Міністерство освіти і науки України
Луцький національний технічний університет
Факультет комп'ютерних та інформаційних технологій
Кафедра інженерії програмного забезпечення

КВАЛІФІКАЦІЙНА РОБОТА

ЗА СТУПЕНЕМ ВИЩОЇ ОСВІТИ «БАКАЛАВР»

РОЗРОБКА ВЕБДОДАТКУ ДЛЯ ОРГАНІЗАЦІЇ ОСОБИСТИХ ПОДОРОЖЕЙ З ІНТЕГРАЦІЄЮ ІНТЕРАКТИВНИХ КАРТ І СИСТЕМИ РЕКОМЕНДАЦІЙ З ВИКОРИСТАННЯМ REACT, NODE.JS ТА MYSQL

DEVELOPMENT OF A WEB APPLICATION FOR ORGANIZING PERSONAL TRIPS WITH INTERACTIVE MAP INTEGRATION AND A RECOMMENDATION SYSTEM USING REACT, NODE.JS AND MYSQL

спеціальність 121 «Інженерія програмного забезпечення»
освітня програма «Інженерія програмного забезпечення»

Виконав: здобувач вищої освіти
групи ІПЗс-21
Чернюк Богдан Іванович

Керівник:
Андрущак Ігор Євгенович

Кваліфікаційну роботу
допущено до захисту
«10» 06 2025 р.
Гарант освітньої програми:
к.т.н., доцент
Ліщина Наталія Миколаївна

Луцьк – 2025 року

ЛУЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

Факультет *комп'ютерних та інформаційних технологій*
Кафедра *інженерії програмного забезпечення*
Ступінь вищої освіти *бакалавр*
Галузь знань: *12 «Інформаційні технології»*
Спеціальність: *121 «Інженерія програмного забезпечення»*
Освітня програма: «*Інженерія програмного забезпечення»*

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____
«20» 12 2024 р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧУ ВИЩОЇ ОСВІТИ

Чернюку Богдану Івановичу
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи *«Розробка вебдодатку для організації особистих подорожей з інтеграцією інтерактивних карт і системи рекомендацій з використанням React, Node.js та MySQL»*
Керівник роботи: *д.т.н., професор Андрущак І. Є.*
затверджені наказом закладу вищої освіти від *«19» грудня 2024 р.* № *474/01-02*
2. Строк подання здобувачем вищої освіти кваліфікаційної роботи бакалавра *«10» червня 2025 р.*
3. Вихідні дані до роботи: *React, Node.js, MySQL, методичні вказівки до виконання кваліфікаційної роботи бакалавра*
4. Зміст розрахунково-пояснювальної записки (перелік питань, що потрібно розробити): *аналіз предметної області, постановка завдань, визначення вимог, проєктування архітектури, реалізація інтерфейсу, програмна логіка, розробка бази даних, тестування, підготовка документації.*
5. Перелік графічного матеріалу: *7 рисунків, 2 таблиці*

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис	
		завдання видав	завдання прийняв
Аналіз предметної області	*Андрущак І. Є.*		
Специфікація вимог до розробленої системи	*Андрущак І. Є.*		
Розробка об'єкта проектування	*Андрущак І. Є.*		
Нормоконтроль	*Вознюк А. В.*		
Гарант ОП	*Ліщина Н. М.*		
Показник запозичень тексту	4,46 %		
Академічна доброчесність	*Андрущак І. Є.*		

7. Дата видачі завдання *«20» грудня 2024 р.*

№	Назва етапів кваліфікаційної роботи бакалавра	Строк виконання етапів роботи	Примітка
1	Огляд літературних джерел по темі кваліфікаційної роботи бакалавра	до 04.02.2025 р.	вик.
2	Аналіз проблеми розробки та впровадження об'єкту проектування	до 01.03.2025 р.	вик.
3	Обґрунтування вибору шляхів, технологій і засобів вирішення поставленого завдання	до 15.03.2025 р.	вик.
4	Розробка функціонально-структурної схеми роботи об'єкта проектування та проектування бази даних	до 29.03.2025 р.	вик.
5	Практична реалізація об'єкта проектування та розробка бази даних	до 26.04.2025 р.	вик.
6	Тестування та налагодження об'єкта проектування	до 03.05.2025 р.	вик.
7	Здача чистового варіанту кваліфікаційної роботи бакалавра на кафедру	до 10.06.2025 р.	вик.

Здобувач вищої освіти _____ Богдан ЧЕРНЮК

Керівник кваліфікаційної роботи _____ Ігор АНДРУЩАК

АНОТАЦІЯ

Чернюк Б. І. Розробка вебдодатку для організації особистих подорожей з інтеграцією інтерактивних карт і системи рекомендацій з використанням React, Node.js та MySQL. Рукопис. Кваліфікаційна робота бакалавра ОП «Інженерія програмного забезпечення» спеціальності «Інженерія програмного забезпечення». Луцький національний технічний університет. Луцьк, 2025.

Кваліфікаційна робота бакалавра складається з вступу, трьох розділів, висновків та списку використаних джерел.

У першому розділі проаналізовано існуючі інформаційні системи для планування подорожей і визначено потребу у створенні інтегрованого вебдодатку, після чого сформульовано цілі та завдання роботи. У другому розділі описано вимоги до системи, обґрунтовано вибір технологій та змодельовано архітектуру додатку. У третьому розділі реалізовано функціонал вебдодатку, протестовано систему та підготовлено супровідну документацію. У висновках підсумовано результати проєкту та окреслено перспективи подальшого розвитку.

Ключові слова: вебдодаток, планування подорожей, інтерактивні карти, система рекомендацій, React, Node.js, MySQL, персоналізація, безпека даних.

ABSTRACT

Cherniuk B. Development of a Web Application for Organizing Personal Trips with Interactive Map Integration and a Recommendation System Using React, Node.js and MySQ. Manuscript. Bachelor's qualification work in the educational program "Software Engineering", specialty "Software Engineering". Lutsk National Technical University. Lutsk, 2025.

The bachelor's qualification work consists of an introduction, three chapters, conclusions and a list of references.

The first chapter analyzes existing travel planning systems and defines the need for a new integrated web application, followed by the formulation of goals and objectives. The second chapter outlines system requirements, justifies the technology stack, and models the application architecture. The third chapter presents the implementation of the application, system testing, and accompanying documentation. The conclusions summarize the project results and outline future development prospects.

Keywords: web application, travel planning, interactive maps, recommendation system, React, Node.js, MySQL, personalization, data security.

ЗМІСТ

ВСТУП

Актуальність теми. Сучасний розвиток інформаційних технологій докорінно змінив підходи до планування та організації подорожей. Туристична індустрія стала однією з найбільш динамічних сфер застосування вебтехнологій, забезпечуючи користувачам можливість зручного та швидкого доступу до інформації, планування маршрутів, бронювання готелів, вибору місць для відпочинку та розваг. Водночас, зростання кількості даних та їх різноманітність вимагають використання сучасних підходів до обробки інформації, зокрема інтеграції інтерактивних карт та систем рекомендацій.

На сьогодні існує багато вебдодатків, що допомагають користувачам планувати свої подорожі, проте значна їх частина є або занадто загальною, або не забезпечує достатнього рівня персоналізації. Крім того, часто відсутні інтегровані рішення, які одночасно дозволяють будувати маршрути, отримувати рекомендації на основі вподобань користувача та взаємодіяти з інтерактивними картами в реальному часі.

Світові тенденції розвитку програмних рішень для планування подорожей демонструють активне впровадження технологій машинного навчання для систем рекомендацій, картографічних сервісів для створення оптимальних маршрутів та інтеграції API для доступу до додаткових сервісів і платформ. Відповідно, актуальність теми кваліфікаційної роботи полягає у необхідності створення ефективного вебдодатка, що забезпечує інтерактивне планування подорожей із використанням картографічних сервісів та рекомендаційних алгоритмів. Такий додаток повинен забезпечити користувачу зручний інтерфейс, точні рекомендації та гнучкі можливості для налаштування подорожі відповідно до індивідуальних потреб.

Мета роботи – розробка вебдодатка для підготовки подорожей з інтеграцією інтерактивних карт та системи рекомендацій на основі сучасних технологій React, Node.js та MySQL.

Об’єкт кваліфікаційної роботи – процес організації та інформаційної підтримки особистих подорожей за допомогою вебзастосунків з використанням сучасних інформаційних технологій.

Предмет кваліфікаційної роботи – архітектура, функціональні компоненти та методи реалізації вебдодатку для створення, збереження і рекомендації маршрутів подорожей на основі інтерактивних карт, геолокаційних сервісів і реляційної бази даних.

У рамках реалізації поставленої мети необхідно:

1) проаналізувати існуючі програмні рішення у сфері планування подорожей;

2) сформулювати технічні та функціональні вимоги до системи;

3) обрати відповідний стек технологій для реалізації клієнтської та серверної частин;

4) реалізувати функціонал побудови маршрутів з інтерацією карти та геокодуванням точок, розробити RESTful API для обміну даними між клієнтом і сервером;

5) провести тестування основних компонентів системи та підготувати супровідну документацію для користувача;

6) спроєктувати структуру реляційної бази даних для збереження маршрутів, точок і користувачів;

7) забезпечити базовий рівень захисту через механізми хешування паролів і авторизацію токенами;

8) розробити документацію для програмного продукту.

Галузь застосування – туристичні агентства, приватні мандрівники, платформи для спільного планування подорожей та додатки для індивідуальних туристичних маршрутів.

Взаємозв’язок роботи з іншими дослідженнями полягає у поєднанні передових технологій веброзробки, картографічних інтерфейсів та систем рекомендацій. Робота базується на дослідженнях у галузі геоінформаційних систем, алгоритмів рекомендацій та архітектури вебдодатків.

