МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ ЕКОНОМІКИ І ТЕХНОЛОГІЙ

ННІ	Економіки та бізнес-освіти
Кафедра	Економіки та цифрового бізнесу
Спеціальність	122 Комп'ютерні науки
Форма навчання	Денна

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

Музики Євгенія Максимовича
(прізвище, ім'я, по батькові)

на тему Розробка віджета онлайн-бронювання для сайту гостинного бізнесу

за матеріалами _____
(повна назва бази дослідження)

Науковий керівник _____ _____ Шокотько Л.М.

Робота допущена до захисту в ЕК
Протокол засідання кафедри
від 12.06.2026р. № 15
Завідувач кафедри _____
(підпис)
к.е.н., доцент Радько В.М.
наук. ступень, вчене звання прізвище, ініціали

Кривий Ріг − 2026

ЗАТВЕРДЖЕНО
Наказ Міністерства освіти і науки, молоді та спорту України
29 березня 2012 року № 384

Форма № Н-9.01

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ ЕКОНОМІКИ І ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ЕКОНОМІКИ ТА БІЗНЕС-ОСВІТИ
(повне найменування вищого навчального закладу)

Кафедра економіки та цифрового бізнесу
Освітній ступінь бакалавр
Спеціальність 122 Комп'ютерні науки

ЗАТВЕРДЖУЮ
Завідувач кафедри _____В.М. Радько

"30" березня 2026 року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ БАКАЛАВРСЬКУ РОБОТУ ЗДОБУВАЧУ

_____Музиці Євгенію Максимовичу_____

1. Тема роботи Розробка віджета онлайн-бронювання для сайту гостинного бізнесу_____
_____ _____
науковий керівник роботи _____Шокотько Людмила Миколаївна,
затверджені наказом вищого навчального закладу від «23» березня 2026 р. № 193-ст

2. Строк подання здобувачем роботи 29.05.2026р.

3. Зміст кваліфікаційної бакалаврської роботи, об'єкт, предмет та мета дослідження:

Розділ 1 Аналіз предметної області та існуючих рішень.
Розділ 2 Проєктування та розробка віджета онлайн-бронювання.
Розділ 3 Тестування та оцінка ефективності системи.

Об'єкт дослідження – процес організації та автоматизації онлайн-бронювання послуг у сфері гостинного бізнесу.

Предмет дослідження - методи, технології та програмні засоби розробки веборієнтованих систем онлайн-бронювання, а також механізми автентифікації, авторизації та управління бронюваннями.

Мета кваліфікаційної бакалаврської роботи – розробка та реалізація віджета онлайн-бронювання для вебсайту підприємства гостинного бізнесу, який забезпечує автоматизацію процесу резервування номерів, управління бронюваннями та гнучке налаштування цінової політики.

4. Дата видачі завдання 03.04.2026р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної бакалаврської роботи	Строк виконання етапів роботи	Відмітка керівника про виконання етапів (дата, підпис)
1	Підготовка розділу 1	до 17.04.2026р.	17.04.2026р.
2	Підготовка розділу 2	до 08.05.2026р.	08.05.2026р.
3	Підготовка розділу 3	до 25.05.2026р.	25.05.2026р
4	Реєстрація завершеної кваліфікаційної роботи	до 29.05.2026р.	29.05.2026р.
5	Отримання відгуку від наукового керівника	04.06.2026р.	04.06.2026р.
6	Отримання зовнішньої рецензії	05.06.2026р.	05.06.2026р.
7	Попередній захист кваліфікаційної роботи на кафедрі	08.06.2026р.	08.06.2026р.
8	Перевірка кваліфікаційної роботи на плагіат	12.06.2026р.	12.06.2026р.
9	Допуск кафедрою кваліфікаційної роботи до захисту	12.06.2026р.	12.06.2026р.
10	Підготовка студента до захисту в ЕК	до 19.06.2026р.	19.06.2026р.

Завдання підготував науковий керівник _____ Шокотько Л. М.

Завдання одержав здобувач _____ Музика Є.М.

(підпис) (прізвище та ініціали)

РЕФЕРАТ

Кваліфікаційна робота бакалавра: 70 стор., 24 рисунки, 3 таблиці, 3 додатки, 27 використаних джерел.

Метою роботи є розробка віджета онлайн-бронювання для сайту підприємства гостинного бізнесу, який забезпечує автоматизацію процесу резервування номерів, управління бронюваннями користувачів та гнучке налаштування цінової політики.

Об’єкт дослідження – процес організації та автоматизації онлайн-бронювання послуг у сфері гостинного бізнесу.

Предмет дослідження – методи, технології та програмні засоби розробки веборієнтованих систем онлайн-бронювання, а також механізми автентифікації, авторизації та управління бронюваннями.

У ході виконання роботи було проведено аналіз особливостей організації онлайн-бронювання у сфері гостинного бізнесу, досліджено існуючі вебплатформи бронювання, визначено функціональні та нефункціональні вимоги до системи, спроєктовано архітектуру програмного забезпечення та структуру бази даних. Розроблено серверну та клієнтську частини віджета онлайн-бронювання, реалізовано механізми автентифікації та авторизації користувачів, систему керування бронюваннями, модуль динамічного ціноутворення та адміністративну панель керування.

Під час розробки програмного забезпечення було використано мову програмування Python, фреймворк FastAPI, систему керування базами даних MySQL, технології HTML, CSS, JavaScript та бібліотеку Bootstrap. Для роботи з базою даних застосовано SQLAlchemy, а для реалізації механізмів автентифікації використано JWT-токени та алгоритми хешування паролів.

Розроблений програмний продукт забезпечує перегляд доступних номерів, перевірку доступності дат, автоматичний розрахунок вартості проживання, створення та скасування бронювань, керування ціновими правилами та адміністрування системи через вебінтерфейс.

Експлуатаційне призначення програмного продукту полягає у використанні його підприємствами готельного та гостинного бізнесу для автоматизації процесу онлайн-бронювання послуг і підвищення ефективності взаємодії з клієнтами.

Ключові слова: авторизація, автентифікація, база даних, онлайн-бронювання, вебзастосунок, віджет бронювання, гостинний бізнес, динамічне ціноутворення, JWT, MySQL, FastAPI.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

API (Application Programming Interface) – програмний інтерфейс застосунку.

CRUD (Create, Read, Update, Delete) – базові операції створення, читання, оновлення та видалення даних.

CSS (Cascading Style Sheets) – каскадні таблиці стилів для оформлення вебсторінок.

DB (Database) – база даних.

HTML (HyperText Markup Language) – мова розмітки гіпертекстових документів.

HTTP (HyperText Transfer Protocol) – протокол передавання гіпертексту.

HTTPS (HyperText Transfer Protocol Secure) – захищена версія протоколу HTTP.

JSON (JavaScript Object Notation) – формат обміну даними.

JWT (JSON Web Token) – стандарт передачі даних для автентифікації та авторизації користувачів.

ORM (Object-Relational Mapping) – технологія об’єктно-реляційного відображення даних.

PMS (Property Management System) – система управління готельним господарством.

REST (Representational State Transfer) – архітектурний стиль побудови вебсервісів.

REST API (Representational State Transfer Application Programming Interface) – програмний інтерфейс, побудований за принципами REST.

SQL (Structured Query Language) – мова структурованих запитів до баз даних.

UI (User Interface) – користувацький інтерфейс.

URL (Uniform Resource Locator) – уніфікований покажчик ресурсу.

UX (User Experience) – досвід взаємодії користувача із системою.

PDO (PHP Data Objects) – інтерфейс доступу до баз даних.

ВСТУП

У сучасних умовах стрімкого розвитку інформаційних технологій цифровізація бізнес-процесів стає невід'ємною складовою ефективного функціонування підприємств сфери гостинності. Готелі, хостели та інші заклади розміщення активно впроваджують веб-технології для автоматизації процесів обслуговування клієнтів, зокрема бронювання номерів. Онлайн-бронювання дозволяє підвищити рівень сервісу, оптимізувати роботу персоналу та забезпечити цілодобову доступність послуг для користувачів.

Актуальність теми обумовлена необхідністю створення гнучких, зручних та інтегрованих рішень для бронювання, які можуть бути легко впроваджені на веб-сайти підприємств гостинного бізнесу. Багато існуючих платформ є складними у налаштуванні, дорогими або надлишковими за функціональністю для малих та середніх підприємств. У зв'язку з цим виникає потреба у розробці компактного віджета онлайн-бронювання, який забезпечує базову функціональність, простоту інтеграції та можливість адаптації під конкретні потреби.

Метою кваліфікаційної роботи є розробка віджета онлайн-бронювання для веб-сайтів підприємств гостинного бізнесу з використанням сучасних веб-технологій, що забезпечує зручну взаємодію користувача з системою, автоматизацію процесу бронювання та гнучке управління цінами.

Для досягнення поставленої мети необхідно вирішити такі завдання:

– проаналізувати предметну область онлайн-бронювання у сфері гостинного бізнесу;

– дослідити існуючі програмні рішення та визначити їх переваги і недоліки;

– сформулювати функціональні та нефункціональні вимоги до системи;

– розробити архітектуру програмного забезпечення;

– спроєктувати структуру бази даних;

– реалізувати серверну частину системи з використанням сучасних технологій;

– розробити клієнтську частину у вигляді інтерактивного веб-віджета;

– забезпечити механізми автентифікації та авторизації користувачів;

– провести тестування програмного продукту та оцінити його ефективність.

Об’єктом дослідження є процес організації онлайн-бронювання у сфері гостинного бізнесу.

Предметом дослідження є методи та засоби розробки веб-орієнтованих систем бронювання.

У роботі використано такі методи дослідження: аналіз та узагальнення наукових джерел, методи системного аналізу, об’єктно-орієнтоване проєктування, моделювання інформаційних систем, а також методи тестування програмного забезпечення.

Практичне значення отриманих результатів полягає у можливості використання розробленого віджета онлайн-бронювання на веб-сайтах підприємств гостинного бізнесу для автоматизації процесу бронювання, підвищення якості обслуговування клієнтів та оптимізації внутрішніх бізнес-процесів.

Структура роботи включає вступ, три розділи, висновки, список використаних джерел та додатки. У першому розділі здійснено аналіз предметної області та існуючих рішень. У другому розділі розглянуто процес проєктування та розробки віджета онлайн-бронювання. Третій розділ присвячено тестуванню та оцінці ефективності розробленої системи.

РОЗДІЛ 1
АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ІСНУЮЧИХ РІШЕНЬ

1.1. Особливості організації онлайн-бронювання у сфері гостинного бізнесу

Сфера гостинного бізнесу є однією з найбільш динамічних галузей, що активно впроваджує сучасні інформаційні технології для підвищення якості обслуговування клієнтів. Одним із ключових напрямів цифровізації є автоматизація процесу бронювання номерів, що дозволяє забезпечити швидкий доступ до послуг, зменшити навантаження на персонал та мінімізувати людський фактор.

Онлайн-бронювання являє собою процес резервування номерів або інших послуг через веб-інтерфейс або спеціалізовані програмні системи. На відміну від традиційних способів бронювання (телефон, електронна пошта, особисте звернення), онлайн-системи працюють у режимі реального часу та забезпечують миттєве підтвердження доступності ресурсів. Це значно підвищує зручність користування для клієнтів та ефективність управління для бізнесу [1].

Основною особливістю організації онлайн-бронювання є необхідність синхронізації даних між користувацьким інтерфейсом та серверною частиною системи. Користувач повинен отримувати актуальну інформацію про доступність номерів, ціни та умови проживання. У свою чергу, система повинна обробляти запити в реальному часі, враховуючи вже існуючі бронювання та обмеження.

Процес онлайн-бронювання можна умовно поділити на кілька етапів:

– вибір користувачем параметрів пошуку (дати, тип номера, кількість гостей);

– перевірка доступності номерів;

– відображення варіантів розміщення з цінами;

– введення персональних даних користувача;

– підтвердження бронювання та збереження інформації у базі даних.

На рисунку 1.1 представлено узагальнену схему процесу онлайн-бронювання.

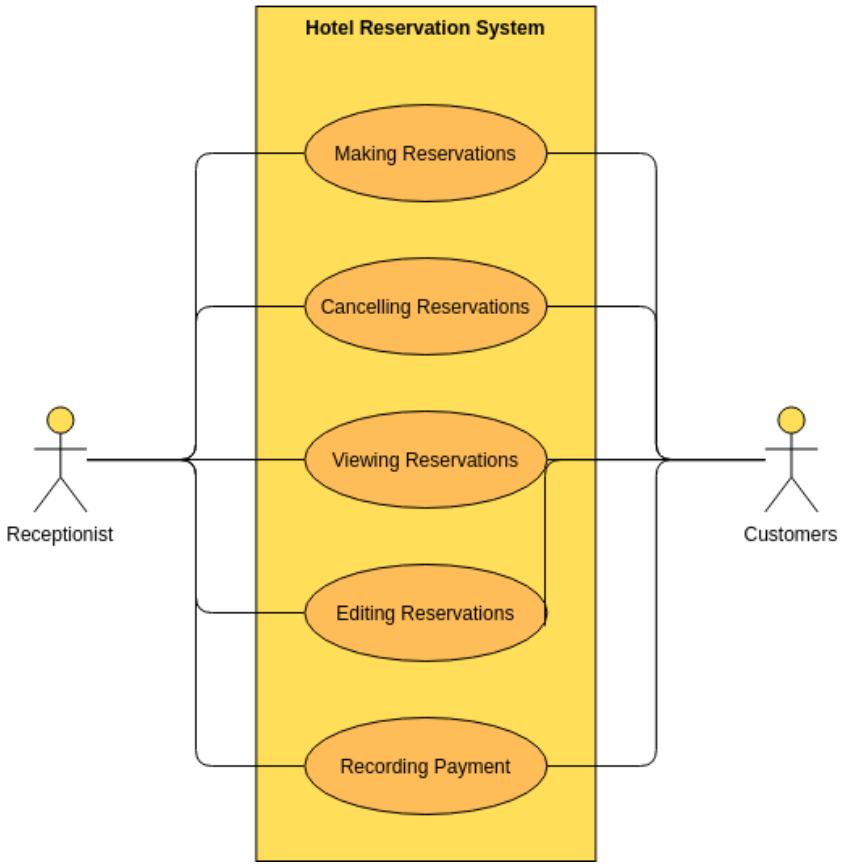

Рис. 1.1. Загальна схема процесу онлайн-бронювання

Примітка. Джерело: Розроблено із використанням [2]

Як видно з рисунка 1.1, ключовими компонентами системи є клієнтська частина (інтерфейс користувача), серверна частина (обробка запитів) та база даних, у якій зберігається інформація про номери, користувачів та бронювання.

Важливою особливістю сучасних систем онлайн-бронювання є використання динамічного ціноутворення. Ціни на номери можуть змінюватися залежно від сезону, попиту, тривалості проживання або

спеціальних акцій. Це вимагає реалізації гнучких механізмів управління цінами, які дозволяють задавати правила для різних періодів часу.

Ще одним важливим аспектом є забезпечення коректної перевірки доступності номерів. Система повинна враховувати всі активні бронювання та запобігати ситуаціям подвійного резервування. Для цього застосовуються алгоритми перевірки перетину дат, які визначають, чи існує конфлікт між новим і вже створеними бронюваннями.

