Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра комп'ютерних наук

Бакалаврська кваліфікаційна робота на тему:

РОЗРОБКА WEB-РЕСУРСУ «ТУРАГЕНТСТВО»

Виконав: студент 4 курсу, групи 2КН-21б
спеціальності 122 – Комп'ютерні науки
(шифр і назва напряму підготовки, спеціальності)
Кравчук А. О.
(прізвище та ініціали)
Керівник: к.т.н., доцент каф. КН
Крилик Л. В.
(прізвище та ініціали)
«04» червня 2025 р.

Рецензент: к.т.н., доцент каф. АІІТ
Гармаш В. В.
(прізвище та ініціали)
«05» червня 2025 р.

Допущено до захисту
Зав. кафедри Яровий А. А.
«06» червня 2025 р.

Вінниця ВНТУ - 2025 рік

Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра комп'ютерних наук
Рівень вищої освіти перший (бакалаврський)
Галузь знань – 12 Інформаційні технології
Спеціальність – 122 Комп'ютерні науки
Освітньо-професійна програма – Комп'ютерні науки

ЗАТВЕРДЖУЮ
Завідувач кафедри КН
проф., д.т.н. Яровий А. А.
«05» травня 2025 р.

ЗАВДАННЯ
НА БАКАЛАВРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТА
Кравчука Андрія Олександровича

1. Тема роботи: Розробка WEB-ресурсу «Турагентство».
керівник роботи: Крилик Людмила Вікторівна, к.т.н., доц.
затверджені наказом вищого навчального закладу «20» 03 2025 року №97

2. Термін подання студентом роботи 30 травня 2025

3. Вихідні дані до роботи: мова програмування – об'єктно-орієнтована; кількість можливих бронювань не менше 1; кількість підтримуваних платформ не менше 3; інтерфейс користувача має бути інтуїтивно зрозумілий; час відгуку не перевищує 0.8 секунд.

4. Зміст текстової частини (перелік питань, які потрібно розробити): вступ; обґрунтування доцільності розробки WEB-ресурсу «Турагентство»; проектування WEB-ресурсу «Турагентство»; програмна реалізація WEB-ресурсу «Турагентство»; висновки; список використаних джерел; додатки.

5. Перелік ілюстративного матеріалу (з точним зазначенням обов'язкових креслень): UML-діаграма прецедетів; UML-діаграма класів; UML-діаграма класів клієнтської частини модуля підбору турів; ER-модель для бази даних WEB-ресурсу «Турагенство»; схема алгоритму роботи програмного модуля WEB-ресурсу «Турагенство»; приклад роботи програми.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1–3	Крилик Л. В., к.т.н., доц. каф. КН		

7. Дата видачі завдання 05 травня 2025 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва та зміст етапу	Термін виконання		Примітка
		початок	закінчення	
1	Вступ. Обґрунтування доцільності розробки WEB-ресурсу «Турагентство»	05.05.25	07.05.25	
2	Проектування WEB-ресурсу «Турагентство»	08.05.25	11.05.25	
3	Програмна реалізація WEB-ресурсу «Турагентство»	12.05.25	15.05.25	
4	Висновки та аналіз отриманих результатів	16.05.25	26.05.25	
5	Оформлення додатків	27.05.25	29.05.25	
6	Розробка інструкції користувача	30.05.25	01.06.25	
7	Оформлення матеріалів до захисту БКР	02.06.25	04.06.25	

Студент _____ **Кравчук А. О**
(підпис) (прізвище та ініціал

Керівник роботи _____ **Крилик Л. В**
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Бакалаврська кваліфікаційна робота складається з 92 сторінок формату А4, які включають 20 рисунків і 6 таблиць. У списку використаних джерел зазначено 25 найменувань.

Метою роботи є розширення функціональних можливостей WEB-ресурсу «Турагентство».

На основі аналізу сучасних технологій, методик та технічних рішень було обґрунтовано доцільність розробки WEB-ресурсу «Турагентство». Створено алгоритм функціонування системи і UML-діаграми, що деталізують структуру і взаємодію між клієнтською та серверною частинами WEB-ресурсу, що суттєво полегшує процес розробки та підтримки програмного забезпечення.

Розробка програмного продукту проводилася в середовищі Visual Studio Code з використанням мови програмування JavaScript. Були реалізовані основні функціональні модулі WEB-ресурсу та проведено його тестування. Результати випробувань підтвердили успішну реалізацію поставлених завдань і правильну роботу системи.

Ключові слова: WEB-ресурс, турагентство, UML-діаграма, JavaScript.

ABSTRACT

The bachelor's thesis consists of 92 A4 pages, including 20 figures and 6 tables. The list of references includes 25 items.

Based on the analysis of modern technologies, methods and technical solutions, the feasibility of developing the WEB resource "Travel Agency" was substantiated. An algorithm for the functioning of the system and UML diagrams were created, detailing the structure and interaction between the client and server parts of the WEB resource, which significantly facilitates the process of software development and support.

The development of the software product was carried out in the Visual Studio Code environment using the JavaScript programming language. The main functional modules of the web resource were implemented and its testing was carried out. The test results confirmed the successful implementation of the tasks and the correct operation of the system.

Keywords: web resource, tour operator, UML diagram, JavaScript.

ЗМІСТ

ВСТУП

Актуальність досліджень.

З розвитком цифрових технологій та широким впровадженням Інтернету в усі сфери життя, туристична індустрія зазнала значних змін. Сучасні туристичні компанії дедалі активніше використовують онлайн-платформи, що дозволяє їм ефективніше взаємодіяти з клієнтами, пропонуючи зручний доступ до своїх послуг.

Припустимо, ви вирішили вирушити у відпустку. Як тільки це рішення прийнято, постають ключові питання: «Яке місце обрати?», «Де зупинитися?» і «Яка вартість відпочинку в обраному місці?». У пошуках відповідей люди звертаються до знайомих, шукають інформацію в Інтернеті або консультуються з туристичними агентствами. Однак кожен із цих способів має певні обмеження. Поради друзів обмежені їхнім власним досвідом подорожей, інформація в Інтернеті часто буває надто загальною і містить багато зайвого, а пропозиції туристичних агентств іноді здаються надмірно дорогими.