Розробка такого вебдодатка дозволить підвищити ефективність процесу планування подорожей, забезпечити кращу якість обслуговування користувачів і створити інструмент, який об’єднує найкращі практики сучасних технологій для туризму.

РОЗДІЛ 1
АНАЛІЗ СИСТЕМ ДЛЯ ПІДГОТОВКИ ПОДОРОЖЕЙ І ПОСТАНОВКА ЗАВДАНЬ НА КВАЛІФІКАЦІЙНУ РОБОТУ

1.1 Аналіз сучасного стану проблеми

Ринок інформаційних систем для планування подорожей демонструє значний розвиток завдяки впровадженню новітніх вебтехнологій і картографічних сервісів. У наукових дослідженнях та практичних розробках підкреслюється важливість інтеграції рекомендаційних систем, які використовують алгоритми машинного навчання для персоналізації результатів, а також інтерактивних карт для оптимізації маршрутів. Проведений огляд літературних джерел показує, що значна частина досліджень зосереджена на вдосконаленні інтерфейсів користувача, підвищенні продуктивності системи та забезпеченні захисту даних.

Б. Л. Шевчук, І. К. Нестерчук [1] досліджують можливості інтерактивних карт, створених на основі геоінформаційних систем, для потреб туризму в Україні. У статті охарактеризовано сильні та слабкі сторони існуючих картографічних проєктів, а також надано рекомендації щодо їх удосконалення для підвищення ефективності використання в туризмі.

В. В. Литвин, О. М. Наум, В. А. Висоцька, М. В. Дверій [2] аналізують архітектуру онлайн-системи для планування подорожей, яка використовує інтелектуальні алгоритми для пошуку, інтеграції та формування контенту. Описано методи агрегації даних для оптимізації вибору маршрутів, готелів і транспортних послуг з урахуванням потреб користувача.

М. В. Грабар [3] розглядає сучасний стан та перспективи інформаційних систем і технологій у туристичній галузі. Визначено основні технологічні тренди, такі як великі дані, віртуальна реальність, блокчейн та мобільні технології, що впливають на розвиток туризму і процеси планування подорожей.

Також одним із основних джерел інформації є наукові статті, присвячені аналізу алгоритмів рекомендацій, таких як Collaborative Filtering та Content-Based Filtering, які дозволяють системам адаптувати свої рекомендації до вподобань користувача. Також значна увага приділяється картографічним сервісам, зокрема Google Maps API, OpenStreetMap та Mapbox, що забезпечують інструменти для інтеграції карт у вебдодатки. Патентний пошук підтверджує активний розвиток технологій, пов’язаних із системами навігації, динамічними картами та оптимізацією маршрутів у реальному часі.

На ринку вже існує низка популярних рішень, серед яких можна виділити Google Maps (рис. 1.1), TripAdvisor, Booking.com та Rome2Rio. Кожен із цих сервісів має свої особливості. Google Maps надає можливість будувати маршрути, переглядати карти та отримувати базову інформацію про місця. TripAdvisor фокусується на оглядах і рекомендаціях, але не забезпечує інтегрованої системи планування маршрутів. Booking.com і Airbnb надають функції бронювання, однак їх можливості обмежені у контексті побудови маршрутів і взаємодії з картами. Водночас Rome2Rio спеціалізується на пошуку оптимальних маршрутів для різних видів транспорту, але не забезпечує персоналізованих рекомендацій і детального планування подорожі.

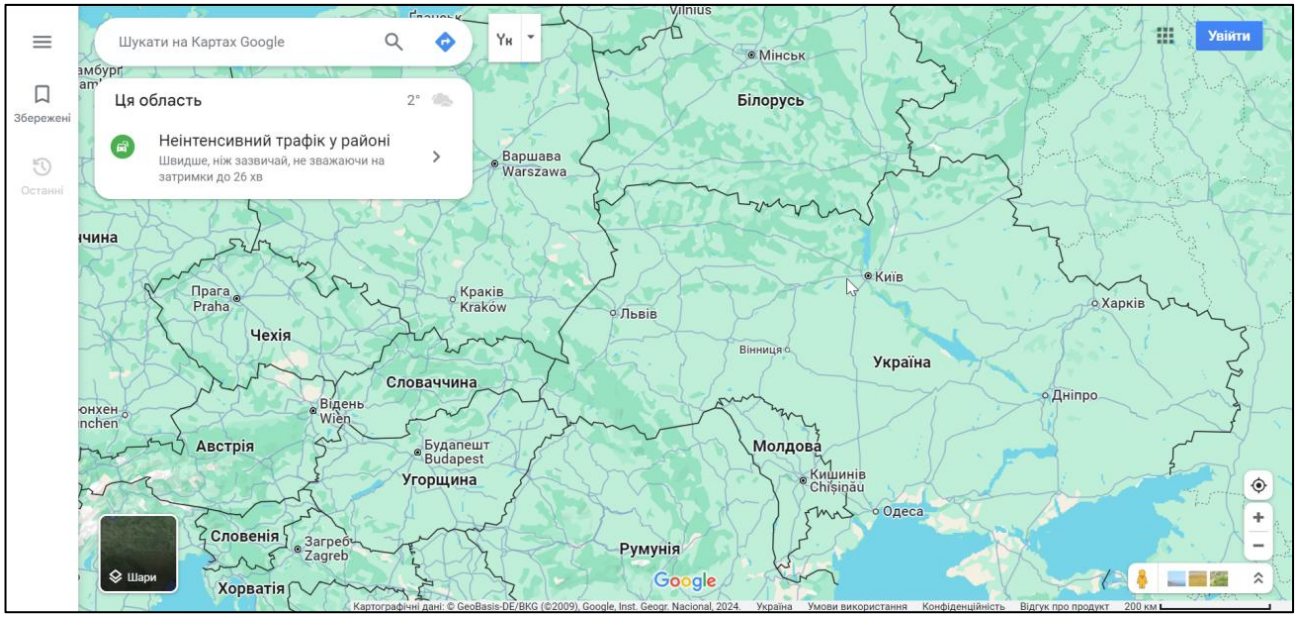

Рисунок 1.1 – Інтерфейс Google Maps [4]

Серед переваг наявних рішень можна відзначити високу популярність, надійність, інтеграцію з численними сторонніми сервісами та простоту у використанні. Проте існує низка недоліків, зокрема відсутність комплексного підходу, фрагментованість функціоналу, обмежена персоналізація та недостатня інтеграція рекомендаційних систем із картографічними сервісами. Нижче наведена таблиця з порівнянням функціоналу описаних аналогів (табл. 1.1).

Таблиця 1.1 – Порівняння існуючих аналогів

Система	Google Maps [4]	TripAdvisor [5]	Booking.com [6]	Airbnb [7]	Rome2Rio [8]
Функція побудови маршрутів	Є	Немає	Немає	Немає	Є
Інтерактивні карти	Є	Немає	Немає	Немає	Є
Система рекомендацій	Немає	Є	Є	Є	Немає
Персоналізація	Немає	Є	Є	Є	Немає
Інтеграція бронювання	Немає	Немає	Є	Є	Немає
Захист даних	Є	Є	Є	Є	Є
Доступність на мобільних пристроях	Є	Є	Є	Є	Є

Доцільність створення нової системи обґрунтовується необхідністю об’єднання в єдиному вебдодатку основних функціональних можливостей: інтерактивних карт, персоналізованих рекомендацій та гнучкого інтерфейсу для користувача. Сучасні технології, такі як React**,** Node.js і MySQL, забезпечують достатню гнучкість і продуктивність для створення ефективного рішення. Розробка такого додатка дозволить усунути існуючі обмеження, забезпечити більш інтуїтивно зрозумілий досвід користування, а також оптимізувати процеси планування подорожей завдяки сучасним алгоритмам і технологіям.

Отже, на основі проведеного аналізу можна зробити висновок, що створення вебдодатка для підготовки подорожей з інтеграцією інтерактивних карт і системи рекомендацій є актуальним і доцільним кроком для подальшого розвитку в цій сфері.

1.2 Постановка завдання на кваліфікаційну роботу бакалавра

Метою кваліфікаційної роботи є розробка вебдодатка для підготовки подорожей з інтеграцією інтерактивних карт і системи рекомендацій, що забезпечить користувачам зручний інструмент для планування маршрутів, отримання персоналізованих рекомендацій і взаємодії з інтерактивними картами.

Для досягнення цієї мети поставлено такі завдання:

1) провести аналіз сучасних інформаційних систем, що використовуються для планування подорожей, визначити їх функціональні можливості, недоліки та тенденції розвитку;

2) сформулювати технічне завдання на розробку вебдодатку, визначити функціональні та нефункціональні вимоги до системи, а також обрати відповідні технології та архітектурні рішення;

3) обґрунтувати вибір методів, бібліотек, API та алгоритмів, які будуть використані для реалізації основних функцій системи, зокрема маршрутизації, рекомендацій та геокодування;

4) реалізувати вебдодаток з використанням технологій React.js для клієнтської частини, Node.js з Express для серверної частини та MySQL для бази даних, забезпечивши можливість створення, збереження та перегляду маршрутів;

5) провести тестування та налагодження вебдодатку для перевірки його працездатності, виявлення та усунення помилок, визначення мінімальних вимог для запуску системи;

6) спроєктувати та реалізувати логічну структуру бази даних у MySQL з урахуванням нормалізації та забезпеченням зв’язків між сутностями;

7) забезпечити базовий рівень захисту інформаційно-комп’ютерної системи, реалізувавши хешування паролів, автентифікацію за допомогою JWT-токенів та обмеження доступу до особистих маршрутів;

8) розробити супровідну документацію до програмного продукту, що включає інструкцію з розгортання, мінімальні системні вимоги, довідкові матеріали для користувача та рекомендації щодо подальшого розвитку системи.

Виконання поставлених завдань дозволить створити ефективний, безпечний і інтуїтивно зрозумілий вебдодаток, що забезпечить користувачам зручний інструмент для планування подорожей, взаємодії з картами та отримання персоналізованих рекомендацій.