Крім того, особливістю онлайн-бронювання є необхідність забезпечення безпеки даних. Оскільки користувачі вводять персональну інформацію (ім'я, контактні дані, іноді платіжні реквізити), система повинна забезпечувати їх захист від несанкціонованого доступу. Це досягається шляхом використання механізмів автентифікації, авторизації, шифрування даних та захищених протоколів передачі інформації [3].

На рисунку 1.2 представлено структуру типової системи онлайн-бронювання.

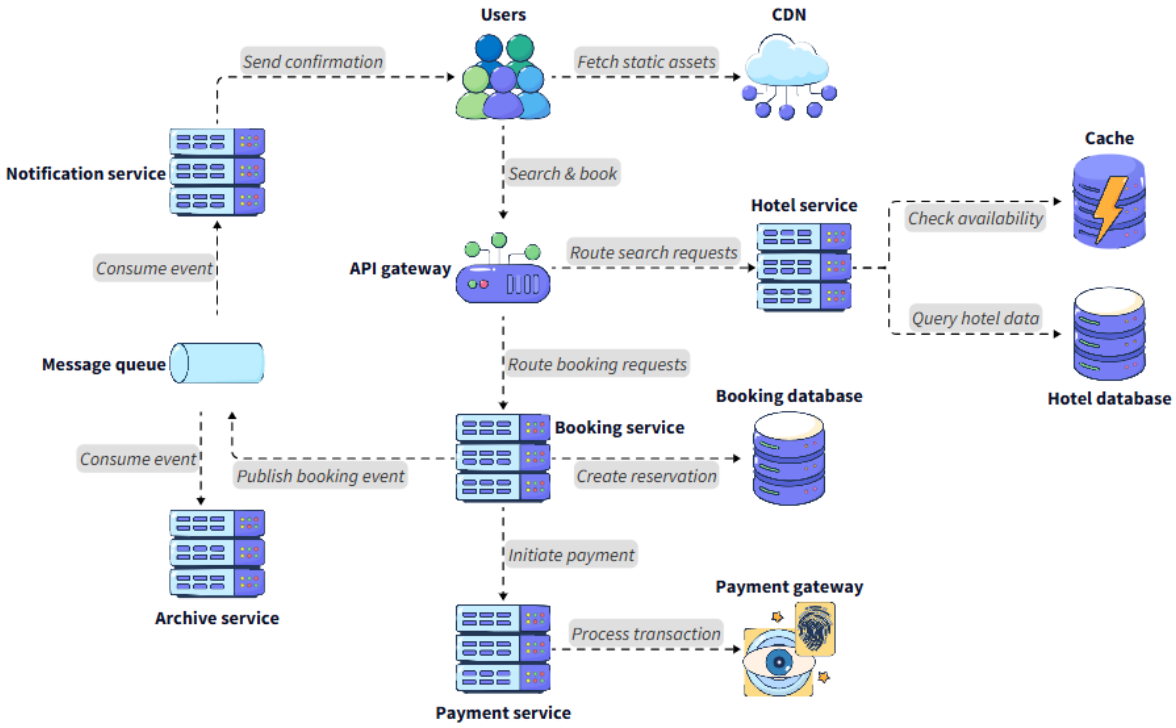

Рис. 1.2. Структура системи онлайн-бронювання

Примітка. Джерело: Розроблено із використанням [3]

Як показано на рисунку 1.2, система має багаторівневу архітектуру, що дозволяє розділити функціональність і забезпечити масштабованість та гнучкість у розробці.

Особливу увагу слід приділити зручності користувацького інтерфейсу. Віджети онлайн-бронювання повинні бути інтуїтивно зрозумілими, швидкими у використанні та адаптованими до різних пристроїв (мобільні телефони, планшети, комп'ютери). Наявність календаря для вибору дат, автоматичного підрахунку вартості та миттєвого відображення результатів є обов'язковими елементами сучасних систем.

Також важливою є інтеграція з іншими системами, такими як системи управління готелем (Property Management System, PMS), платіжні шлюзи та сервіси аналітики. Це дозволяє розширити функціональність системи та підвищити ефективність управління бізнесом.

Отже, організація онлайн-бронювання у сфері гостинного бізнесу має низку особливостей, серед яких ключовими є необхідність обробки даних у реальному часі, забезпечення актуальності інформації, реалізація гнучкого ціноутворення, гарантування безпеки даних та створення зручного користувацького інтерфейсу. Урахування цих аспектів є необхідною умовою для розробки ефективного програмного рішення у даній сфері.

1.2 Аналіз існуючих вебплатформ бронювання

У сучасному гостинному бізнесі важливу роль відіграють глобальні веб-платформи бронювання, які виступають посередниками між постачальниками послуг розміщення та кінцевими користувачами. Найбільш відомими серед них є Booking.com та Airbnb, які забезпечують широкий функціонал для пошуку, бронювання та управління об'єктами розміщення.

Платформа Booking.com є однією з найбільших систем онлайн-бронювання у світі. Вона орієнтована переважно на готелі, хостели та апартаменти і надає користувачам можливість швидкого пошуку житла за

різними параметрами: ціною, розташуванням, рейтингом, зручностями тощо. Однією з ключових особливостей даної платформи є висока ступінь автоматизації процесу бронювання, а також інтеграція з системами управління готелями (PMS) [4].

У свою чергу, платформа Airbnb спеціалізується на короткостроковій оренді житла від приватних осіб. Вона пропонує користувачам більш гнучкі варіанти розміщення, включаючи оренду квартир, будинків або окремих кімнат. Основною відмінністю Airbnb є акцент на соціальній взаємодії між користувачами, системі відгуків та рейтингу господарів [5].

На рисунку 1.3 представлено узагальнену модель взаємодії користувачів із платформами онлайн-бронювання.

Рис. 1.3. Модель взаємодії у веб-платформах бронювання

Примітка. Джерело: Розроблено із використанням [4-6]

Як показано на рисунку 1.3, платформа виступає центральною ланкою, яка забезпечує обробку запитів, управління бронюваннями, а також зберігання та аналіз даних.

Основними перевагами таких платформ є:

– широкий вибір варіантів розміщення;

– зручний інтерфейс користувача;

– наявність системи відгуків і рейтингів;

– підтримка онлайн-оплати;

– глобальна доступність.

Водночас, використання централізованих платформ має і певні недоліки. Зокрема, значні комісійні збори для постачальників послуг, обмежена гнучкість у налаштуванні функціоналу, а також залежність бізнесу від сторонніх сервісів. Для малих та середніх підприємств це може бути критичним фактором, що зумовлює необхідність розробки власних рішень для бронювання [6].

З метою більш детального аналізу було проведено порівняння основних характеристик платформ Booking.com та Airbnb, результати якого наведено в таблиці 1.1.

Таблиця 1.1

Порівняння вебплатформ бронювання

Критерій	Booking.com	Airbnb
Тип розміщення	Готелі, хостели, апартаменти	Приватне житло
Цільова аудиторія	Туристи, бізнес-користувачі	Туристи, індивідуальні мандрівники
Система рейтингу	Відгуки клієнтів	Відгуки клієнтів і господарів
Комісія	Висока	Середня
Гнучкість налаштувань	Обмежена	Середня
Інтеграція з PMS	Так	Обмежена
Онлайн-оплата	Підтримується	Підтримується
Модель взаємодії	B2C	C2C

Як видно з таблиці 1.1, обидві платформи мають схожий базовий функціонал, проте відрізняються за моделлю взаємодії, типом розміщення та рівнем гнучкості. Booking.com більше орієнтований на професійних постачальників послуг, тоді як Airbnb надає можливість будь-якому користувачу виступати в ролі орендодавця.

Аналіз існуючих рішень показує, що, незважаючи на широкий функціонал, такі платформи не завжди є оптимальними для інтеграції безпосередньо у вебсайти окремих підприємств. Це пов’язано з необхідністю перенаправлення користувача на сторонній ресурс, відсутністю повного контролю над процесом бронювання та додатковими витратами.

У зв’язку з цим актуальним є створення власних віджетів онлайн-бронювання, які можуть бути інтегровані безпосередньо на сайт підприємства. Такі рішення дозволяють забезпечити:

– повний контроль над даними користувачів;

– гнучке налаштування бізнес-логіки;

– зниження витрат на комісії;

– покращення користувацького досвіду.

Отже, проведений аналіз показав, що існуючі вебплатформи бронювання мають значні переваги, однак не завжди відповідають потребам окремих підприємств. Це підтверджує доцільність розробки власного віджета онлайн-бронювання, який поєднує необхідний функціонал із можливістю гнучкої адаптації до конкретних умов використання.

1.3 Функціональні та нефункціональні вимоги до системи

Визначення вимог до програмної системи є одним із ключових етапів її розробки, оскільки саме на цьому етапі формується уявлення про майбутній функціонал, обмеження та характеристики якості програмного продукту. Для віджета онлайн-бронювання у сфері гостинного бізнесу вимоги поділяються на функціональні та нефункціональні.

Функціональні вимоги описують перелік можливостей, які система повинна реалізовувати для задоволення потреб користувачів. Нефункціональні вимоги визначають характеристики якості системи, такі як продуктивність, безпека, надійність та зручність використання [7].

У контексті розробки віджета онлайн-бронювання особливого значення набуває узгодження вимог між різними категоріями користувачів системи. Кінцеві користувачі зацікавлені у швидкому та зручному процесі бронювання, тоді як адміністратори потребують інструментів для ефективного управління номерним фондом, ціноутворенням та контролю бронювань. Тому при

формуванні вимог необхідно враховувати як бізнес-процеси підприємства, так і очікування майбутніх користувачів системи.

На рисунку 1.4 представлено узагальнену структуру вимог до системи.

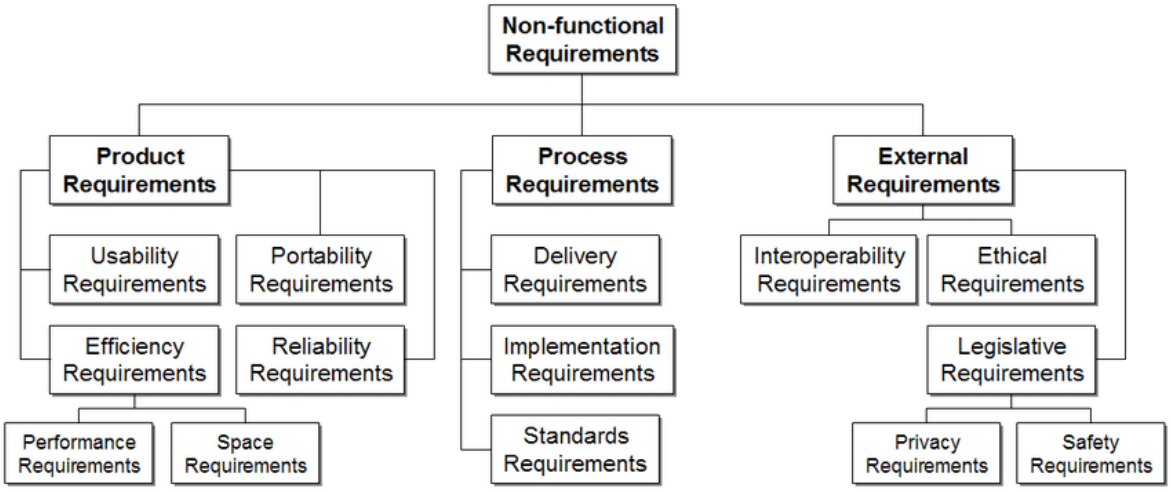

Рис. 1.4. Класифікація вимог до програмної системи

Примітка. Джерело: Розроблено із використанням [7]

Як показано на рисунку 1.4, усі вимоги поділяються на дві основні категорії, кожна з яких відіграє важливу роль у процесі проєктування та реалізації системи.

Функціональні вимоги

Функціональні вимоги визначають основні сценарії використання системи та перелік операцій, які повинні бути доступні користувачам. Для віджета онлайн-бронювання можна виділити такі ключові функціональні можливості:

1. Реєстрація та автентифікація користувачів. Система повинна забезпечувати можливість створення облікового запису, входу до системи та управління сесією користувача. Для цього використовуються механізми автентифікації, зокрема токени доступу.

2. Перегляд доступних номерів. Користувач повинен мати змогу отримати список доступних номерів із зазначенням їх характеристик (назва, ціна, тип).

3. Вибір дат бронювання. Система повинна надавати зручний інтерфейс для вибору дат за допомогою календаря з урахуванням уже зайнятих періодів.

4. Перевірка доступності номерів. Перед створенням бронювання система повинна перевіряти, чи не зайнятий обраний номер у зазначений період.

5. Розрахунок вартості проживання. Система повинна автоматично обчислювати загальну вартість бронювання з урахуванням кількості днів та можливих змін ціни залежно від дати [8].

6. Створення бронювання. Користувач повинен мати можливість оформити бронювання, після чого інформація зберігається у базі даних.

7. Перегляд власних бронювань. Зареєстрований користувач повинен мати доступ до списку своїх бронювань.

8. Скасування бронювання. Система повинна дозволяти користувачу скасовувати раніше створені бронювання.

9. Адміністративне управління. Адміністратор системи повинен мати можливість:

– керувати правилами ціноутворення;

– переглядати всі бронювання;

– редагувати параметри системи.

10. Управління цінами (динамічне ціноутворення). Система повинна підтримувати встановлення різних цін для різних періодів часу.

Наведені функціональні вимоги формують основу логіки роботи системи онлайн-бронювання та визначають взаємодію користувачів із програмним продуктом. Реалізація зазначених можливостей забезпечує повний цикл бронювання — від пошуку доступного номера та вибору дат до оформлення й подальшого керування бронюванням. Особливу увагу приділено автоматичній перевірці доступності номерів і механізмам динамічного ціноутворення, оскільки саме ці функції безпосередньо впливають на ефективність використання номерного фонду та зручність

роботи користувачів. Для наочного відображення взаємодії між користувачами системи та її основними функціями доцільно використати діаграму варіантів використання, яка дозволяє визначити ролі учасників та перелік доступних їм операцій.

На рисунку 1.5 представлено діаграму варіантів використання системи онлайн-бронювання.

Рис. 1.5. Діаграма варіантів використання системи

Примітка. Джерело: Розроблено із використанням [9]

Як видно з рисунка 1.5, система передбачає взаємодію двох основних типів користувачів, кожен із яких має власний набір функцій.

Нефункціональні вимоги

Нефункціональні вимоги визначають якісні характеристики системи та впливають на її ефективність і зручність використання.

1. Продуктивність. Система повинна забезпечувати швидку обробку запитів користувачів. Час відповіді сервера не повинен перевищувати допустимих значень навіть при одночасній роботі декількох користувачів.

2. Надійність. Система повинна забезпечувати коректну роботу без збоїв, а у випадку помилок – мати механізми відновлення. Дані про бронювання не повинні втрачатися.

3. Безпека. Система повинна забезпечувати захист персональних даних користувачів. Для цього застосовуються:

– шифрування паролів;

– використання захищених протоколів передачі даних;

– механізми авторизації доступу [9].

4. Масштабованість. Система повинна мати можливість розширення у разі збільшення кількості користувачів або обсягу даних.

5. Зручність використання (юзабіліті). Інтерфейс користувача повинен бути інтуїтивно зрозумілим, простим у використанні та адаптованим до різних пристроїв.

