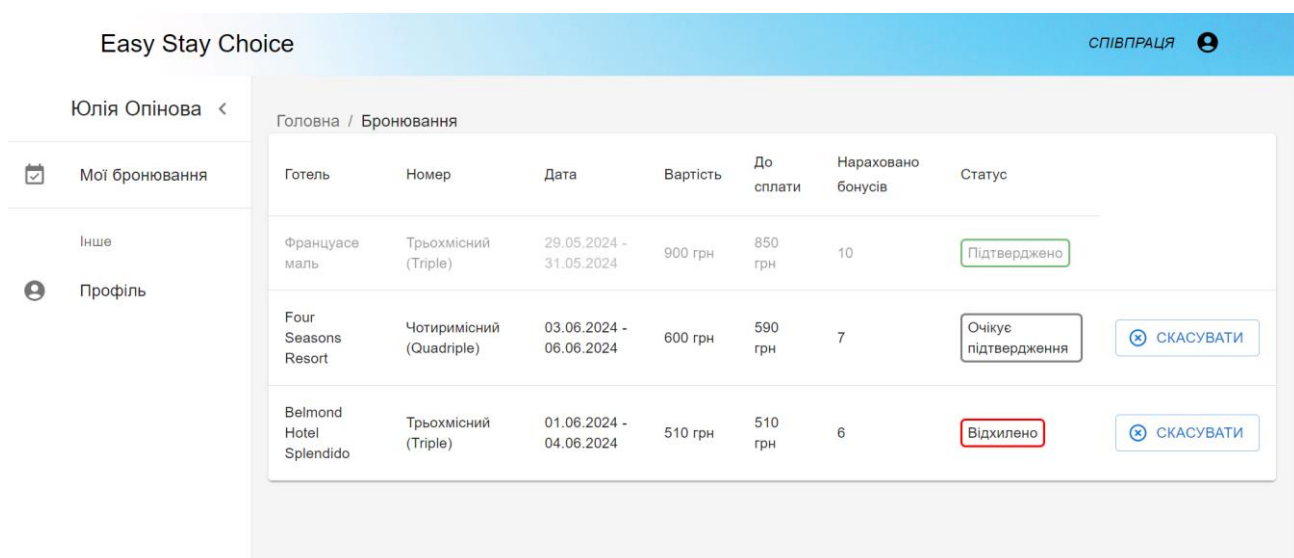


Рис.3.13 Перегляд бронювань

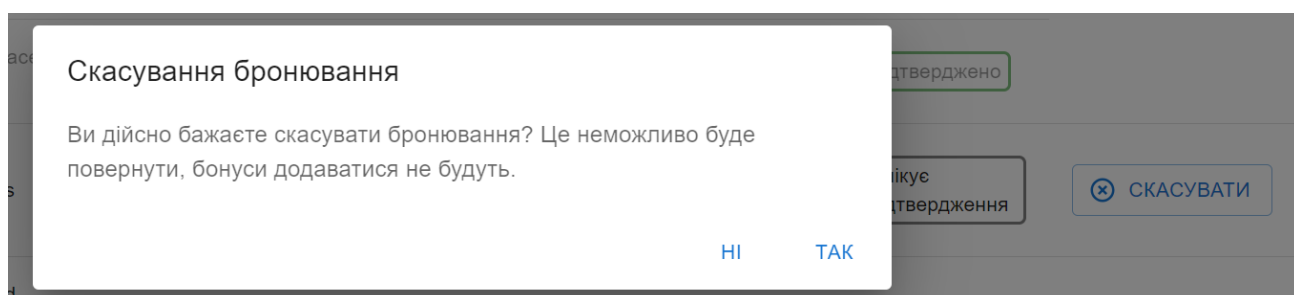


Рис.3.14 Скасування бронювання

Для показу повідомлення про успішне бронювання, скасування помилок і тому подібних, використовуються повідомлення з відповідною інформацією (рис.3.15).

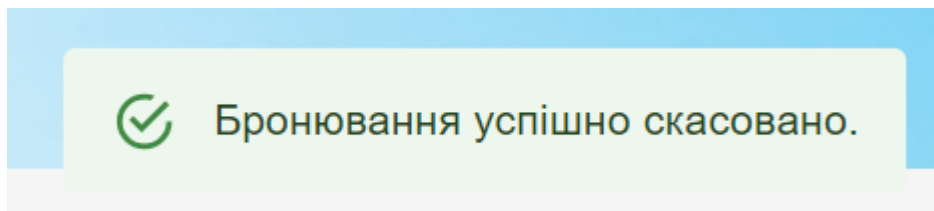


Рис.3.15 Повідомлення про виконання дії/запиту

Платформа також містить розділи з інформацією про бонуси, спеціальні пропозиції, контакти та відповіді на поширені запитання (FAQ). Це допомагає користувачам краще орієнтуватися у можливостях платформи та отримувати максимальну вигоду від її використання. Через підвал сторінки можна потрапити на будь-який із цих розділів (рис.3.16).