У таких умовах онлайн-платформи для туристичних агентств стають надзвичайно затребуваними. Вони забезпечують простоту використання, доступність із будь-якого пристрою, інтеграцію з платіжними системами для бронювання турів, а також можливість персоналізованого пошуку послуг. Крім того, WEB-ресурси сприяють покращенню конкурентоспроможності компаній, надаючи клієнтам інструменти для зручного вибору турів, порівняння пропозицій та швидкого оформлення бронювання.

З огляду на зростаючу конкуренцію в індустрії туризму та необхідність у сучасному сервісі, розробка WEB-ресурсу «Турагентство» є надзвичайно актуальною, це підтверджується такими аспектами:

- зростання конкуренції на ринку туризму: ринок туристичних послуг натепер є надзвичайно конкурентним. Туристичні компанії змушені шукати нові способи залучення клієнтів і збільшення прибутків. Наявність якісного WEB-ресурсу дає можливість забезпечити зручний доступ до послуг для користувачів,

значно підвищуючи ефективність бізнесу. Завдяки цьому, компанії можуть швидко та ефективно надавати послуги, що дозволяє їм виділитися серед конкурентів і залучити нових клієнтів;

- попит на онлайн-бронювання турів та послуг: онлайн-бронювання стає все популярнішим способом вибору та придбання туристичних послуг. З кожним роком збільшується кількість туристів, які обирають для себе тури через інтернет, це підтверджує зростаючу потребу в ефективних WEB-ресурсах. Створення WEB-ресурсу для турагентства сприятиме користувачам зручно знайти, порівняти і забронювати тури з мінімальними витратами часу та зусиль;

- зростання мобільного доступу до сервісів: кількість користувачів, які використовують мобільні пристрої для бронювання турів і пошуку інформації про подорожі, стрімко зростає. WEB-ресурс для турагентств має бути адаптованим для мобільних пристроїв, що дозволяє користувачам бронювати тури з будь-якої точки світу та в будь-який час. Це дозволяє підвищити доступність послуг і зробити їх ще зручнішими для споживачів;

- використання новітніх технологій в управлінні туризмом: зростає важливість використання сучасних технологій для автоматизації та оптимізації процесів управління туристичними послугами. Розробка WEB-ресурсу для турагентства дозволяє не лише полегшити процес бронювання та обробки запитів, але й використовувати аналітичні інструменти для покращення досвіду користувачів, що є важливим аспектом у збереженні конкурентоспроможності.

Отже, тема «Розробка WEB-ресурсу «Турагентство»» є надзвичайно актуальною, оскільки відповідає сучасним вимогам ринку туризму, сприяє розвитку онлайн-послуг і покращує взаємодію з клієнтами, що в свою чергу підвищує ефективність та конкурентоспроможність туристичних компаній.

Метою дослідження є розширення функціональних можливостей WEB-ресурсу «Турагентство».

Об’єктом дослідження є процес пошуку та бронювання турів.

Предметом дослідження є програмні засоби та технології для реалізації WEB-ресурсу «Турагентство».

Задачі дослідження:

1. Обґрунтувати доцільність розробки WEB-ресурсу «Турагентство», враховуючи сучасні тенденції розвитку ринку туризму;

2. Спроектувати WEB-ресурс «Турагентство»;

3. Програмно реалізувати WEB-ресурс «Турагентство»;

4. Виконати тестування WEB-ресурсу «Турагентство» та провести аналіз отриманих результатів.

5. Розробити інструкцію користувача.

Апробація роботи.

Результати досліджень було апробовано на LIV Всеукраїнській науково-технічній конференції підрозділів Вінницького національного технічного університету (НТКП ВНТУ) м. Вінниці у 2025 р. [1].

Публікації: за основними результатами досліджень опубліковано тези доповіді на науково-технічній конференції [1].

1 ОБҐРУНТУВАННЯ ДОЦІЛЬНОСТІ РОЗРОБКИ WEB-РЕСУРСУ «ТУРАГЕНТСТВО»

Подорожі та туризм є невід'ємною частиною індустрії гостинності. Відмінності в часових поясах, відстанях, умовах життя, цілях і тривалості подорожей підкреслюють багатогранність цього явища. Туризм можна розглядати як багатофункціональне явище, що поєднує в собі не лише пригоди, романтику далеких країн та дослідження екзотичних місць, але й практичні аспекти, такі як підприємництво, охорона здоров'я, безпека та захист особистої власності.

На сьогодні співпраця з туристичними агентствами є одним з ключових елементів поточної та стратегічної діяльності будь-якого туроператора. Важко переоцінити значення турагентів для успіху та розвитку бізнесу оператора. Незалежно від популярності туроператорів та уваги до їхніх турів, досягти високих результатів у туристичному секторі неможливо без ефективної та розгалуженої мережі агентств. Тому у кожного туроператора є спеціальний співробітник або навіть цілий відділ, який працює з агентами, розробляючи нові способи мотивації та підвищуючи ефективність роботи агентської мережі [2]. Розглянемо переваги, які забезпечить WEB-ресурс «Турагентство» для користувачів та туристичних платформ:

1. Збільшення залученості користувачів: залучити більше користувачів на платформу, надаючи зручний та ефективний спосіб вибору туристичних пропозицій. Користувачі з більшою ймовірністю проводитимуть більше часу на платформі та користуватимуться нею частіше, якщо отримуватимуть рекомендації, які відповідають їхнім інтересам.

2. Раціональне керування контентом: розробка WEB-ресурсу «Турагентство» забезпечує оптимальне використання інформації про туристичні пропозиції. Завдяки аналізу поведінки користувачів на платформі, можна визначити їхні основні інтереси та вподобання, що дозволяє адаптувати пропозиції та формувати тури, які найбільше відповідають потребам цільової аудиторії.

3. Перевага над конкурентами: у сучасному середовищі конкуренції між туристичними платформами створення WEB-ресурсу «Турагентство» може стати важливим фактором для досягнення лідерства на ринку. Платформи, що пропонують більш інтуїтивні, персоналізовані та ефективні рішення для вибору турів, мають більші шанси залучити нових клієнтів і зберегти лояльність наявних.

4. Оптимізація управління контентом: розробка WEB-ресурсу «Турагентство» дозволяє більш ефективно організовувати туристичний контент. Вивчаючи взаємодію користувачів із пропозиціями турів, можна виявити їхні переваги та адаптувати контент, пропонуючи тури, що найкраще відповідають інтересам цільової аудиторії.