6. Кросплатформність. Система повинна коректно працювати у різних браузерах та на різних платформах.

7. Модульність. Архітектура системи повинна бути побудована таким чином, щоб окремі компоненти могли змінюватися або доповнюватися без впливу на всю систему.

8. Безперервність роботи. Система повинна бути доступною для користувачів у будь-який час (24/7), що є критично важливим для онлайн-сервісів.

Таким чином, визначення функціональних та нефункціональних вимог дозволяє сформувати чітке уявлення про можливості та характеристики майбутньої системи. Урахування цих вимог є основою для подальшого проєктування архітектури та реалізації віджета онлайн-бронювання.

1.4 Огляд технологій та інструментів розробки

Вибір технологій та інструментів розробки є важливим етапом створення програмного забезпечення, оскільки саме він визначає продуктивність, масштабованість, зручність підтримки та подальший розвиток системи. Для реалізації віджета онлайн-бронювання було обрано сучасний стек веб-технологій, який забезпечує ефективну взаємодію між клієнтською та серверною частинами.

Розробка системи базується на клієнт-серверній архітектурі, що передбачає розподіл функціональності між інтерфейсом користувача (frontend) та серверною логікою (backend). Обмін даними між компонентами здійснюється за допомогою HTTP-запитів у форматі JSON [10].

На рисунку 1.6 представлено загальну архітектуру використаного технологічного стеку.

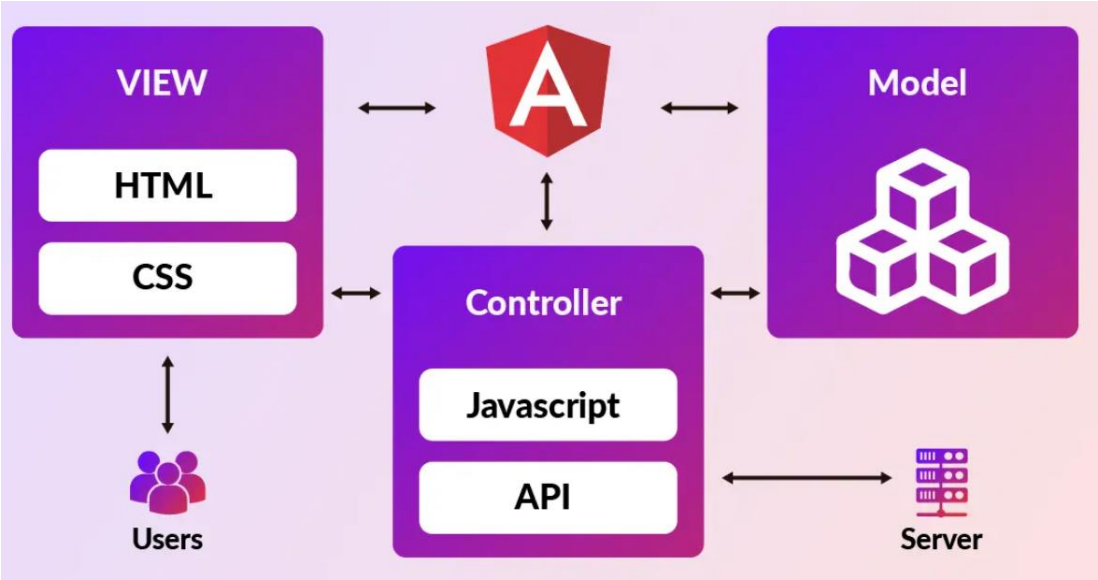

Рис. 1.6. Архітектура технологічного стеку системи

Примітка. Джерело: Розроблено із використанням [10]

Як показано на рисунку 1.6, система складається з декількох взаємопов’язаних рівнів, кожен з яких виконує окремі функції.

Серверна частина (Backend)

Для реалізації серверної частини було обрано мову програмування Python, яка є однією з найпопулярніших мов для розробки вебзастосунків завдяки простоті синтаксису, широкому набору бібліотек та активній спільноті розробників.

У якості веб-фреймворку використано FastAPI – сучасний високопродуктивний фреймворк для створення REST API. Основними перевагами FastAPI є:

– висока швидкість роботи;

– автоматична генерація документації API;

– підтримка асинхронних запитів;

– зручна інтеграція з бібліотеками для роботи з базами даних [11].

Для роботи з базою даних застосовується бібліотека SQLAlchemy, яка реалізує об’єктно-реляційне відображення (ORM). Це дозволяє працювати з даними у вигляді об’єктів мови програмування, що спрощує розробку та підвищує читабельність коду.

Клієнтська частина (Frontend)

Клієнтська частина системи реалізована з використанням мови JavaScript, яка забезпечує динамічну взаємодію користувача з веб-інтерфейсом. Використання чистого JavaScript (без складних фреймворків) дозволяє створити легкий та швидкий віджет, який легко інтегрується у будь-який веб-сайт.

Для оформлення інтерфейсу використовується CSS-фреймворк Bootstrap, який надає готові компоненти та стилі для створення адаптивного дизайну. Це дозволяє забезпечити коректне відображення інтерфейсу на різних пристроях, включаючи мобільні телефони та планшети.

Для реалізації вибору дат застосовується бібліотека Flatpickr, яка забезпечує зручний календар із можливістю вибору діапазону дат, відображення недоступних періодів та додаткової інформації, зокрема вартості проживання.

База даних

У якості системи керування базами даних використано MySQL – одну з найпоширеніших реляційних СКБД. Основними перевагами MySQL є:

– висока продуктивність;

– надійність зберігання даних;

– підтримка транзакцій;

– широке використання у веб-розробці [12].

База даних використовується для зберігання інформації про користувачів, номери, бронювання та правила ціноутворення. Реляційна модель даних дозволяє забезпечити цілісність інформації та ефективний доступ до неї.

Взаємодія компонентів (REST API)

Взаємодія між клієнтською та серверною частинами здійснюється через REST API. Це архітектурний підхід, який базується на використанні стандартних HTTP-методів (GET, POST, PUT, DELETE) для виконання операцій над ресурсами.

Формат обміну даними – JSON, що є зручним для обробки як на стороні клієнта, так і на стороні сервера. Такий підхід забезпечує універсальність і можливість інтеграції з іншими системами.

Автентифікація та безпека

Для забезпечення безпеки доступу до системи використовується механізм автентифікації на основі JSON Web Token (JWT). Після успішного входу користувач отримує токен, який використовується для підтвердження його прав при подальших запитах до сервера.

Основними перевагами використання JWT є:

– відсутність необхідності зберігання сесій на сервері;

– можливість масштабування системи;

– підвищена безпека передачі даних.

Крім того, паролі користувачів зберігаються у зашифрованому вигляді з використанням алгоритмів хешування, що забезпечує додатковий рівень захисту.

Отже, обраний стек технологій забезпечує створення сучасного, ефективного та масштабованого програмного рішення. Використання FastAPI для серверної частини, JavaScript для клієнтської, а також MySQL для зберігання даних дозволяє реалізувати всі необхідні функції системи онлайн-бронювання.

Застосування REST API та механізмів автентифікації забезпечує гнучкість взаємодії між компонентами та безпеку системи. Обрані інструменти є оптимальними для реалізації віджета онлайн-бронювання та відповідають сучасним вимогам до веб-застосунків.

Висновки до розділу 1

У першому розділі кваліфікаційної роботи було проведено комплексний аналіз предметної області онлайн-бронювання у сфері гостинного бізнесу, досліджено існуючі програмні рішення, визначено вимоги до системи, а також розглянуто сучасні технології та інструменти розробки.

У результаті аналізу встановлено, що системи онлайн-бронювання є важливим інструментом цифровізації підприємств гостинного бізнесу, який дозволяє забезпечити автоматизацію процесу обслуговування клієнтів, підвищити ефективність роботи персоналу та надати користувачам можливість здійснювати бронювання у режимі реального часу. Визначено основні особливості таких систем, зокрема необхідність забезпечення актуальності даних, перевірки доступності номерів, реалізації динамічного ціноутворення та гарантування безпеки інформації.

Проведений аналіз існуючих веб-платформ бронювання показав, що такі рішення, як Booking.com та Airbnb, мають широкий функціонал і забезпечують ефективну взаємодію між користувачами та постачальниками

послуг. Водночас вони характеризуються низкою недоліків, серед яких залежність від сторонніх сервісів, обмежена гнучкість налаштування та наявність комісійних витрат. Це підтверджує доцільність розробки власного віджета онлайн-бронювання, який може бути інтегрований безпосередньо у веб-сайт підприємства.

У межах розділу також було сформульовано функціональні та нефункціональні вимоги до системи. До основних функціональних вимог віднесено можливість реєстрації та автентифікації користувачів, перегляду доступних номерів, вибору дат, розрахунку вартості проживання, створення та скасування бронювань, а також реалізацію адміністративного функціоналу. Нефункціональні вимоги охоплюють такі характеристики, як продуктивність, надійність, безпека, масштабованість, зручність використання та сумісність.

Крім того, було проведено огляд сучасних технологій та інструментів розробки, що використовуються для створення веб-застосунків. Обґрунтовано вибір клієнт-серверної архітектури, використання мови програмування Python та фреймворку FastAPI для серверної частини, JavaScript і Bootstrap для клієнтської частини, а також системи керування базами даних MySQL. Розглянуто особливості застосування REST API та механізмів автентифікації на основі JWT.

Таким чином, результати проведеного аналізу дозволили сформувати цілісне уявлення про предметну область, визначити ключові вимоги до системи та обґрунтувати вибір технологічного стеку. Отримані результати є основою для подальшого проєктування архітектури системи та реалізації віджета онлайн-бронювання, що буде розглянуто у наступному розділі.

РОЗДІЛ 2
ПРОЄКТУВАННЯ ТА РОЗРОБКА ВІДЖЕТА ОНЛАЙН-БРОНЮВАННЯ

2.1 Архітектура системи

Архітектура програмної системи визначає її структуру, принципи взаємодії компонентів та розподіл функціональності між ними. Для розробки віджета онлайн-бронювання обрано клієнт-серверну архітектуру з використанням підходу REST, що забезпечує гнучкість, масштабованість та зручність інтеграції із зовнішніми системами [13].

Основна ідея архітектури полягає у розподілі системи на окремі логічні рівні: клієнтський (frontend), серверний (backend) та рівень даних (database). Кожен із цих рівнів виконує окремі функції та взаємодіє з іншими через чітко визначені інтерфейси.

На рисунку 2.1 представлено загальну архітектуру системи онлайн-бронювання.

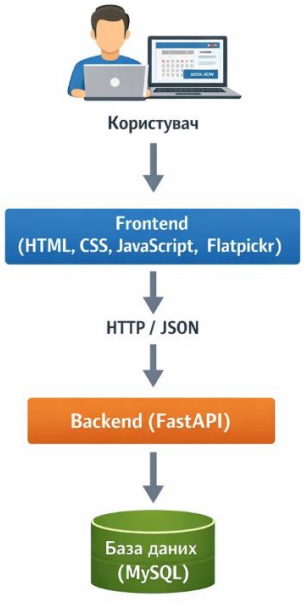

Рис. 2.1. Загальна архітектура системи

(розроблено автором)

Як показано на рисунку 2.1, користувач взаємодіє із системою через веб-інтерфейс, який надсилає запити до серверної частини. Сервер обробляє запити, виконує бізнес-логіку та взаємодіє з базою даних для збереження або отримання інформації.

Клієнтський рівень (Frontend)

Клієнтський рівень реалізований у вигляді веб-віджета, який може бути інтегрований у будь-який сайт гостинного бізнесу. Основними функціями frontend є:

– відображення доступних номерів;

– вибір дат бронювання за допомогою календаря;

– відображення вартості проживання;

– взаємодія з користувачем (введення даних, підтвердження дій).

Особливістю реалізації є використання бібліотеки Flatpickr для створення інтерактивного календаря з можливістю відображення зайнятих дат та динамічних цін. Взаємодія з сервером здійснюється за допомогою асинхронних HTTP-запитів (fetch API), що забезпечує швидке оновлення даних без перезавантаження сторінки [14].

Серверний рівень (Backend)

Серверна частина системи реалізована з використанням фреймворку FastAPI та відповідає за обробку запитів, реалізацію бізнес-логіки та взаємодію з базою даних [14].

Основні функції backend:

– обробка запитів від клієнта;

– перевірка доступності номерів;

– розрахунок вартості бронювання;

– створення, перегляд і скасування бронювань;

– автентифікація та авторизація користувачів;

– управління правилами ціноутворення.

Важливою складовою серверної частини є реалізація REST API, яка дозволяє стандартизувати обмін даними між клієнтом і сервером. Основні ендпоінти системи включають:

– /rooms – отримання списку номерів;

– /bookings – створення бронювання;

– /bookings/my – отримання бронювань користувача;

– /calculate-price – розрахунок вартості;

– /admin/price-rules – управління цінами.

Рівень даних (Database)

Рівень даних реалізований на базі реляційної системи керування базами даних MySQL. Для взаємодії з базою даних використовується ORM-бібліотека SQLAlchemy, що дозволяє працювати з таблицями як з об’єктами [14].

Основними сутностями бази даних є:

– користувачі (users);

– номери (rooms);

– бронювання (bookings);

– правила ціноутворення (price_rules).

Структура бази даних побудована з урахуванням необхідності забезпечення цілісності та узгодженості інформації під час виконання операцій бронювання. Взаємозв’язки між сутностями дозволяють ефективно зберігати дані про користувачів, доступні номери, створені бронювання та правила зміни вартості проживання. Такий підхід спрощує реалізацію бізнес-логіки системи, забезпечує швидкий доступ до необхідної інформації та підтримує масштабування програмного продукту в разі розширення його функціональних можливостей. Використання ORM-технології додатково підвищує зручність розробки, оскільки дає змогу працювати з об’єктною моделлю предметної області без необхідності створення складних SQL-запитів для кожної операції.

На рисунку 2.2 представлено спрощену UML-діаграму класів системи.

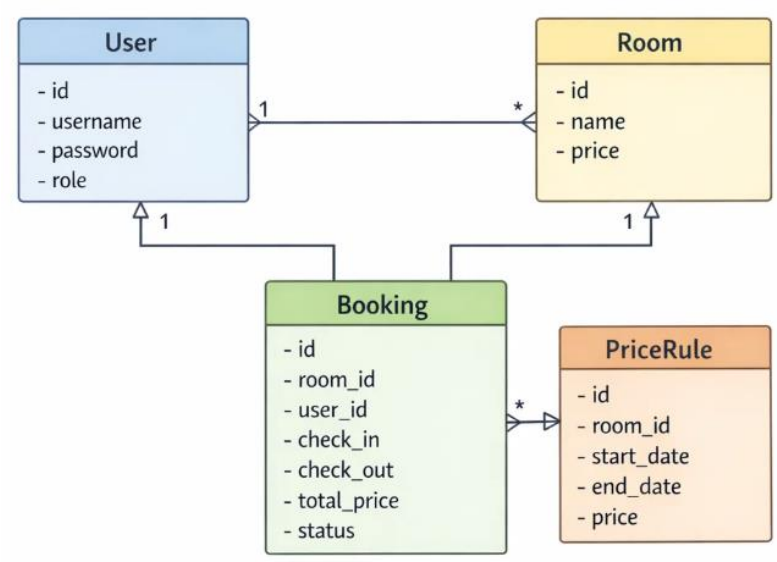

Рис. 2.2. UML-діаграма класів системи

(розроблено автором)

Як показано на рисунку 2.2, структура бази даних відображає основні бізнес-сутності системи та їх взаємозв’язки.