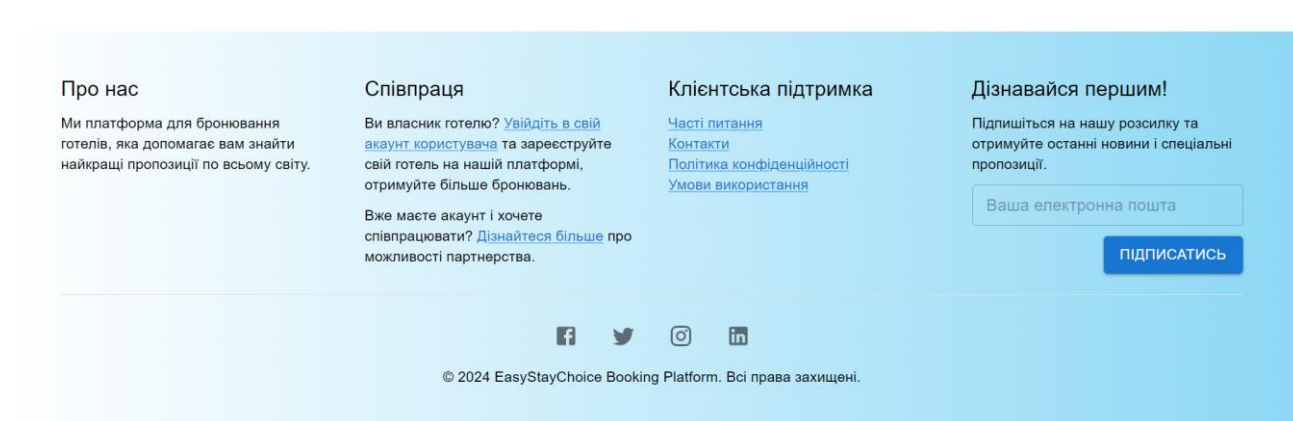


Рис.3.16 Підвал платформи

3.2 Інтерфейс для власника готелю та адміністратора платформи

У додатку присутній розподіл користувачів за ролями. Відповідно основним є адміністратор платформи. В особистому кабінеті такий користувач має доступ до усіх заявок на співпрацю (рис.3.17). Відповідно заявку можна погодити чи відхилити. При відхиленні пропонується відправити листа на пошту власника заявки із вказівками, чому було відхилено та що потрібно змінити, щоб заявку прийняли.