Розробка WEB-ресурсу «Турагентство» пропонує значні переваги для користувачів та туристичних платформ. Завдяки ефективному керуванню контентом і персоналізованим рекомендаціям, ресурс може значно покращити досвід користувачів, допомагаючи їм швидше знаходити найбільш відповідні тури. Взаємодія з платформою дозволяє отримувати індивідуальні пропозиції, які відповідають уподобанням користувачів, що в свою чергу підвищує їх задоволеність і лояльність.

Для туристичних агентств власний WEB-ресурс − це важливий інструмент, який відкриває нові можливості для прямого бронювання. Він може значно спростити процес замовлення, підвищити ефективність обслуговування клієнтів і зменшити залежність від посередників. Інтеграція з платіжними системами та сторонніми сервісами дозволяє автоматизувати рутинні процеси, знизити витрати і підвищити конкурентоспроможність компанії на ринку [3].

Крім того, аналіз поведінки користувачів на платформі дозволяє постійно вдосконалювати туристичну пропозицію. Це робить систему більш привабливою і враховує потреби та вподобання потенційних клієнтів. Таким чином, WEB-ресурс туристичної агенції не лише забезпечує зручний інтерфейс для вибору та бронювання турів, але й створює приємне середовище для взаємодії з користувачами.

Такий підхід допомагає поліпшити якість обслуговування, підвищити лояльність клієнтів і зробити процес організації подорожей більш інтуїтивно зрозумілим та ефективним. Як наслідок, WEB-ресурси стають важливим фактором успіху туристичних організацій у сучасному цифровому світі.

1.1 Огляд середовищ для реалізації програмного продукту

Вибір оптимального середовища для розробки WEB-ресурсу «Турагентство» є критично важливим етапом у створенні ефективного програмного продукту. Оскільки різноманітні платформи надають різні можливості для реалізації функціональності WEB-ресурсів, вибір середовища має ґрунтуватися на вимогах користувачів, специфіці обробки даних, а також масштабованості та надійності системи.

В таблиці 1.1 наведено порівняльну характеристику платформ для створення WEB-ресурсу «Турагентство».

WEB-ресурси − це один з найефективніших способів розробки платформи, яка пропонує користувачам можливість вибирати і переглядати туристичні тури. Це дуже зручно, оскільки користувачі можуть отримати доступ до сервісу з будь-якого пристрою, підключеного до Інтернету.

Основною перевагою WEB-ресурсів є їх універсальність. Вони сумісні з різними операційними системами, що дозволяє використовувати їх як на комп'ютерах, так і на планшетах, і смартфонах, забезпечуючи безперешкодний доступ до платформи в будь-який час та з будь-якого місця.

WEB-ресурси надають достатню функціональність для більшості користувачів і не вимагають встановлення додаткового програмного забезпечення. Для доступу до них достатньо стандартного WEB-браузера на більшості пристроїв.

Таблиця 1.1 – Аналіз існуючих платформ для розробки WEB-ресурсу «Турагентство»

	ПК	Мобільні пристрої	WEB-застосунки
Доступність	Зручний доступ на робочому місці, вдома або в офісі	Мобільний доступ з будь-якого місця	Зручний доступ з будь-якого пристрою з Інтернетом
Операційні системи	Windows, macOS, Linux	Android, iOS	Сумісність з будь-яким сучасним WEB-браузером
Охоплення користувачів	Менше охоплення порівняно з телефонами	Велике охоплення через популярність мобільних пристроїв	Велике охоплення через універсальність доступу
Встановлення	Потребує установки окремого додатку на ПК	Потребує установки мобільного додатку на пристрій	Не потребує установки, доступ через WEB-браузер
Зручність використання	Обмежена мобільність, залежність від ОС	Компактність, портативність, залежність від ОС	Універсальність доступу
Екосистема	Різноманітна, залежить від платформи з фреймворками та інструментами, такими як .NET Core, Electron, Qt	Розділена між Android і iOS, обмеженіша	Велика кількість бібліотек React, Angular, Vue.js, Django, Flask, Laravel
Мови програмування	Не обмежений (Java, C++, C#, Python, JavaScript тощо)	Обмежений − Java, Kotlin (Android), Swift, Objective-C (iOS)	Обмежений для FrontEnd (JavaScript/Typescript, WASM). Необмежений для BackEnd

Аналізуючи дані таблиці 1.1, можна зробити висновок, що WEB-застосунок є корисним та універсальним інструментом для розробки програмних продуктів і

є найкращим вибором для цього проекту. Однак, з метою охоплення ширшого кола користувачів та покращення зручності використання, цей програмний модуль буде реалізовано як у WEB-браузерах так і на мобільних пристроях.

1.2 Огляд систем-аналогів

При розробці WEB-ресурсів потрібно враховувати існуючі програмні рішення, які пропонують схожий функціонал. Для досягнення найкращих результатів важливо провести детальний аналіз подібних програмних продуктів і виявити можливості для вдосконалення програмного продукту, що розробляється. Оцінка конкурентів може допомогти виявити сильні сторони, які можна адаптувати для створення унікального та конкурентоспроможного продукту, і слабкі сторони, які можуть стати точками зростання.

Такий аналіз забезпечує краще розуміння потреб користувачів та мінімізує можливі ризики під час розробки та впровадження WEB-ресурсів. Виявлення слабких місць в існуючих системах створює можливості для створення інноваційних програмних продуктів, які є не тільки функціональними, але й більш зручними для користувачів.

Також, враховуючи досвід конкурентів, можна звернути увагу на покращення інтерфейсу, посилення інтеграції з іншими сервісами та забезпечення високого ступеня персоналізації для користувача. Таким чином, новостворений WEB-ресурс турагентства буде більш привабливим, забезпечить ефективність та відповідатиме потребам кінцевих користувачів.

Для проведення порівняльного аналізу розглянемо такі сервіси, як «Expedia», «KAYAK», «Анекс Тур».

Expedia – глобальний сервіс для бронювання подорожей, включаючи готелі, перельоти, оренду авто та екскурсії (рис. 1.1) [4].