Взаємодія компонентів (UML діаграма послідовності)

Процес бронювання номера можна представити у вигляді UML-діаграми послідовності (рисунок 2.3).

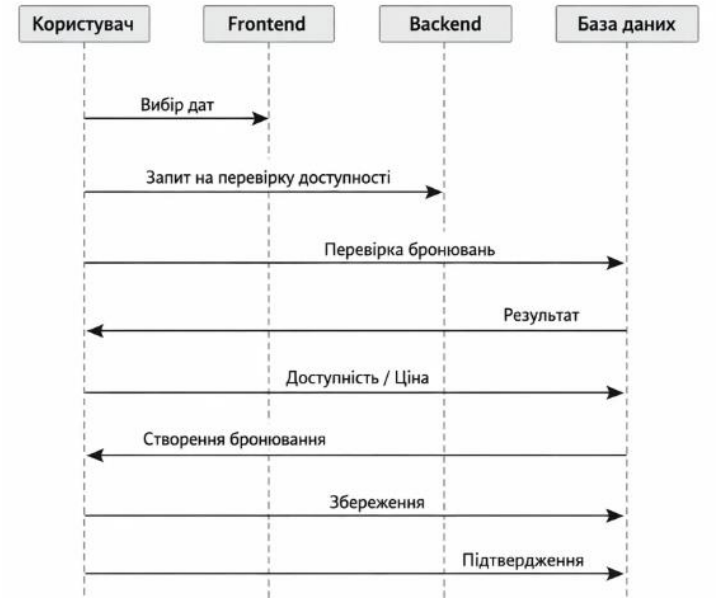

Рис. 2.3. UML-діаграма послідовності процесу бронювання

(розроблено автором)

Ця діаграма демонструє послідовність взаємодії між компонентами системи під час виконання основного сценарію.

Особливості архітектурного рішення

Обрана архітектура має низку переваг:

– модульність (frontend і backend незалежні);

– масштабованість (можливість розширення системи);

– зручність підтримки;

– можливість інтеграції з іншими сервісами;

– використання сучасних стандартів веб-розробки.

Водночас така архітектура потребує врахування певних аспектів, зокрема забезпечення безпеки API, обробки помилок та оптимізації запитів до бази даних.

Таким чином, архітектура розробленої системи онлайн-бронювання забезпечує ефективну взаємодію між компонентами, підтримує реалізацію необхідного функціоналу та відповідає сучасним вимогам до веб-застосунків. Обрана структура є основою для подальшої реалізації та розвитку програмного продукту.

2.2 Проєктування бази даних

Проєктування бази даних є важливим етапом розробки інформаційної системи, оскільки саме структура даних визначає ефективність зберігання, обробки та доступу до інформації. Для реалізації віджета онлайн-бронювання було обрано реляційну модель даних, яка забезпечує цілісність, надійність та зручність роботи з інформацією [15].

У якості системи керування базами даних використано MySQL, що дозволяє реалізувати ефективне зберігання даних, підтримку транзакцій та забезпечення цілісності інформації [16].

Визначення основних сутностей

На основі аналізу предметної області та сформульованих вимог було визначено основні сутності бази даних:

– User (Користувач) – зберігає інформацію про зареєстрованих користувачів системи;

– Room (Номер) – містить інформацію про доступні номери;

– Booking (Бронювання) – відображає факти бронювання номерів;

– PriceRule (Правило ціни) – визначає зміну вартості номера залежно від періоду.

Кожна із зазначених сутностей реалізована у вигляді окремої таблиці бази даних.

Структура таблиць бази даних

Таблиця users призначена для зберігання облікових записів користувачів і має таку структуру:

– id – унікальний ідентифікатор користувача (первинний ключ);

– username – логін користувача (унікальне значення);

– password – зашифрований пароль;

– role – роль користувача (user або admin).

Таблиця rooms містить інформацію про номери:

– id – унікальний ідентифікатор номера;

– name – назва номера;

– price – базова вартість за добу.

Таблиця bookings є центральною у системі та зберігає інформацію про бронювання:

– id – унікальний ідентифікатор бронювання;

– room_id – ідентифікатор номера (зовнішній ключ);

– user_id – ідентифікатор користувача (зовнішній ключ);

– check_in – дата заїзду;

– check_out – дата виїзду;

– total_price – загальна вартість бронювання;

– status – статус бронювання (pending, confirmed, cancelled).

Таблиця price_rules використовується для реалізації динамічного ціноутворення:

– id – унікальний ідентифікатор правила;

– room_id – ідентифікатор номера;

– start_date – початок дії правила;

– end_date – кінець дії правила;

– price – встановлена ціна.

Зв’язки між таблицями

Між таблицями встановлено такі зв’язки:

– один користувач може мати багато бронювань (зв’язок «один до багатьох» між users та bookings);

– один номер може мати багато бронювань (зв’язок «один до багатьох» між rooms та bookings);

– один номер може мати декілька правил ціноутворення (зв’язок «один до багатьох» між rooms та price_rules).

На рисунку 2.4 представлено ER-діаграму бази даних системи.

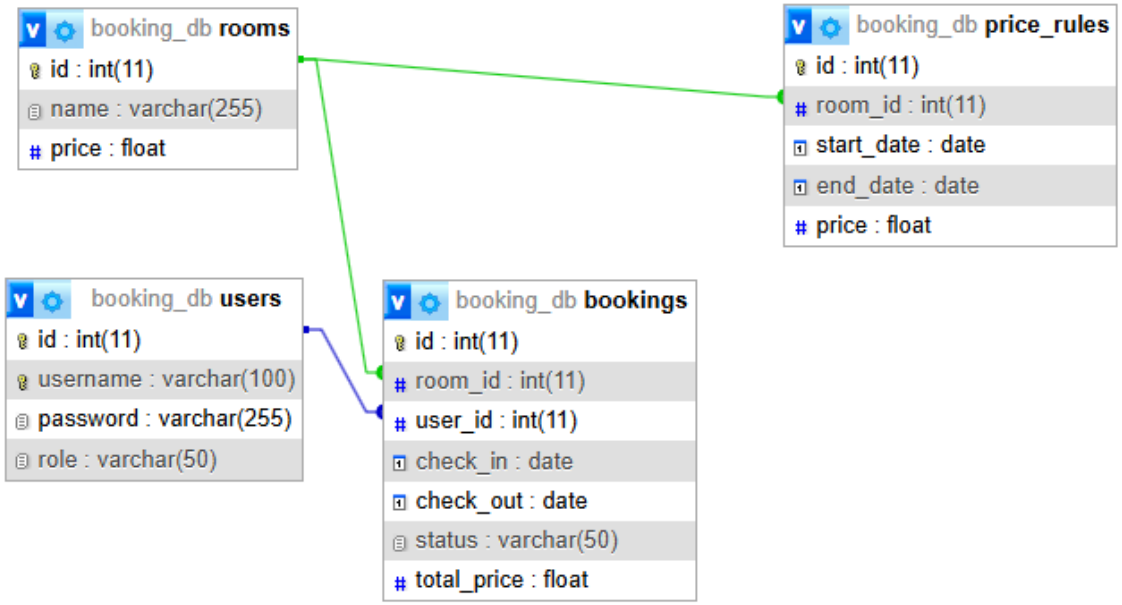

Рис. 2.4. ER-діаграма бази даних

(розроблено автором)

Як показано на рисунку 2.4, таблиця bookings є центральною і пов’язує користувачів та номери.

Нормалізація бази даних

При проєктуванні бази даних було застосовано принципи нормалізації, що дозволяє уникнути надлишковості даних та забезпечити їх цілісність. Зокрема:

– дані про користувачів, номери та бронювання розділені на окремі таблиці;

– відсутнє дублювання інформації;

– використано зовнішні ключі для забезпечення зв’язків між таблицями [17].

Така структура відповідає третій нормальній формі (3NF), що є стандартом для більшості інформаційних систем.

Особливості реалізації

У розробленій системі реалізовано декілька важливих аспектів роботи з базою даних:

1. Перевірка доступності номерів. Перед створенням бронювання виконується запит до таблиці bookings для перевірки перетину дат. Це дозволяє уникнути подвійного бронювання.

2. Динамічне ціноутворення. Таблиця price_rules використовується для визначення ціни на конкретну дату. У разі наявності правила ціна береться з нього, інакше використовується базова ціна номера.

3. Soft delete бронювань. Замість фізичного видалення записів використовується поле status, яке дозволяє позначати бронювання як «cancelled». Це забезпечує збереження історії операцій.

4. Обчислення вартості бронювання. Загальна вартість розраховується на сервері шляхом підсумовування цін за кожен день проживання з урахуванням правил ціноутворення.

Використання ORM

Для взаємодії з базою даних використовується бібліотека SQLAlchemy, яка реалізує об’єктно-реляційне відображення. Це дозволяє:

– працювати з таблицями як з класами Python;

– спростити виконання запитів;

– зменшити кількість помилок при роботі з базою даних.

Кожна таблиця представлена у вигляді окремого класу (User, Room, Booking, PriceRule), що відповідає принципам об’єктно-орієнтованого програмування.

Таким чином, у процесі проєктування бази даних було створено ефективну структуру, яка забезпечує зберігання основних даних системи онлайн-бронювання. Використання реляційної моделі, нормалізації та механізмів зв’язків дозволяє забезпечити цілісність і узгодженість інформації.

Реалізована база даних підтримує всі необхідні функції системи, включаючи управління користувачами, бронюваннями та цінами. Вона є масштабованою та може бути розширена у майбутньому для додавання нових можливостей.

2.3 Реалізація серверної частини (backend)

Серверна частина системи онлайн-бронювання реалізує основну бізнес-логіку застосунку, забезпечує обробку запитів клієнта, взаємодію з базою даних та контроль доступу користувачів. У розробленому проєкті backend побудований із використанням сучасного веб-фреймворку FastAPI, що дозволяє створювати високопродуктивні RESTful API [18].

Загальна структура серверної частини

Серверна частина організована за модульним принципом і складається з наступних основних компонентів:

– main.py – точка входу в застосунок, визначає маршрути (ендпоінти);

– models.py – опис структури таблиць бази даних;

– schemas.py – визначення моделей даних для валідації запитів і відповідей;

– crud.py – реалізація бізнес-логіки (операції створення, читання, оновлення, видалення);

– auth.py – модуль автентифікації та роботи з токенами;

– database.py – налаштування підключення до бази даних.

Така структура відповідає принципам розділення відповідальності (Separation of Concerns), що спрощує підтримку та масштабування системи [19].

Реалізація REST API

У системі реалізовано REST API для взаємодії між клієнтською та серверною частинами. Основні ендпоінти згруповані за функціональним призначенням:

1. Аутентифікація та реєстрація:

– POST /register – створення нового користувача;

– POST /login – автентифікація користувача та отримання токена.

2. Робота з номерами:

– GET /rooms – отримання списку доступних номерів.

3. Бронювання:

– POST /bookings – створення нового бронювання;

– GET /bookings/my – отримання бронювань користувача;

– GET /bookings/{room_id} – отримання бронювань для конкретного номера;

– DELETE /bookings/{booking_id} – скасування бронювання.

4. Ціноутворення:

– GET /calculate-price – розрахунок вартості бронювання;

– GET /prices/{room_id} – отримання цін за період.

5. Адміністративний функціонал:

– GET /admin/price-rules – отримання правил ціноутворення;

– POST /admin/price-rules – створення правила;

– PUT /admin/price-rules/{id} – оновлення правила;

– DELETE /admin/price-rules/{id} – видалення правила.

Використання HTTP-методів (GET, POST, PUT, DELETE) відповідає принципам REST-архітектури та забезпечує зрозумілу структуру API.

Автентифікація та авторизація

У системі реалізовано механізм автентифікації на основі JWT (JSON Web Token). Після успішного входу користувач отримує токен, який використовується для доступу до захищених ресурсів.

Основні етапи роботи:

– перевірка логіна та пароля;

– генерація JWT-токена з даними користувача (username, role);

– передача токена клієнту;

– перевірка токена при кожному запиті.

Для обмеження доступу до адміністративних функцій використовується перевірка ролі користувача. Якщо роль не відповідає вимогам, сервер повертає помилку доступу.

Паролі користувачів зберігаються у зашифрованому вигляді з використанням алгоритму bcrypt, що підвищує рівень безпеки системи.

Реалізація бізнес-логіки

Основна бізнес-логіка реалізована у модулі crud.py. До ключових функцій належать:

1. Перевірка доступності номера (is_room_available). Функція перевіряє наявність конфліктів бронювання шляхом пошуку записів, що перетинаються за датами.

2. Створення бронювання (create_booking). Перед створенням бронювання виконується:

– перевірка коректності дат;

– перевірка доступності номера;

– розрахунок вартості.

Після цього запис додається до бази даних.

3. Розрахунок вартості (calculate_price). Вартість обчислюється для кожного дня окремо з урахуванням правил ціноутворення. Якщо для конкретної дати існує правило, використовується його ціна, інакше – базова ціна номера.

4. Скасування бронювання (delete_booking). Реалізовано механізм «м’якого видалення» (soft delete), при якому статус бронювання змінюється на cancelled.

Валідація даних

Для валідації вхідних даних використовується бібліотека Pydantic, яка дозволяє:

– перевіряти типи даних;

– автоматично перетворювати формати;

– генерувати документацію API.

Моделі schemas.py описують структуру запитів і відповідей, що забезпечує коректність взаємодії між клієнтом і сервером.

Робота з базою даних

Для доступу до бази даних використовується SQLAlchemy. Підключення реалізовано через модуль database.py, який створює сесію для кожного запиту.

Використання ORM дозволяє:

– спростити написання запитів;

– забезпечити переносимість коду;

– зменшити ймовірність помилок.

Усі операції з базою даних виконуються у межах транзакцій, що гарантує цілісність даних.

Обробка помилок

У серверній частині реалізовано обробку помилок за допомогою HTTPException. У разі виникнення помилки користувачу повертається відповідний HTTP-статус та повідомлення.

Основні типи помилок:

– 400 – некоректні дані;

– 401 – помилка автентифікації;

– 403 – відсутність прав доступу;

– 404 – ресурс не знайдено.

Таким чином, серверна частина системи реалізована із використанням сучасних технологій та підходів до розробки веб-застосунків. Використання FastAPI, JWT-автентифікації та ORM забезпечує високу продуктивність, безпеку та зручність розширення системи.

Реалізований backend повністю підтримує функціональні вимоги системи, забезпечує коректну обробку даних та є надійною основою для роботи віджета онлайн-бронювання.

2.4 Реалізація клієнтської частини (frontend)

Клієнтська частина системи онлайн-бронювання відповідає за взаємодію з користувачем, відображення даних та формування запитів до серверної частини. У розробленому проєкті frontend реалізовано у вигляді веб-застосунку з використанням HTML, CSS та JavaScript, що забезпечує кросплатформеність та доступність через браузер [20].