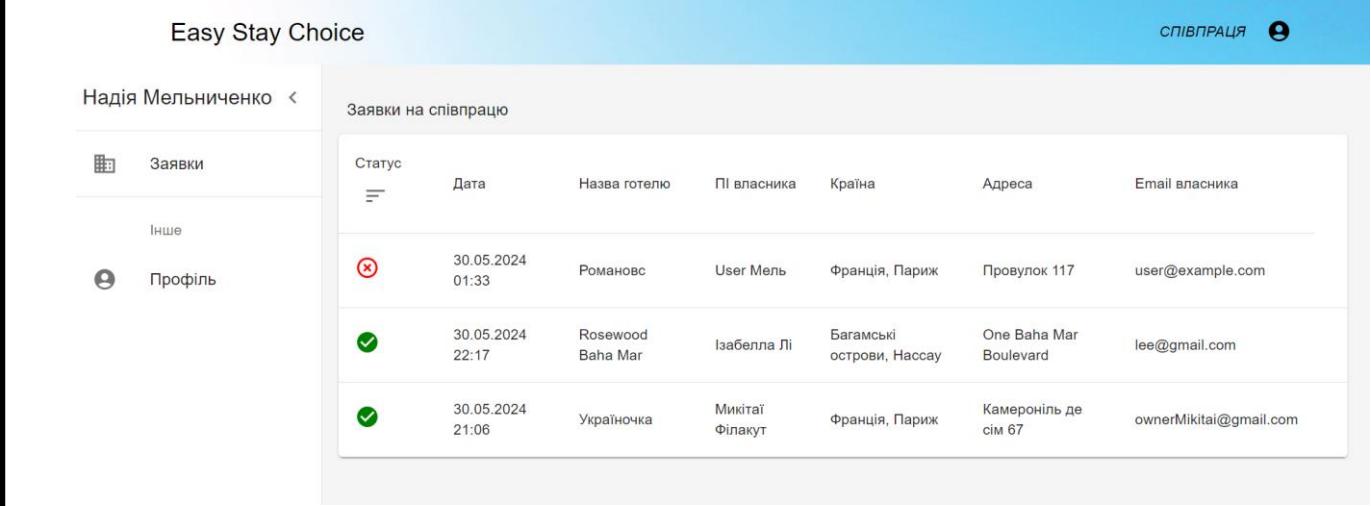


Рис.3.17 Адмінпанель. Заявки на співпрацю

Відповідно кожний користувач може подати заявку на співпрацю, надіславши документи на пошту та заповнивши анкету (рис.3.18).

Зареєструвати власника

Будь ласка, заповніть форму нижче, щоб зареєструвати свій готель.

Надайте точну інформацію, щоб забезпечити безпроблемну реєстрацію.

Назва готелю:

Назва готелю *

Аметист

Кrajна:

Країна *

Місто:

Місто *

Адреса:

Адреса *

Ім'я власника:

Ім'я *

Юлія

Прізвище власника:

Прізвище *

Опінова

Робоча пошта власника:

Email

ulia@gmail.com

Перевірте ваш email. Якщо це не робоча пошта, спочатку зареєструйте необхідну.

Будь ласка, надішліть документи для підтвердження на пошту: easystaychoice.com

Тема листа: Аметист (Юлія Опінова)

☐ Я надіслав документи

НАДІСЛАТИ

Рис.3.18 Анкета заявки на співпрацю

Щойно адміністратор погоджує заявку, роль користувача змінюється, з'являються відповідні права доступу та більш вдосконалений кабінет користувача, налаштований під потреби власника.

Аналогічно до інтерфейсу користувача, у власника є сторінка бронювання, де він може підтвердити чи відхилити (рис.3.19).

Easy Stay Choice

Олег Укків

Мої готелі

Бронювання

Інше

Профіль

Головна / Бронювання

Статус	Дата створення	Готель	Номер	Дата	Повна вартість	До сплати	Завершено	Дії
✓	31.05.2024 14:52	Hotel Martinez	Двомісний (Double)	11.06.2024 - 14.06.2024	300 грн	300 грн	ЗАВЕРШИТИ	
✓	31.05.2024 14:53	Hotel Martinez	Двомісний (Double)	11.06.2024 - 14.06.2024	300 грн	300 грн	ЗАВЕРШИТИ	
✗	31.05.2024 14:53	Hotel Martinez	Двомісний (Double)	05.06.2024 - 10.06.2024	500 грн	500 грн		
⚠	31.05.2024 14:53	Hotel Martinez	Двомісний (Double)	03.06.2024 - 11.06.2024	800 грн	800 грн	ЗАВЕРШИТИ	✓ ПІДТВЕРДИТИ ВІДХИЛИТИ

Рис.3.19 Список бронювань

Додавати готель може лише адміністратор відповідно до наданих документів, проте редагувати, додавати кімнати, приховувати на платформі може власник (рис.3.20 – рис.3.22).

Easy Stay Choice

Олег Укків

Мої готелі

Бронювання

Інше

Профіль

Головна / Готелі

Назва готелю

Назва готелю	Дата реєстрації	Опубліковано	Розташування	НОМЕРИ	
Le Grand Hotel Cannes	30.05.2024	Так	45 Boulevard de la Croisette (Франція, Канни)	НОМЕРИ	✎
Hotel Martinez	30.05.2024	Так	73 La Croisette (Франція, Канни)	НОМЕРИ	✎
Французе малъ	21.05.2024	Ні	Провулок 46 (Франція, Марсель)	НОМЕРИ	✎

Рис.3.20 Список зареєстрованих готелів власника

Олег Укків <

Мої готелі

Бронювання

Інше

Профіль

Головна / Готелі / 27

Редагувати готель

Назва

Le Grand Hotel Cannes

Опис

Le Grand Hotel Cannes пропонує гостям елегантні номери з видом на Середземне море, приватний пляж, ресторан т

Адрес

45 Boulevard de la Croisette (Франція, Канни)

ОБРАТИ ФАЙЛ

ЗАВАНТАЖИТИ ФОТО

Дата створення

30.05.2024

ЗНЯТИ З ПУБЛІКАЦІЇ

ЗБЕРЕГТИ ЗМІНИ

Рис.3.21 Сторінка редагування готелю

Easy Stay Choice

Олег Укків <

Мої готелі

Бронювання

Інше

Профіль

Головна / Готелі / Номери / Редагувати

Редагувати номер

Тип кімнати:

Двомісний (Double)

Ціна за ніч:

130

Кількість таких номерів в готелі:

6

Опис:

Стандартний двомісний номер з окремою ванною. Ідеально підходить для сімейного відпочинку.