Висновки до розділу 1

У першому розділі було проведено детальний аналіз сучасного стану проблеми, пов’язаної з плануванням подорожей за допомогою вебдодатків. Виявлено основні виклики, зокрема надлишок інформації, відсутність персоналізованих рекомендацій, обмеженість інтеграції інтерактивних карт і недостатній рівень захисту даних користувачів. Проведений огляд існуючих аналогів, таких як Google Maps**,** TripAdvisor**,** Booking.com**,** Airbnb та Rome2Rio, показав, що жоден із них не забезпечує комплексного підходу до вирішення всіх завдань планування подорожей.

Аналіз світових тенденцій підтвердив актуальність використання інтерактивних картографічних сервісів, алгоритмів машинного навчання для персоналізованих рекомендацій та сучасних методів захисту даних. Визначено, що більшість існуючих систем зосереджені на вузьких функціональних завданнях і не здатні забезпечити інтегроване рішення, яке б об’єднувало планування маршруту, рекомендації та взаємодію з інтерактивними картами в одному додатку.

На основі проведеного дослідження було сформульовано завдання на кваліфікаційну роботу бакалавра, що включають аналіз існуючих систем, розробку архітектури вебдодатка, інтеграцію картографічних сервісів, створення системи рекомендацій, забезпечення безпеки даних та розробку документації користувача.

Отже, можна зробити висновок, що створення вебдодатка для підготовки подорожей із використанням інтерактивних карт та системи рекомендацій є актуальним та доцільним завданням, що відповідає сучасним вимогам ринку і користувачів.

РОЗДІЛ 2
СПЕЦИФІКАЦІЯ ВИМОГ ДО РОЗРОБЛЮВАНОЇ СИСТЕМИ

2.1 Аналіз, визначення вимог до розроблюваного програмного забезпечення та проектування програмного забезпечення

У рамках розробки вебдодатку для підготовки подорожей з інтеграцією інтерактивних карт і системи рекомендацій було обрано ітераційну інкрементну модель розробки програмного забезпечення. Такий підхід дозволяє поступово реалізовувати та тестувати функціональність системи, своєчасно враховувати зворотний зв'язок користувачів, підвищувати якість і стабільність продукту. Для формального опису моделі архітектури системи було використано UML-нотацію, зокрема діаграми випадків використання, послідовностей і компонентів.

Для збору вимог було використано комплексний підхід, що включав інтерв'ю з цільовими користувачами, серед яких – індивідуальні мандрівники, туристичні менеджери та тревел-блогери. Також проведено детальний аналіз існуючих систем-аналогів, таких як Google Maps, Rome2Rio, Booking та TripAdvisor, з метою виявлення функціональних обмежень і переваг. Для глибшого розуміння ринку було застосовано SWOT-аналіз функціоналу конкурентів. Додатково здійснювалося спостереження за сценаріями використання туристичних планувальників у реальних умовах. На основі зібраної інформації було визначено основні функціональні та нефункціональні вимоги до системи.

Метою є надати користувачам зручний інструмент для планування подорожей, який забезпечить інтерактивну роботу з картами, отримання персоналізованих рекомендацій і управління маршрутами.

Основні задачі:

1) надання користувачам можливості створення особистих маршрутів подорожей шляхом додавання географічних точок на інтерактивній карті з фіксацією координат та їхньою географічною назвою;

2) автоматичне визначення орієнтовного часу проходження маршруту з урахуванням типу пересування (пішки або автомобілем) за допомогою інтеграції з зовнішнім API Geoapify;

3) збереження маршрутів у базу даних з можливістю подальшого перегляду, редагування, завершення або скасування подорожі;

4) реєстрація та автентифікація користувачів із використанням безпечного зберігання паролів (хешування) та механізму авторизації через JWT-токени;

5) реалізація персоналізованої системи рекомендацій, що пропонує маршрути інших користувачів на основі збігів першої точки маршруту (місто або область);

6) клонування рекомендованих маршрутів до власного кабінету користувача для подальшого використання;

7) забезпечення зручного перегляду збережених маршрутів у візуальному вигляді зі списком точок, координатами, адресами, статусом та тривалістю;

8) забезпечення базового рівня захисту інформації, включаючи обмеження доступу до маршрутів тільки для авторизованих користувачів;

9) можливість адаптивного використання системи як на десктопних пристроях, так і на мобільних, завдяки адаптивному інтерфейсу.

Основні функціональні вимоги описано в таблиці 2.1.

Таблиця 2.1 – Функціональні характеристики системи

Назва функціоналу	Опис функціональної можливості	Користувач	Технології реалізації
Реєстрація користувача	Створення облікового запису з введенням email, логіна та пароля	Гість	React, Node.js, bcryptjs, MySQL
Авторизація	Вхід до системи з перевіркою облікових даних, отримання JWT-токена	Гість	React, Node.js, JWT
Побудова маршруту	Додавання точок на карті шляхом кліків, зберігання координат у стані	Користувач	React, Leaflet, useState
Геокодування точок	Визначення назви населеного пункту за координатами через API	Користувач	Geoapify API, fetch
Назва функціоналу	Опис функціональної можливості	Користувач	Технології реалізації

Продовження таблиці 2.1

Назва функціоналу	Опис функціональної можливості	Користувач	Технології реалізації
Розрахунок тривалості маршруту	Отримання орієнтовного часу між точками маршруту	Користувач	Geoapify Routing API
Збереження маршруту	Відправлення точок маршруту на сервер, запис у базу даних	Користувач	React, Node.js, MySQL
Перегляд збережених маршрутів	Виведення маршрутів користувача зі списком точок, статусом і тривалістю	Користувач	React, Node.js, MySQL
Завершення/скасування маршруту	Можливість змінити статус маршруту або видалити його	Користувач	Node.js, MySQL
Отримання рекомендацій	Пошук маршрутів інших користувачів з подібною першою точкою (місто/область)	Користувач	Node.js, MySQL, логіка на основі point_name
Клонування маршруту	Створення копії маршруту іншого користувача в особистому кабінеті	Користувач	Node.js, MySQL
Видалення облікового запису	Повне видалення акаунта користувача з усіма його маршрутами	Користувач	Node.js, MySQL, JWT
Захист доступу	Захист приватних API через JWT-мідлвар і перевірку прав доступу	Користувач	Express Middleware, JWT

Буде розроблено інтуїтивно зрозумілий інтерфейс із трьома основними зонами:

1) панель керування маршрутом;
2) інтерактивна карта;
3) блок рекомендацій.

На рисунку 2.1 зображено діаграму варіантів використання. Вона відображає основні дії користувача: створення маршруту, взаємодія з картою, реєстрація, логін. На діаграмі зображено двох акторів – «Користувача» та «Гостя». «Гість» має доступ лише до функцій реєстрації та авторизації, тоді як «Користувач» може створювати, зберігати, редагувати, видаляти та переглядати маршрути, отримувати рекомендації маршрутів, переглядати виділені елементи на карті та клонувати маршрути.

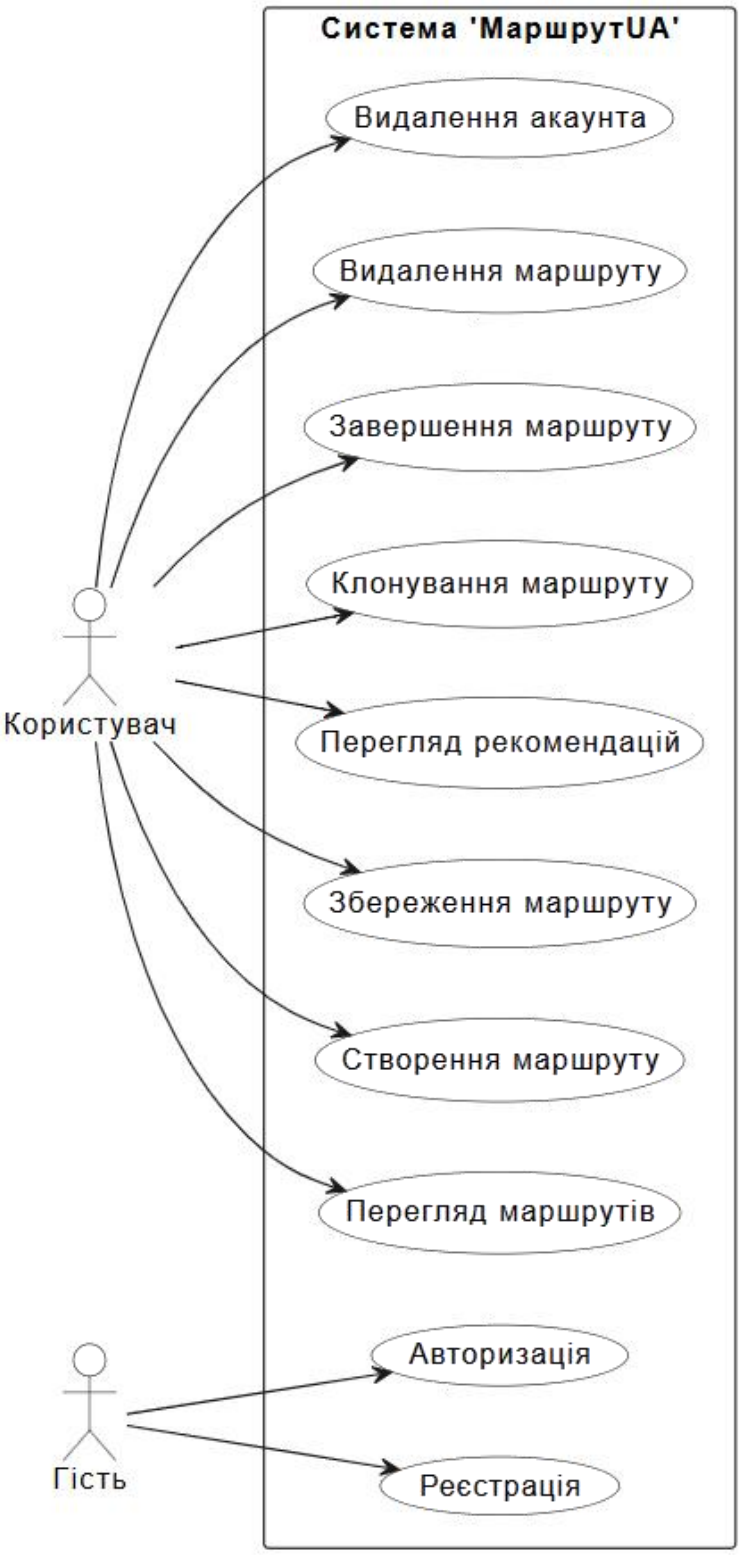

Рисунок 2.1 – Діаграма варіантів використання

На рисунку 2.2 зображено діаграму послідовностей для сценарію «Користувач створює маршрут і зберігає його».

