МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ВАСИЛЯ СТУСА
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ І ПРИКЛАДНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КОНОВАЛЮК ІВАННА ЛЕОНІДІВНА

Допускається до захисту:
в.о. завідувача кафедри
інформаційних технологій,
д-р. техн. наук, професор
_____Наталія ВЕСЕЛОВСЬКА
«____»_____ 2026 р.

РОЗРОБКА ТУРИСТИЧНОГО САЙТУ ДЛЯ БРОНЮВАННЯ ПОДОРОЖЕЙ ТА ПОЇЗДОК

Спеціальність 122 Комп’ютерні науки

Кваліфікаційна (бакалаврська) робота

Керівник:
Юрій ПОРЕМСЬКИЙ,
старший викладач
кафедри
інформаційних технологій
_____

Оцінка: _____ / _____ / _____
(бали/за шкалою ЄКТС/за національною шкалою)

Голова ЕК: _____

Вінниця – 2026

АНОТАЦІЯ

Коновалюк І. Л. Розробка туристичного вебсайту для бронювання подорожей та поїздок. Спеціальність 122 «Комп'ютерні науки». Донецький національний університет імені Василя Стуса, Вінниця, 2026.

У кваліфікаційній (бакалаврській) роботі досліджено принципи розробки вебсайту для організації туристичних послуг та онлайн-бронювання поїздок. У межах роботи розроблено веб сайт, що забезпечує перегляд туристичних пропозицій, реєстрацію користувачів, створення особистого кабінету, вибір, перегляд та бронювання подорожей.

Клієнтська частина реалізована з використанням технологій HTML5, CSS3, JavaScript, бібліотеки jQuery та фреймворку Bootstrap, що забезпечує адаптивність інтерфейсу та зручність використання. Для збереження даних про користувачів, туристичні маршрути та бронювання передбачено використання реляційної бази даних PostgreSQL.

Ключові слова: вебсайт, туристична система, бронювання подорожей, HTML, CSS, JavaScript, Bootstrap, PostgreSQL.

ABSTRACT

Konovaliuk I. L. Development of a tourist website for booking travels and trips. Specialty 122 "Computer Science". Vasyl’ Stus Donetsk National University, Vinnytsia, 2026.

In the bachelor's qualification work, the principles of developing web systems for organizing tourism services and online trip booking are studied. Within the framework of the work, a website was developed that provides viewing of tourist offers, user registration, creation of a personal account, selection, viewing and booking of trips.

The client side of the system is implemented using HTML5, CSS3, JavaScript, the jQuery library and the Bootstrap framework, which ensures interface responsiveness and ease of use. A PostgreSQL relational database is planned to be used for storing data about users, tourist routes and bookings.

Keywords: website, tourism system, travel booking, HTML, CSS, JavaScript, Bootstrap, PostgreSQL.

ЗМІСТ

ВСТУП

Актуальність теми дослідження. У сучасному інформаційному суспільстві цифрові технології відіграють важливу роль у різних сферах людської діяльності. Не менш важливу роль вони відіграють у сфері туризму. З розвитком інтернет-технологій дуже швидкими темпами зросла потреба користувачів у швидкому доступі до відомостей про туристичні послуги, можливості планування мандрівок та онлайн-бронювання поїздок. Традиційні способи організації туристичних поїздок, які передбачають особисте звернення до туристичних агенцій, поступово поступаються місцем сучасним веб-сервісам, що дозволяють здійснювати пошук, вибір та бронювання подорожей безпосередньо через мережу Інтернет лише за кілька хвилин.

Важливий напрямок розвитку інформаційних систем - розробка туристичних вебсайтів.

Такі веб-сайти є дуже ефективними, адже дають змогу автоматизувати процес пошуку та надання інформації про подорожі, реєстрації та бронювання усіх інших туристичних послуг. Застосування таких веб технологій дозволяє зручно отримати доступ до сервісу з різних приладів, це підіймає ефективність взаємодії між користувачем та туристичними фірмами.

Розробка туристичного вебсайту вимагає залучення сучасних інструментів веб-програмування та систем урядування базами даних. Використання технологій HTML5, CSS3, JavaScript, збірки jQuery та фреймворку Bootstrap дає змогу сформувати зручний та адаптивний інтерфейс споживача. Для зберігання та обробки даних про користувачів, туристичні маршрути зручним та необхідним є застосування реляційної системи урядування базами даних PostgreSQL, яка забезпечує стабільність, масштабованість та продуктивну роботу з великими обсягами даних.

Метою дослідження є проектування та подальша розробка туристичного вебсайту, який забезпечує перегляд туристичних пропозицій, реєстрацію користувачів, створення особистого кабінету та бронювання подорожей з використанням сучасних технологій.

Завдання дослідження:

1. Проаналізувати сучасні веб сервіси у сфері туристичних послуг та онлайн-замовлення мандрівок, описати їхні функціональні можливості, переваги та вади з метою формування вимог до системи, що розробляється.
2. Дослідити особливості розробки вебсайтів та обґрунтувати вибір технологічного набору для створення туристичного сайту, зокрема вживання технологій HTML5, CSS3, JavaScript, бібліотеки jQuery та фреймворку Bootstrap.
3. Спроєктувати структуру та логіку роботи туристичного веб-сайту, визначити ключові модулі системи, зокрема модуль огляду туристичних пропозицій, реєстрації клієнтів, персонального кабінету та бронювання турів.
4. Розробити реляційну структуру бази даних із застосуванням PostgreSQL для зберігання відомостей про клієнтів, туристичні шляхи, поїздки та замовлення, гарантуючи цілісність і узгодженість даних.
5. Реалізувати клієнтський інтерфейс вебсайту з використанням сучасних технологій веб-творення та забезпечити адаптивність інтерфейсу для належної роботи на різних пристроях.
6. Інтегрувати веб-інтерфейс із базою даних для забезпечення можливості збереження, обробки та показу відомостей про туристичні пропозиції й замовлення клієнтів.
7. Провести випробування розробленого вебсайту, проаналізувати правильність роботи головних функцій системи та оцінити комфортність використання клієнтського інтерфейсу.

Об’єкт дослідження - процес організації туристичних послуг та онлайн-бронювання подорожей з використанням сучасних інформаційних веб-систем.

Предмет дослідження - методи, моделі та технології створення веб-сторінок для пошуку, вибору та бронювання туристичних подорожей, а також засоби втілення користувацького інтерфейсу і систем зберігання даних на базі сучасних веб-технологій та реляційних систем управління базами даних.

Теоретичне значення роботи полягає у вивченні сучасних шляхів до розроблення вебсайту у сфері туристичних послуг, розгляді засад формування інтерфейсів користувачів, а також способів упорядкування збереження та опрацювання відомостей у механізмах мережевого бронювання. У роботі підсумовано аспекти застосування актуальних веб-технологій для створення інтерактивних та гнучких вебсайтів.

Практична цінність роботи полягає у створенні туристичного вебсайту, який дає можливість користувачам оглядати туристичні пропозиції, реєстрування у системі, формування індивідуального профілю та резервування подорожей. Створений програмний продукт може бути залучений як фундамент для побудови цілісної системи бронювання туристичних послуг у мережі або інтегрований у діяльність туристичних фірм.

Апробація результатів дослідження. Результати кваліфікаційної роботи не були предметом обговорення на наукових конференціях та не публікувалися у наукових виданнях.

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ВИМОГ ДО ТУРИСТИЧНОГО ВЕБСАЙТУ

1.1 Особливості туристичних вебсайтів і систем онлайн-бронювання

Сьогодні туристична галузь приймаю нового вигляду, вона оцифровуємося, а процес планування поїздок прийшов в зовсім інший формат. У минулому людям потрібно було самостійно переглядати туристичні каталоги та обрати підходящий їм маршрут. Переважно, цим займалися туристичні агенства та різні компанії, які надавали свої послуги та виступали у ролі посередників. Зараз їх поступово замінюють вебсайти, адже все більше і більше користувачів надає їм перевагу. Користуючись туристичним вебсайтом, користувач отримує швидкий доступ до всіх доступних поїздок. Вебсайти стали не лише важливим засобом для організації туристичних подорожей, а й дуже важливим засобом між користувачем та компанією [1, 2].

Функціональність туристичних вебсайтів не обмежується лише переглядом каталогу. За допомогою них можна створити для себе повноцінний туристичний план, придбати потрібну поїздку, здійснити пошуку та переглянути інші корисні функції. Тому сучасний туристичний вебсайт слід розглядати як інформаційну систему, яка поєднує відображення послуг, яка має намір покращити та спростити процес взаємодії між користувачем та компанією надання туристичних послуг [1, 3].

Особливість ресурсів туристичних вебсайтів зумовлена складністю туристичних поїздок. Туристична послуга включає в себе дуже багато різних нюансів, серед яких маршрут, тривалість поїздки, види транспорту, проживання, вартість та додаткові послуги, зокрема екскурсії чи страхування. Тому туристичні вебсайти повинні обробляти велику кількість інформації та подавати її в логічно зрозумілій, структурованій і зручній для користувача формі. Тому

хороший туристичний вебсайт повинен мати багаторівневу структуру, картки, можливість фільтрації та пошуку та багато інших функцій [1, 4].

Ще однією ключовою характеристикою туристичних вебсайтів є орієнтація на користувача та можливість порівняння між кількома варіантами. Потенційні клієнти порівнюють різні тури, перед тим як обирати щось одне, тому, важливо зробити систему порівняння зручною, логічно розташувати елементи та подумати над зручною навігацією [2, 5].

Загалом, туристичні вебсайти можна поділити на дві основні групи: інформаційні вебсайти та вебсайти з можливістю онлайн-бронювання. Інформаційні веб-сайти дають можливість переглянути інформацію про туристичні маршрути, інформацію про компанію, зображення, новини та контактні дані. Тоді як вебсайти з онлайн-бронюванням дозволяють користувачам виконувати конкретні дії, наприклад заповнювати заявки або бронювати певні поїздки. Такі вебсайти потребують якісного інтерфейсу користувача, а також важливо забезпечити наявність механізмів обробки даних, зберігання записів про користувачів і замовлення [3, 6].

Вебсайт з онлайн-бронюванням відрізняється від звичайного туристичного вебсайту тим, що він має супроводити користувача до того моменту, поки він не заплатить кошти та не вибрав собі підходящу поїздку.

Типовим процесом є вибір місця, перегляд деталів про поїздку, заповнення форми та внесення особисту інформацію про себе та пасажирів, контактних даних і бронювання замовлення. У більш розвинених вебсайтах є можливість створити власний особистий кабінет, де буде зберігатися інформація про поїздки і заброньовані подорожі. Це дає зрозуміти, що бронювання є не окремим модулем, важливою складовою усього веб-сайту, функції якого мають бути логічними і працювати для зручності користувача [3, 6].

Ефективність будь-якого туристичного вебсайту залежить від того, наскільки швидко користувачможе розібратися з функціоналом, знайти

інформацію про потрібно йому поїздку і натиснути кнопку бронювання. Якщо вебсайт має заплутану навігацію, багато зайвих блоків або потребує великої кількості переходів для виконання основної дії, це робить його менш зручним. Людині яка вперше користується такими сервісами може бути важко прийняти рішення та розібратись. Тому сучасні туристичні вебсайти мають бути простими, логічними і максимально зрозумілим для користувача. Це особливо важливо для туристичної сфери [2].

З огляду на емоційну заангажованість споживачів туристичних послуг, візуальна привабливість посідає ключове місце у дизайні туристичних веб-ресурсів. Якісні фотоматеріали, логічно організовані пропозиції турів, чіткі описи та мінімалістичний інтерфейс здатні зацікавити потенційного клієнта та підвищити шанси на здійснення покупки. Однак, коли акцент робиться виключно на зовнішню ефектність, а структура сайту залишається незручною, це може стати перешкодою для користувача. Таким чином, одним із пріоритетних завдань для туристичного вебсайту є гармонійне поєднання естетичної привабливості з практичною зручністю користування [2, 5].

Для успіху туристичного вебсайту надзвичайно важливо підтримувати актуальність інформації. Дані відносно вартості турів, наявності вільних місць, дат вильоту, умов проживання чи специфіки маршруту мають тенденцію швидко змінюватися. Якщо відвідувач натрапить на застарілу або неповну інформацію, це може підірвати його довіру до ресурсу та негативно позначитися на подальшій комунікації. Тому туристичні веб-платформи мусять бути розроблені з урахуванням ефективної організації зберігання даних та можливості їх миттєвого оновлення. На практиці це вимагає використання структурованих баз даних та чітко визначених системних компонентів [1, 3].

Важливою специфікою туристичних вебсайтів є необхідність забезпечення коректного відображення на різних типах пристроїв. Велика кількість користувачів шукає інформацію про подорожі за допомогою смартфонів чи планшетів, тому вебсайт повинен адекватно масштабуватися під екрани різного

розміру, зберігаючи при цьому повну функціональність у мобільній версії. Адаптивний дизайн у цьому контексті виступає не як додаткова перевага, а як обов'язкова вимога для сучасного веб-ресурсу, адже його відсутність суттєво знижує зручність навігації, читання описів та заповнення форм бронювання [7, 8].

Зважаючи на те, що туристичні вебсайти з функцією онлайн-бронювання працюють з персональними даними користувачів, критично важливо дотримуватися вимог безпеки та чинного законодавства. Під час реєстрації або подання заявки користувач надає своє ім'я, контактний телефон, адресу електронної пошти та іншу конфіденційну інформацію. Обробка цих даних в Україні регламентується Законом України «Про захист персональних даних». Крім того, угоди, що укладаються в електронному форматі, регулюються Законом України «Про електронну комерцію», що безпосередньо впливає на процеси бронювання, підтвердження замовлень та документування взаємодії сторін [9, 10].

Отже, туристичні вебсайти та платформи з онлайн-бронюванням є специфічними веб-ресурсами, що інтегрують інформаційні, презентаційні та функціональні можливості. Вони працюють з комплексним туристичним продуктом, підтримують процеси пошуку та бронювання, потребують постійного оновлення контенту, орієнтуються на широку аудиторію та пред'являють високі вимоги до дизайну інтерфейсу, адаптивності та безпеки. Саме ці характеристики визначають унікальність туристичних вебсайтів і закладають основу для подальшого аналізу існуючих рішень та формування вимог до власного веб-ресурсу [1, 2, 3, 7].

1.2 Аналіз сучасних платформ для бронювання подорожей та поїздок

Сьогодні туристичний сектор переважно функціонує через цифрові платформи, які дають змогу користувачам здійснювати пошук, порівняння та

резервування туристичних продуктів в мережі. Якщо раніше звернення до менеджерів туристичних компаній або відвідування їхніх офісів було нормою для оформлення поїздок, то сьогодні багато мандрівників самостійно планують свої виїзди, користуючись веб-ресурсами та мобільними застосунками. Ця трансформація зумовлена прогресом технологій, ширшим доступом до Інтернету, розповсюдженням систем онлайн-бронювання та зростаючою потребою в оперативному отриманні послуг без залучення посередників [1, 3].

Сучасні платформи для планування подорожей та резервування послуг можна розглядати як унікальні віртуальні середовища. У них користувачі отримують доступ до даних про туристичні пропозиції, можуть застосовувати інструменти для пошуку, аналізувати різні варіанти та підтверджувати замовлення. Від типових інформаційних ресурсів вони відрізняються тим, що не лише надають інформацію про доступні сервіси, а й дозволяють здійснити конкретні дії – забронювати проживання, квитки, пакетні тури, екскурсії чи інші пов'язані послуги. Ключова перевага подібних ресурсів полягає в їхній здатності об'єднати функції надання інформації, технічної підтримки та здійснення транзакцій в єдиному просторі [5].

З практичної точки зору, сучасні туристичні веб-платформи можна класифікувати за декількома основними типами. До першої категорії належать традиційні онлайн-туристичні агенції (OTA), які агрегують пропозиції з різноманітних джерел, надаючи можливість бронювати послуги через єдиний інтерфейс. Другу групу утворюють платформи, де безпосередній зв'язок встановлюється між постачальником послуги та кінцевим споживачем, оминаючи посередників. Прикладом є сервіси для короткострокової оренди житла. Третій тип – це ресурси, що спеціалізуються на публікації відгуків, рейтингів та рекомендацій, допомагаючи користувачам робити вибір, спираючись на досвід інших мандрівників. Окремо слід виділити глобальні системи дистрибуції та бронювання, які слугують технічною базою для взаємодії між туристичними компаніями, агенціями та цифровими сервісами [5, 11, 13].

Серед сучасних платформ для планування подорожей особливої уваги варті Booking.com, Airbnb та Tripadvisor. Кожна з них застосовує індивідуальний підхід до організації туристичних сервісів, презентації пропозицій та взаємодії з користувачем. Booking.com, наприклад, переважно фокусується на пошуку та бронюванні помешкань, авіаквитків та інших похідних туристичних послуг.

На рисунку 1.2.1 подано приклад інтерфейсу вебсайту Booking.com, де користувач може вказати напрямок, дати подорожі, кількість осіб і переглянути доступні варіанти розміщення.

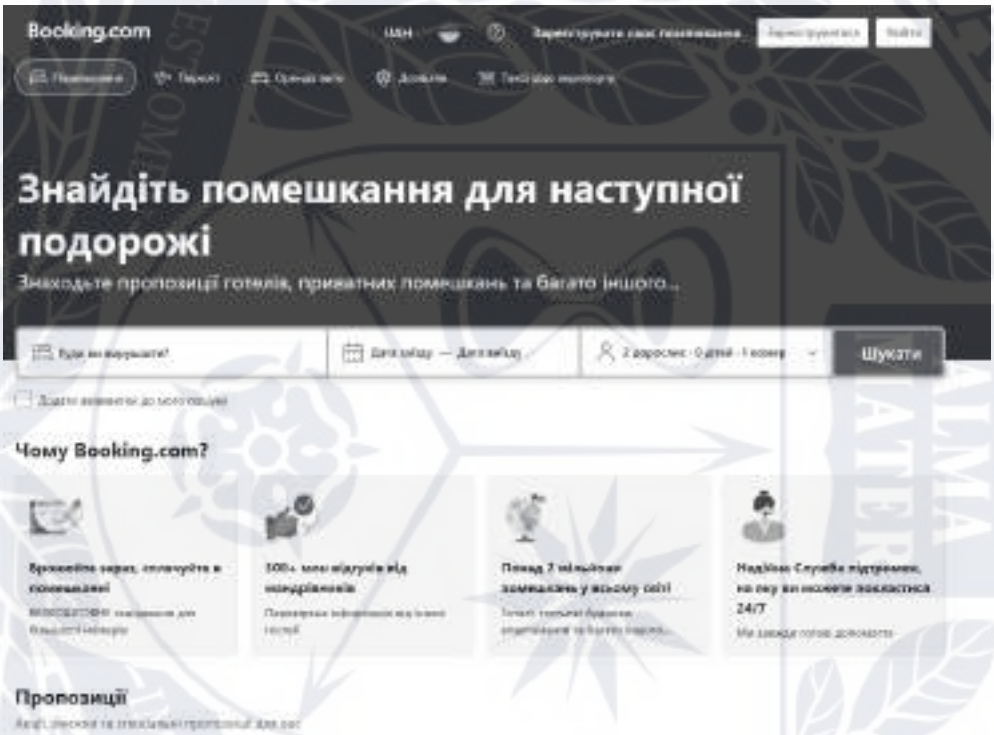

Рисунок 1.2.1 - Фрагмент інтерфейсу вебсайту Booking.com

Airbnb відрізняється тим, що основний акцент робить на оренді житла від приватних власників, а також на унікальних місцях проживання та туристичних враженнях. На рисунку 1.2.2 зображено приклад інтерфейсу Airbnb, який демонструє підхід до представлення житла за допомогою карток, фотографій, ціни, рейтингу та короткого опису.

Рисунок 1.2.2 - Фрагмент інтерфейсу вебсайту Airbnb

Tripadvisor, на відміну від попередніх вебсайтів, поєднує функції пошуку туристичних послуг із великою кількістю відгуків, рейтингів і рекомендацій від користувачів. Це дозволяє туристам не лише знайти готель або подорож, а й оцінити якість послуг на основі досвіду інших людей. На рисунку 1.2.3 подано приклад інтерфейсу Tripadvisor, де важливу роль відіграють рейтинги, відгуки та рекомендації щодо туристичних об’єктів.

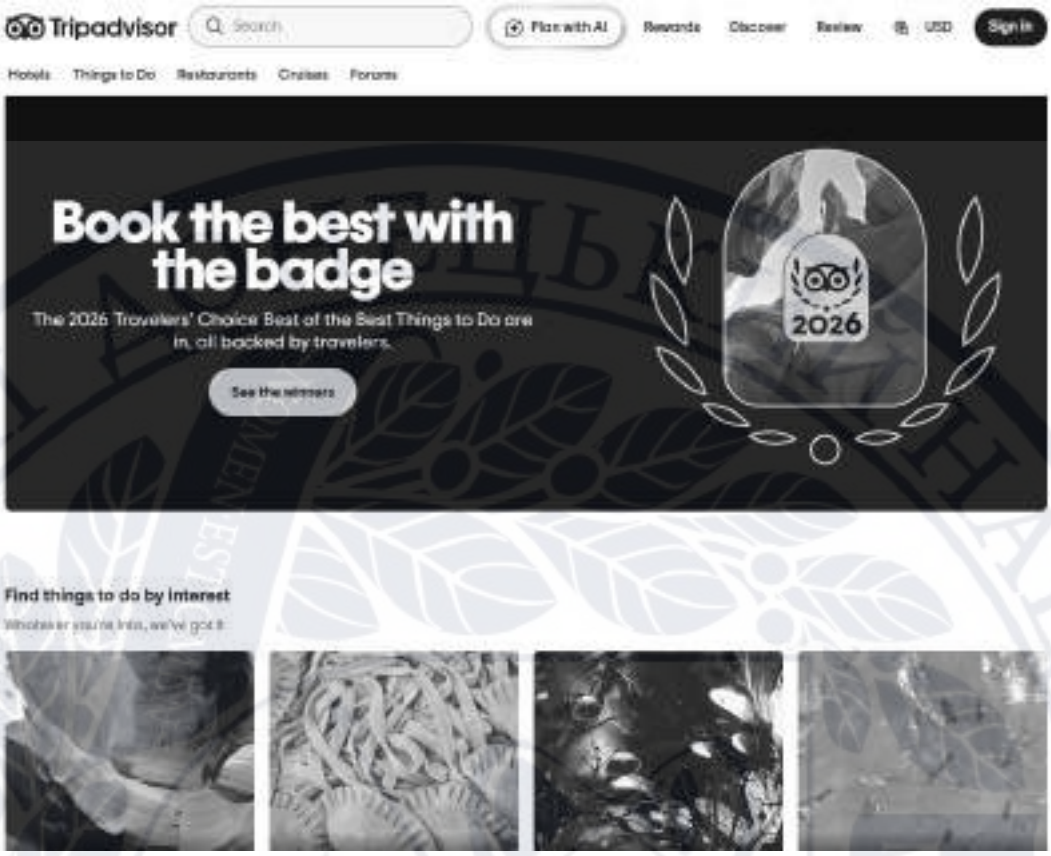

Рисунок 1.2.3 - Фрагмент інтерфейсу вебсайту Tripadvisor

Booking.com - це типовий представник OTA-платформ, що надає можливість шукати, порівнювати та резервувати численні варіанти проживання. Це досягається завдяки зручним фільтрам, системі рейтингів та стандартизованому формату подання даних, хоча деякі користувачі можуть вважати інтерфейс перевантаженим елементами, що дещо ускладнює процес вибору [2, 11, 12, 13]. Натомість, Airbnb акцентує увагу на персональному досвіді подорожей та прямому спілкуванні між користувачами. Ця платформа інтегрує процес бронювання з детальними описами житла, візуальним контентом та системою рекомендацій [11, 12, 13]. Tripadvisor, у свою чергу, виконує переважно інформаційну та аналітичну функцію, допомагаючи користувачам оцінювати туристичні послуги на основі відгуків, рейтингів та порад інших мандрівників [11, 12].

До окремої категорії ринкових гравців належать потужні екосистемні сервіси, як-от Expedia. Ці платформи об'єднують на одному сайті послуги

бронювання житла, транспортних перевезень, пакетних турів та інших супутніх туристичних продуктів. Ключовою перевагою таких рішень є їхня здатність до інтеграції з численними постачальниками, синхронізації даних, роботи з різними каналами реалізації та підримки складних користувацьких сценаріїв. Це демонструє, що розвиток туристичних платформ спрямований не лише на поліпшення користувацького досвіду, а й на технологічне об'єднання різних сегментів туристичної індустрії в єдиному цифровому просторі [11, 13, 14].

Окрім глобальних онлайн-платформ, слід також брати до уваги локальні сервіси, які зосереджуються на задоволенні потреб внутрішнього туризму, враховують специфіку національного ринку та відповідають звичним для користувачів моделям взаємодії. Аналіз сучасних вебсайтів для бронювання подорожей та поїздок показує такі основні тенденції:

1. забезпечення повного циклу взаємодії з користувачем - від пошуку до підтвердження бронювання;
2. персоналізація сервісу;
3. активне використання візуального контенту, рейтингів і відгуків;
4. орієнтація на мобільні пристрої та зручність для користувача.

Багатофункціональність, проте, не завжди гарантує зручність. Наявність перевантаженого інтерфейсу, надлишкових елементів та заплутаної навігації, як правило, ускладнює взаємодію користувача з вебсайтом [2].