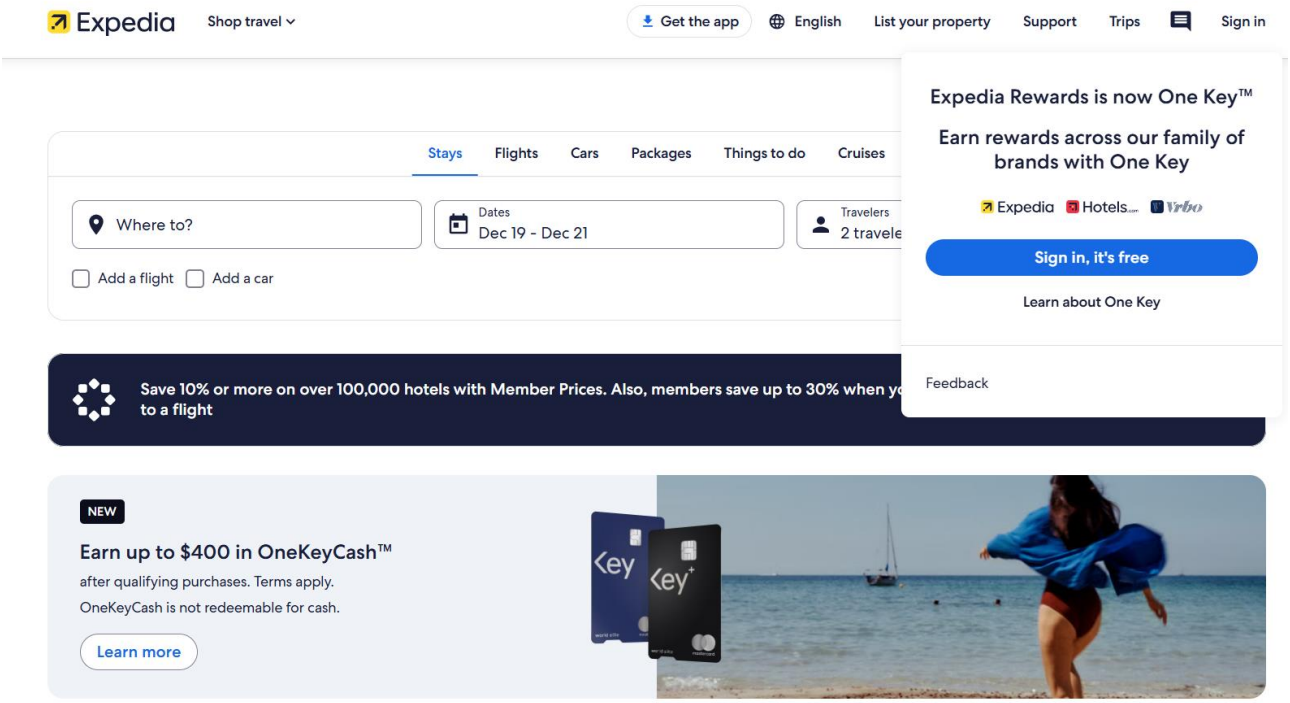

Рисунок 1.1 – Загальний вигляд інтерфейсного вікна WEB-платформи Expedia

Платформа Expedia пропонує такі можливості:

- персоналізований вибір туристичних пропозицій;
- доступ до детальної інформації про кожен тур;
- можливість створювати індивідуальні або групові списки вподобаних турів;
- функція: поділитися списком обраних турів із друзями.

Недоліки:

- відсутність української локалізації інтерфейсу;
- Україна не входить до списку підтримуваних країн;
- наявність реклами, яка займає значну частину екрану.

KAYAK – онлайн-сервіс з пошуку авіаквитків, бронювання готелів, оренди автомобілів, пошуку турпакетів та круїзів (рис. 1.2) [5].

Особливості платформи KAYAK:

- інтуїтивно зрозумілий та простий у використанні інтерфейс;
- наявність інструментів для бюджетних Sarthi Travelів;
- порівнянна цін на різні послуги.

Недоліки платформи KAYAK:

- інформація про ціни не завжди актуальна;

- за необхідності змін чи скасування бронювання потрібно звертатись безпосередньо до постачальників послуг;

- деякі додаткові збори можуть бути приховані до завершення бронювання.

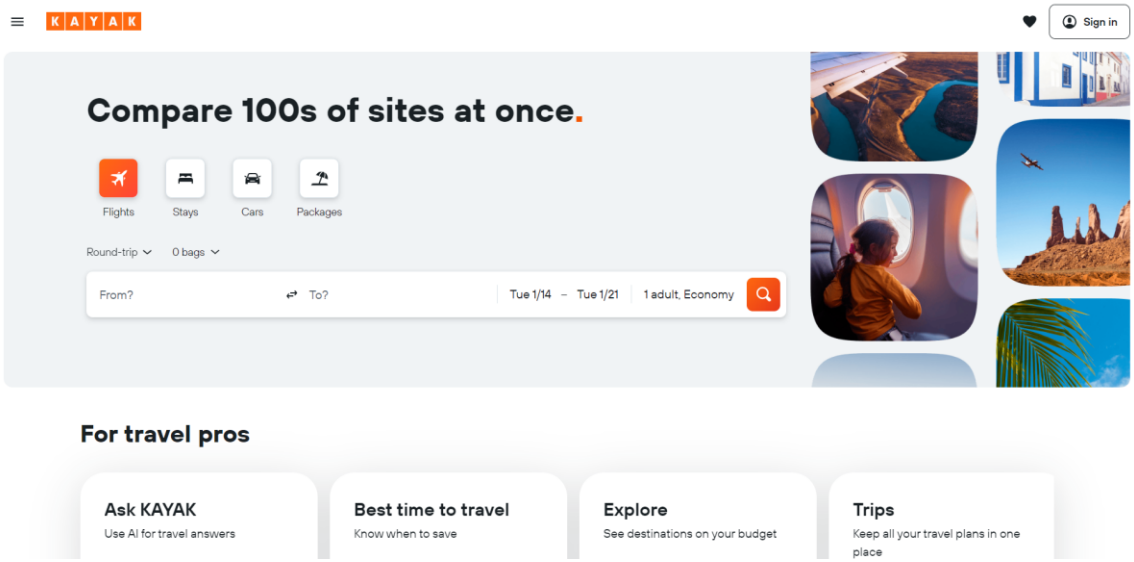

Рисунок 1.2 – Загальний вигляд інтерфейсного вікна платформи KAYAK

Анекс Тур – один із лідерів туристичного ринку, що спеціалізується на пакетних турах та індивідуальних програмах подорожей (рис. 1.3) [6].