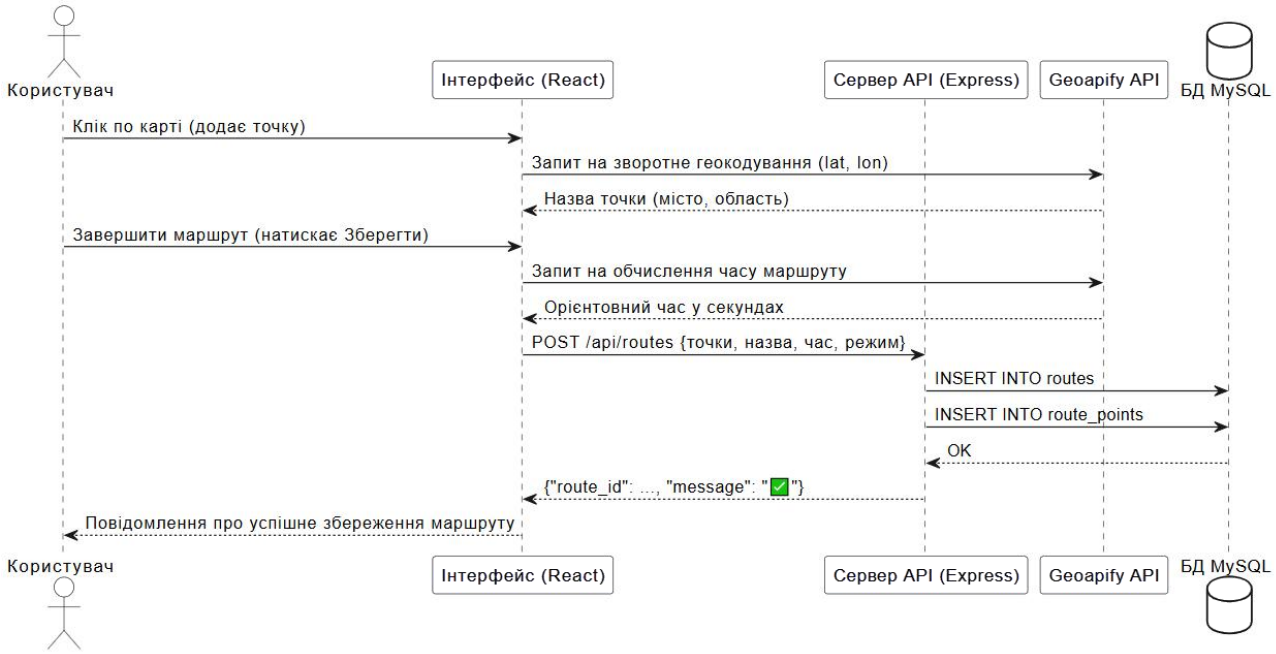

Рисунок 2.2 – Діаграма послідовності

Діаграма на рисунку 2.3 відображає логічну структуру системи: клієнт (frontend), сервер (backend) і база даних.

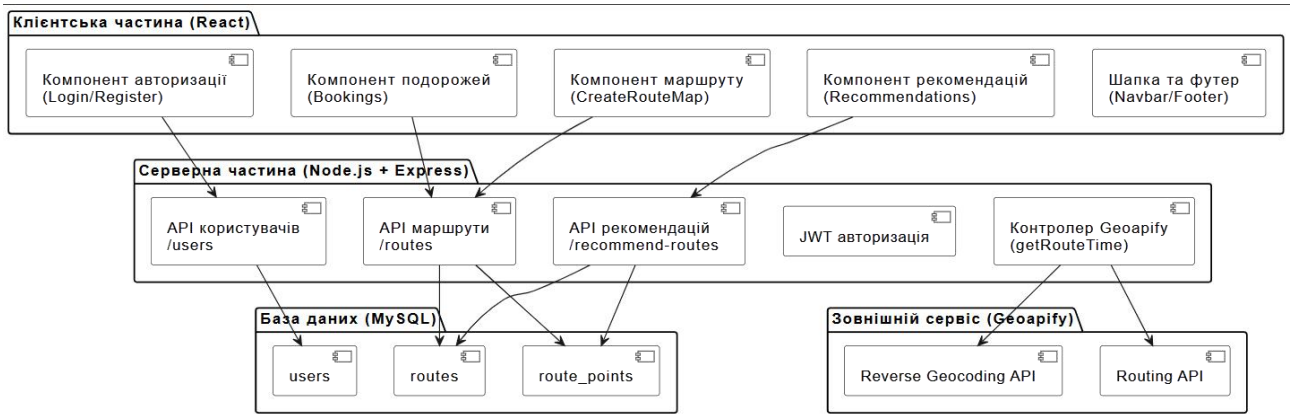

Рисунок 2.3 – Діаграма компонентів системи

Отримана модель підтвердила відповідність архітектури сучасним стандартам: реалізовано трирівневу структуру (клієнт – сервер – БД), забезпечено розділення відповідальності між компонентами, легко масштабувати систему або додавати новий функціонал.

2.2 Вибір засобів, методів і алгоритмів вирішення поставленого завдання

Для ефективної реалізації вебдодатку для підготовки подорожей з інтеграцією інтерактивних карт і системи рекомендацій було проаналізовано низку сучасних інструментів, мов програмування, бібліотек та архітектурних підходів, які відповідають вимогам продуктивності, масштабованості, безпеки та зручності для користувача.

На етапі аналізу можливих технологій було розглянуто такі платформи та фреймворки:

1) фронтенд: React.js, Angular, Vue.js;

1) бекенд: Node.js, Django, Laravel;

2) база даних: MySQL, PostgreSQL, MongoDB;

3) картографічні сервіси: Google Maps API, OpenStreetMap (через Leaflet або Mapbox);

4) системи рекомендацій: бібліотеки машинного навчання (TensorFlow.js, Surprise, Scikit-learn), або кастомна реалізація на основі фільтрації.

Після порівняльного аналізу за критеріями швидкодії, підтримки спільноти, простоти інтеграції, документації та гнучкості було обрано наступні засоби для реалізації проєкту:

React – це одна з найпопулярніших бібліотек JavaScript для створення інтерфейсів користувача. Вона забезпечує компонентно-орієнтовану архітектуру, завдяки якій інтерфейс поділяється на логічні частини (компоненти), які легко повторно використовувати, тестувати та підтримувати [9]. Завдяки віртуальному DOM React мінімізує кількість змін у реальному DOM, що значно пришвидшує рендеринг і оновлення даних у браузері. Бібліотека має потужну екосистему: React Router дозволяє реалізовувати SPA-навігацію без перезавантаження сторінки, Context API забезпечує зручну передачу стану між компонентами [10]. У вебдодатку

реалізовано адаптивний інтерфейс, зручний як для мобільних, так і для десктопних пристроїв, що відповідає сучасним вимогам до UI/UX-дизайну.

Node.js – це подієво-орієнтоване середовище виконання JavaScript на сервері, яке дозволяє створювати високопродуктивні вебсервіси з асинхронною обробкою запитів [11]. Воно ідеально підходить для побудови RESTful API та обробки великої кількості одночасних з'єднань без потреби у багатопоточності. У рамках даного проєкту Express.js використовується як вебфреймворк для побудови маршрутизованого API, обробки запитів від клієнта, взаємодії з базою даних, керування автентифікацією та надання відповіді. Express є мінімалістичним, однак гнучким і розширюваним, що дозволяє впровадити складну бізнес-логіку без надмірних складностей у конфігурації [12].

MySQL – це реляційна система керування базами даних (СУБД), яка відзначається високою продуктивністю, надійністю та широкими можливостями з підтримки транзакцій, індексів, зовнішніх ключів і складних SQL-запитів [13]. Вона ідеально підходить для вебдодатків, які потребують структурованого зберігання взаємопов'язаних даних. У розробленій системі MySQL використовується для збереження інформації про користувачів, маршрути, точки подорожей та інші сутності, пов'язані між собою зв'язками «один до багатьох» [14]. Такий підхід забезпечує цілісність і логічну узгодженість даних, а також дозволяє швидко здійснювати вибірки за допомогою JOIN-запитів.

Основним компонентом системи стала інтеграція з Geoapify API – ефективним хмарним сервісом для роботи з просторовими даними [15]. У проєкті використовуються два основні інструменти Geoapify: зворотне геокодування (reverse geocoding) та маршрутизація (routing). Зворотне геокодування дозволяє автоматично визначати назву населеного пункту та регіону за координатами точки, яку користувач додає на мапі [16]. Це забезпечує персоналізацію імен точок маршруту, що особливо важливо для логіки рекомендацій. Алгоритм маршрутизації використовується для розрахунку орієнтовного часу подорожі між точками, з урахуванням типу

пересування (пішки чи на авто). Завдяки Geoapify додаток здатен надавати користувачу максимально точну і актуальну інформацію без потреби у складній власній реалізації картографічних сервісів.