Таким чином, сучасні сервіси для туристів демонструють різноманітні підходи до організації пропозицій. Вивчення цих онлайн-платформ дає змогу усвідомити, які конструктивні рішення виявилися успішними, а які містять вади, що слід пам'ятати для майбутньої розробки системи [3, 5, 14].

1.3 Аналіз функціональних можливостей існуючих рішень

Практична користь сучасних туристичних веб-платформ безпосередньо залежить від їхніх функціональних можливостей. Якщо раніше такі ресурси здебільшого надавали інформацію, то нині вони охоплюють увесь ланцюжок

взаємодії з туристичним продуктом: від пошуку та огляду доступних пропозицій до здійснення бронювання, отримання підтвердження та подальшого обслуговування клієнта [5, 12]. Саме з цієї причини, досліджуючи наявні рішення, слід оцінювати не лише візуальне оформлення сайту, але й комплекс функцій, що сприяють швидкому знаходженню оптимального варіанту та здійсненню бажаної дії.

Серед ключових можливостей туристичних веб-ресурсів варто виділити пошук та фільтрацію. Користувачі мають змогу знаходити актуальні пропозиції, спираючись на різноманітні критерії: географічні напрямки, дати подорожі, ціновий діапазон, тривалість перебування, кількість осіб, тип розміщення, оціночний рейтинг або специфічні вимоги. Як приклад, система пошуку на Booking.com базується на зазначенні пункту призначення, дат заїзду та виїзду, а також кількості гостей і номерів. Це забезпечує миттєве формування переліку наявних пропозицій, які згодом можна деталізувати за допомогою фільтрів.

На рисунку 1.3.1 подано приклад реалізації пошуку та фільтрації на туристичному вебсайті. Такий підхід спрощує роботу з великою кількістю туристичних пропозицій і зменшує час, необхідний користувачу для вибору відповідного варіанта [12, 13].

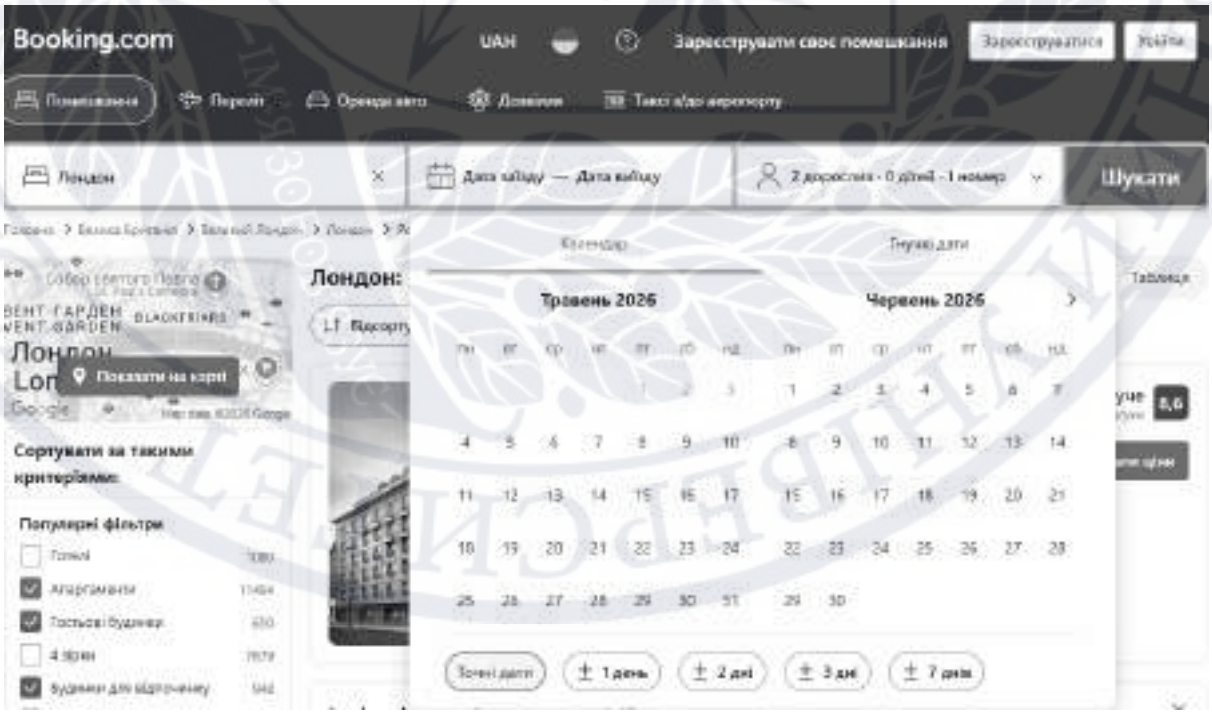

Рисунок 1.3.1 - Приклад пошуку та фільтрації туристичних пропозицій

Ключовим аспектом у створенні привабливих туристичних пропозицій є їхня грамотна візуалізація. Більшість онлайн-платформ, що займаються туризмом, вдаються до формату карток для презентації інформації. Кожна така картка зазвичай містить назву, стислий опис, вартість, фотографію, рейтинг та ключові особливості подорожі, готелю чи місця проживання. Цей підхід дає змогу потенційному мандрівникові швидко зорієнтуватися серед різноманіття варіантів та обрати для детальнішого ознайомлення лише ті пропозиції, які відповідають його інтересам [2, 12]. Як показує досвід Airbnb, використання карток з яскравими зображеннями, зазначенням ціни за ніч, рейтингом та лаконічним описом помешкання забезпечує інтуїтивно зрозумілий та комфортний перегляд пропозицій, навіть коли це відбувається з екранів мобільних гаджетів.

На рисунку 1.3.2 наведено приклад карткового подання туристичних пропозицій. У такому форматі користувач одразу бачить основну інформацію про варіант подорожі та може швидко оцінити його привабливість.

Рисунок 1.3.2 - Приклад карткового подання туристичних пропозицій

Наступною ключовою можливістю є детальна презентація інформації про послугу. Користувач, переходячи на спеціально відведену сторінку, отримує вичерпні відомості про маршрут, тривалість подорожі, умови проживання, харчування, транспортне забезпечення, наявні дати, вартість та процедуру бронювання. Якість та прозорість наявної інформації безпосередньо впливають на рівень довіри до веб-ресурсу та визначають рішення щодо подання заявки [1, 5]. Наприклад, на платформі Tripadvisor сторінки, присвячені туристичним об'єктам, зазвичай не обмежуються лише описом, а й містять візуальний контент (фотографії), рейтинги, відгуки інших відвідувачів, інформацію про географічне

розташування та поради. Це значно полегшує оцінку якості послуги ще до здійснення бронювання.

На рисунку 1.3.3 зображено приклад сторінки детального перегляду туристичної пропозиції. Така сторінка дозволяє користувачу отримати розширену інформацію перед прийняттям рішення щодо бронювання.

Рисунок 1.3.3 - Приклад детального перегляду туристичної пропозиції

Головним призначенням сайтів туристичних компаній є забезпечення процесу бронювання або подання заявки. Саме ця функціональність виокремлює туристичний ресурс від звичайного сайту з інформаційним наповненням. Реалізація цієї можливості може варіюватися: від простої контактної форми до багатокрокового процесу бронювання, остаточного підтвердження замовлення або повноцінної інтеграції з платіжними системами [5]. Візьмемо для прикладу Booking.com: після того, як ви обрали бажаний варіант проживання, система

пропонує розпочати процес бронювання, де необхідно ввести ваші контактні дані, уточнити параметри перебування та підтвердити свою заявку. Щодо туристичних турів, подібний підхід може полягати у використанні форми бронювання, де клієнт зазначає свої ім'я, номер телефону, адресу електронної пошти, кількість осіб, що подорожуватимуть, а також будь-які додаткові побажання.

На рисунку 1.3.4 подано приклад форми бронювання або оформлення заявки. Такий елемент є ключовим для туристичного вебсайту, оскільки саме через нього користувач переходить від перегляду пропозиції до фактичного замовлення послуги.

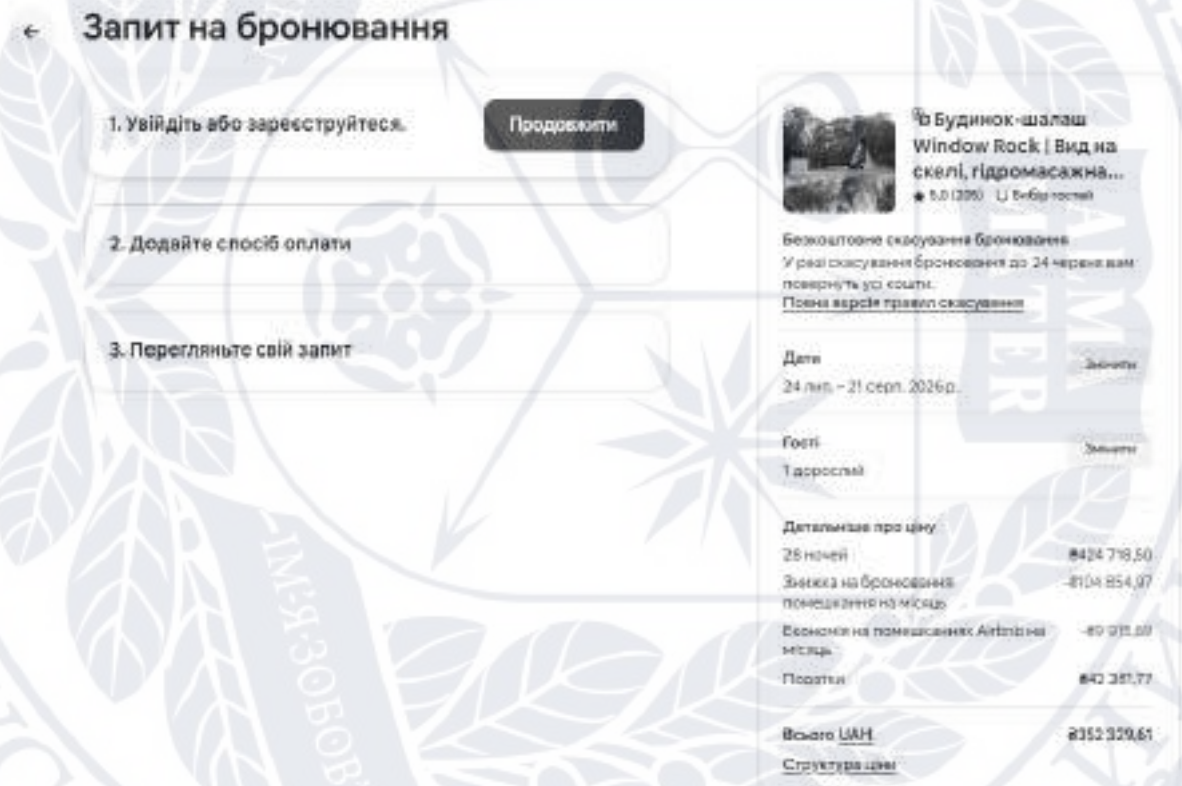

Рисунок 1.3.4 - Приклад форми бронювання туристичної послуги

Більшість сучасних туристичних онлайн-платформ пропонують користувачам реєстрацію, вхід до свого профілю та авторизацію. Такий підхід дозволяє зберігати персональні дані, історію бронювань, вибрані пропозиції та статус запитів. Це значно полегшує повторне використання сервісу та сприяє кращому врахуванню індивідуальних потреб клієнтів [1, 2]. Наприклад, на

великих туристичних порталах користувачі можуть переглядати як майбутні, так і минулі бронювання, оновлювати свої особисті дані, зберігати вподобані варіанти, а також отримувати індивідуально підібрані рекомендації.

Особливого значення надається відгукам, рейтингам та іншим формам соціального підтвердження. Вони допомагають потенційним клієнтам оцінити якість наданих послуг, спираючись на досвід інших мандрівників. Це надзвичайно важливо у сфері туризму, адже рішення про бронювання часто приймаються задовго до фактичного отримання послуги [11]. Так, наприклад, Tripadvisor активно використовує рейтинги, оцінки, коментарі та фотографії від своїх користувачів. Booking.com також надає інформацію про оцінки гостей, переваги помешкання та коментарі, що допомагає сформувати більш обґрунтований вибір.

Окрім функцій, орієнтованих на кінцевих користувачів, онлайн-платформи для туризму також мають адміністративні можливості. До них належать додавання та редагування туристичних пропозицій, зміна цін, оновлення дат, перегляд та обробка запитів, управління обліковими записами користувачів та підтримання актуальності інформації на сайті. Без цих функцій туристичний вебсайт не зможе ефективно функціонувати в реальних умовах, оскільки дані про тури, ціни, наявність місць та доступні дати постійно потребують оновлення [5, 15].

Варто також звернути увагу на адаптивність та зручність використання туристичних вебсайтів на мобільних пристроях. Сучасні користувачі нерідко шукають туристичні послуги за допомогою смартфонів, тому вебсайт має забезпечувати коректне відображення сторінок, легку навігацію, читабельність тексту та зручне заповнення форм незалежно від типу пристрою, який використовується [2, 6, 7]. Наприклад, у мобільних версіях Booking.com чи Airbnb ключові дії користувача, такі як пошук, перегляд пропозицій, застосування фільтрів та здійснення бронювання, залишаються доступними без складних навігаційних елементів.

Отже, аналіз функціоналу сучасних туристичних рішень свідчить про необхідність наявності таких базових можливостей: пошук та фільтрація, представлення пропозицій у вигляді карток, детальний перегляд, можливість бронювання, реєстрація користувачів, розділ з відгуками, адміністративне керування та адаптивний інтерфейс. Саме ці компоненти формують основу ефективного вебсайту для онлайн-бронювання туристичних послуг і можуть слугувати орієнтиром при розробці власної туристичної платформи [15].

1.4 Визначення цільової аудиторії та сценаріїв використання вебсайту

Розуміння того, хто ж є цільовою аудиторією туристичного вебсайту, являє собою фундаментальний крок у процесі його розробки. Адже саме від потреб користувачів залежать каркас системи, спектр її функціоналу та механізми взаємодії. В епоху стрімкої діджиталізації туристичної галузі від відвідувачів сайтів очікують миттєвого доступу до інформації, інтуїтивно зрозумілого пошуку, опцій для порівняння різних варіантів та зручного онлайн-бронювання послуг [16].

Основну аудиторію туристичного вебсайту доцільно поділити на такі групи:

1. індивідуальні мандрівники, які самостійно планують свій відпочинок і потребують простого та зручного способу перегляду й бронювання туристичних послуг;
2. сім’ї та малі групи, для яких важливою є ціна, тривалість подорожі, умови проживання та зручність маршруту;
3. користувачі, які цікавляться внутрішнім туризмом і цінують локальні маршрути, зрозумілий інтерфейс та адаптацію сервісу під національний ринок;

4. адміністратори або менеджери туристичного вебсайту, які відповідають за підтримку системи, оновлення інформації та обробку заявок;
5. власники туристичного сервісу.

Основні категорії користувачів, їхні потреби та типові дії в системі наведено в табл. 1.1.

Таблиця 1.1 Цільова аудиторія туристичного вебсайту та її основні потреби

Категорії користувачів	Основні протреби	Типові дії в системі
Індивідуальні мандрівники	Швидкий пошук, зручне бронювання, перегляд деталей	Пошук туру, перегляд опису, оформлення заявки
Сім'ї та малі групи	Порівняння вартості, умов проживання, тривалості	Вибір пропозиції, аналіз умов, бронювання
Користувачі внутрішнього туризму	Локальні напрямки, зрозумілий інтерфейс, зручна оплата	Перегляд локальних маршрутів, бронювання
Адміністратори або менеджери	Додавання та оновлення інформації, обробка заявок	Редагування турів, зміна цін, перевірка заявок

Як видно з таблиці 1.1, туристичний вебсайт має бути орієнтований не лише на користувачів, які користуються послугами, а й на адміністраторів, які забезпечують актуальність інформації та функціонування системи. Це означає, що вебсайт має підтримувати як функції пошуку та бронювання, так і функції редагування та оновлення даних.

Кожній групі користувачів система має забезпечувати зрозумілі та зручні сценарії використання. Для звичайного користувача основними завданнями є пошук туристичної пропозиції, перегляд короткої інформації, перехід до детального опису та оформлення заявки або бронювання. У сучасних системах важливими також є отримання зворотного зв’язку, підтвердження бронювання та можливість повторної взаємодії із сервісом.

Особлива ситуація виникає, коли система використовується повторно. Це означає, що користувач може увійти на сайт, переглянути історію своїх замовлень, скористатися послугами ще раз або перевірити поточний статус бронювання. Такий підхід значно покращує зручність користування сайтом і дає змогу створити персоналізований досвід для кожного відвідувача [16].

З боку адміністратора або менеджера, найчастіше виконуються такі завдання: створення нових туристичних пакетів, модифікація їх описів, оновлення вартості, дат проведення та наявності турів, а також обробка запитів, що надходять. Саме ці дії гарантують, що інформація на сайті завжди буде актуальною, а сам сайт функціонуватиме належним чином як сервісна платформа [15].

Таким чином, головними користувачами туристичного вебсайту є дві основні категорії: ті, хто шукає подорожі (туристи), та ті, хто ним керує (працівники). Аналіз потреб обох груп дозволяє визначити ключові аспекти, на яких слід зосередитись під час розробки системи: зручний функціонал пошуку, ефективне представлення турів, швидкий процес бронювання, можливість повторного використання сайту та належне управління всіма ресурсами платформи [12, 16].

1.5 Формування функціональних і нефункціональних вимог до вебсайту

Розробка вимог до туристичного вебсайту завершує аналітичний розділ першої частини. Саме ці вимоги слугують орієнтиром для вибору методів

розробки і подальшої реалізації проєкту. У сфері програмної інженерії прийнято розрізняти два типи вимог: функціональні, що окреслюють функції та завдання, які має виконувати система, та нефункціональні, котрі визначають критерії якості її функціонування. Такий підхід до класифікації є цілком виправданим для туристичного вебсайту, адже він дає змогу чітко визначити як ключові функції платформи, так і очікування щодо її юзабіліті, продуктивності, безпеки та надійності [17].

До функціональних вимог туристичного вебсайту належать:

1. відображення переліку туристичних пропозицій, напрямків або поїздок;
2. пошук і фільтрація пропозицій за основними параметрами;
3. перегляд повної інформації про тур, маршрут або послугу;
4. можливість оформлення заявки або бронювання;
5. надання користувачеві контактних даних та засобів зворотного зв’язку;
6. реєстрація, вхід до системи та перегляд історії замовлень;
7. адміністрування контенту, зокрема додавання, редагування та оновлення туристичних пропозицій.

До нефункціональних вимог туристичного вебсайту належать:

1. Зручність використання - інтерфейс має бути простим і логічним для різних груп користувачів.
2. Адаптивність - вебсайт повинен коректно працювати на комп’ютерах, планшетах і мобільних пристроях.
3. Швидкодія - сторінки мають завантажуватися без значних затримок, а пошук і форми повинні працювати стабільно.
4. Надійність - система має коректно обробляти запити користувачів і зберігати актуальні дані.
5. Безпека - персональні дані користувачів мають бути захищеними, а передача інформації повинна здійснюватися відповідно до вимог безпеки та чинного законодавства.

6. Масштабованість і супроводжуваність - вебсайт має забезпечувати можливість розширення функціоналу та оновлення контенту без суттєвих змін у структурі системи.

Функціональні та нефункціональні вимоги тісно переплетені. Наприклад, сам факт можливості здійснення бронювання є функціональною вимогою. Проте, ця функція буде дійсно ефективною лише тоді, коли форма введення даних буде інтуїтивно зрозумілою, сторінки сайту завантажуватимуться блискавично, а відомості користувачів будуть надійно захищені. Так само, функція пошуку туристичних пропозицій має поєднувати в собі насиченість можливостей із простотою взаємодії та бездоганною роботою на найрізноманітніших пристроях [17, 18].

Таким чином, функціональні вимоги до туристичного вебсайту полягають у визначенні його ключових можливостей, що стосуються перегляду, пошуку та оформлення бронювань туристичних послуг. Натомість, нефункціональні вимоги встановлюють стандарти якості його функціонування. Комплекс цих вимог складає фундамент для подальшої розробки архітектури вебсайту, його бази даних та інтерфейсу взаємодії з користувачем [18].

1.6. Аналіз фінансової складової та підходів до монетизації туристичних вебсайтів

Економічний аспект є невід'ємною частиною функціонування сучасних веб-ресурсів туристичної сфери, адже саме від нього залежить обґрунтованість витрат на створення, обслуговування та подальший розвиток проєктів. Більшість онлайн-сервісів для мандрівників виконують не лише інформаційні та операційні функції, але й функціонують як комерційні платформи, що приносять прибуток завдяки різним способам заробітку.

Одним із найпоширеніших методів є комісійна схема. За такою моделлю вебсайт отримує певний відсоток від кожної успішно завершеної транзакції бронювання. У цьому випадку туристичний сервіс виступає посередником між

кінцевим користувачем та надавачем послуг, забезпечуючи ефективну комунікацію та оформлення замовлення. Цей підхід активно використовується великими платформами для бронювання житла та турів, адже дозволяє розширювати бізнес без необхідності прямого володіння туристичними активами.

Іншою популярною формою є модель афілійованого маркетингу. За цією схемою вебсайт отримує винагороду за скеровування користувачів на сайти партнерів. Дохід у цьому випадку генерується завдяки клікам, або вчиненню певних дій користувачами на сторонніх ресурсах. Цей метод часто застосовується інформаційними туристичними порталами та сайтами з відгуками.

Також застосовується модель платного розміщення, або забезпечення пріоритетного показу. За такою схемою, компанії, що надають туристичні послуги, сплачують за те, щоб їхні пропозиції були більш помітними у результатах пошуку. Це сприяє збільшенню кількості замовлень для постачальників та забезпечує додатковий потік коштів для вебсайту.

Окремо варто згадати модель підписки. Вона передбачає регулярну плату за доступ до розширених функцій сервісу. Це може бути застосовано як до кінцевих споживачів, так і до бізнес-клієнтів, зокрема туристичних агентств чи готелів, які отримують доступ до розширеної аналітики, інструментів управління або можливості пріоритетного розміщення.

Слід наголосити, що ефективність монетизації безпосередньо залежить від якості впровадження функціоналу вебсайту, рівня довіри з боку користувачів та загальної зручності використання системи. Високий рівень юзабіліті, швидкість роботи, безпека даних та позитивний користувацький досвід мають прямий вплив на коефіцієнт конверсії та фінансові показники системи.

Таким чином, фінансова складова туристичних вебсайтів формується шляхом комбінування декількох методів монетизації. Найбільш поширеними серед них є комісійна модель, партнерські програми, платне просування та підписка. Вибір конкретної моделі зумовлюється типом ресурсу, його

масштабом та цільовою аудиторією, а також безпосередньо впливає на архітектурні рішення та бізнес-логіку системи [19].

Висновок до розділу 1

У першій частині дипломної роботи було досліджено специфіку туристичних веб-ресурсів та платформ для онлайн-бронювання. Проведено аналіз сучасних сервісів для планування подорожей, оцінено можливості наявних рішень, визначено основні групи користувачів майбутньої системи та сформульовано ключові вимоги до туристичного сайту.

В ході дослідження встановлено, що сьогоднішні туристичні вебсайти виконують роль не тільки інформаційних порталів, а й повнофункціональних цифрових платформ, що забезпечують комплексне спілкування з користувачем. Вони дають змогу презентувати туристичні продукти, проводити пошук та порівняння, здійснювати бронювання та підтримувати подальшу взаємодію з відвідувачем. Виявлено, що для таких систем характерні складність туристичного продукту, необхідність постійного оновлення інформації, високі стандарти щодо адаптивності, зручності інтерфейсу та безпечного опрацювання особистих даних.

Аналіз актуальних платформ для бронювання засвідчив, що головними напрямками їхнього розвитку є забезпечення повного циклу взаємодії з користувачем, персоналізація послуг, активне використання візуального контенту, систем рейтингу та відгуків, а також орієнтація на мобільні пристрої. Водночас, помічено, що надмірна кількість функцій у деяких платформах може ускладнювати їх використання через перевантаженість інтерфейсу та складність навігації.

В процесі дослідження можливостей існуючих рішень встановлено, що ефективний туристичний вебсайт повинен мати наступні ключові функції: пошук та фільтрування пропозицій, структуроване представлення туристичних

продуктів, детальну інформацію, можливість оформити заявку або здійснити бронювання, реєстрацію користувачів, механізми зворотного зв'язку, а також інструменти для адміністраторів для оновлення та збереження актуальності даних.

У цій частині роботи також було визначено цільову аудиторію системи, до якої входять індивідуальні мандрівники, сім'ї, невеликі групи, користувачі, що цікавляться внутрішнім туризмом, а також адміністратори туристичного вебсайту. Це дозволило визначити основні сценарії використання системи та врахувати потреби як кінцевих користувачів, так і відповідальних за наповнення та підтримку сайту осіб.

На основі проведеного аналізу сформульовано функціональні та нефункціональні вимоги до туристичного вебсайту. Функціональні вимоги охоплюють перегляд туристичних пропозицій, пошук, фільтрацію, бронювання, зворотний зв'язок, реєстрацію користувачів та керування контентом. Нефункціональні вимоги включають легкість використання, адаптивність, швидкодію, надійність, безпеку, масштабованість системи та можливість її подальшого технічного супроводу.