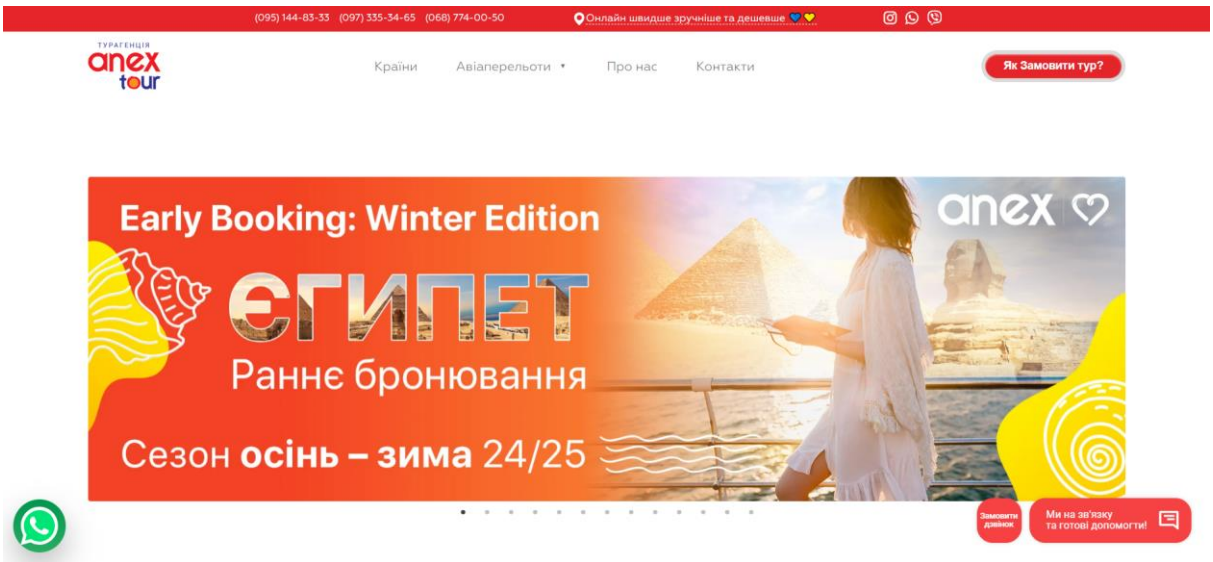

Рисунок 1.3 – Загальний вигляд інтерфейсного вікна платформи Annex Tours

Особливості платформи Annex Tours:

- широкий вибір турів і напрямків;
- детальна інформація про кожен тур;
- підтримка багатьох мов;
- платформа доступна як WEB та мобільний додаток.

Недоліки платформи Annex Tours:

- відсутність гнучкості в налаштуваннях пошуку (наприклад, обмежені фільтри);
- нестабільність рейтингів;
- обмежена інтерактивність;
- неточна інформація про ціни в процесі бронювання.

1.3 Формулювання вимог та постановка задачі

Кількість напрямків і туристичних пропозицій зростає з кожним роком, тому клієнтам стає дедалі складніше обрати найкращий для себе тур. Це пов'язано з тим, що у всіх різні смаки, бюджети і бажані вимоги до відпочинку. Одних цікавлять пляжні курорти, інші хочуть екстремального відпочинку, треті − культурної подорожі. Вибір туру для групи часто призводить до довгих дискусій та суперечок. Розробка програмного забезпечення, яке підбирає тури на основі вподобань користувача та групи, дозволить скоротити час, витрачений на вибір і знайти найкращий варіант для всіх.

Annex Tours, Expedia та KAYAK − провідні онлайн-платформи для планування подорожей, кожна з яких пропонує користувачам унікальні пропозиції. Annex Tours спеціалізується на організації пакетних турів, екскурсій та індивідуальних турів по всьому світу. Платформа пропонує широкий вибір популярних напрямків і надає своїм клієнтам спеціальні пропозиції та знижки. Це корисно для тих, хто шукає пропозиції «все включено», які включають перельоти, готелі та трансфери. Однак обмежена інформація про менш популярні та нетрадиційні напрямки може бути складною для тих, хто прагне дослідити нові

місця. Expedia пропонує широкий спектр туристичних послуг, включаючи перельоти, готелі, оренду автомобілів та екскурсії. Expedia також має програму лояльності для постійних клієнтів, які можуть заробляти бали та отримувати знижки. KAYAK − ще одна популярна туристична платформа. Вона пропонує гнучкі туристичні послуги, включаючи перельоти, готелі, оренду автомобілів та різні туристичні пакети, а також має програму лояльності, яка пропонує знижки постійним користувачам. Однак обмежений вибір спеціальних послуг у деяких напрямках може стати проблемою для користувачів, які шукають персоналізовані варіанти подорожей. Хоча всі ці платформи мають свої переваги, вони також мають свої проблеми, такі як обмежена інформація про менш популярні напрямки, тривалі процедури бронювання та обмежена кількість спеціальних послуг для певних напрямків. Проте всі ці платформи є надійними варіантами для планування подорожей і пропонують широкий вибір варіантів для Sarthi Travelів, які шукають різноманітних вражень від подорожей по всьому світу.

Альтернативою онлайн-платформам для планування подорожей, таким як Anex Tour, Expedia та KAYAK, можуть бути локальні туроператори або спеціалізовані ресурси, що орієнтуються на менш популярні напрямки та пропонують більш спрощений і зручний інтерфейс. Вони можуть запропонувати користувачам доступ до ексклюзивних пропозицій та послуг, які не завжди доступні на великих платформах. Ці сервіси часто мають меншу кількість функцій, що може зробити їх простішими у використанні. Крім того, спеціалізовані платформи можуть пропонувати більш детальну інформацію про конкретні напрямки, місцеві готелі, активності, що дозволяє забезпечити більш точний підбір турів відповідно до інтересів користувачів.

Пошукові системи для вибору турів можуть містити такі функції:

- аутентифікація та авторизація користувачів: можливість створення облікового запису або входу через соціальні мережі для персоналізації пошуку турів;

- створення груп для вибору турів: користувачі можуть створювати групи, щоб спільно обирати напрямки та тури;

- редагування існуючих груп: учасники можуть оновлювати свої плани подорожей, додаючи нові опції та коригуючи деталі;

- пропонування турів на основі вподобань групи: система автоматично підбирає тури, що відповідають потребам та бюджету групи, з урахуванням інтересів кожного учасника;

- сторінка популярних турів: цей розділ відображає найпопулярніші напрямки подорожей із посиланнями на рекомендації, сформовані на основі відгуків та оцінок інших користувачів.