Система рекомендацій реалізована як алгоритм порівняння назв перших точок маршрутів користувача з аналогічними точками маршрутів інших користувачів у базі даних, що дозволяє формувати персоналізовані пропозиції подорожей за принципом географічної близькості – спочатку на рівні міста, а потім області. Нижче наведено блок-схему рекомендаційного модуля (рис. 2.4).

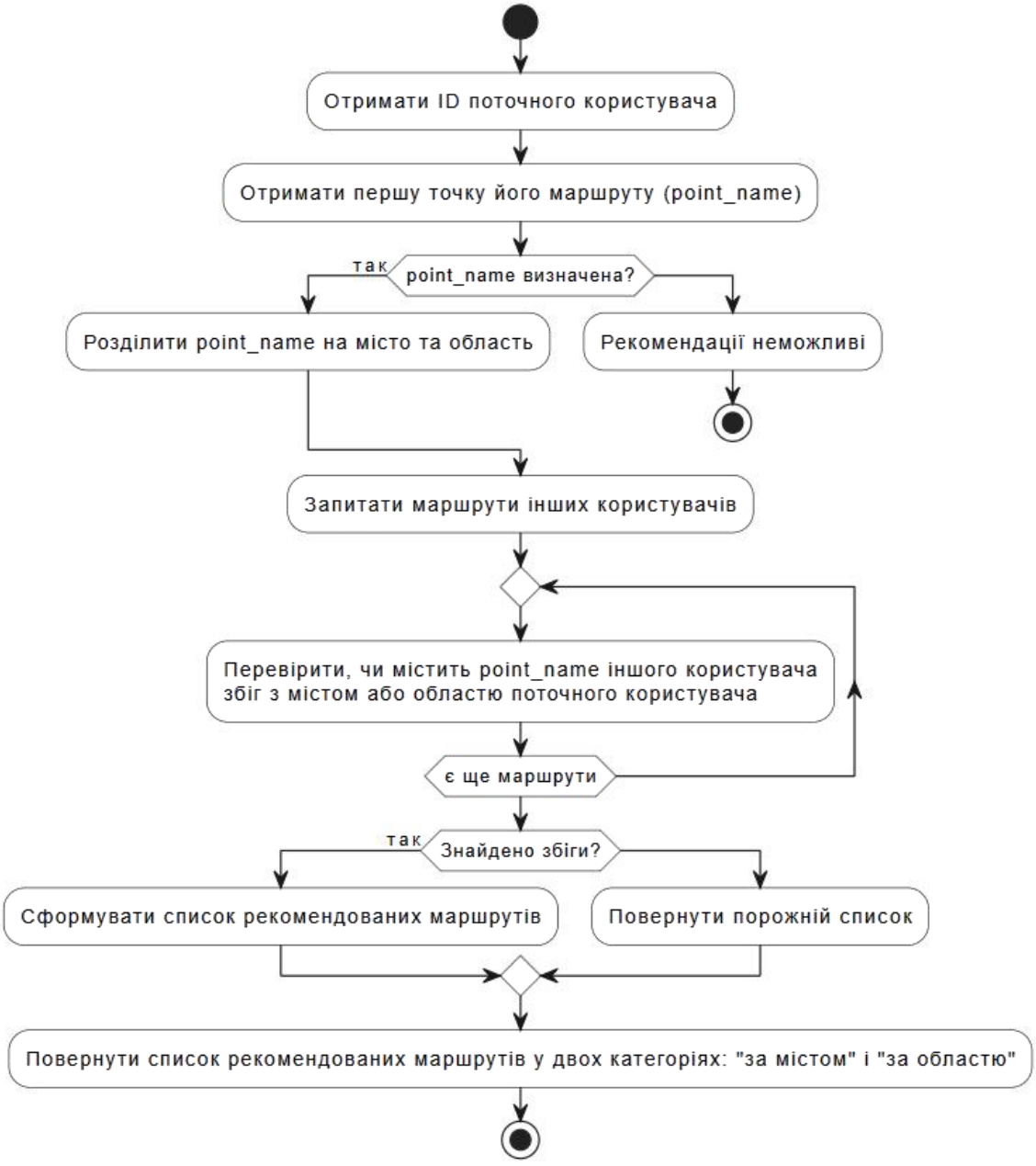

Рисунок 2.4 – Блок-схема роботи рекомендаційного модуля

Діаграма розгортання зображена на рисунку 2.5 – система розгорнута на вебсервері, база даних – на хмарному MySQL.

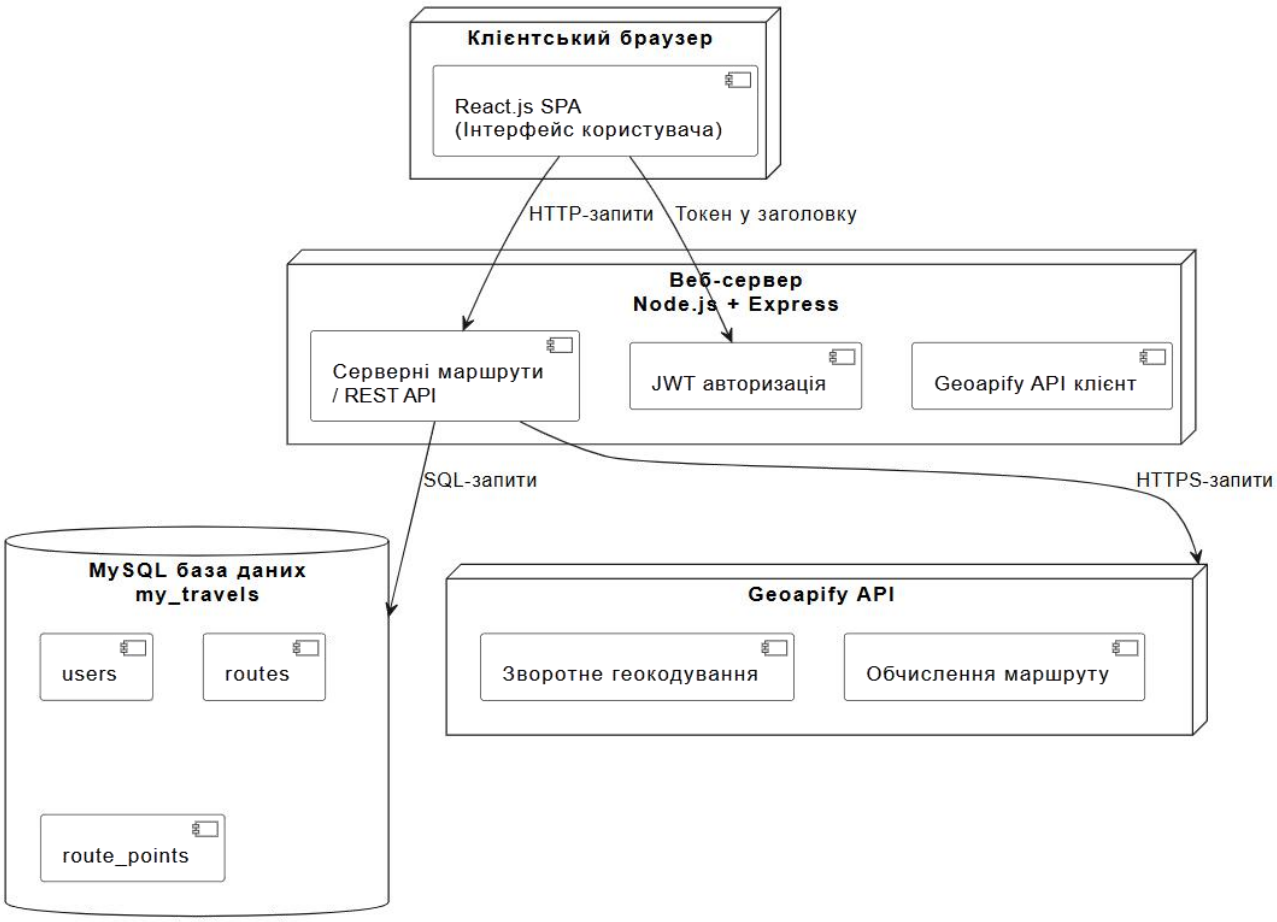

Рисунок 2.5 – Діаграма розгортання

Обрані засоби розробки дозволяють створити масштабований, зручний і безпечний вебдодаток, що відповідає сучасним вимогам до інтерактивних туристичних платформ. Використання перевірених інструментів – таких як React, Node.js, MySQL і Geoapify API – гарантує гнучкість та надійність системи. Алгоритмічне забезпечення рекомендацій дозволяє формувати персоналізований досвід для кожного користувача, що є важливим чинником успіху застосунку.

Висновки до розділу 2

У другому розділі було визначено технічні, функціональні та нефункціональні вимоги до вебдодатку для підготовки подорожей, обґрунтовано вибір моделі розробки програмного забезпечення, а також здійснено проектування системи з використанням UML-діаграм. На основі аналізу проблемної області та вимог користувачів було сформовано чітке технічне завдання, що охоплює основні сценарії взаємодії з системою, функціональні можливості та вимоги до інтерфейсу.

Проведено огляд сучасних інструментів і технологій, які потенційно можуть бути використані для реалізації поставлених задач, а також аргументовано обрано стек технологій: React.js для клієнтської частини, Node.js з Express для серверної логіки, MySQL як систему управління базами даних і Geoapify API для інтерактивної роботи з геоданими.

Моделювання системи за допомогою діаграм UML (випадків використання, класів, компонентів, розгортання) дало змогу візуалізувати структуру, потоки даних та взаємодію між елементами. Отримані результати засвідчують, що архітектура додатка є логічно обґрунтованою, масштабованою і відповідає сучасним вимогам до інформаційних систем у сфері туризму. На основі сформованих вимог і обраних інструментів буде створено надійну основу для реалізації функціонального вебдодатку на наступному етапі роботи.