Загальна структура клієнтської частини

Клієнтська частина організована у вигляді набору HTML-сторінок і JavaScript-файлів, кожен з яких відповідає за окремий функціональний модуль:

– index.html / app.js / index.js – основний інтерфейс бронювання;

– login.html / login.js – сторінка входу;

– register.html / register.js – сторінка реєстрації;

– profile.html / profile.js – сторінка перегляду бронювань;

– admin.html / admin.js – адміністративна панель управління цінами;

– style.css – стилізація інтерфейсу.

На рисунку 2.5 представлено структуру клієнтської частини.

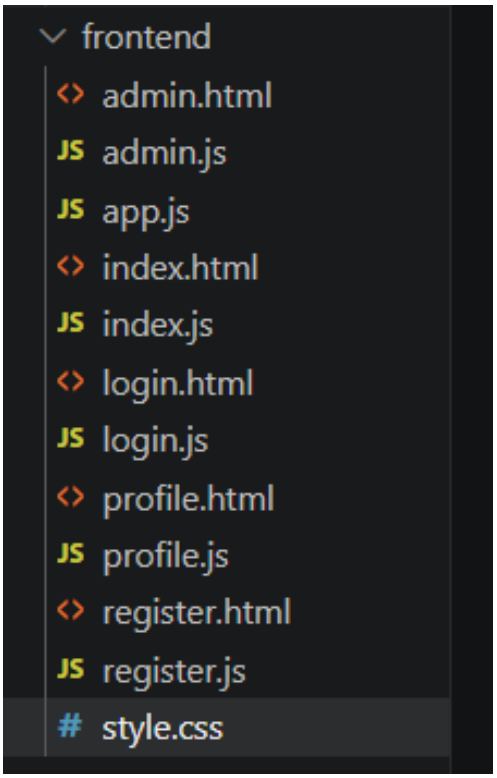

Рис. 2.5. Структура клієнтської частини

(розроблено автором)

Така структура дозволяє розділити логіку за функціональними модулями та спростити підтримку коду.

Основний інтерфейс бронювання

Головна сторінка системи (index.html) реалізує базовий сценарій взаємодії користувача із системою. Основні елементи інтерфейсу:

– список доступних номерів;

– календар вибору дат;

– блок відображення вартості;

– кнопка створення бронювання.

Під час розробки користувацького інтерфейсу особлива увага приділялася зручності використання та мінімізації кількості дій, необхідних для оформлення бронювання. Усі основні елементи сторінки розташовані таким чином, щоб користувач міг послідовно виконати вибір номера, визначити дати проживання, ознайомитися з розрахованою вартістю та

підтвердити бронювання. Додатково інтерфейс забезпечує оперативне оновлення даних без перезавантаження сторінки, що позитивно впливає на швидкість взаємодії із системою та покращує загальний користувацький досвід. Використання сучасних засобів розробки вебінтерфейсів також дозволяє забезпечити коректне відображення сторінки на різних типах пристроїв і розмірах екранів.

На рисунку 2.6 представлено інтерфейс сторінки бронювання.

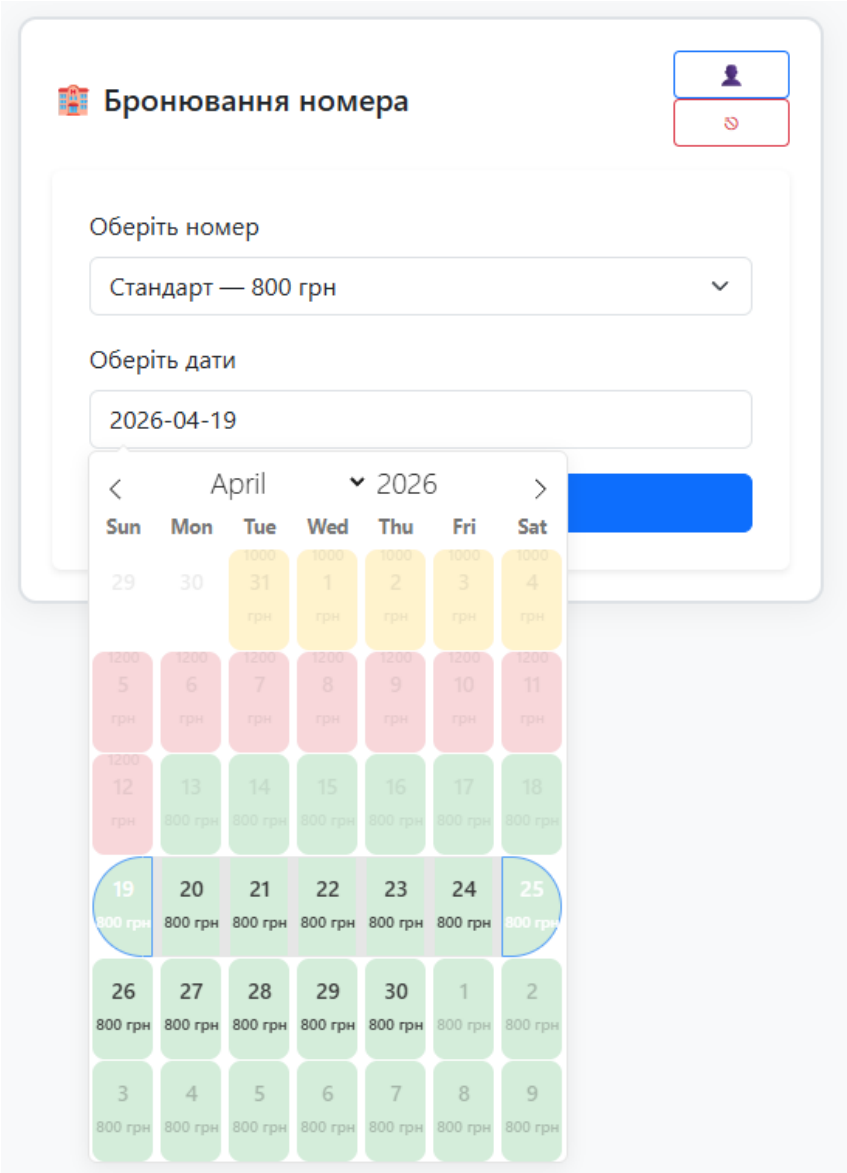

Рис. 2.6. Інтерфейс сторінки бронювання

(розроблено автором)

Як показано на рисунку 2.5, інтерфейс побудований із використанням компонентів бібліотеки Bootstrap, що забезпечує адаптивність та зручність користування.

Робота з календарем

Для реалізації вибору дат використовується бібліотека Flatpickr, яка забезпечує:

– вибір діапазону дат (mode: "range");

– блокування зайнятих періодів;

– відображення цін для кожного дня.

Особливістю реалізації є динамічне підсвічування дат залежно від вартості. Для цього використовується обробник події onDayCreate, який:

– визначає ціну для кожного дня;

– застосовує відповідний CSS-клас (low-price, mid-price, high-price);

– додає тег із ціною.

Це дозволяє користувачу візуально оцінити вартість проживання ще до вибору дат.

Взаємодія з сервером

Клієнтська частина взаємодіє із сервером за допомогою Fetch API. Використовуються асинхронні запити для:

– отримання списку номерів;

– отримання зайнятих дат;

– отримання цін на період;

– створення бронювання;

– авторизації користувача.

Приклад запиту:

– GET /rooms – отримання номерів;

– POST /bookings – створення бронювання;

– GET /calculate-price – розрахунок вартості.

Використання асинхронних запитів дозволяє уникнути перезавантаження сторінки та забезпечує кращий користувацький досвід [21].

Реалізація авторизації

У клієнтській частині реалізовано механізм збереження JWT-токена у локальному сховищі браузера (localStorage). Після успішного входу:

– токен зберігається у localStorage;

– додається до заголовків кожного запиту (Authorization: Bearer token);

– використовується для доступу до захищених ресурсів.

Також реалізовано перевірку авторизації при відкритті сторінок. У разі відсутності токена користувач перенаправляється на сторінку входу.

Сторінка профілю користувача

Сторінка profile.html дозволяє користувачу переглядати свої бронювання. Основні функції:

– отримання списку бронювань через API;

– відображення інформації (номер, дати, ціна, статус);

– можливість скасування бронювання.

Для покращення користувацького досвіду (UX) та швидкого зчитування інформації впроваджено систему кольорової індикації статусів. Це дозволяє користувачу миттєво оцінити стан своїх справ без необхідності детального читання тексту.

Таблиця 2.1

Відповідності статусів бронювання

Статус	Опис	Колір (CSS Class)	Значення
Confirmed	Підтверджено	Зелений (.bg-success)	Оплата отримана або бронь схвалена модератором.
Pending	Очікується	Жовтий (.bg-warning)	Замовлення в черзі на обробку або очікує оплати.
Cancelled	Скасовано	Сірий (.bg-secondary)	Бронювання скасовано користувачем або адміністратором.

На рисунку 2.7 представлено інтерфейс профілю користувача.

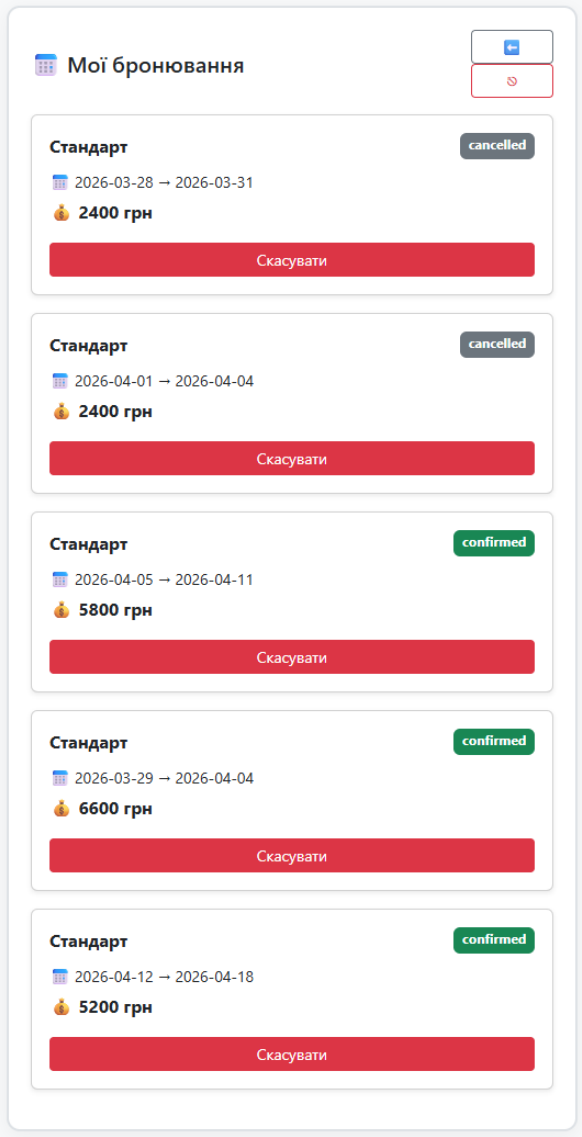

Рис. 2.7. Інтерфейс профілю користувача

(розроблено автором)

Адміністративна панель

Адміністративна частина системи (admin.html) призначена для управління правилами ціноутворення. Основні можливості:

– створення нових правил;

– редагування існуючих;

– видалення правил;

– фільтрація за кімнатами;

– відображення календаря цін.

На рисунку 2.8 представлено інтерфейс адміністративної панелі.

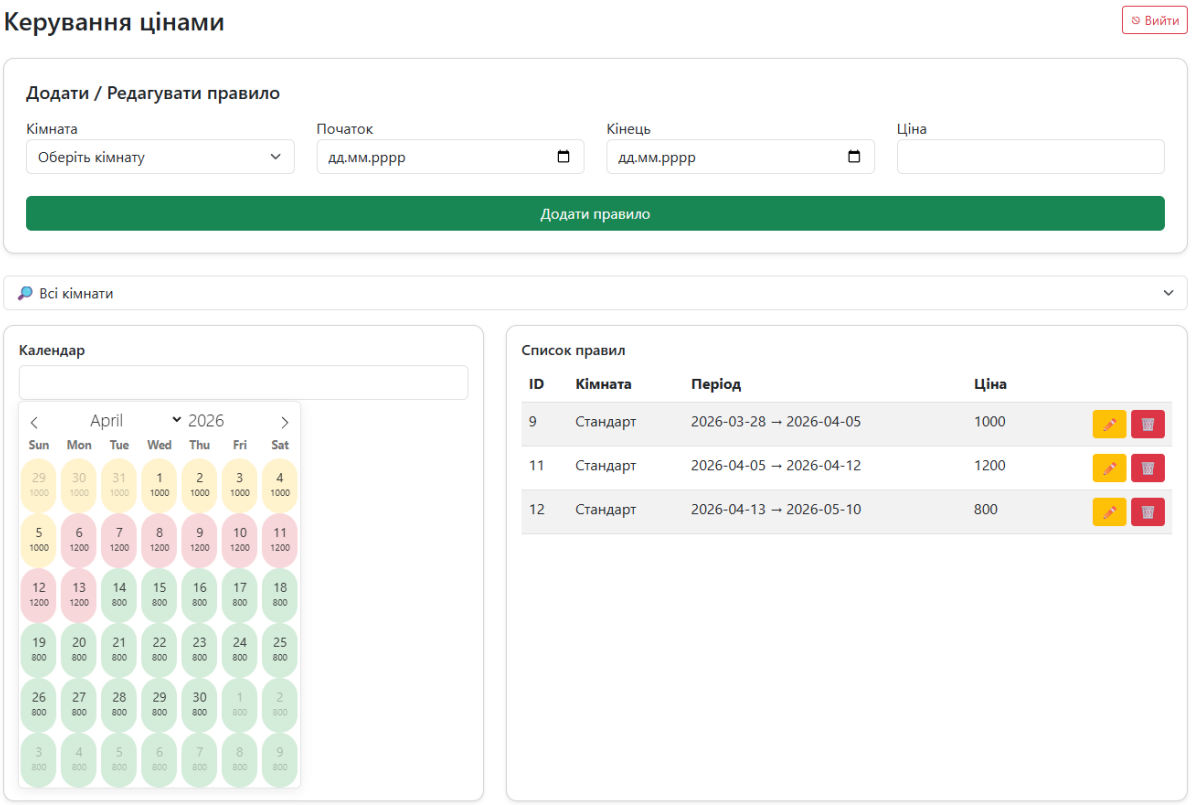

Рис. 2.8. Інтерфейс адміністративної панелі

(розроблено автором)

Особливістю є інтеграція календаря з відображенням вартості для кожного дня, що дозволяє адміністратору швидко аналізувати ціни.

Обробка подій та UI-логіка

У клієнтській частині активно використовується обробка подій:

– зміна вибраного номера;

– вибір дат;

– натискання кнопок;

– введення даних.

UI-логіка реалізована у JavaScript-функціях, що забезпечують:

– оновлення інтерфейсу;

– валідацію даних;

– відображення повідомлень про помилки або успішні дії.

Стилізація інтерфейсу

Для оформлення інтерфейсу використовується CSS та фреймворк Bootstrap. Основні переваги:

– адаптивний дизайн;

– готові UI-компоненти;

– швидка розробка інтерфейсу.

Додатково реалізовано кастомні стилі для:

– календаря;

– кольорового відображення цін;

– карток бронювань.