Таким чином WEB-ресурси дозволяють користувачам не тільки знаходити індивідуальні пропозиції подорожей, але й організовувати спільні поїздки з друзями та колегами.WEB-ресурс та мобільний додаток мають бути інтуїтивно зрозумілими для користувача, тому будуть розроблятись за допомогою користувацького досвіду (UX). Вхідні дані мають легко задаватися користувачем за допомогою інтуїтивного та зрозумілого інтерфейсу. Вихідні дані мають бути зрозуміло представлені, щоб користувач міг без зайвих зусиль знайти потрібну йому інформацію.

Для створення WEB-ресурсу «Турагентство», на основі досвіду платформ Anex Tour, Expedia та KAYAK, потрібно реалізувати рішення, яке забезпечить користувачам зручний інструмент для швидкого та ефективного планування подорожей. Ресурс має враховувати індивідуальні та групові потреби, пропонуючи персоналізовані рекомендації, можливість вибору маршрутів та турів відповідно до вподобань і бюджету.

Для створення коректного інтуїтивного WEB-ресурсу для підбору турів, потрібно вирішити основні завдання:

- реалізувати систему пошуку та підбору турів;

- реалізувати систему замовлення та бронювання турів;

- реалізувати систему створення та управління туроператорськими пропозиціями;

- реалізувати систему оцінок та відгуків;

- здійснити тестування та оптимізацію WEB-ресурсу.

Отже, реалізація цих функцій дозволить створити зручний та ефективний інструмент для користувачів, який забезпечить простий доступ до туристичних пропозицій, покращить взаємодію між турагентством і клієнтами, а також підвищить рівень задоволеності від використання сервісу.

1.4 Висновок

У цьому розділі було проведено огляд доступних платформ для розробки програмних продуктів, спрямованих на спрощення процесу вибору туристичних послуг. В якості основних платформ для реалізації модулів були розглянуті Anex Tour, Expedia та KAYAKA. Однак аналіз показав, що більшість цих платформ орієнтовані на індивідуальні запити і не завжди забезпечують ефективну персоналізацію подорожей.

Розробка WEB-ресурсу «Турагенство», який слує для підбору турів має велике значення, оскільки зростаючий попит на послуги онлайн-планування подорожей вимагає зручних та ефективних інструментів, які дозволять користувачам підбирати тури, що відповідають їхнім різним уподобанням. З огляду на те, що багато людей обирають напрямки, спираючись на рекомендації друзів та родини, можливість створювати групові плани та враховувати думки кожного члена групи було б доцільним. Такий інструмент значно спрощує процес планування подорожі та економить час на пошук і вибір найкращого варіанту.

У цьому розділі визначено завдання для подальшої розробки WEB-ресурсу «Турагенство». WEB-ресурс має на меті забезпечити простоту та ефективність процесу вибору туристичних послуг і допомогти користувачам оптимізувати свої плани подорожей.

2 ПРОЕКТУВАННЯ WEB-РЕСУРСУ «ТУРАГЕНТСТВО»

2.1 Вибір архітектури для розробки WEB -ресурсу «Турагентство»

Для ефективної організації роботи туристичних операторів доцільно розподіляти процеси на окремі етапи. Існують різні підходи до управління туристичними послугами, які допомагають оптимізувати бронювання, логістику та комунікацію з клієнтами. Серед найефективніших моделей – централізоване управління заявками, автоматизовані системи бронювання та інтеграція з партнерами. Вони дозволяють забезпечити зручність для користувачів, спростити управління туристичними пропозиціями та покращити обслуговування клієнтів. Тому варто детально розглянути ці підходи для покращення роботи турагентств.

Архітектура MVC (Model-View-Controller) забезпечує структурований підхід до розробки програмного забезпечення, поділяючи його на три основні компоненти: модель, подання та контролер. Модель відповідає за обробку даних і реалізацію бізнес-логіки, подання – за відображення інформації користувачеві, а контролер – за керування взаємодією між ними. Такий поділ дозволяє розробникам незалежно модифікувати кожен із компонентів, що сприяє гнучкості, масштабованості та зручності супроводу програмного продукту.

Концепція MVC була розроблена ще в 1970-х роках і стала основою багатьох сучасних WEB-ресурсів, особливо тих, що працюють із динамічними даними та потребують швидкої адаптації до змін у бізнес-логіці. Основна перевага цього підходу – чітке розмежування функціональних частин програми, що значно спрощує тестування, підтримку та розширення коду іншими розробниками.

Завдяки такій архітектурі WEB-застосунки стають більш структурованими, зручними для обслуговування та ефективними, що робить використання MVC актуальним для широкого спектра програмних рішень [7, 8].

Архітектура MVC (Model-View-Controller) є одним із найпоширеніших шаблонів проєктування, що забезпечує чіткий розподіл функціональності між

окремими компонентами, спрощуючи розробку, тестування та підтримку програмного забезпечення.

Основні компоненти MVC

Модель (Model) є центральним елементом системи, який містить бізнес-логіку та відповідає за управління даними. Вона забезпечує взаємодію з базою даних, виконує обробку інформації та перевіряє її коректність. Модель не має прямого зв’язку з інтерфейсом користувача, що робить її незалежною від змін у поданні.

Модель може бути представлена у вигляді:

- бізнес-логіки програми, що виконує основні обчислення та перевірку даних;

- окремого шару даних (DAO, БД, XML-файлів), що відповідає за збереження та отримання інформації;

- менеджера бази даних або набору об’єктів, які організовують роботу з інформацією в системі.

Подання (View) відповідає за візуалізацію даних і взаємодію з користувачем. Воно отримує інформацію від моделі та відображає її у зручному форматі. Подання не виконує змін у логіці програми, а лише забезпечує користувацький інтерфейс.

Основні характеристики подання:

- отримує інформацію від моделі та представляє її у вигляді графічного або текстового інтерфейсу;

- може містити певну логіку для форматування даних перед їхнім відображенням;

- може бути реалізоване у вигляді HTML-сторінок, WPF-форм, мобільних додатків або інших інтерфейсів.