РОЗДІЛ 3
РОЗРОБКА ВЕБДОДАТКУ ДЛЯ ПІДГОТОВКИ ПОДОРОЖЕЙ

3.1 Практична реалізація об’єкта проєктування

Розробка вебдодатку для організації особистих подорожей реалізована із застосуванням сучасного стеку вебтехнологій: React.js для клієнтської частини, Node.js з Express для серверної логіки та MySQL для збереження даних. Робота над програмним забезпеченням велась у середовищі розробки Visual Studio Code, а для тестування REST API використовувався інструмент Postman. Компонентний підхід у React забезпечив зручну модульну структуру додатку.

На стороні клієнта (frontend) основна логіка побудови маршрутів реалізована у компоненті CreateRouteMap.jsx. Тут використовується бібліотека Leaflet для взаємодії з інтерактивною картою. Користувач натискає на карту для додавання координат точок маршруту, які зберігаються у стані React (useState). Після завершення побудови маршруту ці точки доповнюються назвами за допомогою запитів до API Geoapify (reverse geocoding), після чого відправляються POST-запитом до серверного API. Логіку обробки цих дій подано у лістингу 3.1.

Лістинг 3.1 – Функція збереження маршруту на клієнті

```
const handleSaveRoute = async () => {
  const pointsWithNames = await Promise.all(
    routePoints.map(async (p, index) => {
      const { point_name } = await getAddressFromCoordinates(p.lat, p.lng);
      return {
        latitude: p.lat,
        longitude: p.lng,
        position_index: index,
        point_name,
      };
    })
  );
  const createRes = await fetch("http://localhost:5000/api/routes", {
    method: "POST",
    headers: {
      "Content-Type": "application/json",
      Authorization: ’Bearer ${token}’,
```

```
    },
    body: JSON.stringify({
      route_name: routeName,
      points: pointsWithNames,
      mode: routeMode,
      estimated_time_seconds: totalSeconds,
    }),
  });
};
```

кінець лістингу 3.1

На сервері (backend) основна логіка реалізована у файлі server.js, який запускає сервер Express. Реєстрація та авторизація користувача виконуються з використанням бібліотек bcryptjs для хешування паролів і jsonwebtoken для генерації токенів доступу. Для захисту приватних маршрутів використовується middleware authenticateToken (ліст. 3.2).

Лістинг 3.2 – Middleware перевірки JWT-токена

```
const authenticateToken = (req, res, next) => {
  const authHeader = req.headers["authorization"];
  if (!authHeader || !authHeader.startsWith("Bearer ")) {
    return res.status(403).json({ error: "Невірний або відсутній токен" });
  }
  const token = authHeader.split(" ")[1];
  jwt.verify(token, SECRET_KEY, (err, decoded) => {
    if (err) return res.status(403).json({ error: "Невірний токен" });
    req.user = decoded;
    next();
  });
};
```

кінець лістингу 3.2

Для отримання тривалості маршруту реалізована окрема функція getRouteTime (ліст. 3.3), яка надсилає запит до API Geoapify, формує URL з параметрами waypoints, mode і apiKey та обробляє результат.

Лістинг 3.3 – Отримання тривалості маршруту через Geoapify

```
const getRouteTime = async (points, mode = "drive") => {
  const waypoints = points.map((p) => `${p.latitude},${p.longitude}`).join("|");
```

```
  const url =
'https://api.geoapify.com/v1/routing?waypoints=${waypoints}&mode=${mode}&apiKey=$
{apiKey}';
  const res = await fetch(url);
  const data = await res.json();
  return data.features?.[0]?.properties?.time || 0;
};
```

кінець лістингу 3.3

На стороні бази даних використано MySQL. Таблиці users, routes, route_points мають зв'язки «один-до-багатьох». Для кожного маршруту користувача зберігаються координати точок із зазначенням порядкового номера (position_index) і назви місцевості (point_name), отриманої з Geoapify.

Отже, практична реалізація включає не лише створення зручного клієнтського інтерфейсу, а й налаштування серверної логіки, обробку запитів до стороннього API, захист даних, а також збереження інформації до реляційної бази даних.

3.2 Тестування та налагодження інформаційно-комп'ютерної системи

У процесі реалізації вебдодатку для організації особистих подорожей було проведено комплексне тестування та налагодження як клієнтської, так і серверної частини. Основною метою тестування стало забезпечення стабільної роботи всіх функцій системи, перевірка коректності обробки даних, а також усунення виявлених помилок. Розробку та налагодження програмного забезпечення здійснено у середовищі Visual Studio Code з використанням внутрішніх інструментів розробника браузера та додаткових утиліт – зокрема, Postman для перевірки API-запитів та Chrome DevTools для налагодження інтерфейсу.

Система проходила перевірку на працездатність у типових сценаріях: реєстрація нового користувача, авторизація, побудова маршруту з інтерактивною картою, обчислення орієнтовного часу подорожі, збереження маршруту до бази даних та подальше його відображення у профілі користувача.

Додатково перевірялась можливість клонування маршрутів інших користувачів та автоматичне формування персоналізованих рекомендацій на основі географічних координат. Усі основні функції успішно пройшли тестування, а виявлені незначні помилки були усунені до завершення розробки.

Однією з проблем, яка виникла на ранніх етапах, стала некоректна обробка ситуації, коли точка маршруту не мала визначеної назви. Через це маршрути не зберігались, або повертали серверну помилку. Це було виправлено шляхом запровадження перевірки та автоматичної підстановки значення за замовчуванням – «Невідома точка». Ще однією особливістю, що потребувала налагодження, стала робота з локальним станом клієнта після збереження маршруту. Спочатку інтерфейс не скидав дані після створення маршруту, однак після додавання перезавантаження сторінки (window.location.reload()) ця проблема була вирішена.

Щодо перевірки надійності серверної частини, було протестовано всі основні API-ендпоїнти, зокрема створення, отримання, оновлення та видалення маршрутів. Також було змодельовано одночасну активність кількох користувачів, щоб перевірити стабільність роботи сервера та бази даних при навантаженні. Серверна логіка коректно обробляє вхідні запити, повертає передбачувані помилки у разі некоректних даних та захищає доступ до приватних маршрутів за допомогою JWT-токенів.

Для коректної роботи системи на клієнтському пристрої достатньо сучасного браузера з підтримкою JavaScript (наприклад, Google Chrome або Mozilla Firefox), мінімум 2 ГБ оперативної пам’яті та доступу до інтернету. Серверна частина потребує встановленого Node.js (версія 18 і вище), MySQL-сервера та наявності дійсного API-ключа Geoapify, що використовується для зворотного геокодування та розрахунку маршрутів.

У результаті виконаного тестування інформаційна система демонструє стабільну роботу, швидкий відгук, передбачувану поведінку інтерфейсу, а також належну обробку виняткових ситуацій. Виявлені недоліки усунено у

процесі розробки, що забезпечує якісну взаємодію користувача із системою у всіх передбачених сценаріях.

3.3 Розробка бази даних

У межах розробки вебдодатку для організації особистих подорожей було створено реляційну базу даних my_travels на платформі MySQL. Для адміністрування бази даних використовувався зручний вебінтерфейс phpMyAdmin, що забезпечив візуальне конструювання таблиць, зв’язків між ними, а також виконання SQL-запитів. На рисунку 3.1 представлено структуру бази даних, яка складається з трьох основних таблиць: users, routes та route_points.

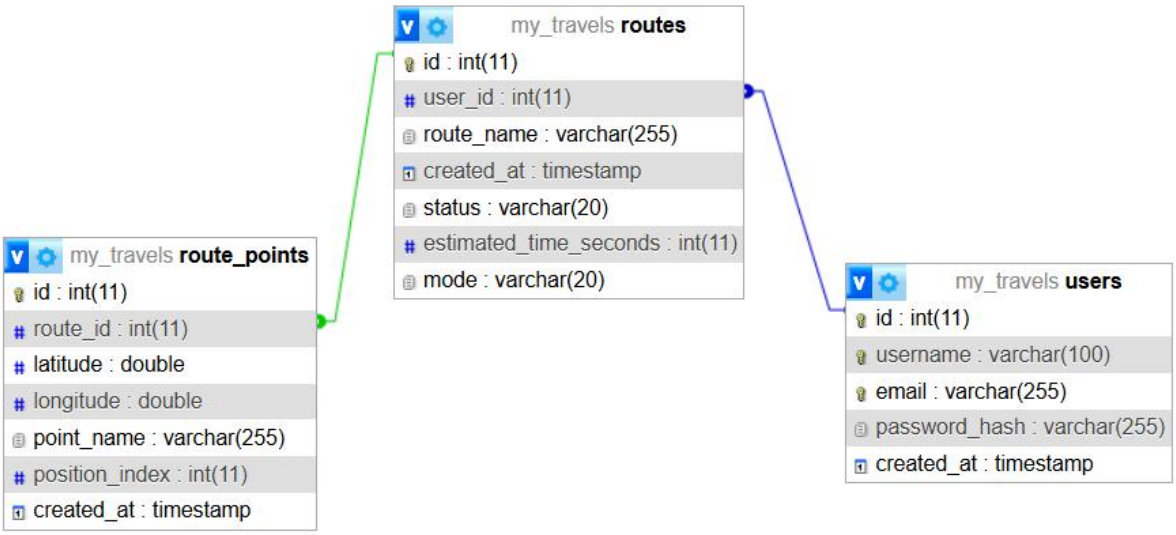

Рисунок 3.1 – Логічна модель бази даних my_travels у MySQL phpMyAdmin

Таблиця users містить облікові записи користувачів. Кожен користувач може мати кілька маршрутів, що реалізується за допомогою таблиці routes. Вона зберігає основні відомості про маршрут: назву, дату створення, режим пересування (пішки або автомобілем), статус і оцінений час у секундах. У свою чергу, таблиця route_points пов’язана з routes відношенням «один до багатьох» і містить координати точок маршруту, їх порядковий номер (position_index) і

назву (point_name), яка визначається автоматично через API Geoapify. Для створення таблиць у середовищі phpMyAdmin використовувались SQL-запити, які наведено нижче (ліст. 3.4).