Таким чином, клієнтська частина системи реалізована як сучасний веб-інтерфейс, що забезпечує зручну взаємодію користувача із системою онлайн-бронювання. Використання HTML, CSS, JavaScript, Bootstrap та Flatpickr дозволило створити функціональний, адаптивний та інтуїтивно зрозумілий інтерфейс.

Реалізований frontend повністю інтегрований із серверною частиною, підтримує всі основні сценарії роботи користувача та відповідає сучасним вимогам до веб-застосунків.

2.5 Реалізація механізмів автентифікації та авторизації

Забезпечення безпеки доступу до системи є одним із ключових аспектів розробки веб-застосунків. У межах даного проєкту реалізовано механізми автентифікації та авторизації, що дозволяють ідентифікувати користувачів та обмежувати доступ до функціоналу відповідно до їх ролей.

Для реалізації цих механізмів використано підхід на основі JWT (JSON Web Token), який є одним із найбільш поширених способів забезпечення безпечної взаємодії між клієнтом і сервером у сучасних веб-системах [22].

Загальна схема автентифікації

Процес автентифікації користувача включає кілька етапів:

– введення користувачем логіна та пароля;

– передача даних на сервер;

– перевірка облікових даних;

– генерація токена доступу;

– використання токена для подальших запитів.

На рисунку 2.9 представлено загальну схему автентифікації у системі.

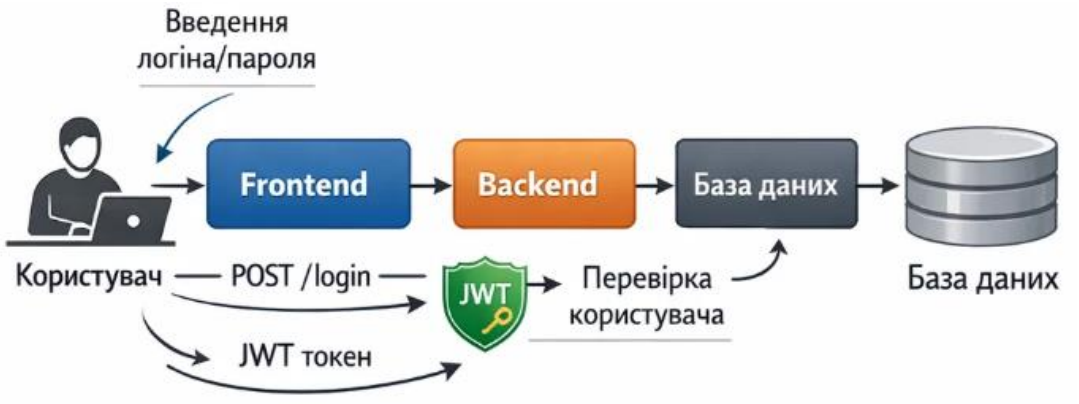

Рис. 2.9. Схема автентифікації користувача

(розроблено автором)

Як показано на рисунку 2.9, після успішної автентифікації клієнт отримує токен, який використовується для доступу до захищених ресурсів.

Реалізація автентифікації

У серверній частині автентифікація реалізована у модулі auth.py. Основні функції:

– hash_password() – виконує хешування пароля з використанням алгоритму bcrypt;

– verify_password() – перевіряє відповідність введеного пароля збереженому хешу;

– create_access_token() – генерує JWT-токен.

Під час реєстрації користувача пароль не зберігається у відкритому вигляді, а проходить процедуру хешування. Це значно підвищує рівень безпеки системи, оскільки навіть у разі компрометації бази даних паролі залишаються захищеними.

Під час входу користувача виконується порівняння введеного пароля із хешем, що зберігається у базі даних. У разі успішної перевірки створюється JWT-токен, який містить:

– ім’я користувача (sub);

– роль користувача (role);

– час дії токена (exp).

Реалізація авторизації

Авторизація у системі базується на перевірці ролі користувача, яка передається у токені. Для цього реалізовано залежності (dependencies) FastAPI:

– get_current_user() – декодує токен та отримує дані користувача; – get_admin() – перевіряє, чи має користувач роль адміністратора.

У разі відсутності або некоректності токена сервер повертає помилку 401 (Unauthorized). Якщо користувач не має необхідних прав доступу, повертається помилка 403 (Forbidden).

На рисунку 2.10 представлено схему авторизації у системі.

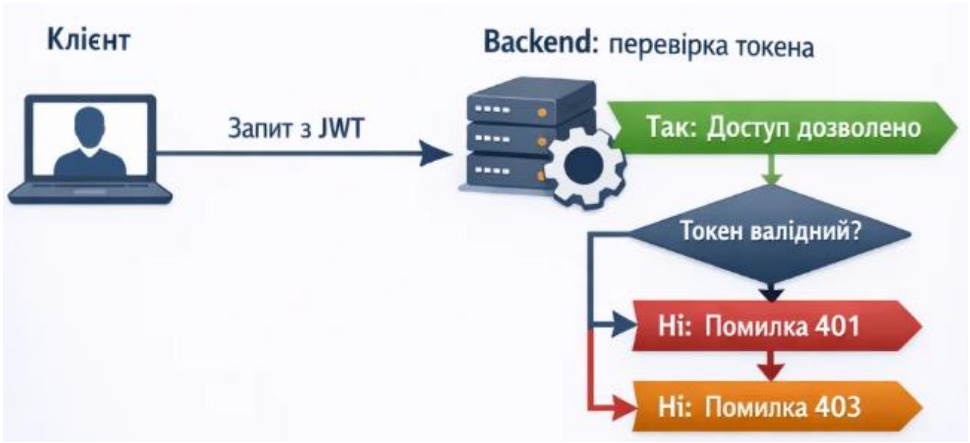

Рис. 2.10. Схема авторизації користувача

(розроблено автором)

Таким чином, система забезпечує контроль доступу до ресурсів відповідно до ролі користувача.

Зберігання та використання токена на клієнті

У клієнтській частині JWT-токен зберігається у локальному сховищі браузера (localStorage). Після входу користувача:

– токен зберігається у localStorage;

– додається до заголовків запитів;

– використовується для доступу до захищених API.

Приклад заголовка:

Authorization: Bearer <token>

Перед виконанням дій, що потребують авторизації, перевіряється наявність токена. У разі його відсутності користувач перенаправляється на сторінку входу.

Захист адміністративних функцій

Адміністративні функції (управління цінами) доступні лише користувачам із роллю admin. Це реалізовано через використання залежності get_admin у відповідних ендпоінтах.

Таким чином, забезпечується розмежування прав доступу:

– звичайні користувачі можуть створювати та переглядати бронювання;

– адміністратори мають додаткові можливості управління системою.

Час життя токена

JWT-токен має обмежений час дії, який задається під час його створення. У розробленій системі використовується параметр ACCESS_TOKEN_EXPIRE_MINUTES, що визначає тривалість сесії користувача.

Після завершення терміну дії токена користувач повинен повторно пройти автентифікацію.

Переваги обраного підходу

Використання JWT має низку переваг:

– відсутність необхідності зберігати сесії на сервері;

– зменшення навантаження на сервер;

– можливість масштабування;

– простота інтеграції з клієнтською частиною.

Водночас необхідно враховувати питання безпечного зберігання токенів та захисту від атак типу XSS [23].

Таким чином, у системі реалізовано сучасні механізми автентифікації та авторизації, що забезпечують безпечний доступ до функціоналу. Використання JWT-токенів, хешування паролів та перевірки ролей дозволяє гарантувати захист даних та контроль доступу.

Реалізовані механізми повністю відповідають вимогам безпеки веб-застосунків та забезпечують надійну роботу системи онлайн-бронювання.

Висновки до розділу 2

У другому розділі кваліфікаційної роботи було розглянуто процес проєктування та реалізації програмної системи онлайн-бронювання, що включає архітектурні рішення, структуру бази даних, серверну та клієнтську частини, а також механізми автентифікації та авторизації.

У ході виконання розділу було обґрунтовано вибір клієнт-серверної архітектури, яка забезпечує розподіл функціональності між компонентами системи та дозволяє реалізувати гнучкий і масштабований програмний продукт. Побудована архітектура забезпечує ефективну взаємодію між клієнтською частиною, сервером та базою даних.

Було здійснено проєктування реляційної бази даних, яка включає основні сутності предметної області: користувачів, номери, бронювання та правила ціноутворення. Визначено зв’язки між таблицями та забезпечено їх відповідність принципам нормалізації, що дозволяє уникнути надлишковості даних та забезпечити їх цілісність.

У межах розділу детально описано реалізацію серверної частини системи з використанням фреймворку FastAPI. Реалізовано REST API, що забезпечує взаємодію між клієнтом і сервером, а також бізнес-логіку системи, включаючи перевірку доступності номерів, розрахунок вартості бронювання та управління даними.

Клієнтська частина системи реалізована у вигляді веб-інтерфейсу з використанням HTML, CSS та JavaScript. Вона забезпечує зручну взаємодію користувача із системою, включаючи вибір номерів, дат бронювання, перегляд вартості та управління бронюваннями. Особливу увагу приділено використанню інтерактивного календаря та динамічного відображення цін.

Окремо було розглянуто реалізацію механізмів автентифікації та авторизації. Використання JWT-токенів, хешування паролів та контролю ролей дозволило забезпечити належний рівень безпеки та захисту даних користувачів.

Таким чином, у другому розділі було створено повноцінну програмну основу системи онлайн-бронювання, яка відповідає сучасним вимогам до веб-застосунків. Реалізовані рішення забезпечують функціональність, надійність, безпеку та можливість подальшого розвитку системи.

РОЗДІЛ 3
ТЕСТУВАННЯ ТА ОЦІНКА ЕФЕКТИВНОСТІ СИСТЕМИ

3.1 Методика тестування програмного забезпечення

Тестування програмного забезпечення є невід’ємною складовою процесу розробки, яка дозволяє забезпечити якість, надійність та відповідність системи встановленим вимогам. У межах даного проєкту було застосовано комплексний підхід до тестування віджета онлайн-бронювання, що охоплює перевірку як серверної, так і клієнтської частин системи [24].

Загальний підхід до тестування

У розробленій системі використано такі основні види тестування:

– функціональне тестування;

– інтеграційне тестування;

– тестування інтерфейсу користувача;

– тестування безпеки.

Тестування проводилось переважно у ручному режимі з використанням реальних сценаріїв роботи користувача, що дозволило оцінити систему з точки зору кінцевого користувача.

Функціональне тестування

Функціональне тестування спрямоване на перевірку відповідності реалізованого функціоналу встановленим вимогам. У рамках даного етапу перевірялися такі основні функції:

– реєстрація користувача;

– авторизація користувача;

– отримання списку номерів;

– вибір дат бронювання;

– розрахунок вартості;

– створення бронювання;

– скасування бронювання;

– робота адміністративної панелі.

Для кожної функції були сформовані тестові сценарії, що включають вхідні дані, очікуваний результат та фактичний результат.

На рисунку 3.1 представлено приклад сценарію функціонального тестування.

Рис. 3.1. Приклад сценарію функціонального тестування

(розроблено автором)

Як показано на рисунку 3.1, результати тестування відповідають очікуваним, що підтверджує коректність роботи функції.

Інтеграційне тестування

Інтеграційне тестування спрямоване на перевірку взаємодії між компонентами системи, зокрема між клієнтською та серверною частинами.

Було перевірено:

– коректність обміну даними через REST API;

– обробку відповідей сервером;

– взаємодію з базою даних;

– обробку помилок.

Особливу увагу приділено тестуванню сценаріїв, у яких відбувається передача даних між кількома компонентами, наприклад, при створенні бронювання або розрахунку вартості.

Тестування інтерфейсу користувача

Тестування UI дозволило оцінити зручність використання системи та правильність відображення елементів інтерфейсу. Перевірялися такі аспекти:

– коректність відображення сторінок;

– робота календаря;

– відображення цін;

– адаптивність інтерфейсу;

– обробка дій користувача.

Особливу увагу приділено роботі календаря, зокрема:

– блокуванню зайнятих дат;

– відображенню цін для кожного дня;

– коректному вибору діапазону дат.

Тестування безпеки

У межах тестування безпеки перевірялися:

– доступ до захищених ресурсів без токена;

– доступ користувача без прав адміністратора до адміністративних функцій;

– коректність перевірки JWT-токена;

– захист паролів (хешування).

Було встановлено, що система коректно обмежує доступ до функціоналу відповідно до ролі користувача та не дозволяє виконувати несанкціоновані дії.

Обробка помилок

Окремо перевірялася робота системи у випадку виникнення помилок:

– введення некоректних даних;

– спроба створення бронювання з некоректними датами;

– спроба бронювання зайнятого номера;

– відсутність підключення до сервера.

У всіх випадках система коректно обробляє помилки та повідомляє користувача про проблему.

Використані інструменти

Для тестування використовувалися такі інструменти:

– браузер (Google Chrome) – для тестування клієнтської частини;

– вбудовані інструменти розробника (DevTools) – для аналізу запитів;

– Postman – для тестування REST API;

– логування сервера – для відстеження помилок.

Таким чином, у процесі тестування було перевірено основні функціональні можливості системи онлайн-бронювання, а також її взаємодію між компонентами. Результати тестування показали, що система працює стабільно, відповідає встановленим вимогам та забезпечує коректну обробку даних.

Застосована методика тестування дозволила виявити та усунути можливі помилки, що підвищує надійність і якість розробленого програмного продукту.

3.2 Тестування функціональності системи

Тестування функціональності системи онлайн-бронювання спрямоване на перевірку коректності реалізації основних сценаріїв роботи користувача та адміністратора. У межах даного етапу було виконано перевірку всіх ключових функцій системи відповідно до сформульованих вимог [25].

Основні сценарії тестування

Для перевірки функціональності системи було визначено такі основні сценарії:

1. реєстрація користувача;
2. авторизація користувача;
3. перегляд списку номерів;
4. вибір дат бронювання;
5. розрахунок вартості;
6. створення бронювання;

7. перегляд власних бронювань;

8. скасування бронювання;

9. робота адміністративної панелі.

Кожен сценарій тестувався окремо з використанням реальних даних.

Тестування реєстрації та авторизації

У процесі тестування було перевірено:

– можливість створення нового користувача;

– обробку помилки при повторній реєстрації;

– правильність перевірки логіна та пароля;

– генерацію JWT-токена.

На рисунку 3.2 представлено результат успішної авторизації.