ОБРАТИ ФАЙЛ

ЗАВАНТАЖИТИ ФОТО

Рис.3.22 Сторінка редагування номеру готелю

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		56

3.3 REST API

REST API [14] забезпечує зв'язок між фронтенд та бекенд частинами веб-платформи. REST API розроблений на основі принципів REST, що дозволяє забезпечити простоту, масштабованість та ефективність системи.

Основні ендпоінти системи:

1. Авторизація та реєстрація (**POST**/api/users)

Для реєстрації нового користувача використовуються наступні параметри: email, пароль, ім'я користувача, прізвище та номер телефону.

Приклад запиту:

```
{
  "email": "user@example.com",
  "password": "stringst",
  "name": "string",
  "surname": "string"
}
```

Рис.3.23 Запит на реєстрацію користувача

Для авторизації використовується запит **POST**/api/login-check, що на основі введених даних генерує токен для користувача.

Приклад запиту:

```
{
  "email": "string",
  "password": "string"
}
```

Рис.3.24 Запит на вхід до кабінету

Приклад відповіді:

```
{
  "token": "string"
}
```

Рис.3.25 Запит на вхід до кабінету

2. Реєстрація готелю.

Даний запит здійснюється після отримання заявки (**GET**/api/cooperations) на співпрацю від користувача, а саме під час підтвердження співпраці адміністратором.

Приклад запиту (**POST**/api/hotels):

```
{
```

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						57
Змн.	Арк.	№ докум.	Підпис	Дата		

```

"name": "string",
"description": "string",
"address": "string"
}

```

Рис.3.26 Запит на додавання готелю

Під час цього запиту за допомогою `CreateHotelAction` автоматично додається ід користувача відповідно до вказаного емейлу в заявці, статус не опубліковано та нульові значення для інших полів, які зможе доповнити власник вже зі свого кабінету (`PUT/api/hotels/{id}`).

Аналогічно додається номери до готелів та оновлюються, використовуючи `api/hotelRooms (/ {id})`.

Не менш важливим є запит отримання готелів та номерів згідно з даними, що користувач зазначив як критерії пошуку. Наприклад: `GET/api/find-hotel-rooms?country=Єгипет&priceFrom=0&priceTo=10000`

Основний запит платформи стосується бронювання номеру. Для цього використовується запит `POST/api/book-room`, що обробляється за допомогою спеціального контроллера, щоб врахувати ід користувача, кімнати та сформувати автозаповнені поля. Відповідно для керування статусом замовлення використовується звичайний `PUT/api/bookings/{id}` запит, запропонований `api platform`.

Керування даними для отримання переліку бронювань здійснюється персональними запитами `GET/api/owner-booking` та `GET/api/user-booking`, де об'єкти формуються за належністю номерів до готелів власника чи ід користувача відповідно.

Приклад запиту:

```

{
  "hotelRoom": 3,
  "checkInDate": "2024-05-10T12:30:32.172Z",
  "checkOutDate": "2024-05-10T12:30:32.172Z",
  "fullPrice" : "1332",
  "paidPrice" : "1300"
}

```

Рис.3.27 Запит на отримання переліку бронювань на об'єкти власника

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						58
Змн.	Арк.	№ докум.	Підпис	Дата		

В загальному, REST API платформи для бронювання готелів враховує широкий спектр функцій для взаємодії з користувачами та власниками готелів. Вищенаведені ендпоінти є основними для реалізації критичних функцій системи. Більшість з усіх інших використовуються для здійснення валідацій, отримання додаткових даних, таких як тип кімнат для номеру, списку готелів за конкретними даними (наприклад рейтингом) чи завантаження зображення готелів/номерів.

3.4 Особливості бронювання номеру

Система бронювання номерів є ключовою функціональністю платформи, що забезпечує зручність та ефективність процесу для користувачів та власників готелів. Тому було детально розроблено особливості реалізації бронювання номерів, зокрема перевірка доступності номерів на обрані дати.