Окремо було проаналізовано фінансову сторону та методи отримання прибутку від туристичних вебсайтів. Встановлено, що основними моделями монетизації є комісійна модель, партнерські програми, платне розміщення пропозицій та модель підписки. Ці підходи дозволяють забезпечити стабільну роботу сервісу та його розвиток без необхідності прямого продажу туристичних послуг. Визначено, що вибір моделі монетизації залежить від типу платформи, її масштабу та цільової аудиторії, а також впливає на бізнес-логіку та архітектуру системи.

Таким чином, результати першого розділу створюють теоретичну та аналітичну базу для подальшого проєктування туристичного вебсайту, розробки його структури, бази даних та інтерфейсу користувача в наступних розділах дипломної роботи.

РОЗДІЛ 2. ПРОЄКТУВАННЯ ТУРИСТИЧНОГО ВЕБСАЙТУ ДЛЯ БРОНЮВАННЯ ПОДОРОЖЕЙ

2.1 Загальна архітектура вебсайту

Розробка загальної архітектури веб-платформи для туристичного бронювання є ключовим етапом у створенні, визначаючи базові компоненти системи, способи їх зв'язку, методи керування даними та потенціал для майбутнього розширення. В галузі туризму, веб-технології та інформаційні системи відіграють вирішальну роль, надаючи споживачам миттєвий доступ до туристичних продуктів, спрощуючи процес пошуку пропозицій, автоматизуючи обробку замовлень та підвищуючи ефективність комунікації між клієнтом і туроператором [1, 4].

Розглядатимемо розроблюваний вебсайт не лише як джерело інформації про туристичні напрямки, але і як повноцінну веб-орієнтовану систему, яка в перспективі має охоплювати весь цикл взаємодії користувача з сервісом: перегляд варіантів подорожей, пошук туру за заданими критеріями, вибір відповішого турпакету, заповнення форми бронювання та запис цієї інформації в базу даних. Такий підхід відповідає сучасним тенденціям в онлайн-бронюванні туристичних послуг, де пріоритет віддається доступності, оперативності обробки запитів, інтуїтивно зрозумілому інтерфейсу та можливості отримати послугу без необхідності фізичного відвідування офісу туристичної компанії [12, 13].

Фундамент архітектури туристичного вебсайту складає клієнт-серверна парадигма. Вона передбачає розділення системи на клієнтську складову, серверний компонент та сховище даних. Клієнтська частина відповідає за візуалізацію вмісту веб-сторінок у браузері користувача, реакцію на взаємодію з елементами інтерфейсу, введення даних через форми та навігацію між різними розділами. Серверна частина виконує обробку запитів, верифікацію введеної інформації, реалізацію логіки пошуку та бронювання, а також керує взаємодією

з базою даних. База даних слугує централізованим сховищем інформації про туристичні напрямки, доступні тури, облікові записи користувачів, здійснені бронювання та повідомлення, отримані через контактну форму.

Загальну клієнт-серверну архітектуру зображено на рисунку 2.1.1.

Рисунок 2.1.1 - Загальна клієнт-серверна архітектура туристичного вебсайту

На рисунку 2.1.1 наведено схему, у якій користувач через браузер взаємодіє з клієнтською частиною вебсайту. Клієнтська частина надсилає запити до серверної, яка їх обробляє й, за потреби, взаємодіє з базою даних. Після опрацювання сервера, результат повертається клієнту, а користувач бачить оновлену інформацію на сайті або повідомлення про завершення дії.

Архітектуру системи можна умовно розділити на три базові рівні: рівень користувацького інтерфейсу, рівень серверної логіки та рівень зберігання даних. Цей підхід дозволяє розподілити відповідальність між компонентами, спростити супровід системи та закласти основу для її подальшого розвитку. З точки зору програмної інженерії, таке розбиття на логічні рівні є критично важливим, оскільки покращує структурованість проєкту, полегшує тестування окремих модулів та дозволяє розширювати функціонал без повної перебудови системи [17].

Першим є рівень користувацького інтерфейсу. Він відповідає за зовнішній вигляд сайту та пряму взаємодію користувача із системою. До цього рівня належать: головна сторінка, сторінка напрямків, сторінка про компанію, сторінка контактів, навігаційне меню, форма пошуку турів, картки з описами турів, блоки з перевагами компанії, відгуки клієнтів та нижній колонтитул. Основне завдання цього рівня — забезпечити зручний, зрозумілий та привабливий візуально доступ до туристичних пропозицій. Для туристичного сайту особливо важливий інтуїтивно зрозумілий інтерфейс, щоб користувач міг швидко знайти потрібний напрямок, переглянути тур та подати заявку [2].

Важливою вимогою до клієнтської частини є адаптивність. Туристичним сайтом користуються з різних пристроїв: ПК, ноутбуків, планшетів та смартфонів. Тому інтерфейс повинен коректно відображатись на екранах будь-якого розміру, а ключові елементи керування мають залишатися зручними для використання як на великих, так і на малих екранах. Адаптивний дизайн підвищує доступність сайту, покращує користувацький досвід та є важливим фактором ефективності сучасних веб-ресурсів [6, 7].

Другим рівнем є рівень серверної логіки. Він призначений для обробки даних, що надходять від користувача через веб-форми. Наприклад, при пошуку туру користувач може вказати напрямок, дати заїзду та виїзду, а також бажаний ціновий діапазон. Після надсилання запиту серверна частина повинна перевірити коректність введених даних, сформувати запит до бази даних, отримати відповідні туристичні пропозиції та повернути їх клієнтській частині. Подібним чином серверна частина обробляє запити на бронювання, контактні повідомлення та інші дії користувача.

Використання серверної логіки є невід'ємною частиною повноцінного бронювання, оскільки статичні HTML-сторінки не можуть самостійно забезпечити надійне збереження заявок, перевірку даних чи обробку інформації про доступні тури. Серверна частина дозволяє реалізувати бізнес-логіку: перевірку обов'язкових полів, обробку форм бронювання, зміну статусу заявки, отримання списку турів та взаємодію з адміністративною частиною сайту. Це

створює основу для трансформації сайту з інформаційного ресурсу у функціональну систему онлайн-бронювання.

Третім рівнем є рівень бази даних. База даних слугує основним сховищем інформації для сайту. Тут доцільно зберігати дані про туристичні напрямки, тури, користувачів, бронювання та контактні форми. Наприклад, таблиця турів може містити назву подорожі, опис, країну чи місто призначення, ціну, тривалість, дати початку й завершення, а також посилання на зображення. Таблиця бронювань може зберігати інформацію про користувача, вибраний тур, кількість осіб, дату створення заявки та її статус. Така організація даних дозволяє уникнути дублювання, забезпечити цілісність і спростити подальшу обробку заявок.

Наявність бази даних є особливо критичною для систем бронювання, оскільки туристична діяльність пов'язана з великим обсягом змінної інформації: напрямками, цінами, датами, кількістю місць, контактами клієнтів і статусами заявок. Автоматизовані системи бронювання в туризмі підвищують оперативність обслуговування, зменшують кількість помилок, спрощують пошук пропозицій і забезпечують зручний облік заявок [3, 5, 14].

У наявній реалізації проєкту особлива увага приділена клієнтській частині вебсайту. Вона складається з набору HTML-сторінок, файлів стилів, JavaScript-файлів, зображень та підключених бібліотек для створення інтерактивних елементів. Сайт має готову структуру сторінок, адаптивне меню, головний банер, форму для пошуку турів, блоки з туристичними напрямками, картки турів, відгуки клієнтів та контактну форму. Це створює візуальну та структурну основу для подальшого підключення серверної частини та бази даних.

Клієнтська частина реалізована за допомогою HTML, CSS, SCSS, JavaScript та додаткових бібліотек. HTML використовується для побудови структури сторінок, CSS та SCSS – для візуального оформлення, Bootstrap – для створення адаптивної сітки та використання готових компонентів інтерфейсу, JavaScript та jQuery – для реалізації інтерактивної поведінки елементів. Додаткові бібліотеки, такі як Owl Carousel, AOS та Datepicker, можуть

використовуватися для створення каруселей, анімацій та полів вибору дат у формах пошуку. Така комбінація технологій дозволяє створити зручний і візуально привабливий інтерфейс, що відповідає потребам користувачів туристичного ресурсу.

З точки зору користувача, архітектура сайту побудована так, щоб шлях до основної дії був максимально простим. Користувач потрапляє на головну сторінку, вивчає популярні напрямки або пропозиції, користується формою пошуку, обирає потрібний тур і переходить до оформлення заявки або зв'язується з представником компанії через контактну форму.

Це схематично зображено на рисунку 2.1.2. Така логіка відповідає типовому сценарію використання туристичного вебсайту, де основними діями є пошук, порівняння, вибір і бронювання подорожі.

Рисунок 2.1.2 - Схема опрацювання запитів

На рисунку 2.1.2 можна зобразити послідовність дій: надходження запиту від користувача → обробка на клієнтській частині → передача на сервер → обробка серверною логікою → взаємодія з базою даних → повернення результату користувачу.

Ключовими компонентами вебплатформи для туризму виступають: блок для представлення загальної інформації, блок, що описує туристичні маршрути,

блок пошуку, блок для оформлення замовлень, блок для комунікації з відвідувачами, блок бази даних та адміністративний блок. Перший блок містить відомості про компанію, що надає туристичні послуги, а також описує її пропозиції та переваги. Блок, який представляє туристичні маршрути, інформує про наявні напрямки для подорожей. За допомогою блоку пошуку користувачі можуть вказувати критерії для майбутньої подорожі. Блок для оформлення замовлень слугує для створення запиту на обраний тур. Через блок для комунікації користувачі можуть надсилати свої повідомлення. Адміністративний блок передбачає можливість керування турами, перегляду запитів та внесення змін до контенту сайту.

Важливою складовою розробки є взаємозв'язок між окремими блоками системи. Всі вони функціонують в єдиному інформаційному просторі, обмінюючись даними через серверну частину та спільну базу даних. Наприклад, коли активується блок пошуку, він звертається до бази даних, аби отримати найсвіжіші пропозиції, що відповідають заданим критеріям. Блок, що описує туристичні маршрути, використовує цю ж інформацію для формування переліку доступних турів. Коли користувач оформлює замовлення, блок бронювання передає сформовану заявку до серверної частини, яка потім записує її до бази даних. Адміністративний блок має розширені права доступу до бази даних, що дозволяє йому редагувати, оновлювати або видаляти інформацію про тури та запити. Таким чином, база даних стає центральним вузлом, що забезпечує інтеграцію всіх блоків системи.

Загальну структуру функціональних модулів та їх взаємодію подано в таблиці 2.1.2.

Таблиця 2.1.1 - Основні функціональні модулі туристичного веб сайту та їх взаємодія

Модуль системи	Вхідні дані	Вихідні дані	Взаємодія з іншими модулями

Інформаційний модуль	Дані про компанію	Інформаційні сторінки	База даних
Модуль туристичних напрямків	Запит на перегляд турів	Список турів	База даних, модуль пошуку
Модуль пошуку	Параметри пошуку (дата, напрямок, ціна)	Відфільтровані тури	База даних, модуль напрямків
Модуль бронювання	Дані користувача, обраний тур	Підтвердження заявки	База даних, серверна логіка
Модуль зворотного зв’язку	Повідомлення користувача	Статус відправки	База даних, адміністративний модуль
Модуль бази даних	Запити від модулів	Збережені/отримані дані	Усі модулі системи
Адміністративний модуль	Команди адміністратора	Оновлений контент	База даних, усі модулі

Під час розробки архітектури вебсайту для туристичної галузі, окрім функціональних аспектів, слід приділяти увагу питанням роботи з персональними даними. Користувачі, здійснюючи бронювання, часто надають свої контактні відомості, такі як ім’я, номер телефону, адресу електронної пошти та іншу подібну інформацію. Тому система повинна бути спроможною надійно зберігати ці дані, обмежувати доступ до них та застосовувати їх виключно для виконання заявки на бронювання. Такий підхід відповідає нормам українського законодавства стосовно захисту персональних даних [8]. Крім того, оскільки

процес онлайн-бронювання туру передбачає інтерактивну взаємодію між клієнтом та постачальником послуг, при подальшому розвитку системи важливо брати до уваги положення законодавства, що регулює електронну комерцію [9].

Однією з ключових переваг обраного клієнт-серверного архітектурного рішення є потенціал для поступового розширення функціоналу системи. У майбутньому вебсайт може бути доповнений новими модулями та можливостями. Завдяки принципу поділу системи на окремі рівні, впровадження таких змін стає можливим поетапно, без необхідності повної реорганізації всього проєкту. З погляду якості програмного забезпечення, це позитивно позначається на таких характеристиках, як підтримуваність, масштабованість, зручність використання та функціональна повнота системи [18].

Таким чином, загальна архітектура туристичного вебсайту спирається на клієнт-серверну модель. цей підхід забезпечує логічне розділення системи на компонент інтерфейсу користувача, логіку серверної сторони та сховище даних. Така структура надає можливість створити зручний, адаптований та масштабований онлайн-ресурс, який може слугувати як платформою для представлення туристичних пропозицій, так і функціональним інструментом для повного процесу бронювання подорожей. Поточна реалізація клієнтської частини закладає основу для майбутнього розвитку системи, а послідовне підключення серверної частини та бази даних дозволить трансформувати вебсайт у повноцінну інформаційну систему для бронювання турів.

2.2 Проєктування структури веб сайту та навігації

Створення чіткої структури сайту та ефективної навігації є ключовим для успіху туристичного порталу. Саме від того, наскільки логічно організовані сторінки, залежить, чи буде сайт зручним у використанні, чи зможе користувач швидко знайти потрібну інформацію, та наскільки ефективною буде його взаємодія з ресурсом. Для туристичного сайту критично важливо забезпечити легкий та оперативний доступ до ключових розділів, адже відвідувачі зазвичай

заходять на такі сайти з конкретною метою: обрати напрямок для подорожі, ознайомитися з пропозиціями турів, дізнатися більше про компанію або ж зробити попереднє замовлення. Інтуїтивно зрозуміла система навігації має прямий вплив на загальне враження від користувацького досвіду [2].

У контексті розробки нашого туристичного вебсайту, інформаційна структура організована за принципом розподілу контенту по окремих тематичних сторінках. Такий підхід запобігає перенасиченню головної сторінки надмірним обсягом інформації та гарантує швидкий перехід до необхідних розділів. Основними компонентами сайту є: головна сторінка, сторінка з пропозиціями туристичних напрямків, сторінка, присвячена компанії, та сторінка з контактними даними. Кожна з цих сторінок має свою визначену функцію, а разом вони створюють цілісну та логічно обґрунтовану структуру всього веб-ресурсу.

На рисунку 2.2.1 схематично зображено узагальнену схему навігації туристичного вебсайту, яка відображає основні сторінки системи та логіку переходів між ними.

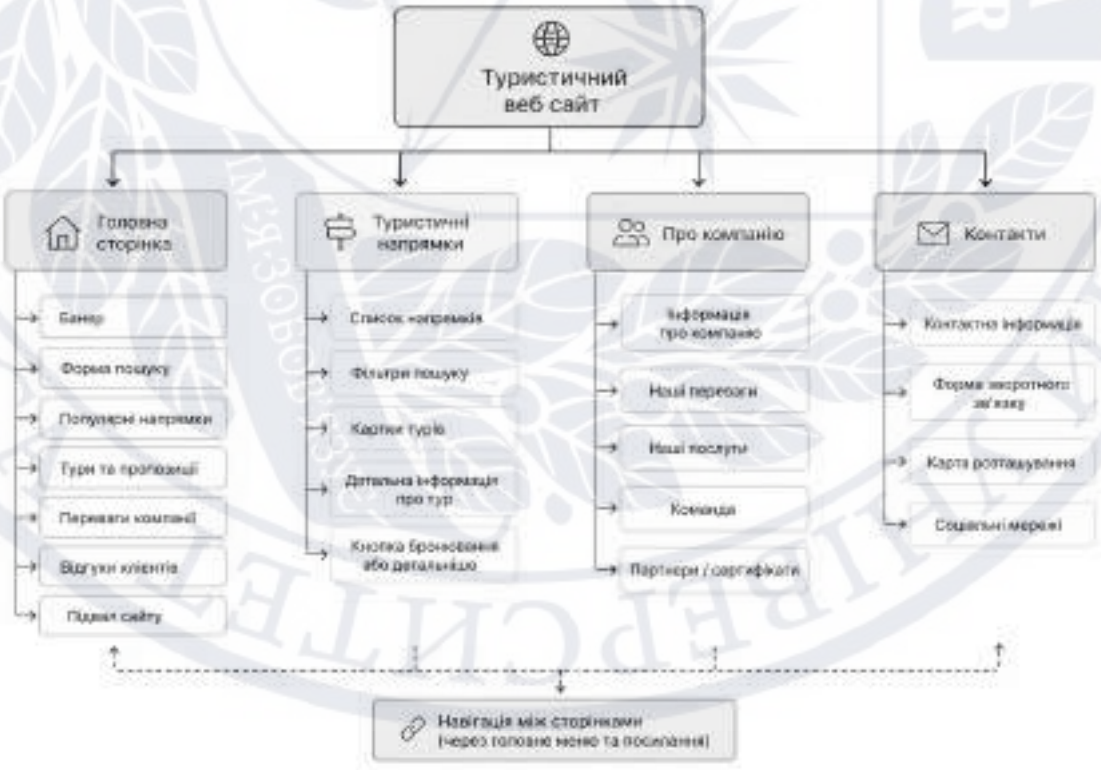

Рисунок 2.2.1 - Узагальнена схема навігації туристичного вебсайту

Цільова сторінка слугує стрижнем онлайн-ресурсу, звідки розпочинається шлях відвідувача в системі. Тут розміщено головне візуальне полотно, блок навігації, поле для пошуку мандрівок, секцію найзатребуваніших маршрутів, картки з туристичними варіантами, відомості про сильні сторони компанії, свідчення задоволених клієнтів і нижній колонтитул. Першочергова мета цієї сторінки – захопити увагу користувача, надати миттєвий доступ до найважливіших опцій та допомогти йому визначитися з туром чи почати процес резервування.

На рисунку 2.2.2 наведено фрагмент головної сторінки туристичного вебсайту, де відображено головний банер, навігаційне меню та форму пошуку подорожей, які є першими елементами взаємодії користувача із системою.

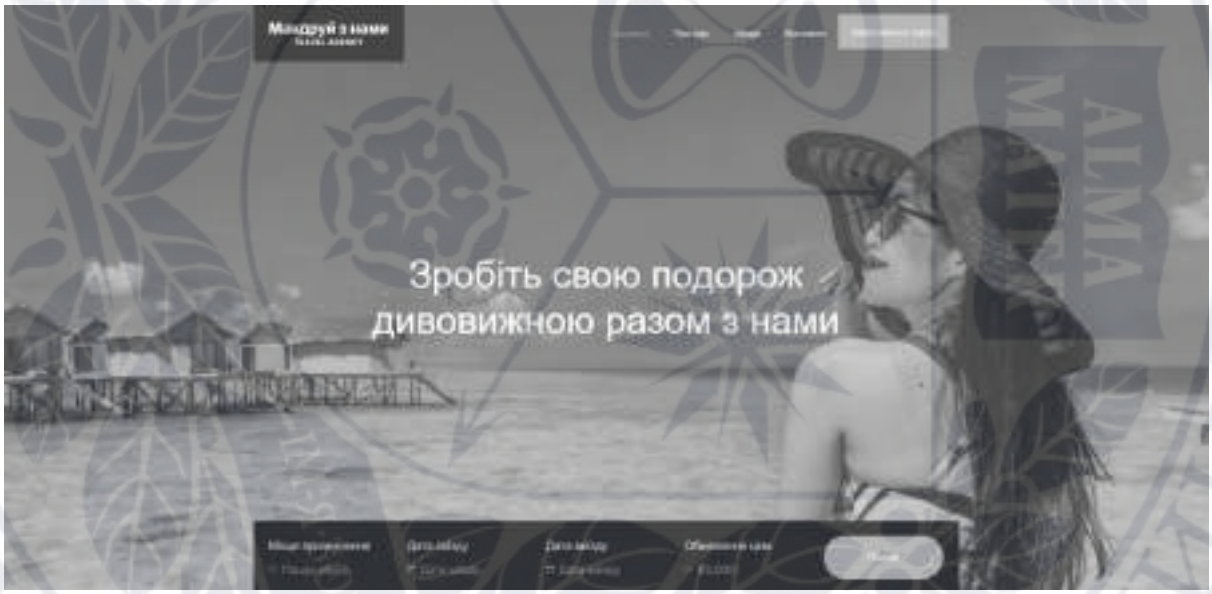

Рисунок 2.2.2 - Фрагмент головної сторінки туристичного вебсайту

На сторінці з напрямками для подорожей представлено вибір доступних варіантів для мандрівників. Відвідувачі мають змогу переглядати туристичні пропозиції, отримуючи стислу інформацію про них, включно з ціною, тривалістю мандрівки та візуальними матеріалами. Ця сторінка слугує своєрідним каталогом туристичних послуг і є невід'ємною частиною сайту, адже

саме через неї потенційні клієнти роблять перший крок до вибору конкретного туру.

Ключову роль у донесенні інформації відіграють картки туристичних пропозицій. Вони дають змогу лаконічно представити найважливіші відомості про тур: фотографію, назву місця призначення, стислий опис, тривалість поїздки, вартість та інші важливі опції. Такий підхід є надзвичайно зручним для користувача, адже дозволяє оперативно порівнювати різні пропозиції та переходити до детального ознайомлення чи оформлення бронювання. Сьогодні візуальна складова має вирішальне значення для туристичних онлайн-платформ, оскільки суттєво впливає на процес прийняття рішення користувачем [2, 16].

На рисунку 2.2.3 наведено фрагмент сторінки туристичних напрямків із картками турів, де представлено основну інформацію про доступні подорожі.

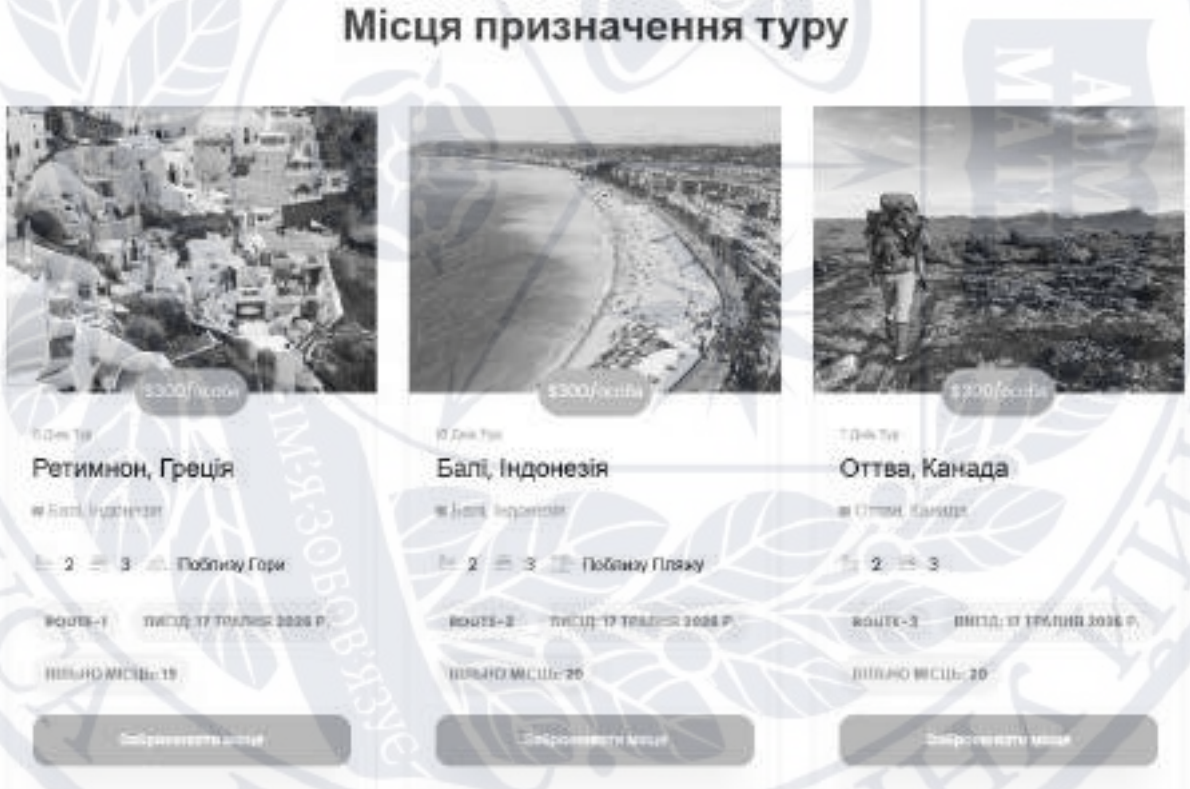

Рисунок 2.2.3 - Фрагмент сторінки туристичних напрямків із картками турів

Розділ «Про нас» слугує для надання інформації, розкриваючи деталі про туристичну агенцію, сферу її діяльності, унікальні пропозиції, набутий досвід та ключові напрямки послуг. Цей елемент є критично важливим для зміцнення

довіри відвідувача до платформи, адже у туристичній індустрії вибір часто залежить не тільки від вартості, але й від репутації та стабільності компанії [1, 4].