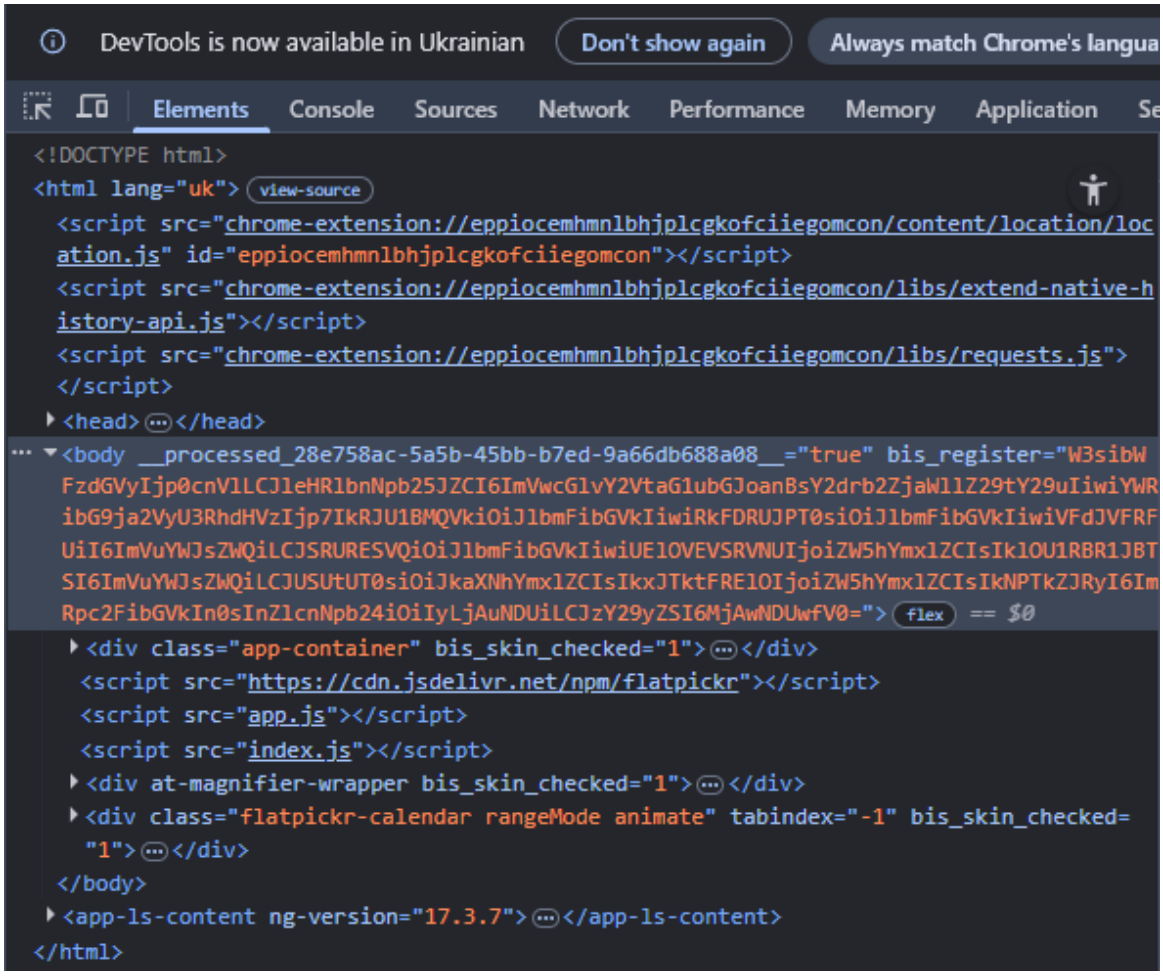

Рис. 3.2. Результат авторизації користувача

(розроблено автором)

Вхідні дані:

– username: user

– password: user

Результат:

– отримано access_token;

– користувач перенаправлений на головну сторінку

Це підтверджує коректність реалізації механізму автентифікації.

Тестування роботи з номерами

Було перевірено отримання списку номерів через API. Результати тестування показали:

– список номерів коректно відображається у випадаючому списку;

– дані відповідають записам у базі даних;

– при зміні номера оновлюється календар.

Тестування календаря та вибору дат

Особливу увагу приділено тестуванню календаря, оскільки він є ключовим елементом інтерфейсу. Перевірено:

– можливість вибору діапазону дат;

– блокування зайнятих періодів;

– відображення вартості для кожного дня;

– коректність підрахунку кількості ночей.

Під час тестування виконано перевірку різних сценаріїв бронювання, включаючи вибір дат у межах одного місяця та періодів, що охоплюють декілька місяців. Особливу увагу приділено коректності роботи алгоритму блокування вже заброньованих дат. Результати показали, що система не дозволяє користувачеві обрати зайнятий період, що запобігає виникненню конфліктних бронювань.

Також було перевірено механізм відображення вартості проживання у календарі. Для кожного дня система отримує актуальну ціну з сервера та відображає її безпосередньо в інтерфейсі користувача. Це дозволяє ще на етапі

вибору дат оцінити вартість проживання та обрати найбільш вигідний період бронювання.

На рисунку 3.3 показано приклад роботи календаря.

Рис. 3.3. Вибір дат у календарі

(розроблено автором)

Результати тестування показали, що календар працює стабільно та відповідає очікуваній поведінці.

Тестування розрахунку вартості

Було перевірено правильність роботи функції розрахунку вартості:

– врахування базової ціни номера;

– застосування правил ціноутворення;

– обчислення загальної вартості за період.

На рисунку 3.4 наведено приклад розрахунку вартості.

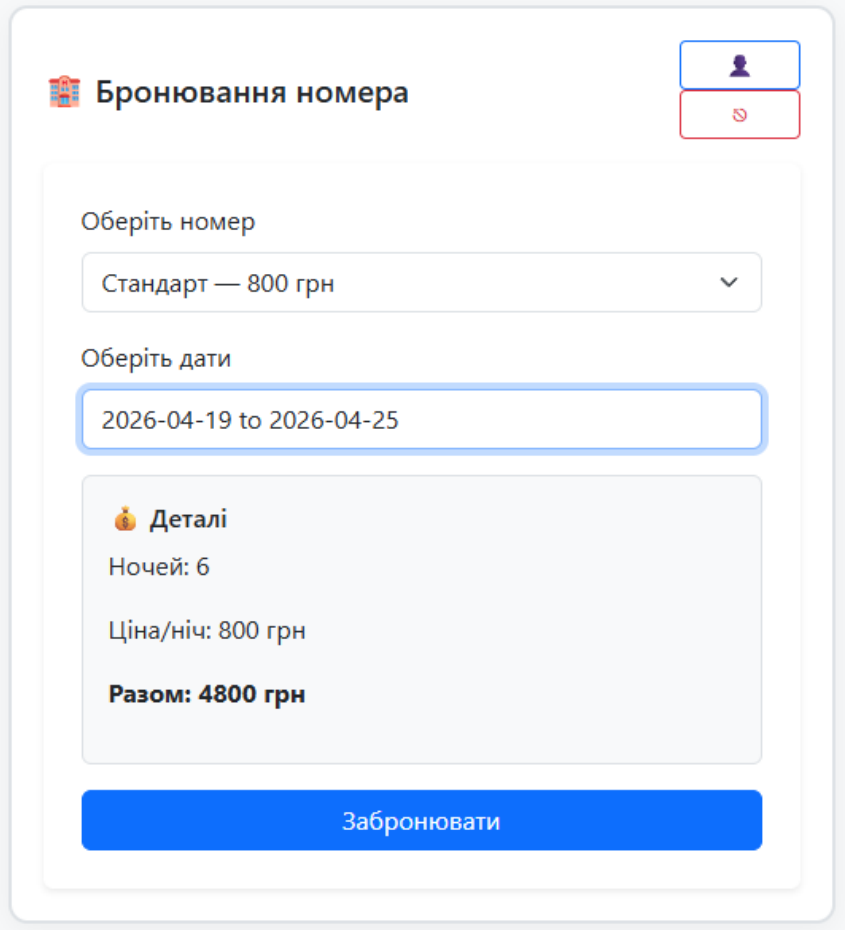

Рис. 3.4. Результат розрахунку вартості

(розроблено автором)

Дані:

– період: 6 ночей

– ціна за ніч: 800 грн

Результат:

– загальна вартість: 4800 грн

Отримані результати відповідають очікуваним значенням.

Тестування створення бронювання

Було перевірено:

– створення бронювання з коректними даними;

– обробку помилки при некоректних датах;

– заборону подвійного бронювання.

На рисунку 3.5 представлено результат успішного бронювання.

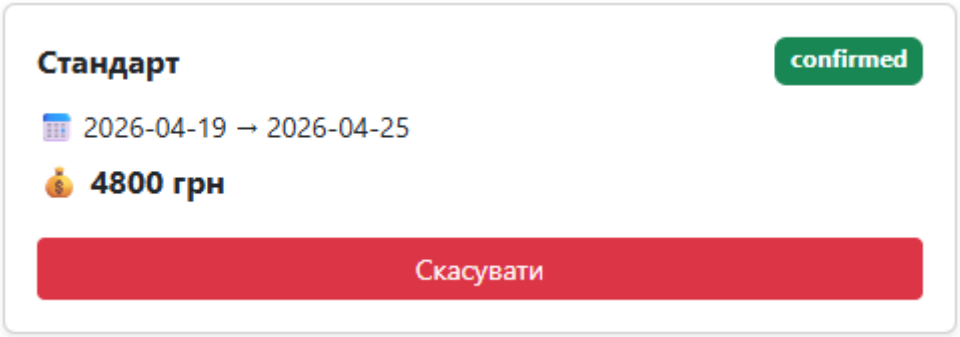

Рис. 3.5. Результат створення бронювання

(розроблено автором)

Результат:

– бронювання створено;

– статус: confirmed;

– дані збережені у базі

Це підтверджує правильність роботи бізнес-логіки.

Тестування профілю користувача

Було перевірено:

– відображення списку бронювань;

– коректність даних;

– відображення статусів;

– можливість скасування бронювання.

При скасуванні бронювання статус змінюється на cancelled, що відповідає реалізації soft delete.

Тестування адміністративної панелі

У межах тестування адміністративного функціоналу було перевірено:

– створення правил ціноутворення;

– редагування правил;

– видалення правил;

– фільтрацію за кімнатами;

– відображення даних у календарі.

На рисунку 3.6 показано інтерфейс адміністративної панелі.

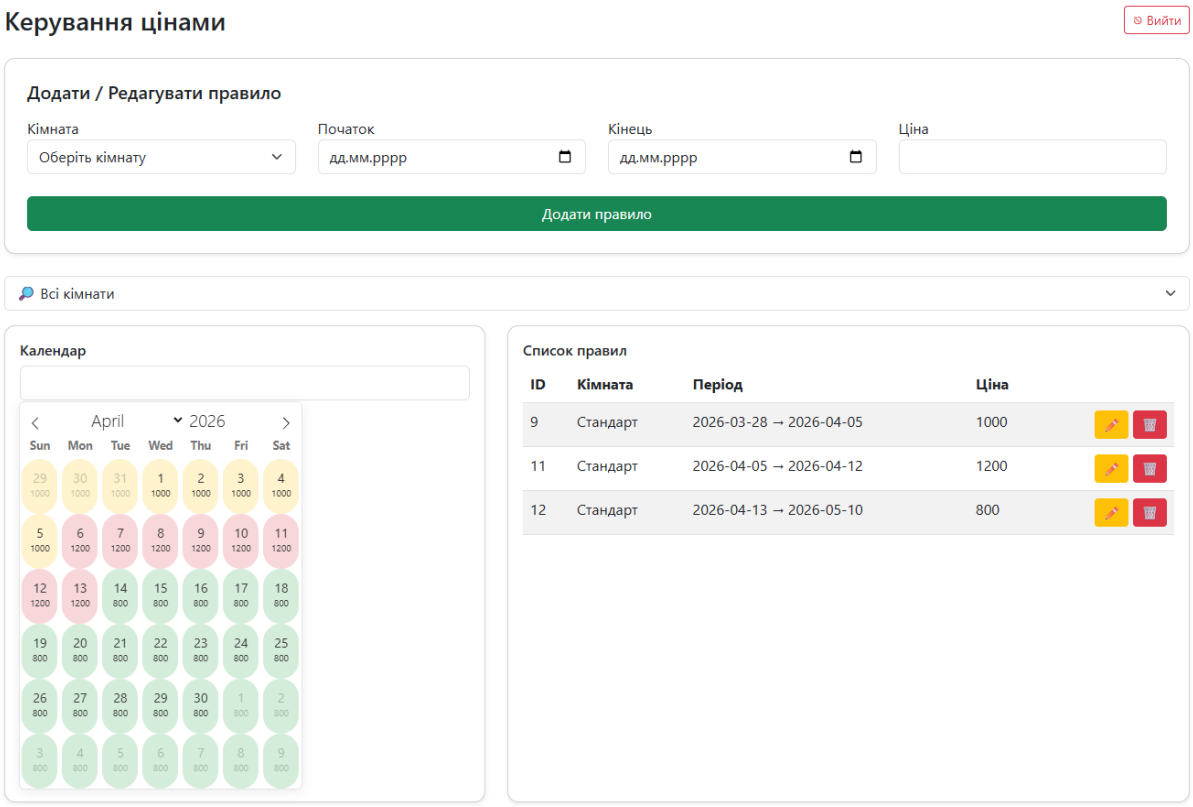

Рис. 3.6. Адміністративна панель управління цінами

(розроблено автором)

Результати тестування показали, що адміністративна частина системи працює коректно та забезпечує управління цінами.

Виявлені помилки та їх усунення

У процесі тестування було виявлено незначні помилки:

– некоректне відображення ціни при відсутності даних;

– необхідність перевірки введення дат;

– дрібні UI-недоліки.

Усі виявлені помилки були виправлені, що дозволило підвищити якість системи.

Таким чином, тестування функціональності системи підтвердило коректність реалізації всіх основних сценаріїв роботи. Система забезпечує

стабільну роботу, правильну обробку даних та зручну взаємодію користувача з інтерфейсом.

Отримані результати свідчать про відповідність реалізованого програмного продукту встановленим вимогам та його готовність до практичного використання.

3.3 Аналіз безпеки та надійності системи

Безпека та надійність є критично важливими характеристиками сучасних веб-застосунків, особливо у сфері онлайн-бронювання, де обробляються персональні дані користувачів та фінансова інформація. У даному підрозділі проведено аналіз реалізованих механізмів захисту та забезпечення стабільності роботи системи відповідно до сучасних підходів [26].

Аналіз безпеки системи

У розробленій системі реалізовано базові механізми захисту, які забезпечують безпечну взаємодію користувача з веб-застосунком.

Автентифікація та авторизація

Система використовує механізм автентифікації на основі JWT-токенів. Після успішного входу користувач отримує токен доступу, який передається у заголовку HTTP-запитів. Це дозволяє:

– уникнути повторної передачі логіна та пароля;

– забезпечити контроль доступу до захищених ресурсів;

– реалізувати рольову модель доступу.

Авторизація реалізована на основі ролей користувачів (user, admin), що дозволяє обмежити доступ до адміністративних функцій.

Захист паролів

Паролі користувачів не зберігаються у відкритому вигляді. Для їх захисту використовується алгоритм хешування bcrypt. Це забезпечує:

– стійкість до атак перебору;

– захист від компрометації бази даних;

– відповідність сучасним вимогам безпеки.

Контроль доступу

Доступ до API-методів обмежено за допомогою перевірки токена. У разі відсутності або некоректності токена система повертає відповідну помилку.

Для адміністративних функцій додатково перевіряється роль користувача, що виключає можливість виконання дій без необхідних прав.

Обробка помилок

Система коректно обробляє помилки та не розкриває внутрішню інформацію про структуру сервера або бази даних. Це зменшує ризик використання вразливостей зловмисниками.

Потенційні загрози та їх мінімізація

У процесі аналізу було визначено основні потенційні загрози:

– несанкціонований доступ до API;

– підбір паролів;

– перехоплення токенів;

– атаки типу SQL Injection;

– некоректне введення даних.

Для мінімізації цих загроз застосовано:

– використання ORM (SQLAlchemy), що знижує ризик SQL-ін’єкцій;

– валідацію вхідних даних через Pydantic;

– перевірку доступу через JWT;

– хешування паролів.