Перед тим, як користувач може забронювати номер, система повинна перевірити, чи доступний цей номер на обрані дати. Ця перевірка включає кілька кроків:

1. Отримання даних про номер та обрані дати в'їзду/виїзду.
2. Перевірка наявності номеру в базі даних.
3. Підрахунок кількості вже заброньованих номерів на вказані дати (так як в готелі можуть бути декілька однакових за функціональністю номерів і відповідно можуть бути вільно декілька номерів на однакові дати.
4. Порівняння кількості заброньованих номерів з доступними номерами.

Нижче наведено приклад реалізації цієї перевірки на серверній стороні:

```
#[Route('/api/check-booking-availability', name: 'check_booking_availability',
methods: ['POST'])]
public function checkBookingAvailability(Request $request, BookingRepository
$bookingRepository, HotelRoomRepository $hotelRoomRepository): JsonResponse
{
    $data = json_decode($request->getContent(), true);
    $hotelRoomId = $data['hotelRoomId'];
    $startDate = new \DateTime($data['checkInDate']);
    $endDate = new \DateTime($data['checkOutDate']);

    $hotelRoom = $hotelRoomRepository->find($hotelRoomId);
    if (!$hotelRoom) {
        return new JsonResponse([
```

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						59
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        'available' => false,
        'message'   => 'Номер не знайдено.'
    ));
}

$totalRooms = $hotelRoom->getAmount();

$bookings = $bookingRepository->findNotConfirmedBookings($hotelRoomId);

$bookedCount = 0;

foreach ($bookings as $booking) {
    $bookingStartDate = $booking->getCheckInDate();
    $bookingEndDate = $booking->getCheckOutDate();

    $startDateFormat = $startDate->format('Y-m-d');
    $endDateFormat = $endDate->format('Y-m-d');
    $bookingStartDateFormat = $bookingStartDate->format('Y-m-d');
    $bookingEndDateFormat = $bookingEndDate->format('Y-m-d');

    if (!($startDateFormat >= $bookingEndDateFormat) && !($endDateFormat <=
    $bookingStartDateFormat)) {
        $bookedCount++;
    }
}

if ($bookedCount >= $totalRooms) {
    return new JsonResponse([
        'available' => false,
        'message'   => 'На вказані дати всі номери зайняті.'
    ]);
}

return new JsonResponse([
    'available' => true,
    'message'   => 'Номер доступний для бронювання.'
]);
}

```

Рис.3.28 Лістинг коду для перевірки на доступність

Опис процесу:

- Отримання даних:** Отримання даних з запиту користувача, включаючи ID номеру та дати перебування.
- Перевірка наявності номеру:** Перевірка, чи існує номер з вказаним ID у базі даних.

Перевірка доступності номеру включає порівняння обраних дат з уже існуючими бронюваннями. Основна мета полягає в тому, щоб визначити, чи перетинаються обрані дати з будь-якими іншими бронюваннями. Ця перевірка визначає, чи перетинаються обрані дати з іншими бронюваннями, за наступним алгоритмом:

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						60
Змн.	Арк.	№ докум.	Підпис	Дата		

- **Форматування дат:** Дати обраного бронювання та існуючих бронювань форматуються до виду Y-m-d для порівняння.

- **Порівняння дат:** Якщо обрана дата початку бронювання (\$startDateFormat) більша або дорівнює даті завершення існуючого бронювання (\$bookingEndDateFormat), або дата завершення обраного бронювання (\$endDateFormat) менша або дорівнює даті початку існуючого бронювання (\$bookingStartDateFormat), то ці бронювання не перетинаються.

- **Врахування перетинів:** Якщо перетину не відбувається, збільшується лічильник заброньованих номерів (\$bookedCount)

3. **Підрахунок бронювань:** Визначення кількості існуючих бронювань, які перетинаються з обраними датами.

4. **Повернення результату:** Якщо кількість заброньованих номерів не перевищує доступну кількість, номер вважається доступним для бронювання.

Для інтеграції перевірки доступності на фронтенді використано функцію, що робить запит до відповідного API ендпоінту:

```
const checkBookingAvailability = (startDate, endDate) => {
  if (!startDate || !endDate) {
    return;
  }

  setLoading(true);
  axios.post(`${ApiUrl}/api/check-booking-availability`, {
    hotelRoomId: params.hotelRoomId,
    checkInDate: startDate,
    checkOutDate: endDate
  }).then(response => {
    setBookingAvailable(response.data.available);
    setNotification({
      visible: true,
      type: response.data.available ? "success" : "error",
      message: response.data.message
    });
  }).catch(error => {
    setNotification({
      visible: true,
      type: "error",
      message: "Помилка перевірки доступності."
    });
  }).finally(() => {
    setLoading(false);
  });
};
```

Рис.3.29 Лістинг коду виклику контроллера для перевірки

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						61
Змн.	Арк.	№ докум.	Підпис	Дата		

Також важливим є реалізація `CreatHotelAction`, що відповідає за створення нового готелю в системі. Цей клас є частиною API, реалізованого за допомогою `Symfony`.

Метод `__invoke` обробляє запити на створення нового готелю. Відбувається пошук локації по ід, пошук користувача за електронною адресою, перевірка чи має користувач роль власника готелю. Якщо ні, роль додається.

Цей клас забезпечує основну логіку для створення готелю, включаючи валідацію даних та асоціацію готелю з власником і локацією [додаток Б].

Клас `UploadController` відповідає за завантаження фотографій готелів та номерів у систему [додаток В]. Він є частиною API, реалізованого за допомогою `Symfony`, і обробляє POST-запити на завантаження файлів.

Файли завантажуються за допомогою об'єкта `Request`. Всі файли з запиту витягуються методом `$request->files->all()`. Якщо файли не завантажені, повертається JSON-відповідь з повідомленням про помилку і статусом 400 (BAD REQUEST). Директорія для збереження файлів визначається за допомогою параметра `uploads_directory`, який зазвичай налаштовується у конфігурації `Symfony`. Кожен завантажений файл обробляється окремо, генерується назва та переміщується у вказану директорію з новим ім'ям.

Після успішного збереження всіх файлів, формується список URL-адрес завантажених файлів.

Висновки до третього розділу

У розділі 3 було детально розглянуто всі аспекти взаємодії користувачів з платформою, включаючи користувацький інтерфейс, інтерфейс для власників готелів та адміністраторів, REST API та особливості бронювання номерів.

Користувацький інтерфейс платформи було ретельно розроблено з урахуванням зручності та естетичності. Користувачам надано зручні інструменти для пошуку готелів за різними критеріями, перегляду детальної інформації про готелі та номери, а також для бронювання номерів.

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						62
Змн.	Арк.	№ докум.	Підпис	Дата		

Інтерфейс для власників готелів та адміністраторів розроблено з урахуванням їхніх потреб у керуванні готелями, номерами та бронюваннями. Вони мають доступ до інструментів для додавання та редагування інформації про готелі, керування доступністю номерів та перегляду бронювань.

REST API платформи надає зручний інтерфейс для взаємодії з платформою з інших програмних систем. Він підтримує всі необхідні операції, такі як отримання списку готелів, перегляд доступних номерів, створення нових бронювань та багато іншого.

Особливості бронювання номерів ретельно розглянуті, включаючи логіку перевірки доступності номерів на певні дати. Код реалізації цієї логіки було показано і детально описано, як саме відбувається процес бронювання та як система контролює доступність номерів.

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						63
Змн.	Арк.	№ докум.	Підпис	Дата		

ВИСНОВКИ

В процесі виконання кваліфікаційної роботи бакалавра було реалізовано веб-платформу для бронювання курортних готелів.

Аналізуючи проблему розробки такої платформи, було встановлено, що наявні аналоги, такі як Booking та Expedia, успішно виконують свої функції, проте, популярна платформа Airbnb не надає можливості бронювання готелів. Основною метою роботи стало створення зручного та ефективного інструменту для бронювання готельних номерів, обходячи традиційних тур-операторів.