Розділ «Контакти» надає інструментарій для комунікації між потенційним клієнтом та туристичним бізнесом. Він охоплює контактну інформацію, включаючи фізичну адресу, номер телефону, електронну адресу, посилання на онлайн-спільноти, а також інтегровану форму для надсилання запитів. Цей розділ є невід'ємним елементом навігації, що дозволяє користувачеві негайно зв'язатися з фірмою при появі будь-яких запитань про подорожі чи процес бронювання.

Структуру основних сторінок туристичного веб сайту подано в таблиці 2.2.1.

Таблиця 2.2.1 - Основні сторінки туристичного веб сайту

Сторінка	Файл сторінки	Призначення
Головна сторінка	index.html	Представлення сайту, форми пошуку, популярних напрямків, турів і переваг компанії
Про компанію	about.html	Надання інформації про туристичну компанію, її послуги та переваги
Туристичні напрямки	destination.html	Перегляд доступних напрямків і туристичних пропозицій

Контакти	contact.html	Відображення контактної інформації та форми зворотного зв’язку

Структура вебсайту для зручного переміщення користувачів побудована на основі навігаційного меню. Його ви знайдете в шапці кожної сторінки, що гарантує постійний доступ до ключових розділів, незалежно від того, де ви перебуваєте на сайті. У меню представлені посилання на головну, секцію з напрямками подорожей, інформацію про компанію та контакти.

Унікальність навігації туристичного порталу полягає у врахуванні типових шляхів поведінки відвідувачів. Зазвичай, процес пошуку виглядатиме так: головна сторінка → ознайомлення з пропозиціями → вибір подорожі → подання заявки або зв’язок з нами. Отже, навігаційна схема повинна бути максимально оптимізованою, щоб користувач міг виконати бажану дію з мінімальними зусиллями.

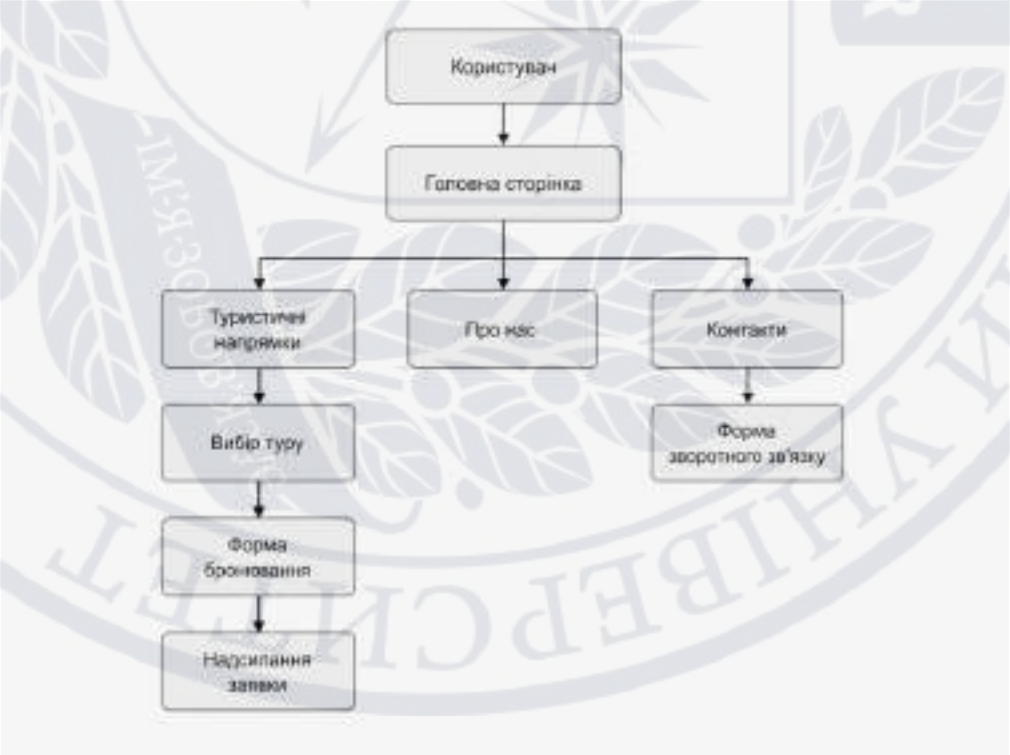

Рисунок 2.2.4 - Схема навігації користувача туристичним веб сайтом

При розробці структури за основу було взято принцип лаконічності. Кількість головних розділів є мінімальною, а їхні найменування - зрозумілими та легкими для сприйняття. Це допомагає запобігти надмірному навантаженню інтерфейсу та гарантує комфорт користування як на комп'ютерах, так і на смартфонах.

Опрацювання системи навігації також передбачає гнучкість інтерфейсу. На широких екранах меню може бути представлене у вигляді горизонтального ряду, а на компактних пристроях воно трансформується в лаконічне випадаюче меню. Такий метод забезпечує легкість у користуванні незалежно від гаджета, який використовується [6, 7].

Таким чином, побудова вебсайту для туризму базується на кількох ключових сторінках, що дають змогу користувачам повноцінно взаємодіяти з туристичними пропозиціями. Навігація вибудувана логічно, просто та з урахуванням адаптивності, що дозволяє оперативно шукати потрібні відомості та виконувати ключові дії, наприклад, ознайомлення з турами та подання запиту. Запропонована структура відповідає стандартам сучасних веб-сервісів і залишає простір для подальшого розвитку через додавання нових функціональних блоків, зокрема персонального розділу для клієнта та адміністративної панелі.

2.3 Проєктування бази даних та опис зв’язків між сутностями

Створення бази даних – це важливий етап в розробці туристичного вебсайту. Саме база даних відповідає за зберігання, впорядкування та подальшу обробку відомостей про відвідувачів, доступні туристичні тури, їх резервування, а також звернень, що надходять через форму зворотнього зв'язку. Для вебсайту, що надає послуги бронювання подорожей, база даних має не лише зберігати дані, але й забезпечувати логічні взаємозв'язки між основними елементами системи. Правильно розроблена структура даних допомагає уникнути повторень

інформації, покращити результативність запитів та гарантувати цілісність даних під час функціонування системи [17, 18].

У сфері туризму процес резервування вимагає обробки значного обсягу різноманітної інформації, такої як найменування турів, географічні маршрути, вартість поїздок, їх тривалість, кількість доступних місць, а також особистих даних клієнтів. З огляду на це, оптимальним рішенням є застосування реляційної моделі бази даних, де інформація зберігається у формі таблиць, що мають зв’язки між собою. Такий підхід забезпечує найкращу структуру даних та можливість встановлення чітких взаємозв’язків між різними елементами [3, 5, 14].

У межах розроблюваного туристичного вебсайту було спроєктовано базу даних, яка складається з шести основних таблиць: users, tours, bookings, booking_passengers, booking_seats та messages.

1. Таблиця users містить дані про користувачів системи.
2. Таблиця tours використовується для збереження туристичних пропозицій.
3. Таблиця bookings містить заявки на бронювання.
4. Таблиця booking_passengers зберігає інформацію про пасажирів у межах бронювання.
5. Таблиця booking_seats містить дані про заброньовані місця.
6. Таблиця messages використовується для збереження звернень користувачів.

Основні сутності бази даних туристичного вебсайту наведено в таблиці 2.3.1.

Таблиця 2.3.1 - Основні сутності бази даних туристичного веб сайту

Сутність	Призначення
users	Збереження даних користувачів, які реєструються на сайті або створюють бронювання

tours	Збереження інформації про туристичні пропозиції
bookings	Збереження заявок на бронювання турів
message	Збереження повідомлень, надісланих через форму зворотного зв’язку
booking_passengers	Збереження інформації про пасажирів у межах бронювання
booking_seats	Збереження заброньованих місць у межах бронювання

Для візуалізації структури бази даних та зв’язків між таблицями використано програмне середовище DBeaver. Воно дозволяє переглядати структуру таблиць, типи даних, ключі та автоматично формувати ER-діаграми. У межах роботи DBeaver застосовується для наочного відображення логічної структури бази даних туристичного вебсайту.

На рисунку 2.3.1 подано ER-діаграму бази даних туристичного вебсайту, створену в середовищі DBeaver.

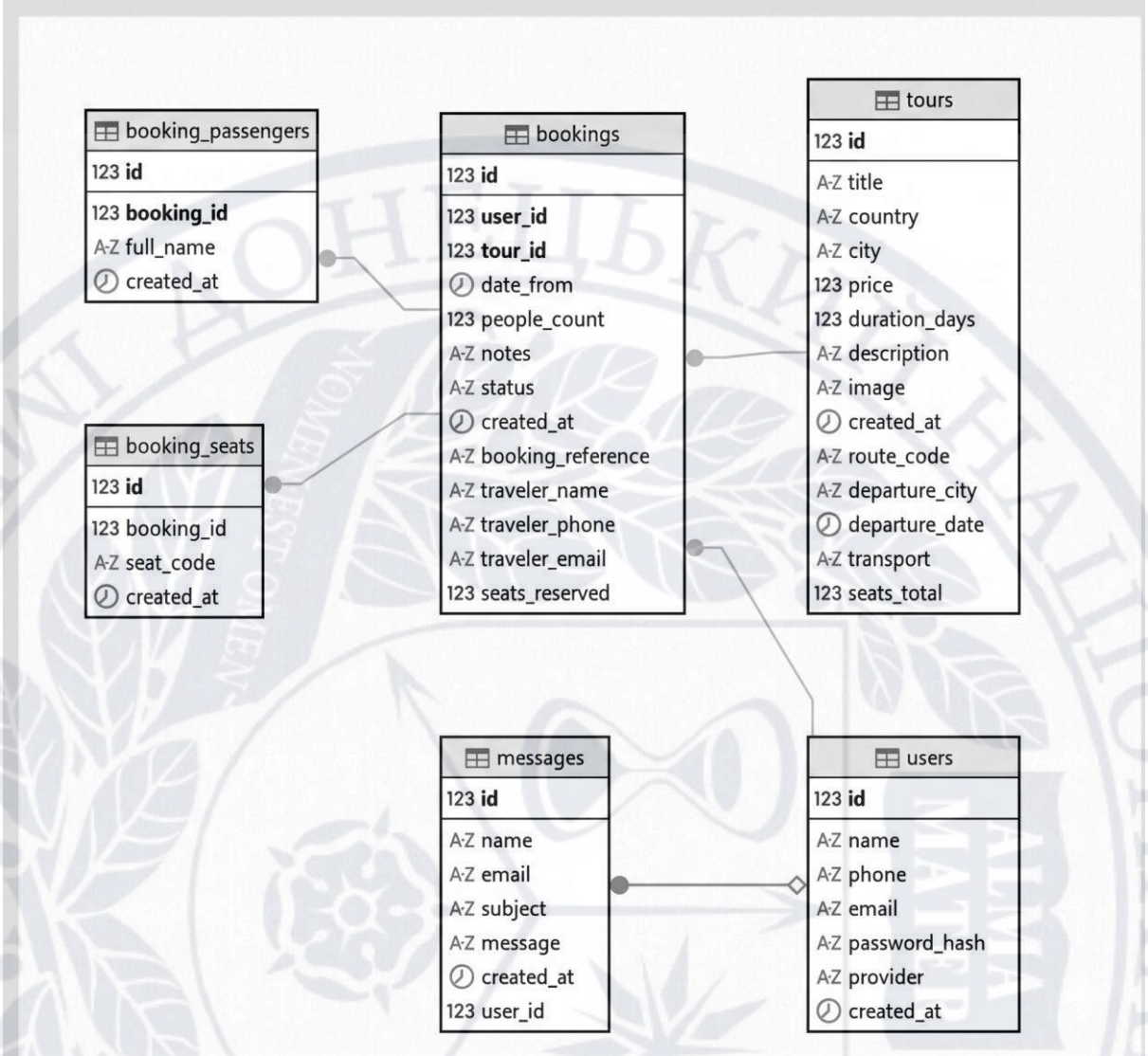

Рисунок 2.3.1 - ER-діаграма бази даних туристичного веб сайту в DBeaver

Таблиця users містить персональні дані користувачів: id, name, phone, email, password_hash, provider, created_at. Поле email має унікальне обмеження, що запобігає дублюванню акаунтів. Паролі зберігаються у вигляді криптографічного хешу, що підвищує рівень безпеки системи. Поле provider визначає тип авторизації користувача.

Оскільки таблиця users містить персональні дані, під час проєктування вебсайту необхідно враховувати вимоги інформаційної безпеки. Захист персональних даних передбачає:

1. використання хешування паролів;

2. обмеження доступу до чутливої інформації на рівні серверної частини;
3. використання захищеного протоколу HTTPS;
4. контроль доступу до адміністративних функцій;
5. мінімізацію зберігання надлишкових персональних даних.

Ці заходи відповідають вимогам законодавства України щодо захисту персональних даних [8].

Таблиця tours є основною для збереження туристичних пропозицій і містить поля: id, route_code, title, country, city, price, duration_days, description, image, departure_city, departure_date, transport, seats_total, created_at. Поле route_code є унікальним і дозволяє ідентифікувати конкретний тур.

У таблиці tours країна, місто, місто відправлення та транспорт зберігаються як текстові поля. Такий підхід спрощує структуру бази даних і є доцільним для навчального проєкту та веб сайту з обмеженою кількістю туристичних пропозицій. У разі подальшого масштабування ці дані можуть бути винесені в окремі довідникові таблиці, однак для поточної реалізації обрана структура є достатньою та зручною.

Таблиця bookings призначена для збереження заявок на бронювання турів. Вона містить унікальний ідентифікатор бронювання id, зовнішній ключ user_id, який посилається на користувача, зовнішній ключ tour_id, який посилається на обраний тур, унікальний номер бронювання booking_reference, ім’я мандрівника traveler_name, телефон traveler_phone, електронну пошту traveler_email, дату початку подорожі date_from, кількість заброньованих місць seats_reserved, додаткові примітки notes, статус заявки status та дату створення запису created_at.

Поле booking_reference має обмеження унікальності, що дозволяє ідентифікувати кожне бронювання за окремим кодом. Це зручно як для користувача, так і для працівників туристичної компанії, оскільки бронювання можна швидко знайти за його номером. Поле status за замовчуванням має значення Pending confirmation, що означає очікування підтвердження заявки. Надалі статус може змінюватися після обробки заявки працівником компанії.

У таблиці bookings додатково зберігаються контактні дані мандрівника: traveler_name, traveler_phone та traveler_email. Хоча частина цієї інформації може збігатися з даними в таблиці users, її збереження в бронюванні є практично доцільним. Це дозволяє зафіксувати контактні дані саме на момент оформлення заявки. Якщо користувач у майбутньому змінить телефон або електронну пошту у профілі, історичні дані конкретного бронювання залишаться незмінними.

Таблиця booking_passengers використовується для збереження інформації про пасажирів, які входять до конкретного бронювання. Вона містить унікальний ідентифікатор запису id, зовнішній ключ booking_id, повне ім’я пасажира full_name та дату створення запису created_at. Винесення пасажирів в окрему таблицю є правильним рішенням, оскільки в одному бронюванні може бути декілька пасажирів. Завдяки цьому не потрібно зберігати імена пасажирів одним текстовим полем у таблиці bookings.

Таблиця booking_seats призначена для збереження місць, які були заброньовані в межах конкретного бронювання. Вона містить унікальний ідентифікатор запису id, зовнішній ключ booking_id, код місця seat_code та дату створення запису created_at. Для таблиці передбачено обмеження UNIQUE (booking_id, seat_code), яке унеможливлює повторення одного й того самого місця в межах одного бронювання. Такий підхід дозволяє структурувати інформацію про місця та уникнути дублювання записів.

Таблиця messages використовується для збереження повідомлень, які користувачі надсилають через сторінку контактів. Вона містить унікальний ідентифікатор повідомлення id, необов’язковий зовнішній ключ user_id, ім’я користувача name, електронну пошту email, тему повідомлення subject, текст повідомлення message та дату створення запису created_at. Поле user_id може мати значення NULL, якщо повідомлення надсилає незареєстрований користувач. Для цього використовується правило ON DELETE SET NULL: якщо користувача буде видалено, повідомлення залишиться в базі даних, але зв’язок із користувачем буде очищено.

Структуру таблиць бази даних подано в таблиці 2.3.2.

Таблиця 2.3.2 - Структура таблиць бази даних туристичного вебсайту

Таблиця	Основні поля
users	id, name, phone, email, password_hash, provider, created_at
tours	id, route_code, title, country, city, price, duration_days, description, image, departure_city, departure_date, transport, seats_total, created_at
bookings	id, user_id, tour_id, booking_reference, traveler_name, traveler_phone, traveler_email, date_from, seats_reserved, notes, status, created_at
booking_passengers	id, booking_id, full_name, created_at
booking_seats	id, booking_id, seat_code, created_at
messages	id, user_id, name, email, subject, message, created_at

Основними зв’язками в базі даних є зв’язки між таблицею bookings та таблицями users і tours. Таблиця bookings містить поле user_id, яке є зовнішнім ключем і посилається на поле id таблиці users. Це означає, що одне бронювання належить одному конкретному користувачу, а один користувач може мати декілька бронювань. Таким чином, між таблицями users і bookings реалізовано зв’язок “один до багатьох”.

Також таблиця bookings містить поле tour_id, яке є зовнішнім ключем і посилається на поле id таблиці tours. Це означає, що кожне бронювання стосується одного конкретного туру, а один тур може мати декілька заявок на бронювання. Отже, між таблицями tours і bookings також реалізовано зв’язок “один до багатьох”.

Таблиця booking_passengers пов’язана з таблицею bookings через поле booking_id. Один запис бронювання може мати декілька пасажирів, тому між таблицями bookings і booking_passengers реалізовано зв’язок “один до багатьох”. Аналогічний зв’язок існує між таблицями bookings і booking_seats.

Таблиця messages має необов’язковий зв’язок із таблицею users. Це означає, що повідомлення може бути пов’язане із зареєстрованим користувачем, але також може бути створене без такого зв’язку, якщо звернення надсилає гість сайту.

Основні зв’язки між таблицями бази даних подано в таблиці 2.5.

Таблиця 2.3.3. - Зв’язки між таблицями бази даних

Таблиця 1	Таблиця 2	Тип зв’язку	Опис
users	bookings	Один до багатьох	Один користувач може створити декілька бронювань
tours	bookings	Один до багатьох	Один тур може мати декілька заявок на бронювання
bookings	booking_passengers	Один до багатьох	Одне бронювання може містити декілька пасажирів
bookings	booking_seats	Один до багатьох	Одне бронювання може містити декілька заброньованих місць
users	messages	Один до багатьох, необов’язковий	Один користувач може надіслати декілька повідомлень, але повідомлення може

			існувати і без користувача

На рисунку 2.3.3. подано логічну структуру зв’язків між таблицями бази даних. Така схема дозволяє наочно показати, що таблиця bookings є центральною для процесу бронювання, оскільки вона поєднує користувача, обраний тур, пасажирів і заброньовані місця.

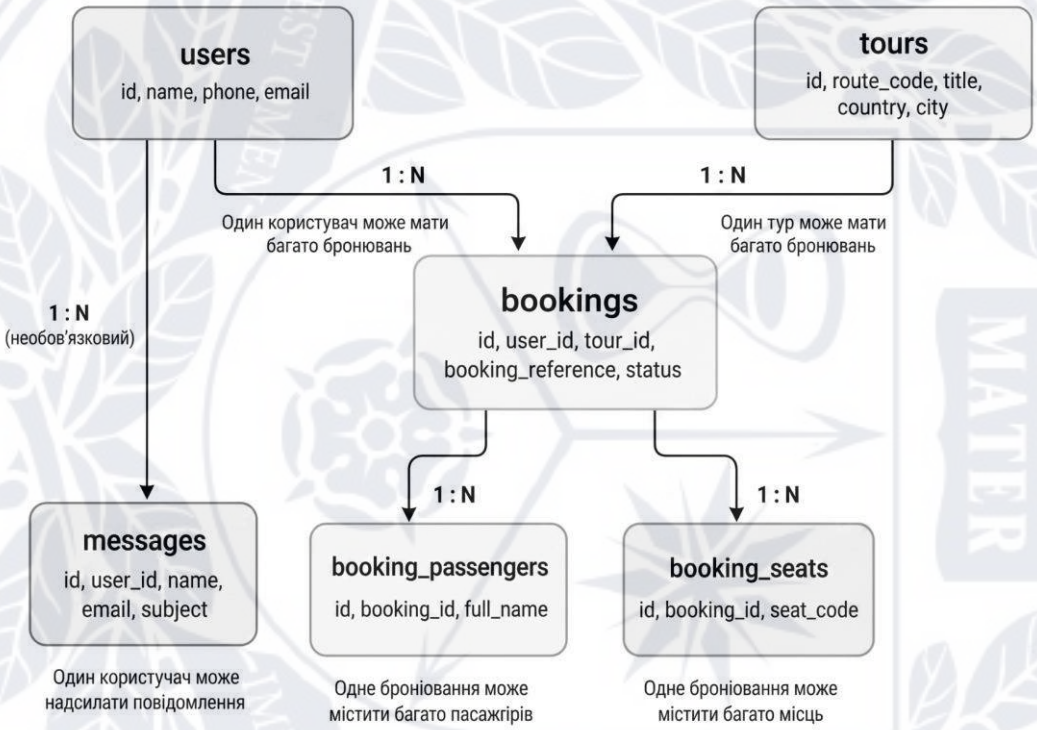

Рисунок 2.3.2 - Логічна структура зв’язків між таблицями бази даних

Під час проєктування бази даних важливо було забезпечити цілісність даних. Для цього у таблицях використовуються первинні та зовнішні ключі. Первинний ключ id дозволяє однозначно ідентифікувати кожен запис у таблиці. Зовнішні ключі user_id, tour_id та booking_id забезпечують зв’язок між користувачами, турами, бронюваннями, пасажирами та місцями.

Для зовнішніх ключів у таблицях bookings, booking_passengers і booking_seats використано правило ON DELETE CASCADE. Це означає, що у

випадку видалення користувача, туру або бронювання пов’язані з ними записи також будуть видалені автоматично. Такий підхід дозволяє підтримувати узгодженість бази даних і не залишати записи, які посилаються на неіснуючі дані.

Окремо в базі даних передбачено індекси. Індекс idx_tours_route_code створений для поля route_code, що дозволяє швидше знаходити тур за кодом маршруту. Індекс idx_tours_departure_date використовується для пришвидшення пошуку турів за датою відправлення. Індекси idx_bookings_user_id та idx_bookings_tour_id створені для швидкого пошуку бронювань за користувачем або туром. Індекс idx_bookings_reference пришвидшує пошук бронювання за унікальним номером. Індекси для таблиць booking_passengers, booking_seats і messages дозволяють ефективніше працювати з пасажирами, місцями та повідомленнями.

Особливу увагу під час проєктування бази даних потрібно приділити таблиці bookings, оскільки вона безпосередньо пов’язана з основним призначенням туристичного вебсайту - бронюванням подорожей. Ця таблиця поєднує дані про користувача, обраний тур, контактні дані мандрівника, кількість місць і статус заявки. Після заповнення форми бронювання на сайті введені дані мають передаватися на серверну частину, перевірятися та зберігатися у таблиці bookings. Після цього працівник туристичної компанії може переглянути заявку й змінити її статус.

Для коректної роботи бронювання важливо передбачити поле status, яке дозволяє відстежувати стан заявки. У поточній структурі бази даних нове бронювання автоматично отримує статус Pending confirmation. Надалі статус може бути змінений, наприклад, на “Confirmed”, “Cancelled” або “Completed”. Такий підхід дозволяє впорядкувати роботу з бронюваннями та забезпечити прозорість обробки заявок.

На рисунку 2.3.3 зображено процес збереження заявки на бронювання в базі даних. Схема демонструє послідовність дій: користувач заповнює форму бронювання, дані передаються на серверну частину, після перевірки створюється

запис у таблиці bookings, а за потреби додаються записи до таблиць booking_passengers і booking_seats.

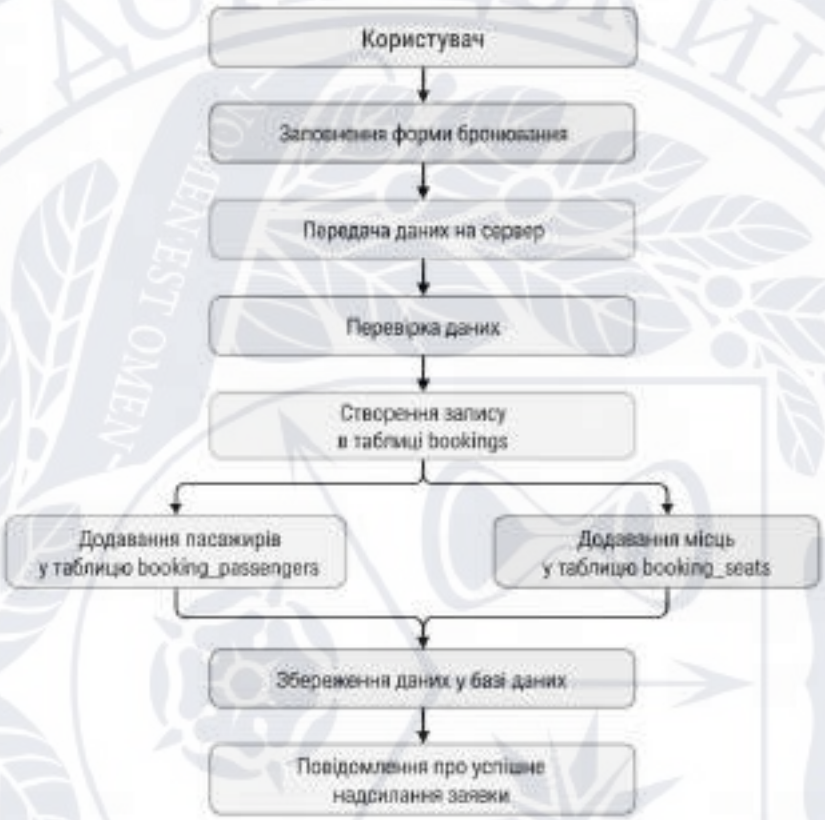

Рисунок 2.3.3 - Процес збереження заявки на бронювання в базі даних

Пропонована схема бази даних цілком підходить для втілення ключових можливостей веб-сайту туристичної компанії. Вона дозволяє вести облік відомостей про клієнтів, пропоновані тури, здійснені бронювання, пасажирів, вибрані місця та отримані повідомлення через форму зворотного зв'язку. У перспективі, ця структура готова до доповнення, зокрема, шляхом створення окремих сховищ даних для відгуків, платіжних операцій, даних адміністраторів, класифікації турів або історії змін статусу замовлень.