Лістинг 3.4 – Запити створення таблиць

```
CREATE TABLE users (
  id INT AUTO_INCREMENT PRIMARY KEY,
  username VARCHAR(100) NOT NULL UNIQUE,
  email VARCHAR(255) NOT NULL UNIQUE,
  password_hash VARCHAR(255) NOT NULL,
  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
CREATE TABLE routes (
  id INT AUTO_INCREMENT PRIMARY KEY,
  user_id INT NOT NULL,
  route_name VARCHAR(255),
  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
  status VARCHAR(20) DEFAULT 'active',
  estimated_time_seconds INT,
  mode VARCHAR(20),
  FOREIGN KEY (user_id) REFERENCES users(id) ON DELETE CASCADE
);
CREATE TABLE route_points (
  id INT AUTO_INCREMENT PRIMARY KEY,
  route_id INT NOT NULL,
  latitude DOUBLE,
  longitude DOUBLE,
  point_name VARCHAR(255),
  position_index INT,
  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
  FOREIGN KEY (route_id) REFERENCES routes(id) ON DELETE CASCADE
);
```

кінець лістингу 3.4

Для вибірки всіх маршрутів певного користувача із зазначенням кількості точок можна скористатися наступним запитом (ліст. 3.5).

Лістинг 3.5 – Вибірка всіх маршрутів певного користувача із зазначенням кількості точок

```
SELECT
  r.id,
  r.route_name,
```

```
  r.created_at,
  COUNT(rp.id) AS points_count
FROM routes r
LEFT JOIN route_points rp ON r.id = rp.route_id
WHERE r.user_id = 1
GROUP BY r.id;
```

кінець лістингу 3.5

А щоб отримати всі точки маршруту з порядком проходження, використовується наступний запит (ліст. 3.6).

Лістинг 3.6 – Запит отримання всіх точок маршруту з порядком проходження

```
SELECT
  rp.latitude, rp.longitude, rp.point_name
FROM route_points rp
WHERE rp.route_id = 101
ORDER BY rp.position_index ASC;
```

кінець лістингу 3.6

Також передбачено можливість видалення маршруту разом із усіма його точками, що реалізується через зовнішній ключ з опцією ON DELETE CASCADE, тобто при видаленні маршруту автоматично видаляються пов’язані з ним точки.

Створена база даних повністю відповідає логіці застосунку, забезпечуючи збереження маршрутів у структурованому вигляді, швидкий доступ до інформації та підтримку рекомендаційної системи за місцем розташування першої точки.

3.4 Захист інформаційно-комп’ютерної системи

У межах розробки вебдодатку для організації особистих подорожей було реалізовано базові, але досить важливі механізми захисту інформаційно-комп’ютерної системи, які забезпечують автентифікацію, авторизацію, шифрування паролів користувачів та обмеження доступу до приватних

маршрутів. Захист реалізовано як на рівні серверної частини, так і у клієнтському середовищі.

Першим важливим елементом є хешування паролів, що здійснюється за допомогою бібліотеки bcryptjs. Під час реєстрації пароль користувача ніколи не зберігається у відкритому вигляді – замість цього він перетворюється у стійкий до зворотного обчислення хеш. У додатку застосовується алгоритм хешування з використанням солі (salt), яка генерується автоматично, що підвищує стійкість до атак типу rainbow table. Алгоритм хешування виглядає наступним чином (ліст. 3.7).

Лістинг 3.7 – Алгоритм хешування

```
const hashedPassword = bcrypt.hashSync(password, 8);
```

кінець лістингу 3.7

Хеш зберігається в полі password_hash таблиці users, що унеможливлює компрометацію реального пароля навіть у разі витоку бази.

Для автентифікації та авторизації реалізовано механізм JWT (JSON Web Token), який генерується після успішного входу користувача у систему. Сервер створює токен, у який вбудовується інформація про id, username та role користувача, і підписується секретним ключем (SECRET_KEY). Токен передається клієнту, зберігається у localStorage і надсилається з кожним запитом до захищених API. Процес перевірки токена реалізований у вигляді middleware-функції authenticateToken, яка аналізує заголовок авторизації у запиті, перевіряє підпис і термін дії токена (ліст. 3.8).

Лістинг 3.8 – Процес перевірки токена

```
jwt.verify(token, SECRET_KEY, (err, decoded) => {
if (err) return res.status(403).json({ error: "Невірний токен" });
req.user = decoded;
next();
});
```

кінець лістингу 3.8

Цей підхід гарантує, що доступ до персональних маршрутів користувача можливий лише за умови наявності чинного токена. Також це дозволяє реалізовувати контроль доступу за ролями (наприклад, admin), хоча на поточному етапі реалізовано тільки базову перевірку автентифікації.

Щодо захисту каналу передачі даних, у локальному середовищі під час розробки використовувався протокол HTTP, однак у разі розгортання системи в продуктивному середовищі передбачається застосування протоколу HTTPS з SSL-сертифікатом, що забезпечить шифрування всього трафіку між клієнтом і сервером.

Механізми захисту від DDoS-атак, обмеження частоти запитів, обфускації JavaScript-коду, а також захисту від декомпіляції на даному етапі не реалізовані, оскільки це виходить за межі базового функціоналу і передбачає застосування додаткових інструментів, таких як reverse proxy (наприклад, Nginx), модулі обмеження запитів (express-rate-limit) або серверні фільтри.

Загалом, у вебдодатку впроваджено надійний фундаментальний рівень захисту: використовується хешування паролів, перевірка авторизації через JWT, контроль доступу до даних користувача та базова обробка помилок. Це забезпечує безпечну взаємодію між користувачем і системою, що відповідає сучасним стандартам розробки веборієнтованих застосунків.

3.5 Розробка документації для програмного продукту

У процесі реалізації вебдодатку «МаршрутUA» для організації особистих подорожей було підготовлено технічну та користувацьку документацію, яка охоплює системні вимоги, інструкції з розгортання проєкту на сервері, з’єднання з базою даних, налаштування інтеграції з API, а також поради для кінцевих користувачів. Документація допомагає як розробникам, так і звичайним користувачам ефективно взаємодіяти з системою.

Для коректної роботи вебзастосунку необхідно дотримуватись певних вимог. Для клієнта (браузер):

1) сучасний браузер з підтримкою ES6 (Google Chrome 90+, Mozilla Firefox 88+);

2) підключення до Інтернету;

3) дозвіл екрана не менше 1024×768;

4) підтримка JavaScript і localStorage.

Для сервера (локальний або хостинг):

1) Node.js v18.0.0 або новіша;

2) менеджер пакетів npm;

3) MySQL Server 8.0+;

4) операційна система: Windows 10/11 або Ubuntu 20.04+;

5) API-ключ для Geoapify (безкоштовний або платний тариф).

Для полегшення роботи з додатком користувачам рекомендується ознайомитися з його основними можливостями. Зокрема, система дозволяє створювати власні маршрути: для цього достатньо натиснути кнопку «Побудувати маршрут» і додати необхідні точки на інтерактивній мапі. Після завершення побудови користувач може зберегти маршрут одним натисканням. Усі збережені маршрути доступні для перегляду в окремому розділі «Мої подорожі», де вони відображаються разом із інформацією про точки та статус виконання.

Кожен активний маршрут можна завершити або скасувати, що дозволяє керувати історією подорожей. Крім того, реалізована функція клонування маршрутів інших користувачів. У розділі «Рекомендації» система автоматично підбирає варіанти на основі геолокації першої точки маршруту користувача. Також додаток підтримує базові функції автентифікації та авторизації, що включають реєстрацію нового користувача, вхід у систему, а також можливість видалення облікового запису за запитом.

Опишемо інструкцію з розгортання вебдодатку на сервері. Першим кроком потрібно встановити всі необхідні залежності (ліст. 3.9).

Лістинг 3.9 – Встановлення залежностей

```
cd backend
npm install
cd ../travel-agency
npm install
```

кінець лістингу 3.9

Далі переходимо до налаштування змінних середовища. У корені backend потрібно створити файл .env (ліст. 3.10).

Лістинг 3.10 – Вміст файлу .env

```
DB_HOST=localhost
DB_USER=root
DB_PASSWORD=yourpassword
DB_NAME=my_travels
GEOAPIFY_KEY=ваш_ключ
```

кінець лістингу 3.10

Наступний крок – створення бази даних – через MySQL / phpMyAdmin виконати SQL-інструкції для створення таблиць users, routes, route_points. Переходимо до запуску серверної частини (ліст. 3.11).

Лістинг 3.11 – Запуск бекенду

```
cd backend
node server.js
```

кінець лістингу 3.11

Наступним кроком іде запуск клієнтської частини (ліст. 3.12).

Лістинг 3.12 – Запуск фронтенду

```
cd travel-agency
npm start
```

кінець лістингу 3.12

Після запуску вебдодаток буде доступний за адресою http://localhost:3000, серверна частина – на http://localhost:5000.

У цьому проєкті довідкова інформація не винесена у вигляді окремого файлу-довідки (наприклад, .chm або .pdf), однак її основні елементи реалізовані інтерактивно у самій системі:

1) в інтерфейсі компонента CreateRouteMap присутня підказка («Натискайте на карту, щоб додати точки маршруту»);

2) у розділі «Про нас» міститься загальна інформація про призначення додатку та його можливості;

3) додаткову довідкову функціональність можна реалізувати у майбутньому у вигляді окремого компонента типу FAQ або Help.

В цілому, документація охоплює всі важливі аспекти роботи з додатком як з боку користувача, так і з боку розробника чи адміністратора системи. Це сприяє швидкому впровадженню, налаштуванню та ефективному використанню системи для особистого планування подорожей.