Архітектурний вибір упередивши SPA з мікросервісами був обґрунтований бажанням поєднати переваги мікросервісної архітектури з швидкістю та зручністю SPA. Symfony був обраний для забезпечення потужності та ефективності на серверній стороні, а API Platform разом з React використовувалися для створення потужного та дружнього інтерфейсу користувача. Висновок з аналізу варіантів використання, розробки бази даних, а також проектування та реалізації алгоритмів роботи платформи дозволив визначити ключові напрямки для успішної реалізації проекту.

Детальний огляд інтерфейсу та робочого процесу з онлайн-платформою для бронювання готелів у розділі 3 дозволив зрозуміти, яким чином користувачі та власники готелів взаємодіють з системою. Особливу увагу було приділено користувацькому інтерфейсу, адмінпанелі, REST API та особливостям бронювання номерів. Всі ці аспекти були детально проаналізовані та описані, щоб забезпечити зручність та ефективність користування платформою для всіх зацікавлених сторін.

Отже, на основі аналізу та реалізації різних аспектів проекту можна зробити висновок, що розроблена платформа відповідає вимогам та потребам користувачів та власників готелів, забезпечуючи зручність, ефективність та надійність у користуванні.

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						64
Змн.	Арк.	№ докум.	Підпис	Дата		

Література

1. Airbnb / Офіційний сайт [Електронний ресурс] – Режим доступу до ресурсу: <https://www.airbnb.com/>
2. API Platform [Електронний ресурс] – Режим доступу до ресурсу: <https://api-platform.com>
3. Architecture [Електронний ресурс] – Режим доступу до ресурсу: <https://rubygarage.org/blog/monolith-soa-microservices-serverless>
4. Booking.com / Офіційний сайт [Електронний ресурс] – Режим доступу до ресурсу: <https://www.booking.com>
5. CRUD [Електронний ресурс] – Режим доступу до ресурсу: <https://symfony.com/bundles/EasyAdminBundle/current/crud.html>
6. EntityListener [Електронний ресурс] – Режим доступу до ресурсу: <https://symfony.com/doc/current/doctrine.html#entity-listeners>
7. Entities [Електронний ресурс] – Режим доступу до ресурсу: <https://symfony.com/doc/current/doctrine.html#creating-an-entity-class>
8. Entity actions [Електронний ресурс] – Режим доступу до ресурсу: <https://symfony.com/doc/current/doctrine.html#entity-lifecycle-events>
9. Extensions [Електронний ресурс] – Режим доступу до ресурсу: <https://symfony.com/doc/current/templates.html#creating-a-new-template-extension>
10. Expedia.com / Офіційний сайт [Електронний ресурс] – Режим доступу до ресурсу: <https://www.expedia.com/>
11. Nominatim [Електронний ресурс] – Режим доступу до ресурсу: <https://nominatim.openstreetmap.org/ui/search.html>
12. React.JS / Documentation [Електронний ресурс] – Режим доступу до ресурсу: <https://react.dev/>
13. React.JS / Огляд хуків [Електронний ресурс] – Режим доступу до ресурсу: <https://uk.legacy.reactjs.org/docs/hooks-overview.html>

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						65
Змн.	Арк.	№ докум.	Підпис	Дата		

14. REST [Електронний ресурс] – Режим доступу до ресурсу:
https://symfony.com/doc/current/controller/restful_api.html
15. Symfony / Офіційний сайт [Електронний ресурс] – Режим доступу до ресурсу: <https://symfony.com/>

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						66
Змн.	Арк.	№ докум.	Підпис	Дата		

ДОДАТКИ

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						67
Змн.	Арк.	№ докум.	Підпис	Дата		

Розширене порівняння веб-додатків аналогів

Характеристика	Booking	Expedia	Airbnb	Easystaychoice
Тип Сервісу	Онлайн- бронювання готелів та інших видів помешкань	Подорожі та бронювання готелів, авіаквитків, автомобілів	Короткостро кова оренда житла від приватних осіб	Онлайн- бронювання готелів
Інтерфейс Користувача	Простий та легко зрозумілий інтерфейс	Інтуїтивно зрозумілий	Цікавий зрозумілий інтерфейс	Простий та інтуїтивно зрозумілий
Географічні Обмеження	Доступний у всьому світі	Доступний у всьому світі	Доступний у всьому світі, але можливі обмеження в окремих країнах або регіонах	Доступний у всьому світі
Функціональніс ть	Спеціалізація на бронюванні готелів	Широкий спектр послуг, включаючи авіа	Оренда житла від приватних осіб	Спеціалізація на бронювання готелів в курортних містах
Технології	Використання різних технологій для реалізації	Використанн я різних технологій, таких як Java, JavaScript (React), Scala	Використанн я технологій, таких як Ruby on Rails (backend), JavaScript (React) для фронтенду.	Використання php (Symfony) для backend, JavaScript (React) для фронтенду.

Лістинг CreatHotelAction:

<?php