Отже, розробка бази даних туристичного порталу включає в себе підбір ключових таблиць, визначення їх полів і побудову взаємозв'язків. Ядром системи виступають таблиці users, tours та bookings, оскільки саме вони формують

логічний зв'язок між відвідувачем сайту, туристичним продуктом та запитом на бронювання. Таблиці booking_passengers та booking_seats надають деталізовану інформацію щодо бронювання, зберігаючи дані про осіб, що подорожують, та вибрані ними місця. Таблиця messages слугує для фіксації звернень користувачів, надісланих через контактну форму. В цілому, дана архітектура задовольняє поточні функціональні вимоги веб-ресурсу, є прозорою для подальшого обслуговування та надає гнучкість для розширення відповідно до нових потреб.

2.4 Опис алгоритмів роботи вебсайту

Опис алгоритмів функціонування туристичного онлайн-ресурсу є ключовим етапом проєктування. Саме алгоритми визначають послідовність дій, які виконує користувач, порядок обробки введеної інформації, логіку пошуку турів, формування запиту на бронювання та збереження даних у базі. Для вебсайту, присвяченого бронюванню мандрівок та подорожей, першочерговими є такі алгоритми: перегляд сторінок, пошук туру, бронювання мандрівки, збереження повідомлення з контактної форми та обробка даних на сервері.

Розробка алгоритмів надає змогу формалізувати логіку функціонування вебсайту та продемонструвати взаємодію між його окремими елементами. Це стає особливо актуальним для платформ бронювання, де клієнт має пройти низку послідовних дій: ознайомитися з доступними пропозиціями, вибрати тур, заповнити форму бронювання, надіслати запит та отримати підтвердження його отримання. Автоматизація цих процесів сприяє зменшенню часу на обробку запитів, мінімізації помилок та підвищенню комфорту при взаємодії користувача з платформою [3, 5, 14].

Загальна схема роботи туристичного вебсайту розпочинається з моменту, коли користувач потрапляє на головну сторінку. На цьому етапі йому доступні навігаційне меню, основний банер, поле для пошуку, актуальні туристичні пропозиції та інші інформаційні блоки. Звідти він може або переглянути доступні тури безпосередньо на головній сторінці, або перейти до розділу з

туристичними напрямками, або ж скористатися пошуковою формою. Така структура алгоритму дає змогу користувачеві самостійно обрати найзручніший спосіб взаємодії з ресурсом.

На рисунку 2.4.1 схематично зображено загальний алгоритм роботи туристичного вебсайту. Схема демонструє основні етапи взаємодії користувача із сайтом: відкриття головної сторінки, перегляд або пошук туру, вибір туристичної пропозиції, оформлення бронювання та отримання повідомлення про результат дії.

Рисунок 2.4.1 - Загальний алгоритм роботи туристичного вебсайту

Першим ключовим алгоритмом є той, що відповідає за пошук турів. Його функціонування починається з моменту, коли користувач заходить на головну сторінку або ж на сторінку з туристичними напрямками, де він заповнює спеціальну форму. У цій формі користувач має можливість вказати місце призначення своєї подорожі, бажану дату початку або ж вильоту, дату завершення поїздки, а також приблизний бюджет. Після того, як користувач натискає кнопку "Пошук", вказані дані повинні бути відправлені на сервер, де вони пройдуть перевірку на правильність.

У випадку, коли користувач залишив невипълненими обов'язкові поля або ввів неточну інформацію, вебсайт має повідомити його про виявлені помилки та запропонувати внести корективи. Якщо ж усі дані введені коректно, серверна частина формує відповідний запит до бази даних і здійснює пошук турів, що відповідають заданим критеріям, у таблиці tours. Пошук може бути здійснений за такими параметрами, як країна, місто, дата відправлення, вартість або ж тривалість подорожі. Відтак, знайдені пропозиції повертаються назад на клієнтську частину для відображення користувачеві у вигляді окремих карток з туристичними пропозиціями.

Алгоритм пошуку туру можна представити наступною послідовністю кроків: користувач відкриває форму пошуку, вносить необхідні параметри подорожі, активує кнопку пошуку, дані передаються на сервер, відбувається верифікація введених значень, формується запит до бази даних, здійснюється пошук релевантних турів, і, нарешті, результати виводяться на сторінці сайту. Такий методичний підхід забезпечує швидкий та інтуїтивно зрозумілий процес пошуку подорожі для кінцевого користувача.

На рисунку 2.4.2 подано алгоритм пошуку туру на туристичному вебсайті. У схемі доцільно показати послідовність дій від заповнення форми пошуку до відображення знайдених туристичних пропозицій.

Рисунок 2.4.2 - Алгоритм пошуку туру на туристичному вебсайті

Наступним важливим етапом є алгоритм оформлення туристичної путівки. Він активується тоді, коли клієнт вже визначився з конкретною подорожжю. На цьому етапі клієнт переходить до спеціального бланку, де вносить особисту інформацію про себе як мандрівника, зазначує кількість необхідних місць, додає

будь-які додаткові коментарі та, якщо це потрібно, вказує дані супутніх пасажирів. Після заповнення всіх полів, клієнт надсилає запит на бронювання.

Системна частина приймає інформацію з цього бланку та проводить її верифікацію. Першочергово проводиться перевірка заповнення обов'язкових полів: ім'я мандрівника, контактний телефон, електронна адреса, кількість місць та вибраний тур. Крім того, може здійснюватися перевірка коректності формату електронної адреси, допустимої кількості місць та наявності обраної путівки в репозиторії даних. У випадку виявлення невідповідностей, клієнту надсилається сповіщення з вимогою виправити помилки. Якщо ж усі введені дані відповідають вимогам, формується новий запис у таблиці bookings.

Після створення первинного запису щодо бронювання, залежно від обставин, можуть додаватися записи до таблиць booking_passengers та booking_seats. Таблиця booking_passengers слугує для зберігання імен осіб, які входять до складу даного бронювання, тоді як таблиця booking_seats використовується для фіксації кодів зарезервованих місць. В результаті, бронювання отримує унікальний ідентифікаційний номер booking_reference та первісний статус Pending confirmation. Цей статус вказує на те, що заявка перебуває на розгляді та очікує підтвердження від представника туристичної агенції.

На рисунку 2.4.3 зображено алгоритм оформлення бронювання туру. Схема має відображати основні етапи: вибір туру, заповнення форми бронювання, перевірку даних, створення запису в таблиці bookings, додавання пасажирів і місць, а також відображення повідомлення про успішне надсилання заявки.

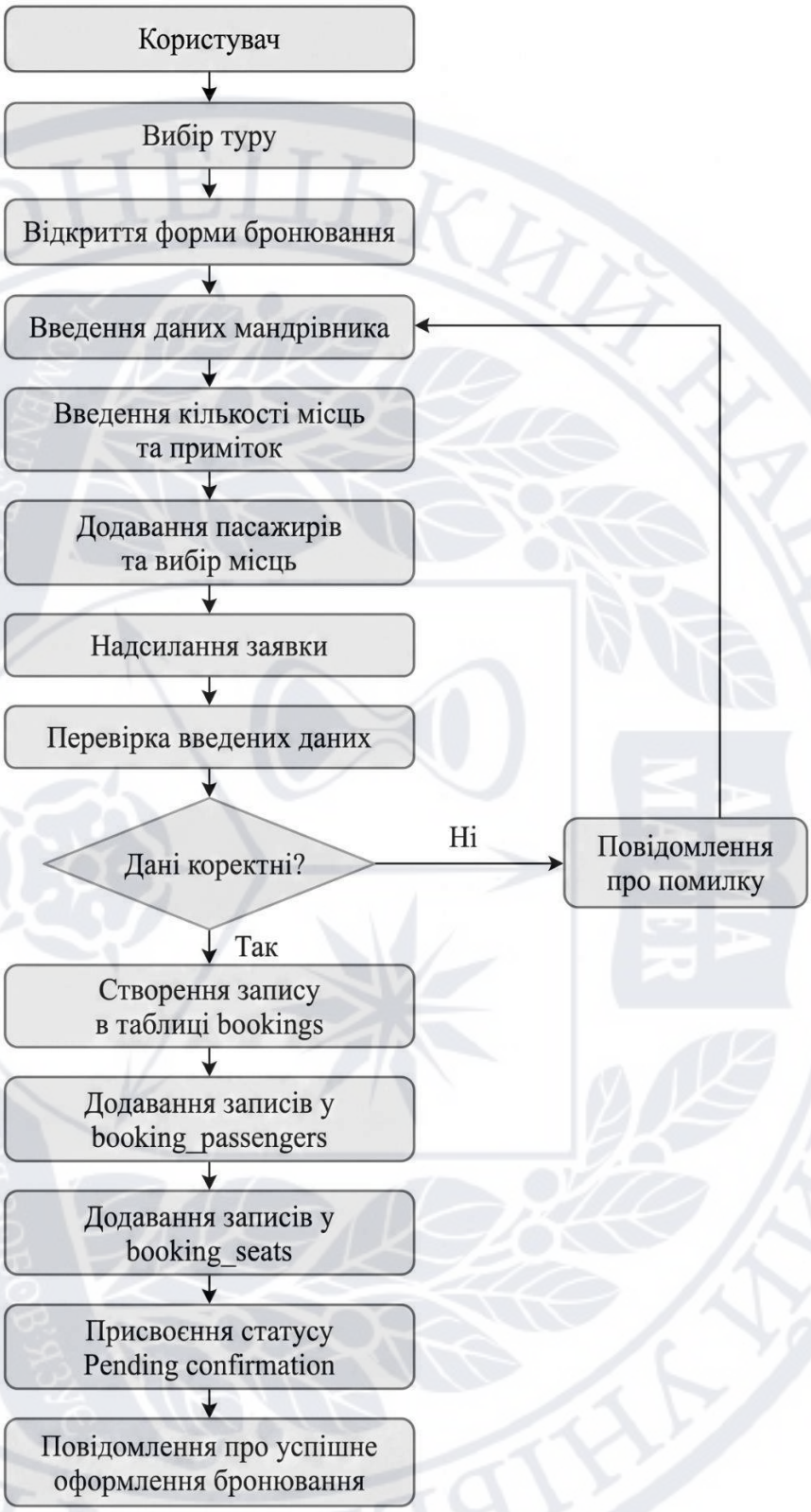

Рисунок 2.4.3 - Алгоритм оформлення бронювання туру

Обробка контактної форми є окремим етапом функціонування вебсайту. Ця функція активується, коли відвідувач бажає поставити запитання, отримати додаткові деталі щодо подорожі або зв'язатися з туристичною компанією. Хоча принцип роботи контактної форми менш складний, аніж процес бронювання, він також вимагає валідації введеної інформації та запису отриманого повідомлення до бази даних.

Відвідувач переходить на сторінку контактів, заповнює поля "Ім'я", "Електронна пошта", "Тема" та "Текст повідомлення". Після кліку на кнопку "Надіслати", дані потрапляють на сервер. Там відбувається перевірка заповнення обов'язкових полів та коректності формату електронної пошти. В разі успішної перевірки, повідомлення записується до таблиці `messages`. Якщо користувач пройшов авторизацію, його унікальний ідентифікатор може бути збережений у полі `user_id`. Для повідомлень від неавторизованих користувачів (гостей сайту) поле `user_id` залишається порожнім.

Застосування такого підходу забезпечує централізоване зберігання всіх звернень користувачів у базі даних, що значно полегшує їх подальшу обробку співробітниками туристичної компанії. Наявність контактної форми суттєво покращує користувацький досвід, дозволяючи швидко надсилати запити без необхідності використовувати зовнішні засоби комунікації.

На рисунку 2.4.4 подано алгоритм обробки повідомлення з контактної форми. У схемі можна показати послідовність дій: відкриття сторінки контактів, заповнення форми, перевірка даних, збереження повідомлення в таблиці messages і відображення повідомлення про успішне надсилання.

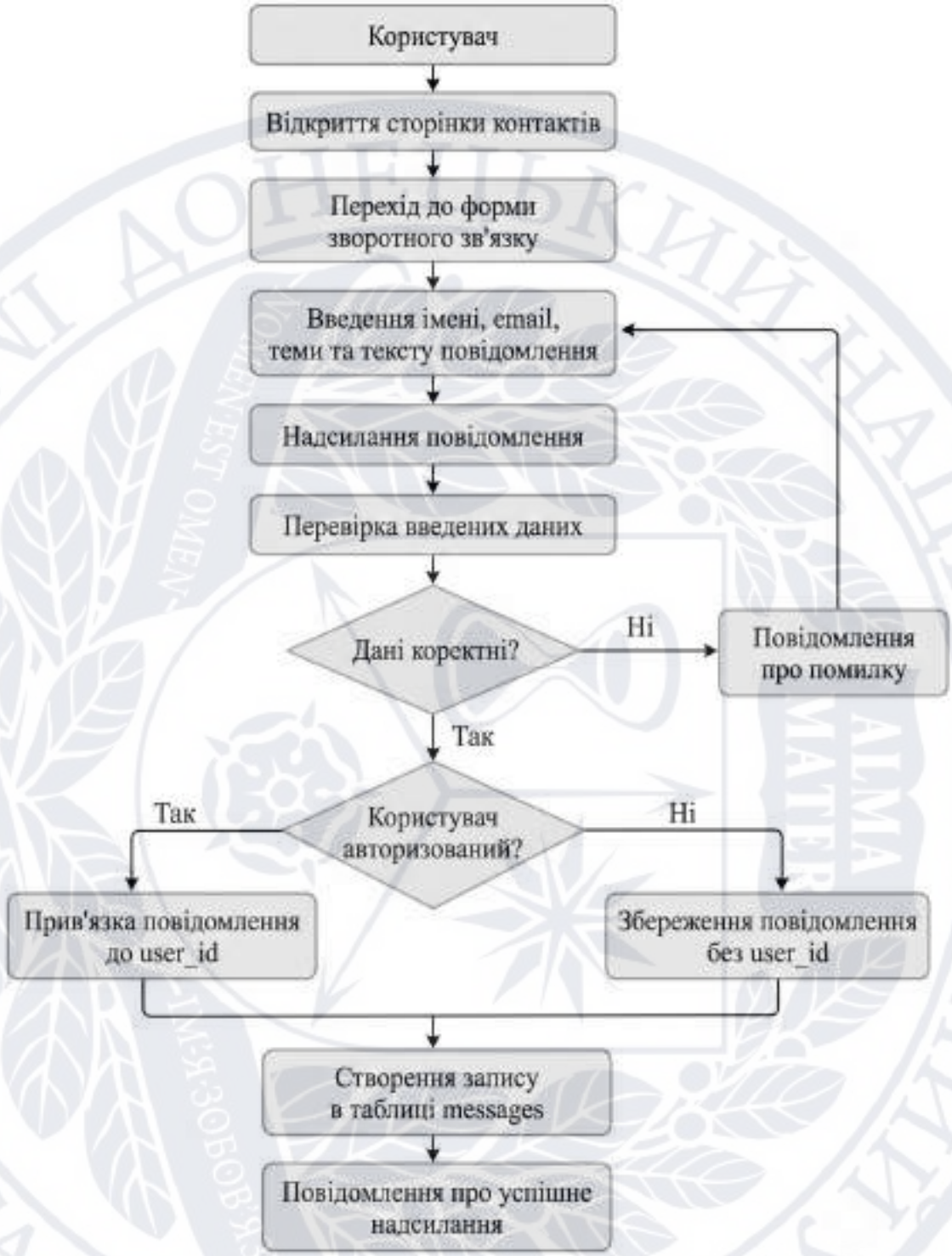

Рисунок 2.4.4 - Алгоритм обробки повідомлення з контактної форми

Беззаперечно, важливим елементом функціонування інтернет-представництва є верифікація наданої інформації. Валідація даних необхідна для

унеможливлення запису неточних чи неповних відомостей. Приміром, коли йдеться про резервування, треба перевіряти: чи контактні дані користувача внесені, чи електронна адреса вказана коректно, чи кількість вибраних місць перевищує нуль, чи обраний тур насправді існує у системі. Ця процедура підвищує надійність роботи сайту та мінімізує ризик помилок при обробці запитів.

При роботі з персональними даними відвідувачів слід пам'ятати, що ім'я, номер телефону та електронна пошта відносяться до контактної інформації. Їх слід використовувати суто з метою бронювання або для встановлення зворотного зв'язку. Пов'язано це з тим, що серверна частина повинна гарантувати належне зберігання таких відомостей, а також обмеження доступу до них, згідно з вимогами законодавства щодо захисту персональних даних [8].

Алгоритмічні механізми функціонування інтернет-порталу мають також враховувати цифрову природу процесу взаємодії як користувача, так і туристичної компанії. Після відправлення користувачем заявки, він мусить отримати чітке сповіщення про те, що його запит взято до розгляду. Згодом, працівник компанії може як підтвердити, так і скасувати цю заявку. Такий підхід цілком відповідає логіці онлайн-резервування туристичних послуг, де ключову роль відіграють прозорість дій, зрозумілість умов та можливість подальшої комунікації з клієнтом [9, 12, 13].

З точки зору взаємодії з базою даних, алгоритм бронювання стає центральним елементом, адже він забезпечує зв'язок між таблицями: users, tours, bookings, booking_passengers та booking_seats. Користувач вибирає маршрут з таблиці tours, після чого формується запис у таблиці bookings. Якщо бронювання передбачає кілька пасажирів, їхні імена будуть занесені до таблиці booking_passengers. Якщо ж користувач обирає конкретні місця, їхні ідентифікатори будуть збережені в таблиці booking_seats. Така послідовність дозволяє деталізувати процес резервування та зберігати не тільки загальну заявку, але й пов'язану з нею додаткову інформацію.

Окремо варто наголосити, що алгоритми роботи інтернет-представництва повинні бути зручними не тільки для клієнта, але й для подальшого технічного супроводу. Чітке розмежування процесів пошуку, резервування та надсилання сповіщень полегшує процеси розробки, тестування та майбутнього розширення функціоналу сайту. Зокрема, в майбутньому можна буде реалізувати особистий кабінет для користувача, історію його замовлень, можливість зміни статусу запиту, дистанційну оплату або ж адміністративну панель для перегляду всіх заявок.

Отже, алгоритми роботи туристичного вебсайту окреслюють ключові сценарії взаємодії користувача з ресурсом: пошук подорожі, вибір туристичної пропозиції, завершення процесу бронювання та надсилання повідомлення через контактну форму. Найбільш значущим є алгоритм бронювання, оскільки він забезпечує формування заявки та збереження необхідних даних у базі. Запропоновані алгоритмічні рішення дають змогу побудувати логічну, структуровану та комфортну роботу сайту, що повністю відповідає головній меті дипломного проекту – створення туристичного веб-порталу для резервування подорожей та поїздок.

Висновки до розділу 2

У другому розділі було здійснено проєктування веб-ресурсу для туристичної галузі, призначеного для бронювання подорожей та поїздок. Основна увага була зосереджена на визначенні загальної архітектури платформи, структури її сторінок, організації навігації, моделюванні бази даних та описі ключових робочих алгоритмів.

У ході проєктування було обґрунтовано вибір клієнт-серверної архітектури, яка передбачає розділення системи на клієнтську частину, серверну частину та базу даних. Такий підхід дозволяє ізолювати користувацький інтерфейс від логіки обробки даних і створює можливості для подальшого розширення функціоналу веб-ресурсу. Клієнтська частина відповідає за

візуалізацію сторінок, навігацію та взаємодію з користувачем. Серверна частина займається обробкою запитів та валідацією введених даних, тоді як база даних забезпечує зберігання інформації про користувачів, тури, бронювання, пасажирів, місця та повідомлення.

Також було проаналізовано структуру туристичного веб-ресурсу та його головні сторінки. До складу веб-ресурсу входять: головна сторінка, сторінка з туристичними напрямками, інформаційна сторінка про компанію та сторінка для зворотного зв'язку. Кожна сторінка має специфічне функціональне призначення і надає користувачеві доступ до необхідної інформації. Особливої уваги було приділено головній сторінці, формі для пошуку подорожей, карткам туристичних пропозицій та основному навігаційному меню, оскільки ці елементи формують ключовий сценарій взаємодії користувача з платформою.

В рамках проєктування бази даних було визначено такі основні таблиці: users, tours, bookings, booking_passengers, booking_seats та messages. Таблиця `users` містить дані користувачів, `tours` - інформацію про туристичні пропозиції, `bookings` - заявки на бронювання, `booking_passengers` - дані пасажирів стосовно бронювання, `booking_seats` - інформацію про заброньовані місця, а `messages` - повідомлення з форми зворотного зв'язку. Для візуального представлення структури бази даних та взаємозв'язків між таблицями використовувалася програма DBeaver.

Описані зв'язки між таблицями забезпечують цілісність даних та коректне функціонування механізму бронювання. Ключовими є зв'язки між користувачами та бронюваннями, турами та бронюваннями, бронюваннями та пасажирами, а також бронюваннями та місцями. Завдяки застосуванню первинних та зовнішніх ключів база даних підтримує логічну узгодженість записів, а індекси сприяють пришвидшенню пошуку інформації за основними полями.

Окремо було описано алгоритми функціонування туристичного веб-ресурсу. До основних алгоритмів належать: загальний алгоритм взаємодії користувача з платформою, алгоритм пошуку туру, алгоритм оформлення

бронювання та алгоритм обробки повідомлень з форми зворотного зв'язку. Найбільш вагомим є алгоритм бронювання, оскільки він відповідає за створення заявки, перевірку введених даних та збереження інформації в базі даних. Після вибору туру користувач заповнює форму бронювання; дані передаються на сервер, де проходять перевірку, після чого створюється запис у таблиці `bookings`, а за потреби додаються записи про пасажирів та місця.

Отже, другий розділ заклав проєктувальну основу для туристичного веб-ресурсу. Розроблена архітектура, структура сторінок, модель бази даних та робочі алгоритми формують фундамент для подальшої програмної реалізації сайту. Запропоновані рішення забезпечують логічну організацію веб-ресурсу, зручність його використання, можливість зберігання заявок на бронювання та подальше розширення функціональних можливостей.

РОЗДІЛ 3. РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ ТУРИСТИЧНОГО ВЕБСАЙТУ

3.1. Реалізація користувацького інтерфейсу вебсайту

Фронтенд туристичного порталу – це перше, що зустрічає відвідувача, а водночас і стартовий етап практичного втілення всієї системи. Дизайн було створено з урахуванням адаптивності та динамічності веб-ресурсу: основою стали HTML5, CSS3 з препроцесором SCSS та JavaScript. Bootstrap додали на початковому етапі, адже він суттєво пришвидшує верстку, усуває проблеми з сумісністю браузерів та надає готову сітку для компонентів.

Бекенд функціонує на Python Flask, тому для генерації сторінок логічним вибором став вбудований шаблонізатор Jinja2. Він дозволив розділити інтерфейс на окремі модулі – header.html, navbar.html, footer.html, позбавивши від потреби дублювати однаковий код на кожній сторінці. Головний шаблон base.html визначає загальний каркас: підключення стилів, скриптів та структури. Усі решта сторінок просто розширюють його.

Під час розробки було доопрацьовано кілька ключових секцій сайту:
Довідник туристичних напрямків (destination.html): Пропозиції представлені у вигляді карток, організованих в адаптивну сітку. Кожна картка є самостійним елементом, що містить зображення, вартість, тривалість подорожі, ідентифікатор маршруту, дату вильоту та кількість доступних місць. Тут же розташована кнопка "Забронювати місце", щоб користувачеві не довелося шукати, де подати запит. Вигляд сторінки показано на рисунку 3.1.1.