Контролер (Controller) є сполучною ланкою між користувачем і системою. Він обробляє введені користувачем команди, передає їх моделі для обробки та оновлює відповідне подання. Контролер дозволяє змінювати модель і визначати,

яке подання потрібно відобразити в конкретний момент часу. Основні особливості контролера:

- обробляє запити користувача та передає їх моделі для виконання необхідних дій;

- визначає, яке подання потрібно використовувати для відображення результатів обробки;

- один контролер може керувати кількома поданнями, забезпечуючи гнучкість і масштабованість системи.

Розглянемо взаємодію компонентів та UML-діаграму. Завдяки чіткому розділенню обов'язків між компонентами MVC, розробники отримують можливість легше оновлювати, тестувати та розширювати систему. Для наочного представлення взаємодії між Model, View та Controller доцільно використовувати UML-діаграми. На рисунку 2.1 наведено UML-діаграму, що ілюструє зв'язки між основними компонентами MVC та їхню взаємодію [7 –10].

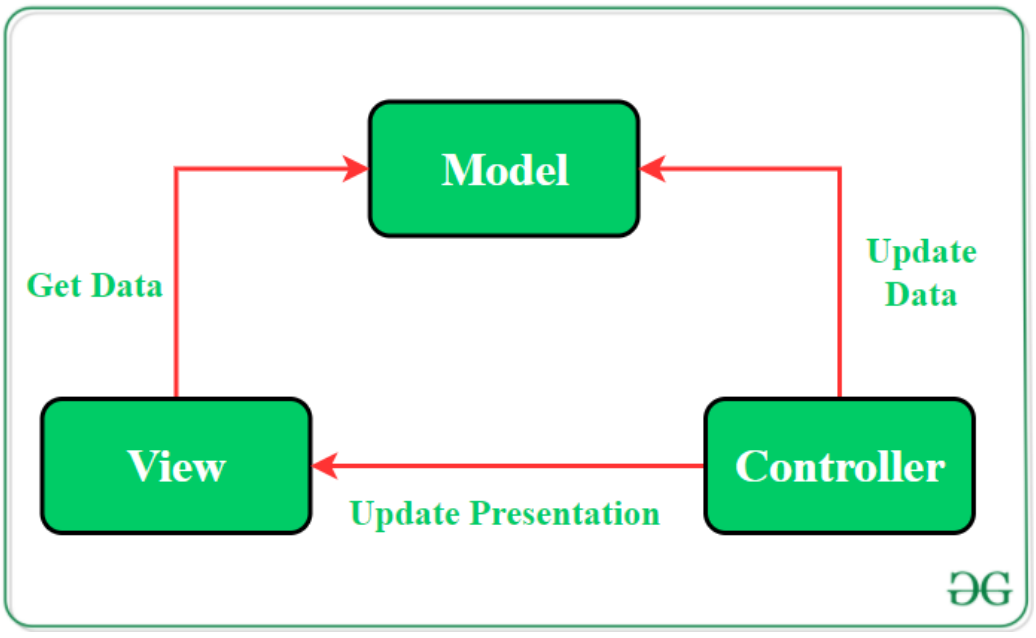

Рисунок 2.1 – UML-діаграма взаємодії ключових компонентів архітектури MVC

Шаблон MVP є подібним до архітектури MVC, але замість контролера (Controller) використовується презентатор (Presenter). Ця архітектура розподіляє додаток на три основні компоненти: Model (модель), View (вигляд) та Presenter (презентатор). На UML діаграмі, що представлена на рисунку 2.2, відображено взаємодію цих основних компонентів архітектури MVP.

Модель (Model) являє собою набір класів, що визначають бізнес-логіку та структуру даних, а також встановлюють правила маніпулювання даними і доступу до них.

Вигляд (View) складається з елементів інтерфейсу користувача, які відповідають за відображення даних, отриманих від презентатора.

Презентатор (Presenter) бере на себе відповідальність за обробку всіх подій інтерфейсу користувача. Він отримує вхідні дані через View, обробляє їх за допомогою Model і передає результати до View для оновлення інтерфейсу [11].

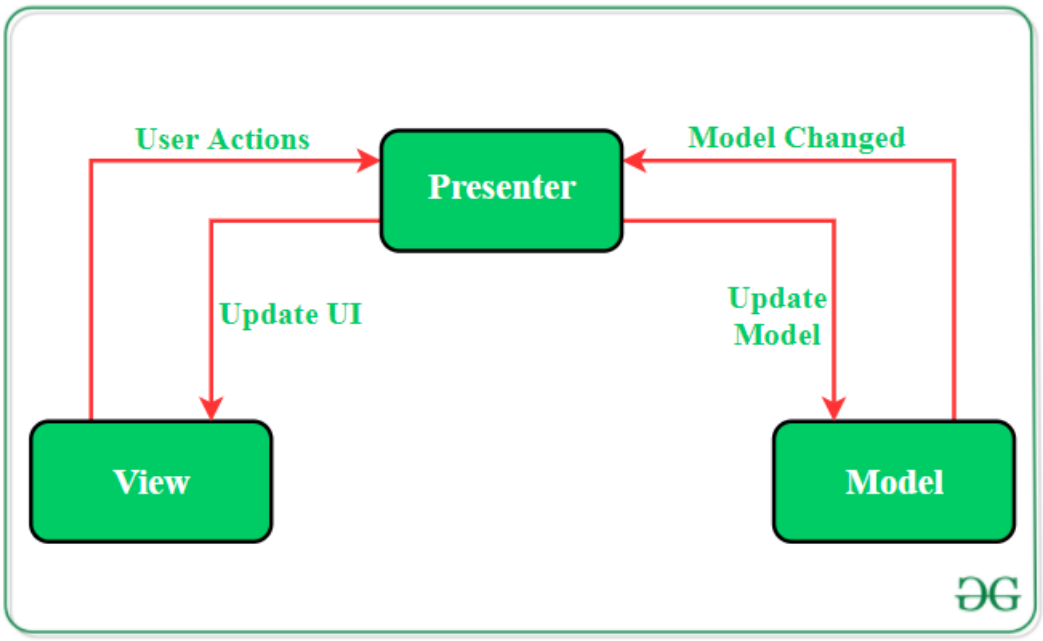

Рисунок 2.2 – UML-діаграма, що ілюструє взаємодію ключових компонентів архітектури MVP

Шаблон «Модель-Вигляд-Модель представлення» (MVVM)

Архітектурний шаблон MVVM має подібність до MVP (Model-View-Presenter), оскільки ViewModel виконує функції, аналогічні Presenter. Однак MVVM усуває певні недоліки MVP, пропонуючи чітке розділення логіки відображення даних (View або UI) та бізнес-логіки програми (рис. 2.3).