```

namespace App\Action;

use ApiPlatform\Validator\ValidatorInterface;
use App\Entity\Hotel;
use App\Entity\Location;
use App\Entity\User;
use DateTime;
use Doctrine\ORM\EntityManagerInterface;
use Symfony\Component\HttpFoundation\JsonResponse;
use Symfony\Component\HttpFoundation\Request;
use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\Serializer\SerializerInterface;
use Symfony\Contracts\HttpClient\Exception\ClientExceptionInterface;
use Symfony\Contracts\HttpClient\Exception\DecodingExceptionInterface;
use Symfony\Contracts\HttpClient\Exception\RedirectExceptionInterface;
use Symfony\Contracts\HttpClient\Exception\ServerExceptionInterface;
use Symfony\Contracts\HttpClient\Exception\TransportExceptionInterface;

class CreateHotelAction
{
    /**
     * @param EntityManagerInterface $entityManager
     * @param ValidatorInterface $validator
     * @param SerializerInterface $serializer
     */
    public function __construct(
        private EntityManagerInterface $entityManager,
        private ValidatorInterface $validator,
        private SerializerInterface $serializer,
    )
    {
    }

    /**
     * @param Request $request
     * @return Response
     * @throws ClientExceptionInterface
     * @throws DecodingExceptionInterface
     * @throws RedirectExceptionInterface
     * @throws ServerExceptionInterface
     * @throws TransportExceptionInterface
     */
    public function __invoke(Request $request): Response
    {
        $data = json_decode($request->getContent(), true);
    }

```

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						69
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        $locationUrl = $data['location'];
        $locationId = $this->extractLocationId($locationUrl);

        if (!$locationId) {
            return new JsonResponse(['error' => 'Оберіть локацію'],
Response::HTTP_BAD_REQUEST);
        }

        $ownerEmail = $data['ownerEmail'];

        $location = $this->entityManager->getRepository(Location::class)-
>find($locationId);

        $owner = $this->entityManager->getRepository(User::class)-
>findOneBy(['email' => $ownerEmail]);

        if (!$owner) {
            return new Response('Такого користувача не існує',
Response::HTTP_NOT_FOUND);
        }

        if (!in_array('ROLE_OWNER', $owner->getRoles(), true)) {
            $owner->setRoles(['ROLE_OWNER']);
            $this->entityManager->persist($owner);
            $this->entityManager->flush();
        }

        $hotel = $this->serializer->deserialize($request->getContent(),
Hotel::class, 'json');

        $hotel->setOwner($owner);
        $hotel->setLocation($location);

        date_default_timezone_set('Europe/Kiev');
        $date = new DateTime();
        $hotel->setCreatedAt($date);

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

        if ($errors !== null && count($errors) > 0) {
            return new Response((string)$errors, Response::HTTP_BAD_REQUEST);
        }

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

        return new Response('Готель створено', Response::HTTP_CREATED);
    }

    private function extractLocationId(string $locationUrl): ?int

```

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						70
Змн.	Арк.	№ докум.	Підпис	Дата		

```

{
    $parts = explode('/', $locationUrl);
    return is_numeric(end($parts)) ? (int) end($parts) : null;
}

```

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						71
Змн.	Арк.	№ докум.	Підпис	Дата		

Лістинг UpdateController (для завантаження фото готелів та номерів):

```
<?php

namespace App\Controller;

use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
use Symfony\Component\HttpFoundation\Request;
use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\Routing\Annotation\Route;
use Symfony\Component\HttpFoundation\JsonResponse;
use Symfony\Component\HttpFoundation\File\Exception\FileException;

class UploadController extends AbstractController
{
    #[Route('/api/upload', name: 'api_upload', methods: ["POST"])]
    public function upload(Request $request): JsonResponse
    {
        $uploadedFiles = $request->files->all();

        if (empty($uploadedFiles)) {
            return new JsonResponse(['message' => 'No files uploaded'],
Response::HTTP_BAD_REQUEST);
        }

        $uploadsDirectory = $this->getParameter('uploads_directory');
        $uploadedUrls = [];

        foreach ($uploadedFiles as $uploadedFileArray) {
            foreach ($uploadedFileArray as $uploadedFile) {
                try {
                    $originalFilename = pathinfo($uploadedFile-
>getClientOriginalName(), PATHINFO_FILENAME);
                    $newFilename = uniqid() . '.' . $uploadedFile-
>guessExtension();
                    $uploadedFile->move($uploadsDirectory, $newFilename);
                    $uploadedUrls[] = '/uploads/' . $newFilename;
                } catch (FileException $e) {
                    return new JsonResponse(['message' => 'Failed to upload file'],
Response::HTTP_INTERNAL_SERVER_ERROR);
                }
            }
        }
        return new JsonResponse(['urls' => $uploadedUrls], Response::HTTP_OK);
    }
}
```

		Мельниченко Н.М.			ІПЗ.КР.Б-121-24-ПЗ	Арк.
						72
Змн.	Арк.	№ докум.	Підпис	Дата		