На рисунку 3.7 представлено загальну схему механізму безпеки.

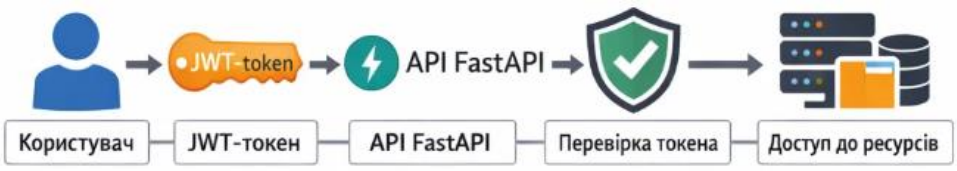

Рис. 3.7. Схема забезпечення безпеки системи

(розроблено автором)

Як показано на рисунку 3.7, усі запити проходять перевірку автентичності перед обробкою.

Аналіз надійності системи

Надійність системи визначається її здатністю стабільно працювати протягом тривалого часу без збоїв та помилок.

Обробка винятків

У серверній частині реалізовано обробку винятків, що дозволяє:

– уникнути аварійного завершення роботи;

– забезпечити коректну відповідь клієнту;

– виконати rollback транзакцій у разі помилки.

Робота з базою даних

Використання транзакцій у базі даних забезпечує цілісність даних. У разі помилки під час створення бронювання виконується скасування змін.

Перевірка бізнес-логіки

Реалізовано перевірки:

– коректності діапазону дат;

– доступності номера;

– унікальності користувача.

Це дозволяє уникнути логічних помилок та некоректних даних.

Стійкість до помилок користувача

Система коректно реагує на некоректні дії користувача:

– повідомляє про помилки;

– не допускає створення некоректних записів;

– забезпечує стабільну роботу інтерфейсу.

Обмеження та можливості покращення

Незважаючи на реалізовані механізми, система має потенціал для вдосконалення:

– впровадження HTTPS для захисту передавання даних;

– використання refresh-токенів;

– обмеження кількості спроб входу;

– додавання CAPTCHA;

– розширене логування та моніторинг;

– резервне копіювання бази даних.

Проведений аналіз показав, що система онлайн-бронювання має базовий рівень захисту та забезпечує достатню надійність для практичного використання. Реалізовані механізми автентифікації, авторизації та обробки даних дозволяють захистити систему від основних загроз.

Водночас існують можливості для подальшого підвищення рівня безпеки та надійності, що може бути реалізовано у процесі розвитку системи.

3.4 Оцінка ефективності впровадження

Оцінка ефективності впровадження програмного забезпечення є важливим етапом, що дозволяє визначити доцільність використання розробленої системи, її вплив на бізнес-процеси та рівень покращення обслуговування користувачів. У даному підрозділі проведено аналіз ефективності впровадження віджета онлайн-бронювання для сайту підприємства сфери гостинності [27].

Критерії оцінки ефективності

Для оцінювання ефективності було визначено такі основні критерії:

– функціональна ефективність;

– економічна ефективність;

– зручність використання;

– швидкодія системи;

– надійність та стабільність роботи.

Ці критерії дозволяють комплексно оцінити вплив впровадження системи на діяльність підприємства.

Функціональна ефективність

Розроблений віджет забезпечує автоматизацію основних процесів бронювання, що раніше могли виконуватись вручну або за допомогою сторонніх сервісів. Зокрема:

– користувач може самостійно обрати номер та дати;

– система автоматично перевіряє доступність;

– виконується розрахунок вартості з урахуванням правил ціноутворення;

– забезпечується можливість керування бронюваннями.

Це дозволяє зменшити навантаження на персонал та підвищити швидкість обслуговування клієнтів.

Економічна ефективність

Впровадження власного віджета онлайн-бронювання дозволяє:

– зменшити витрати на використання сторонніх платформ;

– уникнути комісій за бронювання;

– підвищити прибутковість за рахунок прямого продажу послуг;

– оптимізувати управління цінами.

На рисунку 3.8 наведено порівняння витрат до та після впровадження системи.

Рис. 3.8. Порівняння витрат на бронювання

(розроблено автором)

Таким чином, використання власного рішення є економічно доцільним у довгостроковій перспективі.

Зручність використання

Інтерфейс системи розроблено з урахуванням принципів UX/UI дизайну, що забезпечує:

– інтуїтивно зрозумілий процес бронювання;

– швидкий доступ до функцій;

– наочне відображення інформації;

– мінімальну кількість дій для користувача.

Застосування інтерактивного календаря з відображенням цін значно покращує користувацький досвід.

Швидкодія системи

Система демонструє високу швидкість обробки запитів завдяки:

– використанню FastAPI;

– оптимізації запитів до бази даних;

– асинхронній обробці запитів (на рівні фреймворку).

Час відповіді сервера у більшості випадків не перевищує кількох сотень мілісекунд, що є прийнятним для веб-застосунків.

Надійність та стабільність

У процесі тестування було встановлено, що система:

– стабільно працює при різних сценаріях використання;

– коректно обробляє помилки;

– забезпечує цілісність даних;

– не допускає конфліктів бронювання.

Це свідчить про високий рівень надійності реалізованого рішення.

Порівняльна оцінка

Для узагальнення результатів проведено порівняння основних показників до та після впровадження системи (таблиця 3.1).

Таблиця 3.1

Порівняння ефективності впровадження системи

Показник	До впровадження	Після впровадження
Спосіб бронювання	Ручний / сторонні сервіси	Власний веб-віджет
Час обробки заявки	10–30 хв	1–2 хв
Комісія	10–20%	0%
Навантаження на персонал	Високе	Середнє / низьке
Контроль цін	Обмежений	Повний

(розроблено автором)

Як видно з таблиці 3.1, впровадження системи значно покращує ключові показники ефективності.

Отже, проведена оцінка ефективності показала, що розроблений віджет онлайн-бронювання є доцільним та ефективним рішенням для підприємств сфери гостинності. Він дозволяє автоматизувати процеси, зменшити витрати, підвищити якість обслуговування та забезпечити конкурентні переваги.

Отримані результати підтверджують практичну цінність розробленої системи та можливість її використання у реальних умовах.

Висновки до розділу 3

У третьому розділі кваліфікаційної роботи було розглянуто питання тестування, безпеки, надійності та ефективності впровадження розробленої системи онлайн-бронювання.

У межах підрозділу 3.1 було визначено методику тестування програмного забезпечення, що включає функціональне, інтеграційне тестування, тестування інтерфейсу користувача та безпеки. Запропонований підхід дозволив комплексно оцінити якість реалізованої системи та забезпечити її відповідність встановленим вимогам.

У підрозділі 3.2 проведено тестування функціональності системи на основі основних сценаріїв використання. Було підтверджено коректність роботи таких ключових функцій, як реєстрація та авторизація користувачів, вибір дат бронювання, розрахунок вартості, створення та скасування бронювань, а також робота адміністративної панелі.

У підрозділі 3.3 здійснено аналіз безпеки та надійності системи. Встановлено, що реалізовані механізми автентифікації, авторизації, хешування паролів та обробки даних забезпечують базовий рівень захисту. Також підтверджено стабільність роботи системи та її здатність коректно обробляти помилки та виключні ситуації.

У підрозділі 3.4 проведено оцінку ефективності впровадження системи, яка показала, що використання власного віджета онлайн-бронювання дозволяє автоматизувати бізнес-процеси, зменшити витрати, підвищити швидкість обслуговування клієнтів та забезпечити більший контроль над ціноутворенням.

Таким чином, результати, отримані у третьому розділі, підтверджують, що розроблена система є працездатною, надійною та ефективною. Вона відповідає сучасним вимогам до веб-застосунків і може бути успішно використана у практичній діяльності підприємств сфери гостинності.

ВИСНОВКИ

У кваліфікаційній роботі було вирішено актуальне завдання розробки віджета онлайн-бронювання для сайту підприємства сфери гостинного бізнесу. У ході виконання роботи було проведено комплексне дослідження предметної області, проаналізовано існуючі рішення та реалізовано власну програмну систему.

У першому розділі було розглянуто особливості організації онлайн-бронювання, проведено аналіз сучасних веб-платформ, визначено їх переваги та недоліки. На основі проведеного аналізу сформовано функціональні та нефункціональні вимоги до системи, що дозволило визначити основні напрями подальшої розробки.

У другому розділі було здійснено проєктування та реалізацію системи онлайн-бронювання. Обґрунтовано вибір клієнт-серверної архітектури, розроблено структуру бази даних, реалізовано серверну частину з використанням сучасного фреймворку FastAPI та клієнтську частину у вигляді веб-інтерфейсу. Також було впроваджено механізми автентифікації та авторизації, що забезпечують захист даних та контроль доступу.

У третьому розділі проведено тестування розробленої системи, яке підтвердило коректність реалізації функціоналу та стабільність роботи. Виконано аналіз безпеки та надійності, що показав відповідність системи базовим вимогам захисту. Оцінка ефективності впровадження продемонструвала доцільність використання власного віджета онлайн-бронювання, зокрема з точки зору зниження витрат та підвищення якості обслуговування клієнтів.

У результаті виконання роботи було досягнуто поставленої мети — розроблено повноцінний віджет онлайн-бронювання, який забезпечує автоматизацію процесів взаємодії з клієнтами, підвищує ефективність діяльності підприємства та відповідає сучасним вимогам до веб-застосунків.

Отримані результати мають практичну цінність і можуть бути використані для впровадження у діяльність підприємств сфери гостинності, а також слугувати основою для подальшого розвитку та вдосконалення системи.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. FastAPI Documentation [Електронний ресурс]. – Режим доступу: https://fastapi.tiangolo.com/ (дата звернення: 15.04.2026).

2. Visual Paradigm Online. Hotel Booking System Use Case Diagram [Електронний ресурс]. – Режим доступу: https://online.visual-paradigm.com/ru/diagrams/templates/use-case-diagram/hotel-booking-system-use-case-diagram/ (дата звернення: 16.04.2026).

3. System Design Handbook. Design Hotel Booking System: A Step-by-Step Guide [Електронний ресурс]. – Режим доступу: https://www.systemdesignhandbook.com/guides/design-hotel-booking-system/ (дата звернення: 17.04.2026).

4. MySQL Documentation [Електронний ресурс]. – Режим доступу: https://dev.mysql.com/doc/ (дата звернення: 18.04.2026).

5. SQLAlchemy Documentation [Електронний ресурс]. – Режим доступу: https://docs.sqlalchemy.org/ (дата звернення: 19.04.2026).

6. Mozilla Developer Network. REST API Introduction [Електронний ресурс]. – Режим доступу: https://developer.mozilla.org/en-US/docs/Glossary/REST (дата звернення: 20.04.2026).

7. Altamira. Functional and Nonfunctional Requirements: The Detailed Guide [Електронний ресурс]. – Режим доступу: https://www.altamira.ai/blog/functional-and-nonfunctional-requirements-the-detailed-guide/ (дата звернення: 21.04.2026).

8. Flatpickr Documentation [Електронний ресурс]. – Режим доступу: https://flatpickr.js.org/ (дата звернення: 22.04.2026).

9. Creately. Online Hotel Reservation System [Електронний ресурс]. – Режим доступу: https://creately.com/diagram/example/hkxgmux72/online-hotel-reservation-system (дата звернення: 23.04.2026).

10. Bacancy. Bootstrap vs Angular [Електронний ресурс]. – Режим доступу: https://www.bacancytechnology.com/blog/bootstrap-vs-angular (дата звернення: 24.04.2026).

11. Bootstrap Documentation [Електронний ресурс]. – Режим доступу: https://getbootstrap.com/docs/5.3/getting-started/introduction/ (дата звернення: 25.04.2026).

12. JSON Web Tokens Introduction [Електронний ресурс]. – Режим доступу: https://jwt.io/introduction (дата звернення: 26.04.2026).

13. OWASP Foundation. JSON Web Token Cheat Sheet for Java [Електронний ресурс]. – Режим доступу: https://cheatsheetseries.owasp.org/cheatsheets/JSON_Web_Token_for_Java_Cheat_Sheet.html (дата звернення: 27.04.2026).

14. Pydantic Documentation [Електронний ресурс]. – Режим доступу: https://docs.pydantic.dev/latest/ (дата звернення: 28.04.2026).

15. Lucidchart. Database Design Guide [Електронний ресурс]. – Режим доступу: https://www.lucidchart.com/pages/database-diagram/database-design (дата звернення: 29.04.2026).

16. Richards M. Fundamentals of Software Architecture: An Engineering Approach. – Sebastopol : O'Reilly Media, 2020. – 432 p.

17. Kleppmann M. Designing Data-Intensive Applications. – Sebastopol : O'Reilly Media, 2021. – 614 p.

18. Pressman R. Software Engineering: A Practitioner's Approach. – New York : McGraw-Hill Education, 2020. – 880 p.

19. Sommerville I. Software Engineering. – Boston : Pearson, 2021. – 816 p.

20. Duckett J. HTML and CSS: Design and Build Websites. – Indianapolis : Wiley, 2021. – 490 p.

21. Duckett J. JavaScript and JQuery: Interactive Front-End Web Development. – Indianapolis : Wiley, 2021. – 640 p.

22. Silva M. Bootstrap 5 By Example. –Birmingham: Packt Publishing, 2022.–328 p.

23. Mozilla Developer Network. Fetch API [Електронний ресурс]. – Режим доступу: https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API (дата звернення: 05.05.2026).

24. Auth0. JWT Structure and Security [Електронний ресурс]. – Режим доступу: https://auth0.com/docs/secure/tokens/json-web-tokens (дата звернення: 12.05.2026).

25. IBM Cloud Education. What is Client-Server Architecture? [Електронний ресурс]. – Режим доступу: https://www.ibm.com/topics/client-server (дата звернення: 18.05.2026).

26. Python Software Foundation. Python Documentation [Електронний ресурс]. – Режим доступу: https://docs.python.org/3/ (дата звернення: 24.05.2026).

27. Swagger. OpenAPI Specification [Електронний ресурс]. – Режим доступу: https://swagger.io/specification/ (дата звернення: 29.05.2026).

ЗГОДА здобувача(чки) вищої освіти

Державного університету економіки і технологій про
перевірку кваліфікаційної роботи на прояви
академічного плагіату
та розміщення в Репозитарії Університету

Я, **Музика Євгеній Максимович**,

підтримую політику Державного університету економіки і технологій з академічної доброчесності і відкритого доступу.

Засвідчую, що кваліфікаційна бакалаврська робота **«Розробка віджета онлайн-бронюванная для сайту гостинного бізнесу»** виконана самостійно та не містить академічного плагіату. Я не надавав(ла) і не одержував(ла) недозволену допомогу під час підготовки цієї роботи. Робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

Із чинним Положенням про запобігання та виявлення академічного плагіату в роботах здобувачів вищої освіти Державного університету економіки і технологій ознайомлений(а). Чітко усвідомлюю, що в разі виявлення у кваліфікаційній роботі порушення норм академічної доброчесності робота не допускається до захисту або оцінюється незадовільно.

Також я поінформований(на), що відповідно до «Положення про Репозитарій (електронну базу даних) Державного університету економіки і технологій» зазначена робота буде розміщена в Електронному архіві Університету (Репозитарії ДУЕТ). З умовами такого розміщення ознайомлений(на).

15.06.2026 Музика Є. М.