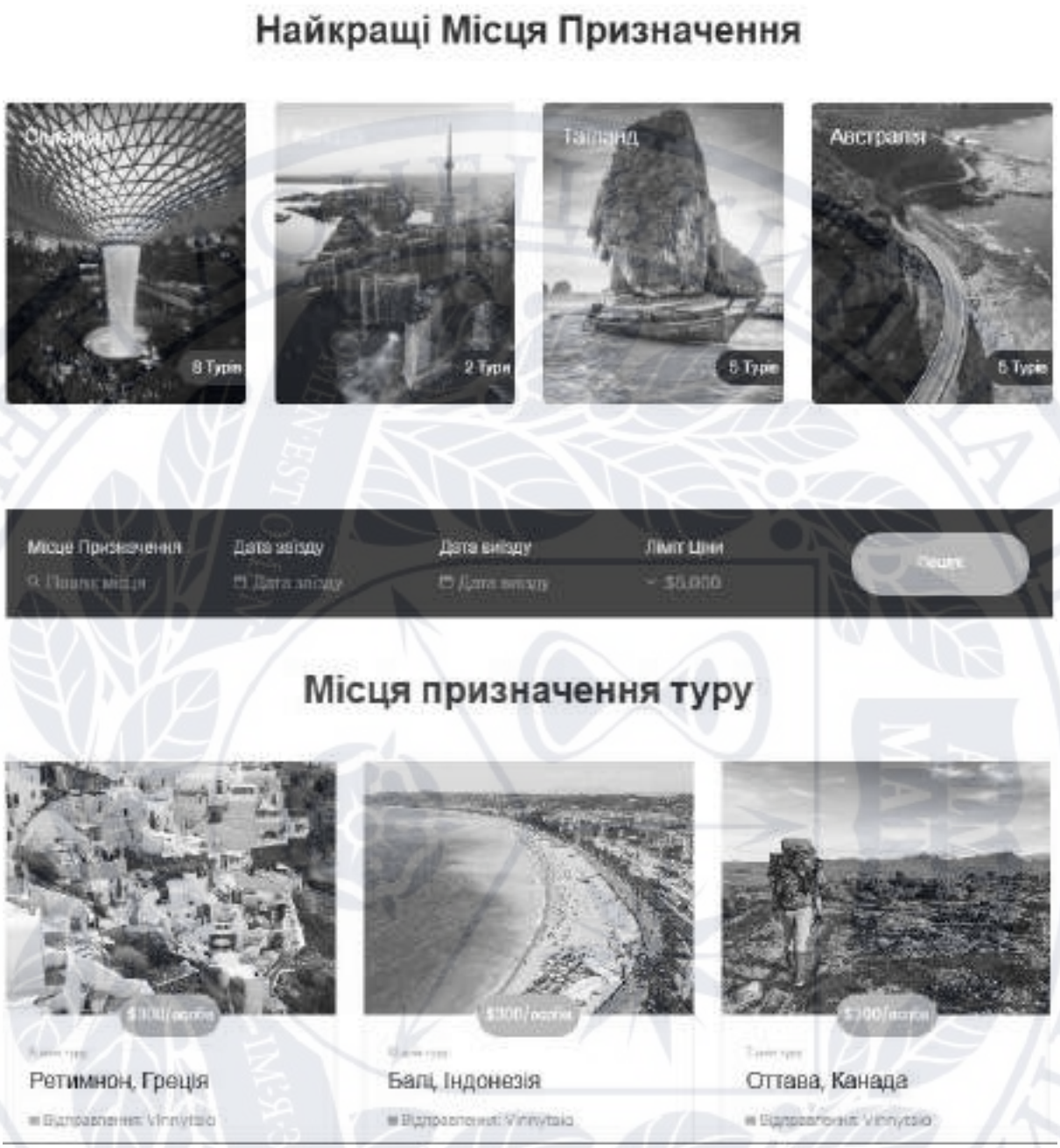

Рисунок 3.1.1 - Графічний інтерфейс сторінки каталогу пропозицій розробленого вебсайту

Особистий кабінет (cabinet.html): Закритий розділ - зайти можна лише після входу в акаунт. Інтерфейс розбитий на вкладки Bootstrap: персональні дані, коротка статистика по заявках і бронюванням, і повна таблиця з історією. У таблиці видно код бронювання booking_reference, назву туру, обрані місця, статус заявки і стан заповнення даних пасажирів. Інтерфейс кабінету показано на рисунку 3.1.2.

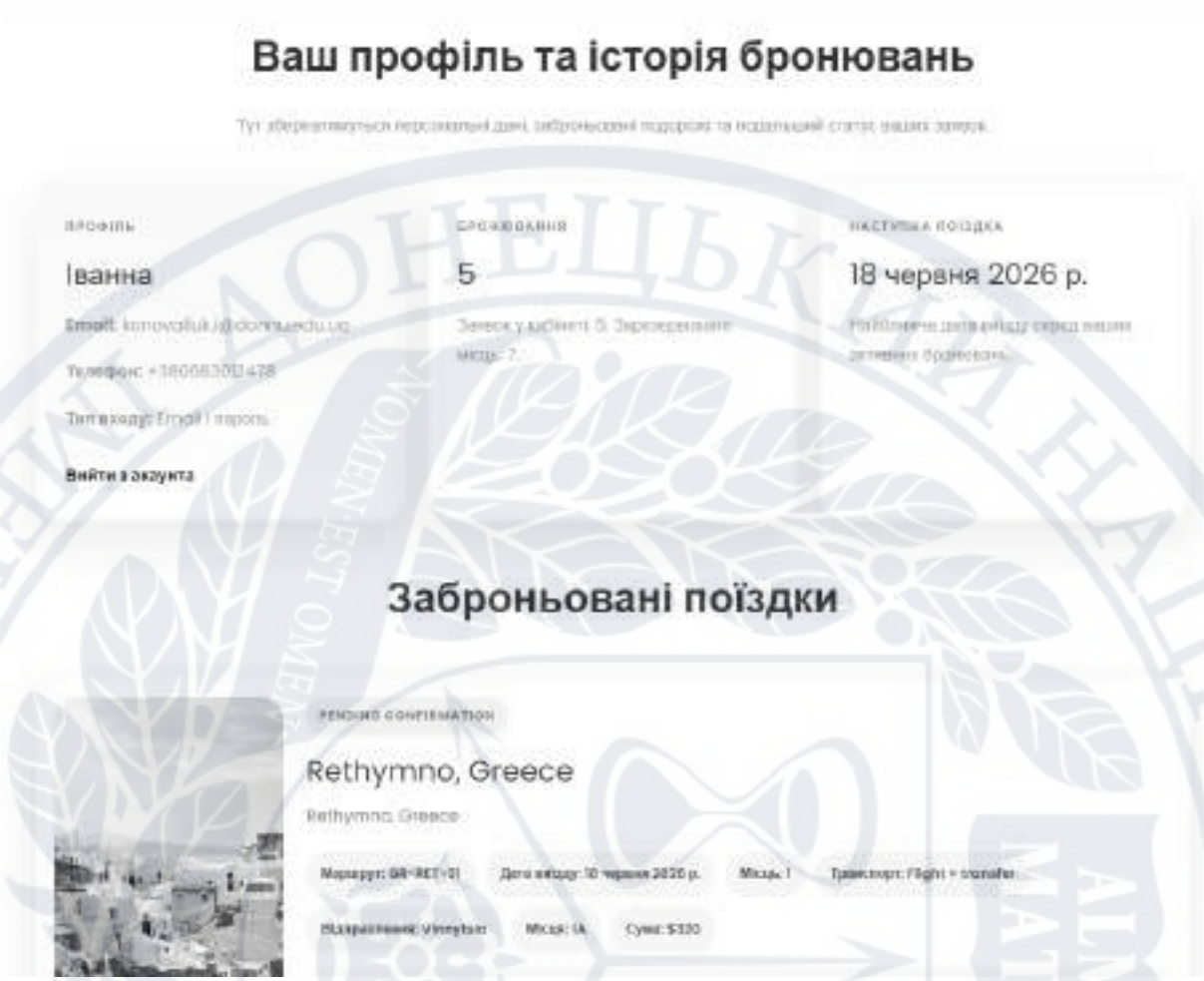

Рисунок 3.1.2 - Графічний інтерфейс особистого кабінету користувача вебсайту

Контакти та зворотний зв'язок (contact.html): Контактна інформація компанії, карта з адресою офісу і форма для повідомлень менеджерам - все на одній сторінці. Вигляд показано на рисунку 3.1.3.

Рисунок 3.1.3 - Графічний інтерфейс сторінки контактів зворотного зв'язку вебсайту

Адаптивність - одна з ключових вимог до проєкту. Сайт однаково нормально виглядає і на широкому моніторі, і на телефоні. Реалізовано це через сітку Bootstrap на базі CSS Flexbox і медіазапити. Каталог на великому екрані показує три колонки, на планшеті - дві, на смартфоні картки розгортаються на всю ширину. Навігація на мобільних згортається у кнопку-гамбургер - щоб не захаращувати обмежений простір екрана.

Для динамічності і зручності інтегрували кілька JavaScript-бібліотек:

1. jQuery - для роботи з DOM і обробки подій: кліки, введення, перемикання вкладок.
2. AOS (Animation on Scroll) - плавна поява елементів під час скролу, що робить сайт візуально живішим.
3. Owl Carousel - каруселі у блоках відгуків і популярних напрямків із підтримкою свайпів на сенсорних екранах.

4. Bootstrap Datepicker - зручний вбудований календар у формах вибору дат подорожі.

У підсумку вийшов цілісний клієнтський інтерфейс - візуально охайний, зручний у використанні і повністю готовий до підключення до серверних ендпоінтів Flask API для обміну даними і виконання бізнес-логіки.

3.2. Реалізація функціоналу бронювання та модуля безпечного розділеного введення даних пасажирів

Модуль бронювання – це серцевина вебсайту. Він забезпечує автоматизовану взаємодію між клієнтом, супутниками та базою даних. Ключовий принцип, що лежить в основі його архітектури, – відсутність статичного контенту. Весь контент генерується динамічно, враховуючи тип транспортного засобу та специфічні умови поїздки. Технічно цей модуль побудований на двох основах: асинхронних JavaScript-скриптах на стороні клієнта та обробниках Flask на сервері.

Процес бронювання починається безпосередньо з каталогу, де кожен тур прив'язаний до унікального ідентифікатора. Коли користувач переходить до форми замовлення, скрипт зчитує цей ідентифікатор і надсилає запит до бекенду. Сервер отримує необхідні дані з PostgreSQL: назву, тривалість, код маршруту та технічні характеристики транспортного засобу.

Одна з унікальних рис сайту – інтерактивна схема салону для вибору місць. Замість стандартного поля «кількість осіб» тут використовується графічний компонент. JavaScript визначає тип транспортного засобу, пов'язаного з маршрутом (автобус, вагон поїзда чи літак), і на основі цієї інформації динамічно будує відповідну сітку місць безпосередньо на екрані.

Кожне місце на схемі є окремим DOM-елементом з унікальним ідентифікатором та трьома можливими станами, які контролюються за допомогою CSS: вільне (для вибору), зайняте (заблоковане згідно з даними бази,

щоб уникнути подвійного продажу) та обране (для позначення поточного вибору користувача). Вигляд схеми показано на рисунку 3.2.1.

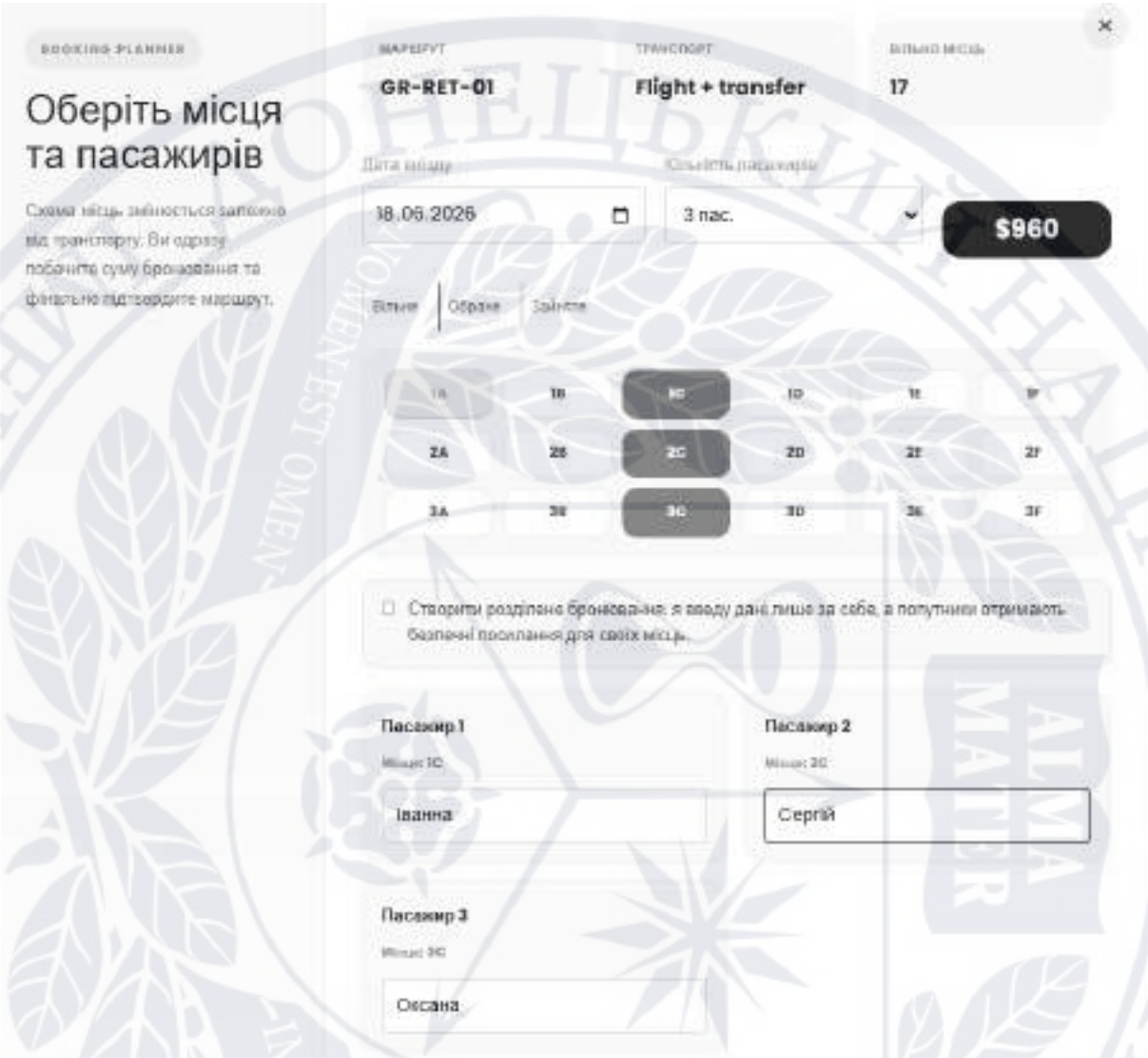

Рисунок 3.2.1 - Графічний інтерфейс підсистеми вибору місць залежно від типу транспорту на вебсайті

Водночас із візуалізацією схеми відбувається автоматичне відслідковування вибору місць. Кількість обраних користувачем крісел безпосередньо відповідає числу пасажирів, що подорожують. Для кожного зарезервованого місця система миттєво формує в формі окремий блок із полями для внесення анкетних даних особи, яка подорожує.

Однак, тут з'являється певна практична складність: замовник не завжди володіє інформацією про своїх супутників на момент здійснення бронювання.

Для вирішення цієї ситуації передбачено спеціальний функціональний модуль — Split-Booking, який дозволяє роздільно вводити дані пасажирів. У формі є специфічна позначка «Розділене бронювання», встановлення якої впливає на подальшу взаємодію як на стороні клієнта, так і на стороні сервера. Це показано на рисунку 3.2.2.

Рисунок 3.2.2 - Інтерфейс підключення модуля розділеного введення даних пасажирів на вебсайті

У розділеному режимі головний замовник заповнює лише своє ПІБ для першого місця. Поля для інших крісел залишаються порожніми. Після натискання «Завершити бронювання» сформований JSON-пакет іде POST-запитом на сервер, де проходить кілька етапів перевірки:

1. Валідація структури - перевіряється наявність обов'язкових полів, коректність ідентифікатора туру і масиву обраних місць.
2. Перевірка формату email - контактна адреса перевіряється регулярним виразом на відповідність стандартам.
3. Захист від race condition - сервер відкриває ізольовану транзакцію і повторно перевіряє, чи не встиг хтось інший зайняти ці місця за долі секунди до поточного запиту.

Якщо всі перевірки пройдено, в таблиці замовлень створюється новий запис з унікальним буквено-цифровим кодом і початковим статусом Partially filled.

Далі відбувається ключова частина механізму: для кожного незаповненого місця Flask-алгоритм генерує унікальний криптографічний токен на базі UUIDv4. Токен записується в базу і слугує ключем доступу. На основі цих токенів формуються окремі посилання для кожного попутника - вони з'являються на екрані після завершення замовлення і дублюються в особистому кабінеті, що показано на рисунку 3.2.3.

Рисунок 3.2.3 - Відображення генерованих токенів доступу для розділеного заповнення анкет на вебсайті

Попутник відкриває своє посилання у браузері. Сервер приймає токен, перевіряє його в базі і - якщо він валідний - повертає спрощену сторінку. Там видно лише назву туру, маршрут, дату виїзду і конкретне місце, яке за ним заброньоване. Єдине активне поле - ПІБ. Нічого зайвого. Вигляд цієї сторінки показано на рисунку 3.2.4.

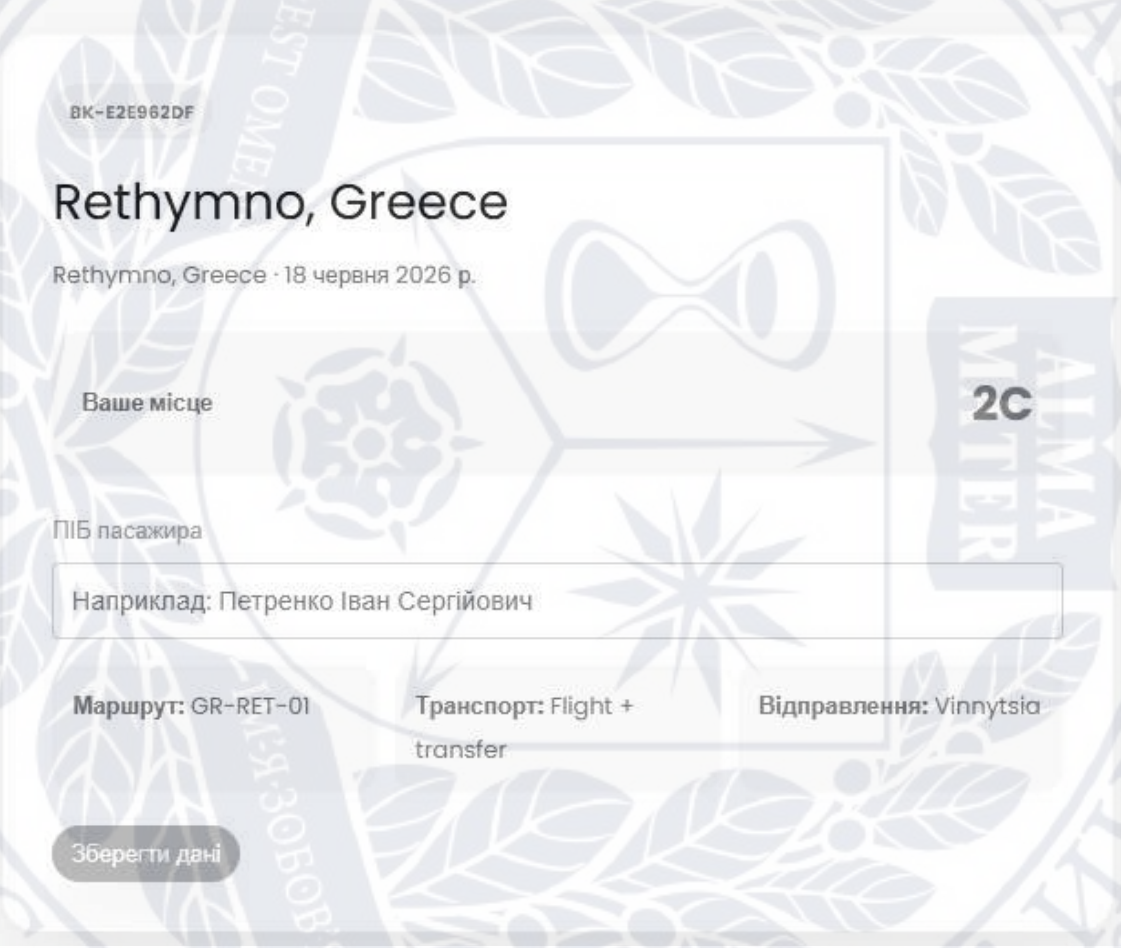

Рисунок 3.2.4 - Вебсторінка самостійної реєстрації персональних даних супутником на вебсайті

Після того як попутник зберігає свої дані, бекенд перевіряє загальний стан замовлення: порівнює кількість заповнених анкет із кількістю заброньованих місць. Коли останній попутник вносить своє ПІБ - статус бронювання автоматично змінюється з Partially filled на Ready for processing. Весь прогрес -

хто вже заповнив анкету, хто ще ні - відображається в особистому кабінеті замовника в реальному часі, що видно на рисунку 3.2.5.

Рисунок 3.2.5 - Моніторинг стану розділеного бронювання в особистому кабінеті клієнта на вебсайті

Реалізований механізм вирішує конкретну практичну задачу: збір даних пасажирів при групових поїздках більше не вимагає, щоб один замовник знав усе про всіх. Кожен вносить свої дані сам, у зручний час, через захищене персональне посилання.

3.3. Програмна реалізація адміністративного модуля аналітики та прогнозування завантаженості вебсайту

Для забезпечення комфортної роботи менеджерів з системою, замість ручного введення даних, у структуру веб-сайту інтегровано приватну адміністративну панель. Цей блок виступає як внутрішній інструмент для

ефективного управління операціями та підтримки прийняття рішень [5, 11]. Доступ до панелі обмежений і базується на рольовій моделі управління доступом (RBAC) [2], що вимагає авторизації за допомогою специфічних облікових даних, зафіксованих у конфігураційних файлах системи безпеки. Успішна верифікація логіна та пароля призводить до того, що Flask вважатиме сесію адміністративною та надасть доступ до захищеного ресурсу admin.html [2, 14].

Інтерфейс адмін-панелі розділений на два основні функціональні модулі: модуль для оперативного управління процесом бронювань та аналітичний інформаційний щиток (дашборд), що включає функціонал прогнозування попиту [11, 15].

Перший блок - операційний моніторинг замовлень. Менеджер бачить усі транзакції в одній інтерактивній таблиці [5]. Бекенд збирає дані SQL-запитами одразу з кількох таблиць: bookings, booking_passengers і booking_seats [14]. У таблиці відображається:

1. унікальний код замовлення (booking_reference),
2. персональні та контактні дані головного замовника,
3. назва туристичного маршруту,
4. коди зарезервованих місць у транспорті,
5. список зареєстрованих пасажирів,
6. індикатор прогресу заповнення розділеного замовлення (Split-Booking).

Окремо реалізоване керування станами замовлень [14]. Залежно від того, що робить клієнт і його попутники, а також від рішень менеджера, статус бронювання змінюється:

1. Partially filled - початковий стан розділеного режиму: попутники ще не внесли свої дані [13];
2. Ready for processing - автоматично виставляється Flask, коли останній пасажир заповнив анкету;
3. Confirmed - менеджер виставляє вручну після перевірки даних;
4. Awaiting payment - проміжний стан перед оплатою;
5. Cancelled і Completed — фінальні архівні стани [5].

Вигляд таблиці операційного моніторингу показано на рисунку 3.3.1.

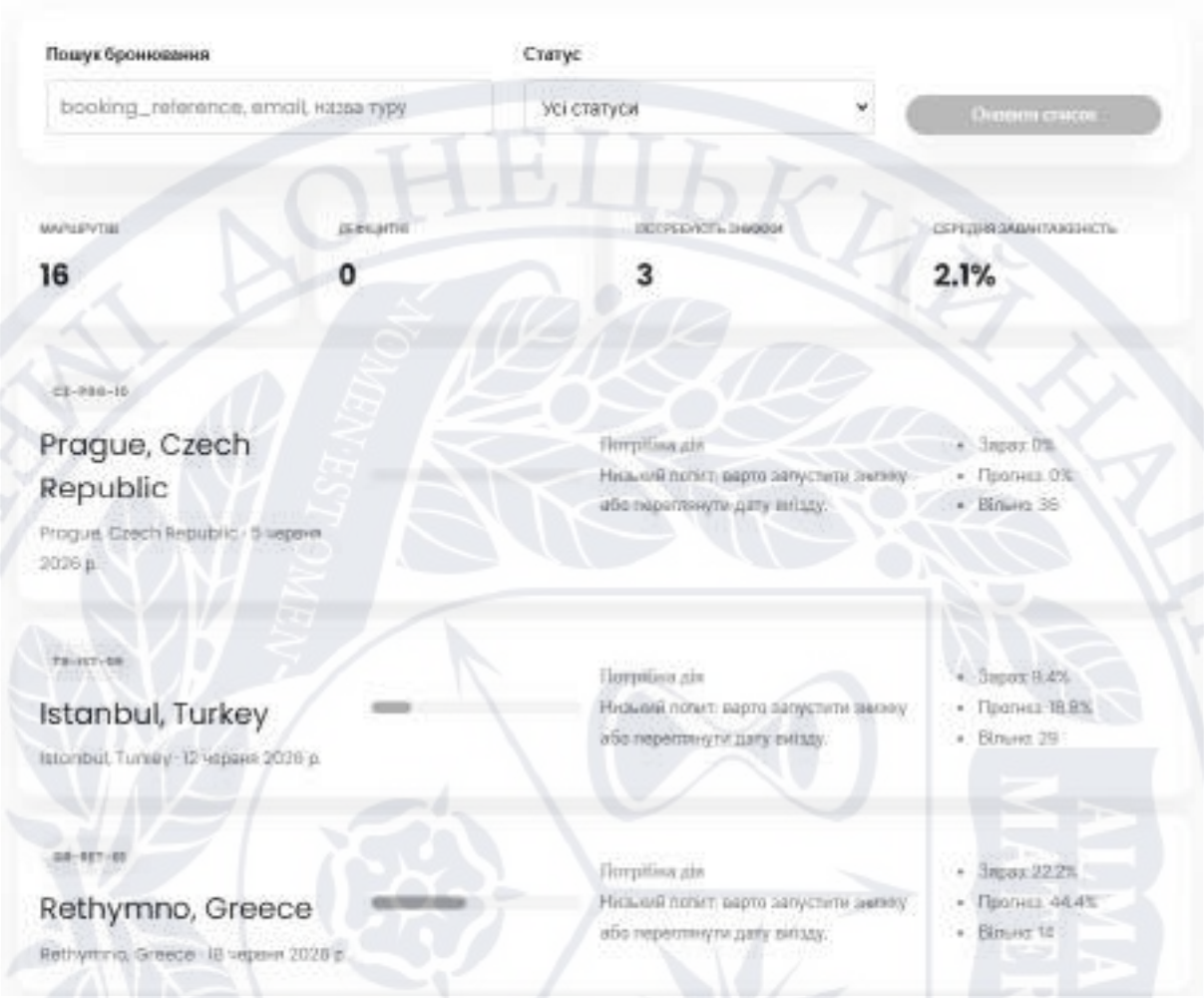

Рисунок 3.3.1 - Інтерфейс операційного керування бронюваннями в адміністративній панелі вебсайту

Для наочного представлення логіки функціонування підсистеми керування замовленнями та візуалізації моделі кінцевого автомата, на якому базується бізнес-логіка серверної частини вебсайту, було розроблено діаграму станів та переходів. Графічну модель життєвого циклу бронювання від моменту його первинної ініціалізації користувачем до фінальної архівації або скасування менеджером компанії наведено на рисунку 3.3.2.