Висновки до розділу 3

У результаті виконання практичного етапу кваліфікаційної роботи було повністю реалізовано вебдодаток для організації особистих подорожей з інтерацією інтерактивних карт і системи персоналізованих рекомендацій. Реалізація охоплює як клієнтську, так і серверну частину, що взаємодіють через REST API.

Для реалізації інтерфейсу користувача застосовано бібліотеку React.js, що дозволило створити динамічний, зручний та адаптивний інтерфейс із можливістю інтерактивної взаємодії з картою. Завдяки використанню Leaflet забезпечено візуалізацію точок маршруту на мапі, а API-сервіс Geoapify дозволив реалізувати автоматичне геокодування координат і розрахунок орієнтовного часу подорожі.

Серверна частина, реалізована на базі Node.js та Express, відповідає за авторизацію, збереження даних до бази, обробку маршрутів, а також рекомендаційні алгоритми на основі географічної близькості. Досягнуто безпечного зберігання паролів шляхом їх хешування через bcrypt, а захист маршрутів реалізовано через механізм авторизації з використанням JWT-токенів.

База даних створена на платформі MySQL. Її структура спроєктована відповідно до вимог нормалізації, що забезпечує логічну організацію збереження даних. Таблиці users, routes та route_points пов’язані між собою зв’язками «один до багатьох», що дозволяє зручно зберігати як загальні відомості про маршрути, так і деталі точок.

Проведено тестування функціоналу додатку, яке підтвердило його працездатність, коректність обробки даних та стабільну взаємодію між усіма компонентами системи. Було усунуто виявлені недоліки, вдосконалено валідацію даних та обробку виняткових ситуацій. Також було підготовлено документацію, яка містить системні вимоги, інструкції з розгортання проєкту, приклади використання та інструкції для користувача. Це дозволяє забезпечити простоту впровадження та подальшу підтримку розробленого вебдодатку.

Загалом розроблений програмний продукт відповідає поставленим функціональним вимогам, демонструє приклад ефективної побудови сучасного вебзастосунку та може бути використаний як база для створення комерційних або навчальних рішень у сфері туристичних онлайн-сервісів.

ВИСНОВКИ

У кваліфікаційній роботі було повністю виконано поставлене завдання щодо розробки вебзастосунку для організації особистих подорожей. Розроблена інформаційна система дозволяє користувачам створювати маршрути на інтерактивній мапі, зберігати їх, отримувати рекомендації на основі географічного розташування, а також керувати своїми подорожами в особистому кабінеті. Реалізовані функціональні можливості відповідають сучасним вимогам до web-сервісів у сфері персонального планування поїздок.

Було проведено аналіз популярних сервісів планування подорожей, таких як Google Maps, Rome2Rio, TripIt, Sygic Travel тощо. Визначено їх основні функціональні можливості, включно з побудовою маршрутів, бронюванням, інтеграцією з календарем, підтримкою офлайн-режиму. Серед недоліків виявлено складність адаптації під індивідуальні потреби користувача, обмеження безкоштовного функціоналу та низьку гнучкість для розробників. На основі дослідження сформовано уявлення про поточні тенденції: розвиток систем на базі штучного інтелекту, використання даних користувача для персоналізації, інтеграція з транспортними API.

Було сформульовано технічне завдання на розробку вебдодатку для планування подорожей. Визначено функціональні вимоги: створення маршрутів, авторизація, збереження, редагування та перегляд маршрутів, використання інтерактивної карти. До нефункціональних вимог віднесено: швидкодію, безпеку, масштабованість, кросплатформеність. Обрано сучасні технології: React.js для клієнтської частини, Node.js (Express) для серверної логіки, MySQL як СКБД, що забезпечують ефективну та гнучку архітектуру застосунку.

Було обґрунтовано вибір ключових бібліотек і технологій: для маршрутизації – використання бібліотеки Mapbox Directions API; для геокодування – використання OpenStreetMap Nominatim API; для рекомендацій – базовий фільтраційний підхід із можливістю подальшої інтеграції алгоритмів

машинного навчання. Обрані рішення забезпечують достатню точність та адаптивність під потреби користувачів при збереженні контролю з боку розробника.

Було реалізовано повнофункціональний вебдодаток: клієнтська частина на React.js забезпечує інтерфейс користувача з інтерактивною картою, формою маршруту та переглядом збережених подорожей; серверна частина на Node.js з Express відповідає за бізнес-логіку, обробку запитів, автентифікацію та роботу з базою даних; збереження інформації здійснюється в СКБД MySQL. Додаток підтримує створення, редагування, видалення маршрутів, а також автентифікацію користувачів.

Було здійснено тестування функціоналу додатку як вручну, так і з використанням модульних тестів (Jest для серверної частини). Виявлено та усунено низку помилок, зокрема у валідації даних та обробці некоректних запитів. Визначено мінімальні вимоги для запуску: Node.js ≥ v18, MySQL ≥ v8, браузери з підтримкою ES6. Проведено перевірку на відповідність технічним і функціональним вимогам.

Було спроєктовано логічну структуру бази даних у СКБД MySQL. Застосовано принципи нормалізації до 3-ї нормальної форми для уникнення надлишкових даних. Реалізовано таблиці users, routes, locations, saved_routes із необхідними зв’язками: один до багатьох і багато до багатьох. Забезпечено узгодженість та можливість масштабування системи.

Було реалізовано базовий рівень захисту системи: паролі користувачів хешуються за допомогою бібліотеки bcrypt перед збереженням у базі даних; для автентифікації використовується JWT (JSON Web Tokens), що дозволяє обмежити доступ до маршрутів лише авторизованим користувачам; реалізовано захист від деяких типових атак, зокрема XSS та SQL-ін’єкцій, за рахунок використання валідаторів і параметризованих запитів.

Було створено повну супровідну документацію до вебдодатку. Документація включає: покрокову інструкцію з розгортання (локально та на сервері), опис структури проєкту, мінімальні вимоги до середовища, базовий

посібник користувача (інтерфейс, можливості, приклади маршрутів), а також рекомендації для розширення системи – додавання рекомендацій на базі штучного інтелекту, підтримка офлайн-доступу та інтеграція з зовнішніми сервісами бронювання.

Практичне використання розробленого продукту можливе у вебсервісах туристичних агентств, локальних гідів, волонтерських платформ для координації переміщень, а також у навчальних проєктах з вивчення основ геоінформаційних технологій. Враховуючи адаптивність, система може бути розгорнута як на локальному сервері, так і на хмарних платформах, включаючи інтеграцію з CDN та захист через HTTPS.

Подальше вдосконалення системи може включати реалізацію мобільної версії, розширення функціоналу рекомендацій за допомогою машинного навчання, додавання фільтрів за категоріями маршрутів, розширення бази POI (Points of Interest), підтримку кількох мов інтерфейсу, а також інтеграцію з соціальними мережами для обміну маршрутами між користувачами.

Отже, результати, отримані в ході виконання кваліфікаційної роботи, мають прикладне значення, підтверджують здатність вирішувати реальні практичні задачі в галузі інформаційних систем і можуть бути використані як основа для подальших наукових, навчальних та комерційних ініціатив.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Шевчук Б. Л., Нестерчук І. К. Аналіз можливостей інтерактивних карт, створених на базі ГІС, для потреб туризму в Україні. Таврійський науковий вісник. Серія: Економіка. 2020. № 3. С. 147–154. URL: https://doi.org/10.32851/2708-0366/2020.3.18 (дата звернення: 05.03.2025).

2. Литвин В. В., Наум О. М., Висоцька В. А., Дверій М. В. Архітектура системи онлайн-туризму для пошуку та планування подорожей з урахуванням потреб користувача. Вісник Національного університету" Львівська політехніка". Інформаційні системи та мережі. 2019. №6. С. 13-29. URL: https://doi.org/10.23939/sisn2019.02.013 (дата звернення: 05.03.2025).

3. Грабар М. В. Інформаційні системи та технології на туристичному ринку: сучасність та перспективи. Market Infrastructure. 2020. №39. URL: https://doi.org/10.32843/infrastruct39-5 (дата звернення: 05.03.2025).

4. Bevor Sie zu Google Maps weitergehen. Google. URL: https://www.google.com/maps (date of access: 20.03.2025).

5. TripAdvisor. URL: https://www.tripadvisor.com/ (date of access: 22.12.2024).

6. Booking.com. URL: https://www.booking.com/ (date of access: 20.03.2025).

7. Airbnb | Помешкання для відпустки, зруби, будинки на пляжі тощо. Airbnb. URL: https://www.airbnb.com.ua/ (date of access: 20.03.2025).

8. Rome2Rio. URL: https://www.rome2rio.com/ (date of access: 20.03.2025).

9. React. URL: https://react.dev/ (date of access: 07.04.2025).

10. Dinku Z. React. js vs. Next. js, 2022. 36 p.

11. Node.js – Run JavaScript Everywhere. Node.js – Run JavaScript Everywhere. URL: https://nodejs.org/en (date of access: 07.04.2025).

12. Anh V. Real-time backend architecture using Node. js, Express and Google Cloud Platform, 2021. 51 p.

13. MySQL. URL: https://www.mysql.com/ (date of access: 07.04.2025).

14. Grippa V. M., Kuzmichev S. Learning MySQL. " O'Reilly Media, Inc.", 2021. 632 p.

15. Geoapify Location Platform: Maps, Geocoding, Routing, and APIs. Geoapify. URL: https://www.geoapify.com/ (date of access: 08.04.2025).

16. Orgoványi P., Hammer T., Karches T. Geocoding Applications for Enhancing Urban Water Supply Network Analysis. Urban Science. 2025. Vol. 9, no. 2. P. 51. URL: https://doi.org/10.3390/urbansci9020051 (date of access: 08.04.2025).