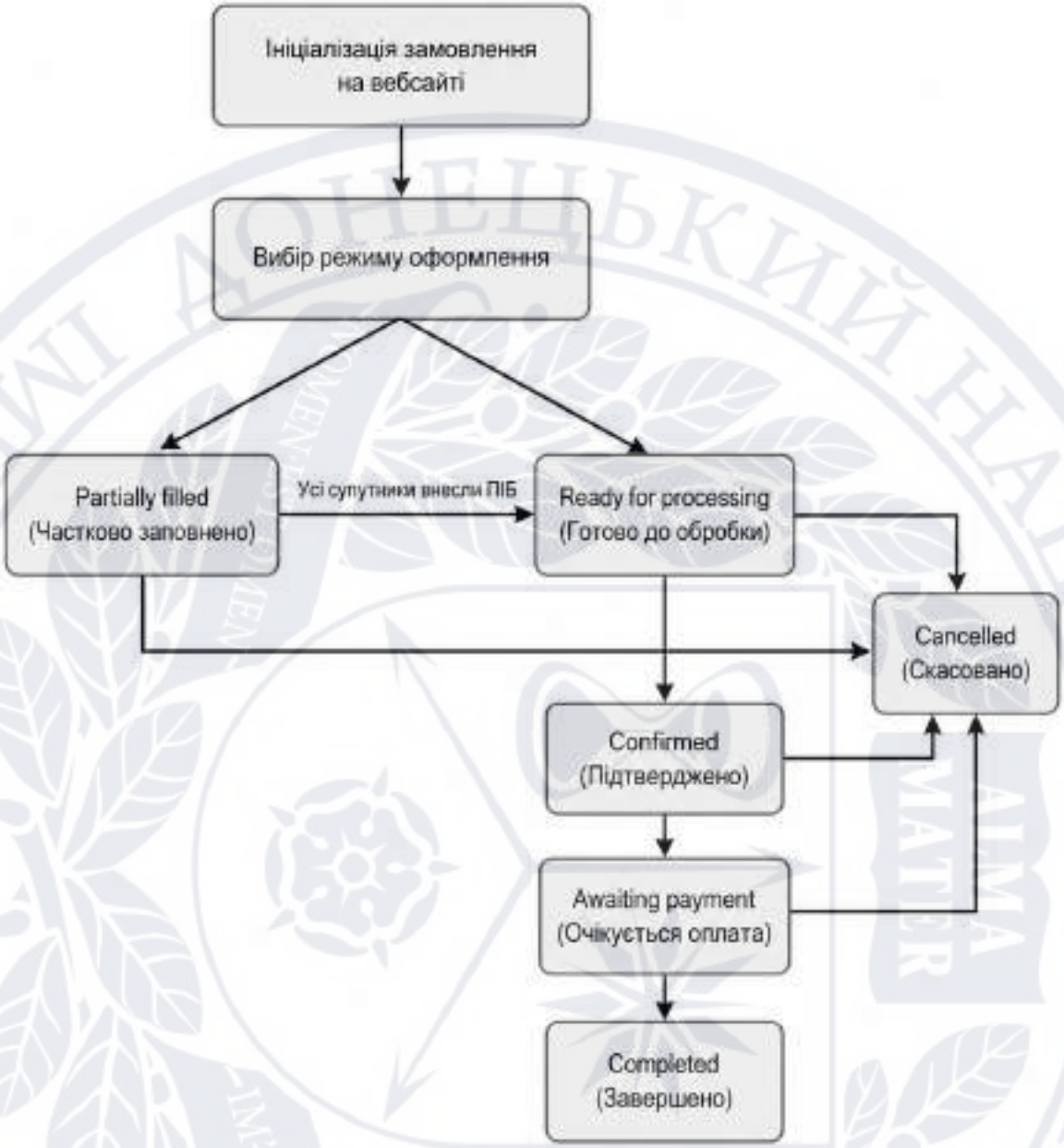

Рисунок 3.3.2 - Діаграма станів та переходів життєвого циклу бронювання подорожей

Після запису змін у статусах заявки до бази даних, менеджер набуває змоги повноцінно контролювати кожен етап життєвого циклу запиту. Інтерфейс таблиці оперативного моніторингу... (далі за твоїм текстом)

Другий елемент - аналітична панель. Це найпривабливіша секція адміністративної панелі з погляду можливостей [12, 15]. Алгоритми на стороні сервера в реальному часі опрацьовують інформацію з PostgreSQL щодо

запланованих рейсів [14] та розраховують середній рівень заповнюваності кожного транспортного засобу, тобто співвідношення вже проданих або зарезервованих місць до їх загальної кількості.

На підставі цих даних, алгоритм порівнює поточну завантаженість, динаміку надходження нових заявок та кількість днів до відправлення, і автоматично визначає для кожного туру один з трьох можливих статусів [11, 15]:

"Дефіцит" - місця реалізуються швидше, ніж очікувалося, залишається мало вільних. Сигнал для менеджера: попит високий, є можливість збільшити вартість [12].

"Потрібні дії" - продажі надзвичайно мляві, а дата виїзду наближається. Підказка: час для запуску рекламної кампанії, запровадження знижок або обмірковування скасування рейсу з метою мінімізації збитків [15].

"Стабільно" - ситуація відповідає плану, жодних невідкладних заходів не потребує [11].

Вигляд аналітичного дашборду з графіками і прогностичними статусами показано на рисунку 3.3.2.

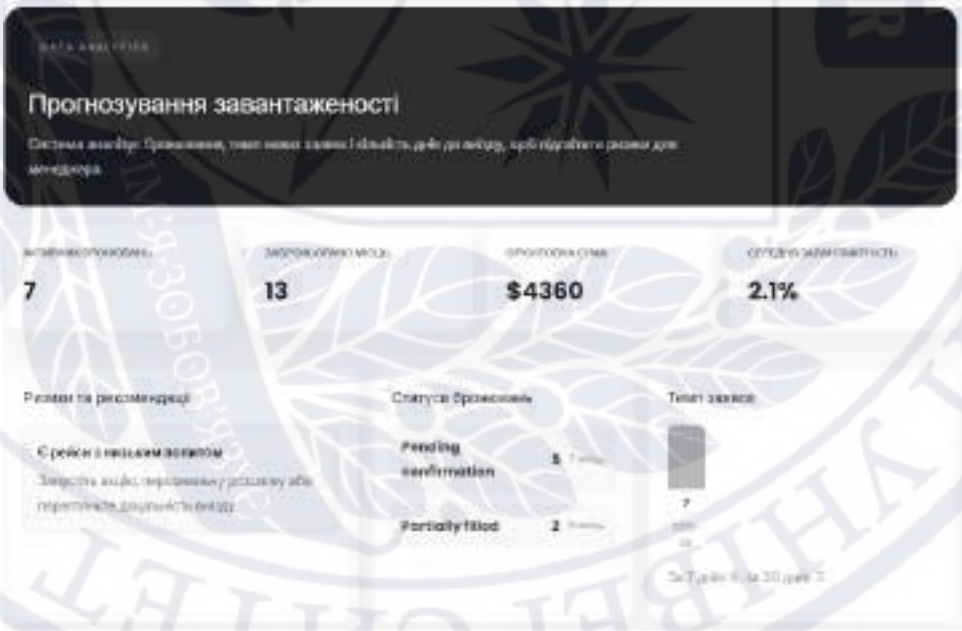

Рисунок 3.3.2 - Інтерфейс аналітичної панелі прогнозування завантаженості рейсів на вебсайті

Завдяки реалізації адміністративного модуля сайт перестав бути просто формою для збору заявок. Тепер це повноцінний інструмент керування туристичним бізнесом: менеджер бачить стан кожної транзакції, отримує автоматичні підказки по кожному рейсу і може приймати рішення на основі реальних даних, а не інтуїції [5, 15].

3.4. Реалізація збереження даних користувачів і транзакцій у базі даних

Наступний етап - підключення серверної частини сайту до реляційної СУБД PostgreSQL [2, 5]. Цей модуль відповідає за те, щоб уся інформація зберігалася надійно і цілісно: облікові записи користувачів, характеристики маршрутів, повідомлення зі сторінки контактів і всі транзакційні записи про бронювання та пасажирів [11, 14].

Flask взаємодіє з PostgreSQL через низькорівневий драйвер-адаптер [8]. Конфігурація доступу до бази побудована з урахуванням сучасних вимог безпеки [2]: жодних паролів і параметрів підключення у вихідному коді - усе зчитується з змінних оточення під час запуску. Це відокремлює конфіденційні дані від коду і унеможливлює їх витік при розгортанні [5, 14].

Модуль збереження даних розбитий на кілька пов'язаних підсистем [8, 11]: підсистема збереження даних користувачів. Відповідає за реєстрацію та авторизацію [2]. Пароль ніколи не зберігається у відкритому вигляді - перед записом він проходить одностороннє криптографічне хешування з випадковою сіллю на стороні сервера [5]. У таблицю потрапляє вже готовий хеш, тому навіть у разі злому бази реальні паролі залишаються недоступними [2, 8]. Вигляд таблиці користувачів у СУБД показано на рисунку 3.4.1.

	id [PK] bigint	name character varying (255)	phone character varying (50)	email character varying (255)	password_hash text
1	1	Іванна	+380683012478	konovaliuk.ig@donnu.edu.ua	scrypt:32768:8:1
2	2	Менеджер системи		manager@travelagency.lo...	scrypt:32768:8:1

Рисунок 3.4.1 - Відображення структури та даних таблиці користувачів у СУБД під час роботи вебсайту

Система керування зверненнями. Опрацьовує форми, що надсилаються з контактної сторінки [11]. Якщо повідомлення надходить від зареєстрованого користувача, воно автоматично пов'язується з його обліковим записом за допомогою зовнішнього ключа [2]. У випадку надсилання анонімним відвідувачем, поле ідентифікатора залишається порожнім, проте зміст звернення та контактна інформація все одно зберігаються для менеджера [14].

Система обробки транзакцій бронювання. Найбільш комплексний елемент модуля, особливо під час використання режиму Split-Booking, коли одне бронювання впливає одночасно на кілька пов'язаних таблиць [5, 14]. Для забезпечення незмінної цілісності даних, усі операції виконано згідно з принципами ACID [2, 8].

Коли користувач завершує оформлення, сервер відкриває єдину транзакцію і послідовно виконує записи:

1. спочатку створюється головний рядок у таблиці замовлень із кодом booking_reference і початковим статусом [14];
2. потім у таблицю місць записуються заброньовані крісла під конкретний рейс [8];
3. паралельно вносяться наявні дані пасажирів [14].

Якщо всі операції завершено успішно, сервер ініціює команду COMMIT, що призводить до остаточного збереження даних [8]. Однак, у разі виникнення будь-яких збоїв на будь-якому з етапів, примусово активується ROLLBACK. Це гарантує повернення бази даних до її первинного стану, виключаючи можливість збереження будь-яких неповних або неузгоджених записів [2, 14].

Було знайдено рішення для типової проблеми "race condition", яка виникає, коли двоє користувачів одночасно намагаються забронювати один і той самий квиток [8]. Для цього на рівні схеми бази даних було впроваджено унікальне обмеження для комбінації ідентифікатора туру та коду місця [2]. Додатково, в

транзакційних запитах на стороні бекенду використовується механізм блокування рядків [14]. Як наслідок, система управління базами даних (СУБД) обробляє запит першого користувача, а другому для того ж місця видає повідомлення про його зайнятість [8, 14].

Приклад пов'язаних записів по одному замовленню в таблицях bookings, booking_seats і booking_passengers показано на рисунку 3.4.2.

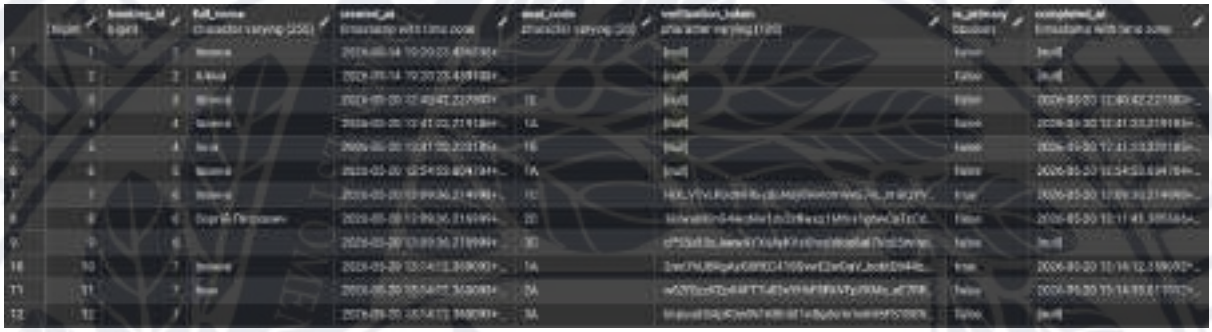

Рисунок 3.4.2 - Результат транзакційного збереження даних групового бронювання в СУБД вебсайту

У режимі Split-Booking база також зберігає криптографічні токени [2, 13]. Згенеровані UUID записуються у відповідні рядки пасажирів [13]. Коли попутник переходить за своїм посиланням, сервер шукає запис саме за цим токеном, перевіряє його актуальність і після введення ПІБ оновлює рядок у таблиці пасажирів [13, 14]. Агрегатні функції бази в реальному часі відстежують, скільки анкет уже заповнено, і як тільки надходить остання - статус замовлення автоматично змінюється на готовий до обробки [14, 15].

3.5. Комплексне тестування працездатності вебсайту

Фінальним актом у процесі створення є всебічне тестування кінцевого продукту. Згідно з актуальними методологіями розробки програмного забезпечення, тестування вважається необхідним етапом, оскільки воно дозволяє виявити відхилення між фактичною поведінкою системи та її очікуваним

функціонуванням відповідно до специфікацій, а також оцінити її стійкість та безпеку [2, 5]. Зважаючи на багатокомпонентну клієнт-серверну структуру веб-сайту та розподілену обробку транзакцій, було застосовано комплексний підхід, що включає функціональне, інтеграційне та регресійне тестування [5, 8, 11].

Початковий етап включав перевірку механізмів автентифікації та захисту сеансів. Ця перевірка здійснювалася за принципом "чорної скриньки" [5]: навмисно вводилися дані, що не відповідали встановленим правилам – адреса електронної пошти без необхідних елементів, пароль, який не задовольняв критеріям складності, а також робилася спроба реєстрації користувача з уже наявним обліковим записом. У всіх подібних випадках контролери Flask успішно перехоплювали некоректні запити, повертаючи відповідні повідомлення про помилку валідації, котрі негайно відображалися користувачеві у вигляді попереджень на інтерфейсі [2, 13]. Результат тесту показано на рисунку 3.5.1.

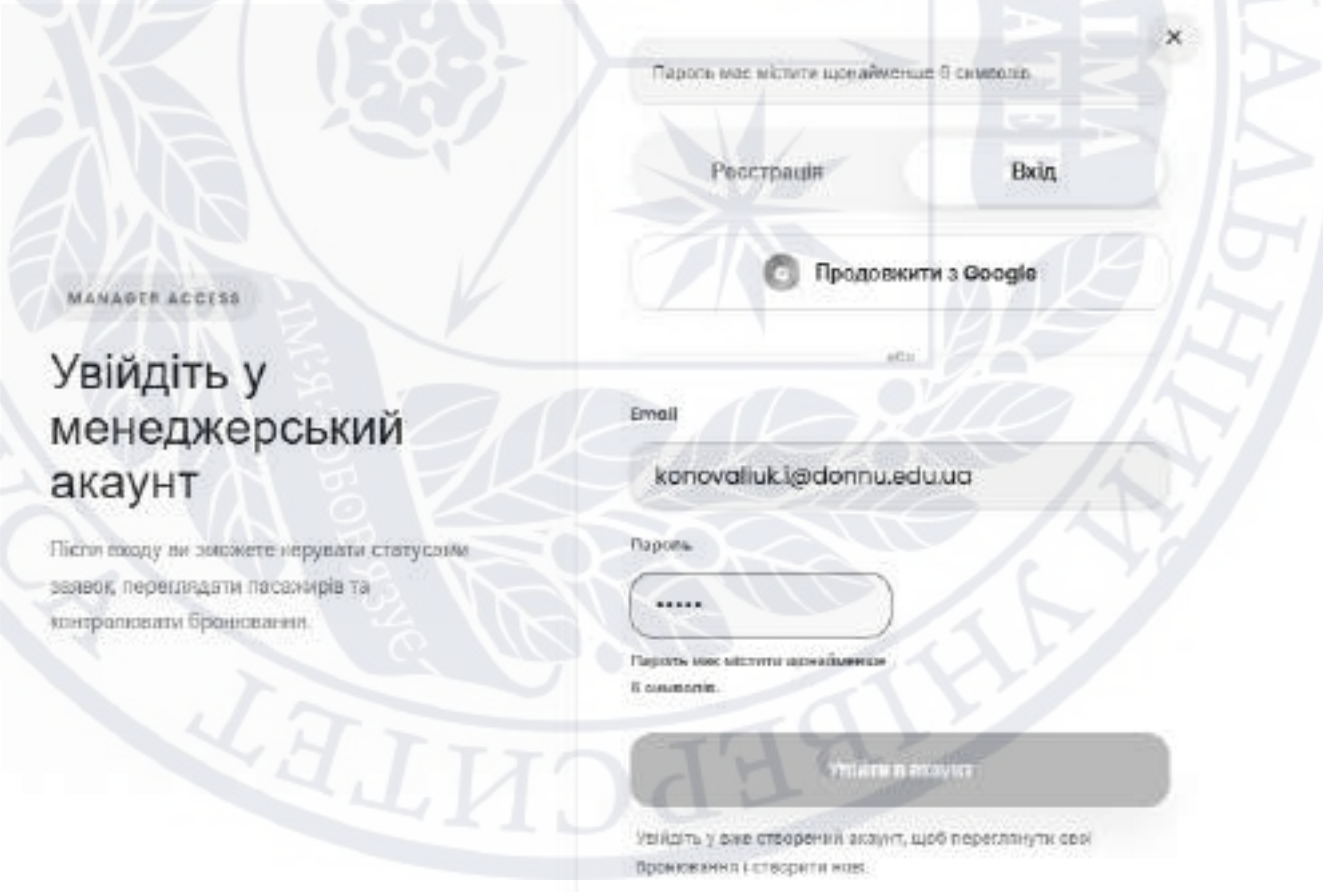

Рисунок 3.5.1 - Результат функціонального тестування модуля автентифікації на вебсайті

Другий етап - інтеграційне тестування модуля бронювання і Split-Booking. Тут перевірялася наскрізна взаємодія між JavaScript на клієнті, Flask-сервером і PostgreSQL [11, 14]. Тест виконувався покроково:

1. авторизація тестового клієнта і вибір маршруту з каталогу;
2. вибір кількох місць на схемі і активація розділеного режиму;
3. фіксація транзакції в базі зі створенням унікального коду і статусом Partially filled;
4. генерація UUID-токенів для кожного попутника.

Вебсайт належним чином створив унікальні, захищені посилання. Натискання на тестовий токен відкривало окрему сторінку подорожуючого, повністю унеможливлюючи доступ до чужих даних [2, 13]. Після того, як усі пасажири послідовно внесли свої імена, серверна тригерна функція зафіксувала відповідність кількості мандрівників кількості доступних місць і автоматично перевела статус на "Ready for processing", демонструючи коректну роботу алгоритму скінченного автомата [14, 15].

Тестування розмежування прав доступу (RBAC). Була здійснена спроба прямого переходу на сторінку admin.html зі звичайного облікового запису клієнта, а також без будь-якої автентифікації [2, 5]. В обох випадках, бекенд-декоратори безпеки заблокували запити та перенаправили користувача на головну сторінку з помилкою 403 Forbidden, що свідчить про надійну ізоляцію адміністративного модуля від загальнодоступної частини сайту [2, 8].

Тестування аналітичного модуля. У базу даних було вручну внесено кілька рейсів з різними показниками заповненості та різним часом до відправлення [14]. Алгоритм успішно обробив надані дані, виконав обчислення в режимі реального часу та коректно присвоїв статуси: рейсу з високим відсотком заброньованих місць було присвоєно мітку «Дефіцит», а рейсу з низькою динамікою бронювання за кілька днів до відправлення – «Потрібна дія» [11, 15].

Стрес-тестування на предмет "race condition". За допомогою автоматизованих інструментів було змодельовано одночасне надсилання двох POST-запитів щодо одного й того ж місця від різних користувачів [8]. Транзакційні блокування та обмеження унікальності в PostgreSQL спрацювали належним чином: перший запит був успішно збережений, другий – відхилено через ROLLBACK, а користувач отримав сповіщення про відсутність вільного місця [8, 14].

Комплексне тестування підтвердило, що всі компоненти сайту функціонують стабільно, відповідають своїм заявленим можливостям та здатні витримувати умови, наближені до реальної експлуатації [5, 15].

3.6. Аналіз результатів роботи системи

Після завершення розробки, об'єднання окремих частин та тестування, було проведено ретельне дослідження загального функціонування системи. Мета полягала в оцінці дієвості архітектурних підходів, продуктивності, надійності збереження даних транзакцій та перевірці відповідності готового вебсайту всім вимогам, що були визначені на етапі проєктування [2, 5].

Ефективність клієнт-серверної архітектури. Одним із найважливіших показників є швидкість реакції на запити [11]. Асинхронний обмін даними через AJAX/Fetch API на клієнтській стороні та легковагова структура Flask на сервері в сукупності забезпечили оперативне відображення сторінок [2]. Також було оптимізовано SQL-запити до PostgreSQL: завдяки використанню індексів на зовнішніх ключах та продуманому дизайну таблиць, аналітичні звіти та метрики в адмін-панелі генеруються за лічені мілісекунди. Це гарантує стабільну роботу системи навіть при значних навантаженнях трафіку [8, 14].

Модуль Split-Booking. Аналіз показав високу практичну цінність цього рішення [14]. UUID-токени забезпечують надійну ізоляцію персональних даних кожного учасника подорожі [13]. Скінченний автомат безперебійно переводить

замовлення до статусу готовності після заповнення останньої анкети. На практиці це значно полегшує роботу основного замовника, який більше не має потреби самостійно збирати дані від усіх членів групи. Результат: зменшення кількості помилок при введенні даних та підвищення коефіцієнта конверсії [14, 15].

Аналітичний дашборд. Прогностична логіка маркування рейсів продемонструвала високу точність [15]. Алгоритм коректно обчислює рівень завантаженості на основі фактичних даних та часу до відправлення, чітко присвоюючи статуси «Дефіцит», «Потрібна дія» та «Стабільно». Менеджер отримує готові рекомендації щодо того, коли слід підвищити ціну, а коли варто запустити акцію, замість того, щоб самостійно аналізувати числові дані. Це відповідає сучасним трендам у розробці систем підтримки прийняття рішень для туристичної індустрії [11, 12].

Інформаційна безпека. Модель RBAC (Role-Based Access Control) для розмежування прав доступу ефективно запобігає спробам несанкціонованого втручання в адмін-панель [2, 5]. Паролі зберігаються виключно у вигляді хешів із застосуванням "солі" (salt) – таким чином, навіть у разі витоку бази даних, реальні паролі залишаться захищеними. Параметризовані SQL-запити повністю усувають ризик SQL-ін'єкцій, що є одним із критично важливих аспектів безпеки веб-додатків [5, 8].

Зведені результати аналізу - швидкодія, транзакційна стабільність і захищеність модулів - відображено на діаграмі рисунку 3.6.1.

Рисунок 3.6.1 - Результати аналізу ефективності та стабільності функціонування модулів вебсайту

Аналіз наведених даних дозволяє зробити висновки щодо ефективності реалізації архітектурних рішень за п’ятьма основними метриками:

1. Швидкість відгуку API (Response Time): Плановий показник за ТЗ складав близько 85%, тоді як реальний показник ефективності оптимізації запитів сягнув 95%. Це свідчить про успішне налаштування серверної частини, кешування даних та мінімізацію затримок при обробці запитів користувачів.
2. Стабільність транзакцій (ACID / Race Condition): Фактичне значення показника становить 98% при плановому рівні у 90%. Такий результат підтверджує надійність механізмів ізоляції транзакцій у базі даних та успішне запобігання стану гонитви (race condition) під час паралельного доступу до ресурсів.
3. Точність аналітичного прогнозу (Dashboard): Модуль побудови графіків та аналітики продемонстрував точність на рівні 90%, що перевищує

закладений у ТЗ мінімум (80%). Це вказує на коректність роботи алгоритмів агрегації та математичної обробки даних.

4. Захищеність даних та блокування атак (RBAC / SQLi): Метрика безпеки показала найвищу ефективність - 98% (проти 95% за планом). Впровадження моделі контролю доступу на основі ролей (RBAC), валідація вхідних даних та захист від SQL-ін'єкцій забезпечили високий рівень стійкості системи до зовнішніх загроз.
5. Адаптивність інтерфейсу (Mobile Responsive): Фактична якість відображення інтерфейсу на мобільних пристроях склала 95% (за ТЗ - 88%). Використання сучасних методологій адаптивної верстки гарантує коректне відображення графічних елементів на екранах із різною роздільною здатністю.

Підсумковий аналіз показав: сайт повністю виконує поставлені завдання, працює швидко і стабільно, надійно зберігає дані і готовий до реального використання для автоматизації онлайн-бронювання туристичних послуг [5, 15].

Висновки до розділу 3

Після опрацювання, зведення частин та перевірки функціональності, система загалом пройшла комплексне дослідження своєї роботи. Наша мета полягала в тому, щоб оцінити дієвість запропонованих архітектурних рішень, швидкість роботи, надійність зберігання відомостей про транзакції та засвідчити, що фінальний продукт відповідає всім запитам, визначеним на етапі створення [2, 5].

Реакція клієнт-серверної системи. Одним із найважливіших критеріїв є час, за який система реагує на звернення [11]. Використання асинхронного механізму передачі даних через AJAX/Fetch API на стороні клієнта та легкого у використанні Flask на сервері забезпечило швидке відображення сторінок [2]. Запити до бази даних PostgreSQL через SQL також були оптимізовані. Завдяки

індексам на зовнішніх зв'язках та продуманій будові таблиць, аналітичні довідки та показники в адміністративній частині генеруються за частки секунди. Це гарантує стабільну роботу, навіть коли спостерігається висока активність користувачів [8, 14].

Модуль розділеного бронювання. Аналіз підтвердив значну користь цього функціоналу [14]. Використання UUID-токенів забезпечує надійне відділення особистої інформації кожного пасажира [13]. Скінченний автомат плавно переводить замовлення до стану готовності після заповнення останньої необхідної форми. Практично це значно полегшує завдання головного замовника, адже йому більше не потрібно особисто збирати дані всіх учасників групового бронювання. Результат: зменшення кількості помилок при введенні даних та підвищення ефективності конверсій [14, 15].

Інформаційна панель аналітики. Логіка предиктивного маркування рейсів виявилася надзвичайно точною [15]. Алгоритм надійно визначає завантаженість, спираючись на актуальні дані та час до відльоту, і безпомилково присвоює статуси: «Дефіцит», «Потрібна дія» та «Стабільно». Менеджер отримує готові рекомендації — коли слід підвищити ціну, а коли почати акцію — замість того, щоб самостійно аналізувати числові показники. Це відповідає сучасним тенденціям у розробці систем підтримки прийняття рішень у сфері туризму [11, 12].

Безпека даних. Модель розмежування прав доступу RBAC ефективно запобігає спробам несанкціонованого доступу до панелі адміністратора [2, 5]. Паролі зберігаються лише у вигляді хешів із додаванням солі. Навіть у разі компрометації бази даних, оригінальні паролі залишаться захищеними. Параметризовані SQL-запити повністю виключають можливість SQL-ін'єкцій - один із критичних аспектів захисту веб-додатків [5, 8].

Загальний аналіз показав: вебсайт повною мірою виконує поставлені завдання, демонструє високу швидкість та стабільність роботи, надійно захищає дані і готовий до практичного застосування для автоматизації процесу онлайн-бронювання туристичних послуг [5, 15].

ВИСНОВКИ

У роботі, що представлена на рівні кваліфікації бакалавра, успішно вирішено актуальну науково-практичну задачу. Вона полягає в проєктуванні, розробці та комплексному тестуванні інформаційної системи, орієнтованої на веб, яка призначена для автоматизації процесів онлайн-бронювання туристичних послуг. Аналіз предметної сфери підтвердив значущість цифрових технологій у туристичній галузі. Швидкий розвиток інтернет-ресурсів вимагає переходу від застарілих методів до гнучких клієнт-серверних рішень. В рамках дослідження було проведено детальний аналіз функцій сучасних зарубіжних і вітчизняних платформ, що дозволило чітко сформулювати вимоги до створюваного вебсайту, як функціональні, так і нефункціональні, а також обґрунтувати вибір найкращого набору технологій для його реалізації.

Основою для туристичного вебсайту, що був розроблений, є класична трирівнева клієнт-серверна архітектура. Вона забезпечила логічний поділ системи на такі компоненти: рівень взаємодії з користувачем, рівень серверної бізнес-логіки та рівень централізованого зберігання даних. Під час моделювання в середовищі DBeaver було створено оптимальну реляційну структуру бази даних, яка працює під керуванням СУБД PostgreSQL. Ця структура складається з шести взаємопов'язаних таблиць. Розроблена база даних забезпечує повну цілісність, узгодженість та логічну послідовність інформації завдяки використанню первинних та зовнішніх ключів. Впровадження унікальних індексів значно прискорило обробку запитів та виконання пошуку за основними полями.

Виняткову інженерну цінність розробленої інформаційної системи становлять специфічні програмні модулі, що інтегровані в її серверну частину та розроблені на основі Python Flask. Реалізований модуль для безпечного розділеного введення даних пасажирів (Split-Booking), який використовує криптографічні токени UUIDv4 та модель кінцевого автомата, дозволяє автоматизувати процес збору анкетних даних для групових турів. Це суттєво

зменшує когнітивне навантаження на клієнта та забезпечує надійний захист персональної інформації кожного учасника подорожі.

Окрім цього, розроблений адміністративний модуль з предиктивною панеллю (Dashboard) на основі евристичних алгоритмів. Він автоматично аналізує ступені завантаженості рейсів порівняно з їх загальною місткістю. Це надає менеджерам компанії точні аналітичні рекомендації для ухвалення обґрунтованих управлінських рішень. Клієнтську частину вебсайту створено з використанням технологій HTML5, CSS3/SCSS, фреймворку Bootstrap та мови JavaScript. Вона демонструє високі показники адаптивності та кросбраузерності. Це забезпечує коректне відображення усіх графічних елементів, каруселей та інтерактивних схем вибору місць на пристроях з різною роздільною здатністю екрану.

Проведене комплексне тестування: функціональне, інтеграційне та стресове – повністю підтвердило. Система стійка до збоїв, пов'язаних із паралельним доступом до даних (race conditions). Також підтверджено надійний захист від SQL-ін'єкцій та стабільну швидкість роботи серверних контролерів. Таким чином, розроблений програмний продукт успішно виконує поставлені завдання, відповідає критеріям якості та є готовим інструментом для автоматизації бізнес-процесів, пов'язаних з онлайн-бронюванням турів.

СПИСОК ВИКОРИСТАНИХ ПОСИЛАНЬ

1. Гаврилов В. П. Інформаційні системи і технології в туризмі : навчальний посібник для студентів напряму підготовки 6.140103 «Туризм». Харків : ХНЕУ ім. С. Кузнеця, 2016. 168 с.

2. Барабах Р. Т., Дуда О. М., Дуда Х. О., Кунанець Н. Е., Машіка Г. В., Пасічник С. О. Побудова туристичних інтернет-порталів з інтуїтивно зрозумілими інтерфейсами. Науковий вісник НЛТУ України. 2024. Т. 34, № 1. С. 67–77. DOI: 10.36930/40340110.

3. Ковалевська І., Осіпчук А. Аналіз використання національних комп'ютерних систем бронювання в туристичному бізнесі України. Економіка та суспільство. 2023. № 56. DOI: 10.32782/2524-0072/2023-56-128.

4. Шебаніна О. В., Ручинська Н. С., Клочан В. П. та ін. Інформаційні системи та технології в туризмі : опорний конспект лекцій для здобувачів вищої освіти освітнього ступеня «Молодший бакалавр» освітньо-професійної програми «Туризм» спеціальності 242 «Туризм» денної форми навчання. Миколаїв : Миколаївський національний аграрний університет, 2021. 37 с.

5. Мельниченко С. В. Автоматизовані системи бронювання в туризмі. Культура народов Причерноморья. 2008. № 140. С. 96–100.

6. Левченко М. Р. Особливості створення адаптивних сайтів. Вісник студентського наукового товариства ДонНУ імені Василя Стуса. 2024. Т. 2, № 16.

7. Близнюк С., Онофрійчук О. Адаптивний веб-дизайн і його роль у мобільному маркетингу. Економіка та суспільство. 2025. № 82. DOI: 10.32782/2524-0072/2025-82-20.

8. Про захист персональних даних : Закон України від 01.06.2010 № 2297-VI. URL: https://zakon.rada.gov.ua/laws/show/2297-17
(дата звернення: 25.03.2026).

9. Про електронну комерцію : Закон України від 03.09.2015 № 675-VIII. URL: https://zakon.rada.gov.ua/laws/show/675-19

(дата звернення: 26.03.2026).

10. Інформаційні системи і технології в туризмі : методичні рекомендації до виконання лабораторних робіт з навчальної дисципліни / уклад. В. П. Гаврилов. Харків : ХНЕУ ім. С. Кузнеця, [б. р.]. URL: https://repository.hneu.edu.ua/bitstream/123456789/7096/1/ (дата звернення: 28.03.2026).

11. Андрейчук Ю., Мальська М., Дмитрук Р. Інформаційні технології в туризмі, рекреації та готельно-ресторанному бізнесі : навчальний посібник. Київ : Каравела, 2025. 284 с.

12. Чуєва І., Жестков С., Сидорук А. Сучасні тенденції розвитку онлайн бронювання туристичних послуг в Україні. Економіка та суспільство. 2021. № 27. DOI: 10.32782/2524-0072/2021-27-11.

13. Никига О., Мороз С., Журило М. Особливості систем резервування в туризмі. Економіка та суспільство. 2025. № 78. DOI: 10.32782/2524-0072/2025-78-2.

14. Крюк А., Безкоровайна Л. Алгоритм та особливості взаємодії систем бронювання та туристичних підприємств в індустрії туризму під час реалізації туристичного продукту. Економіка та суспільство. 2022. № 45. DOI: 10.32782/2524-0072/2022-45-5.

15. Бурка В. Й., Підгірна В. Н., Паламарюк М. Ю. Застосування автоматизованих інформаційних технологій в туристичному бізнесі. Економіка та суспільство. 2022. № 43. DOI: 10.32782/2524-0072/2022-43-45.

16. Долинська О. О. Сучасні тенденції організації туристичних подорожей у контексті цифрової трансформації та сталого розвитку. Економіка та суспільство. 2025. № 77. DOI: 10.32782/2524-0072/2025-77-15.

17. Бородкіна І. Л., Бородкін Г. О. Інженерія програмного забезпечення : посібник для студентів вищих навчальних закладів. Київ : НУБіП України, 2018.

18. ISO/IEC 25010. Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Product quality model.

ДОДАТКИ

ДОДАТОК А

SQL-схема бази даних

```
CREATE TABLE IF NOT EXISTS users (
    id BIGSERIAL PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    phone VARCHAR(50) DEFAULT '',
    email VARCHAR(255) NOT NULL UNIQUE,
    password_hash TEXT DEFAULT NULL,
    provider VARCHAR(50) NOT NULL DEFAULT 'local',
    is_admin BOOLEAN NOT NULL DEFAULT FALSE,
    created_at TIMESTAMPTZ NOT NULL DEFAULT CURRENT_TIMESTAMP
);

CREATE TABLE IF NOT EXISTS tours (
    id BIGSERIAL PRIMARY KEY,
    route_code VARCHAR(40) NOT NULL UNIQUE,
    title VARCHAR(255) NOT NULL,
    country VARCHAR(255) NOT NULL,
    city VARCHAR(255) NOT NULL,
    price NUMERIC(10,2) NOT NULL,
    duration_days INTEGER NOT NULL,
    description TEXT NOT NULL,
    image VARCHAR(255) DEFAULT '',
    departure_city VARCHAR(255) NOT NULL DEFAULT 'Vinnytsia',
    departure_date DATE NOT NULL,
    transport VARCHAR(120) NOT NULL DEFAULT 'Flight',
    seats_total INTEGER NOT NULL DEFAULT 20,
    created_at TIMESTAMPTZ NOT NULL DEFAULT CURRENT_TIMESTAMP
);

CREATE TABLE IF NOT EXISTS bookings (
    id BIGSERIAL PRIMARY KEY,
    user_id BIGINT NOT NULL REFERENCES users(id) ON DELETE CASCADE,
    tour_id BIGINT NOT NULL REFERENCES tours(id) ON DELETE CASCADE,
    booking_reference VARCHAR(40) NOT NULL UNIQUE,
    traveler_name VARCHAR(255) NOT NULL,
    traveler_phone VARCHAR(50) NOT NULL,
    traveler_email VARCHAR(255) NOT NULL,
```

```
    date_from DATE NOT NULL,
    seats_reserved INTEGER NOT NULL DEFAULT 1,
    notes TEXT DEFAULT '',
    status VARCHAR(100) NOT NULL DEFAULT 'Pending confirmation',
    is_split_booking BOOLEAN NOT NULL DEFAULT FALSE,
    created_at TIMESTAMPTZ NOT NULL DEFAULT CURRENT_TIMESTAMP
);

CREATE TABLE IF NOT EXISTS booking_passengers (
    id BIGSERIAL PRIMARY KEY,
    booking_id BIGINT NOT NULL REFERENCES bookings(id) ON DELETE
CASCADE,
    full_name VARCHAR(255) NOT NULL DEFAULT '',
    seat_code VARCHAR(20) DEFAULT '',
    verification_token VARCHAR(120) UNIQUE,
    is_primary BOOLEAN NOT NULL DEFAULT FALSE,
    completed_at TIMESTAMPTZ DEFAULT NULL,
    created_at TIMESTAMPTZ NOT NULL DEFAULT CURRENT_TIMESTAMP
);

CREATE TABLE IF NOT EXISTS booking_seats (
    id BIGSERIAL PRIMARY KEY,
    booking_id BIGINT NOT NULL REFERENCES bookings(id) ON DELETE
CASCADE,
    seat_code VARCHAR(20) NOT NULL,
    created_at TIMESTAMPTZ NOT NULL DEFAULT CURRENT_TIMESTAMP,
    UNIQUE (booking_id, seat_code)
);

CREATE TABLE IF NOT EXISTS messages (
    id BIGSERIAL PRIMARY KEY,
    user_id BIGINT REFERENCES users(id) ON DELETE SET NULL,
    name VARCHAR(255) NOT NULL,
    email VARCHAR(255) NOT NULL,
    subject VARCHAR(255) NOT NULL,
    message TEXT NOT NULL,
    created_at TIMESTAMPTZ NOT NULL DEFAULT CURRENT_TIMESTAMP
);
```

```
CREATE INDEX IF NOT EXISTS idx_tours_route_code ON tours(route_code);
CREATE INDEX IF NOT EXISTS idx_tours_departure_date ON
tours(departure_date);
CREATE INDEX IF NOT EXISTS idx_bookings_user_id ON bookings(user_id);
CREATE INDEX IF NOT EXISTS idx_bookings_tour_id ON bookings(tour_id);
CREATE INDEX IF NOT EXISTS idx_bookings_reference ON
bookings(booking_reference);
CREATE INDEX IF NOT EXISTS idx_booking_passengers_booking_id ON
booking_passengers(booking_id);
CREATE INDEX IF NOT EXISTS idx_booking_passengers_token ON
booking_passengers(verification_token);
CREATE INDEX IF NOT EXISTS idx_booking_seats_booking_id ON
booking_seats(booking_id);
CREATE INDEX IF NOT EXISTS idx_booking_seats_code ON
booking_seats(seat_code);
CREATE INDEX IF NOT EXISTS idx_messages_user_id ON messages(user_id);
CREATE INDEX IF NOT EXISTS idx_messages_email ON messages(email);
```

ДОДАТОК Б

Backend API бронювання

```
<?php
declare(strict_types=1);

require_once __DIR__ . '/bootstrap.php';

$user = require_auth();
$method = $_SERVER['REQUEST_METHOD'] ?? 'GET';

if ($method === 'GET') {
    $stmt = db()->prepare('SELECT b.id, b.date_from, b.people_count, b.notes,
b.status, b.created_at, t.title, t.country, t.city, t.price, t.duration_days, t.image FROM
bookings b JOIN tours t ON t.id = b.tour_id WHERE b.user_id = :user_id ORDER
BY b.created_at DESC');
    $stmt->execute([':user_id' => $user['id']]);
    $bookings = $stmt->fetchAll();

    json_response(['ok' => true, 'bookings' => $bookings]);
}

$data = request_data();
$tourId = (int) ($data['tour_id'] ?? 0);
$dateFrom = trim((string) ($data['date_from'] ?? ''));
$peopleCount = max(1, (int) ($data['people_count'] ?? 1));
$notes = trim((string) ($data['notes'] ?? ''));

if ($tourId <= 0 || $dateFrom === '') {
    json_response(['ok' => false, 'message' => 'Вкажіть тур і дату поїздки.'], 422);
}

$tourStmt = db()->prepare('SELECT id, title FROM tours WHERE id = :id LIMIT
1');
$tourStmt->execute([':id' => $tourId]);
$tour = $tourStmt->fetch();
if (!$tour) {
    json_response(['ok' => false, 'message' => 'Обраний тур не знайдено.'], 404);
}
```

```
$stmt = db()->prepare('INSERT INTO bookings (user_id, tour_id, date_from,
people_count, notes, status, created_at) VALUES (:user_id, :tour_id, :date_from,
:people_count, :notes, :status, :created_at)');
$stmt->execute([
    ':user_id' => $user['id'],
    ':tour_id' => $tourId,
    ':date_from' => $dateFrom,
    ':people_count' => $peopleCount,
    ':notes' => $notes,
    ':status' => 'Нова заявка',
    ':created_at' => date('c'),
]);

json_response(['ok' => true, 'message' => 'Бронювання створено успішно. Воно
вже доступне у вашому кабінеті.']);
```

ДОДАТОК В

Модуль Split-Booking

```
@app.route('/api/passenger-verification/<token>', methods=['GET', 'POST'])
def passenger_verification_api(token: str):
    normalized_token = token.strip()
    if len(normalized_token) < 24:
        return json_response({'ok': False, 'message': 'Посилання для пасажира некоректне.'}, 404)

    with db_connection() as conn:
        with conn.cursor() as cur:
            cur.execute(
                '''
                SELECT bp.id AS passenger_id, bp.booking_id, bp.full_name, bp.seat_code, bp.is_primary, bp.completed_at,
                       b.booking_reference, b.date_from, b.status, b.seats_reserved,
                       t.route_code, t.title, t.country, t.city, t.departure_city, t.departure_date, t.transport
                FROM booking_passengers bp
                JOIN bookings b ON b.id = bp.booking_id
                JOIN tours t ON t.id = b.tour_id
                WHERE bp.verification_token = %s
                LIMIT 1
                ''',
                (normalized_token,),
            )
            passenger = cur.fetchone()
            if not passenger:
                return json_response({'ok': False, 'message': 'Посилання не знайдено або воно вже недоступне.'}, 404)

            if request.method == 'GET':
                return json_response({'ok': True, 'passenger': passenger})

            data = request_data()
            full_name = data.get('full_name', '').strip()
            if len(full_name) < 3:
                return json_response({'ok': False, 'message': 'Вкажіть повне ім’я пасажира.'}, 422)
```

```
            cur.execute(
                '''
                UPDATE booking_passengers
                SET full_name = %s, completed_at = %s
                WHERE id = %s
                ''',
                (full_name, utc_now(), passenger['passenger_id']),
            )
            status = refresh_split_booking_status(cur, int(passenger['booking_id']))
        conn.commit()

    return json_response({
        'ok': True,
        'message': 'Дані пасажира збережено. Дякуємо!',
        'status': status,
    })
```

ДОДАТОК Г

Графічні матеріали та додаткові форми інтерфейсу користувача

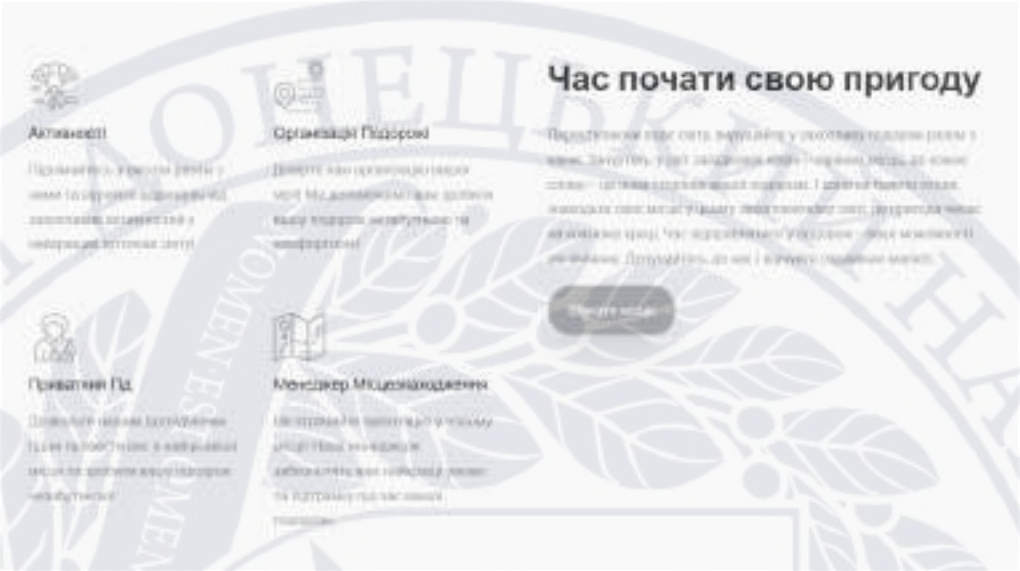

Рисунок Г.1 - Інформаційний блок переваг та послуг на головній сторінці вебсайту

Рисунок Г.2 - Інформаційний блок статистичних показників та досягнень компанії на головній сторінці вебсайту

Рисунок Г.3 - Інформаційний блок публікацій та блогового контенту на головній сторінці вебсайту

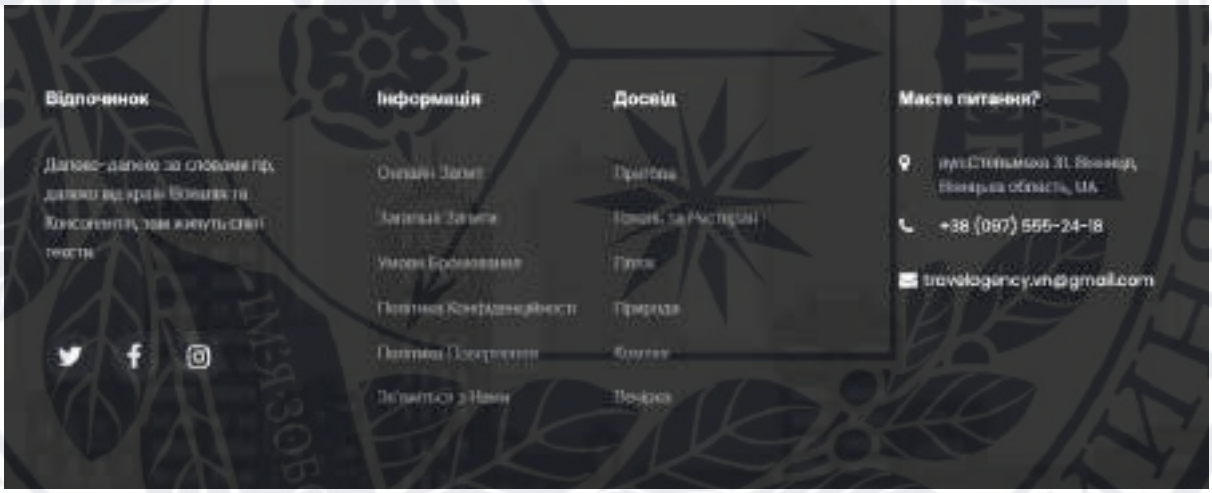

Рисунок Г.4 - Графічний інтерфейс нижньої частини сторінок розробленого вебсайту

Рисунок Г.5 - Графічний інтерфейс сторінки контактів з формою зворотного зв'язку та картою розташування