Модель (Model) – відповідає за управління джерелами даних і їх абстрагування. Вона взаємодіє з ViewModel для отримання та збереження інформації.

Вигляд (View) – цей рівень слугує для передачі подій користувача у ViewModel. Він спостерігає за ViewModel, не містячи в собі логіки програми.

Модель представлення (ViewModel) –містить необхідні потоки даних, що використовуються у View та виступає посередником між виглядом і моделлю, забезпечуючи їхню взаємодію [8].

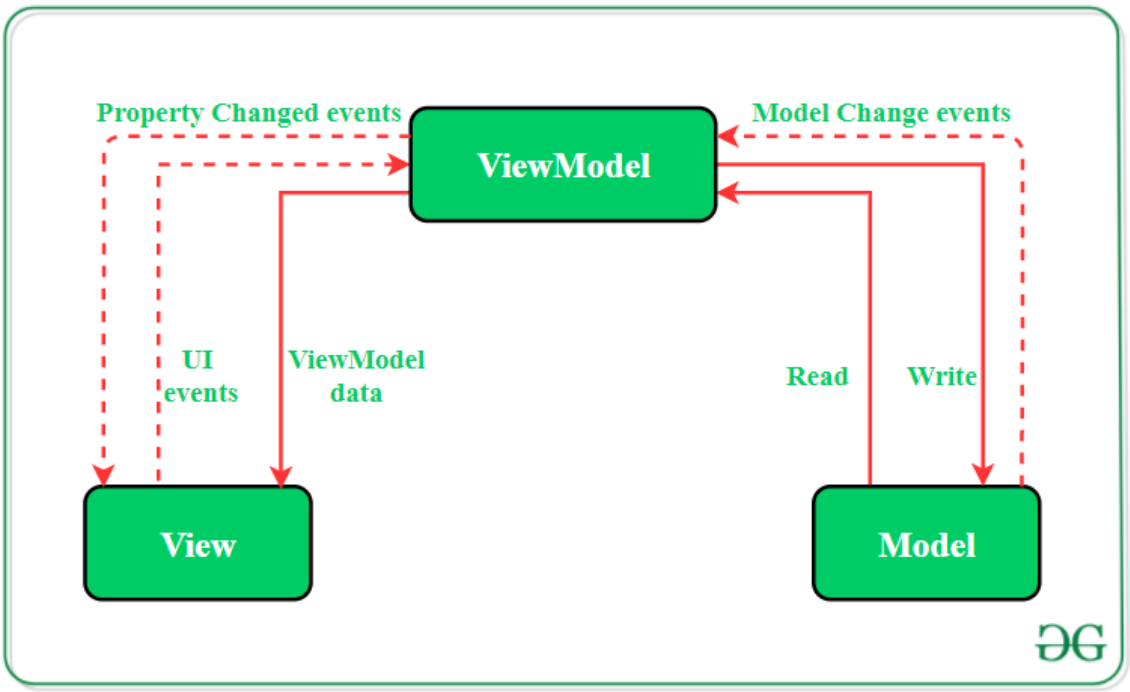

Рисунок 2.3 – UML-діаграма, що відображає взаємодію ключових компонентів архітектури MVVM

Підсумовуючи аналіз трьох архітектурних шаблонів, можна зробити висновок, що найбільш ефективним є використання MVC. Це пояснюється тим,

що він застосовує патерн спостерігача, який забезпечує синхронізацію та актуальність даних. Крім того, цей підхід сприяє розподілу програмного коду на окремі логічні частини, що полегшує його розуміння, тестування та подальшу підтримку.

2.2 Розробка WEB-ресурсу «Турагентсво» із застосуванням UML-діаграм

Розробка WEB-ресурса починається з фази проєктування, котра охоплює детальний опис майбутнього продукту. Головна мета на цьому етапі - чітко окреслити елементи системи, спроектувати навігацію між інтерфейсами користувачів, визначити класи для обміну інформацією, обдумати візуальну складову додатку, а також закласти фундамент для наступної фази - програмування.

Діаграми активності UML – це графічні засоби, що використовуються для моделювання динамічної поведінки систем. Вони дають можливість візуалізувати порядок дій, і їх можна виконувати та формально аналізувати за допомогою обчислення контекстно-залежних середовищ (CCA). Уніфікована мова моделювання (UML) визнана фактичним промисловим стандартом для моделювання програмних систем. Вона використовує набір графічних позначень, організованих у вигляді діаграм та активно застосовується в багатьох галузях. Незважаючи на переваги візуального представлення, UML не надає формального визначення семантики своїх діаграм, що обмежує її застосування в системах з підвищеними вимогами до безпеки, таких як медичні, авіаційні або енергетичні, де навіть незначні помилки можуть призвести до серйозних наслідків [10].

Однією з ключових діаграм UML є діаграма активності, яка використовується для представлення поведінкових аспектів систем. В цій роботі запропоновано формалізоване тлумачення діаграм активності UML на основі числення контекстно-залежних середовищ (CCA). Розроблено алгоритм - семантична функція, яка перетворює будь-яку UML-діаграму активності у

відповідний процес в межах CCA, який описує її поведінку. Цей процес можна реалізувати та формально перевірити, використовуючи інструменти моделювання та верифікації CCA, зокрема ccaPL та ccaRV. Це дозволяє знаходити помилки проектування на ранніх етапах розробки. Ефективність запропонованого підходу показано на прикладі системи електронної комерції.

Діаграми UML використовуються з різною метою, зокрема для:

- визначення й структурування функціональних вимог до системи;
- розподілу завдань та контролю за їх виконанням;
- проектування інформаційних систем;
- передачі проектної документації на етап розробки;
- моделювання основного сценарію взаємодії у випадках використання;
- візуалізації цілей взаємодії між користувачем і системою [11].

Серед найбільш поширених типів UML-діаграм можна виділити:

- діаграму випадків використання (прецедентів);
- діаграму класів;
- діаграму активностей;
- діаграму послідовностей;
- діаграму розгортання;
- діаграму об’єктів;
- діаграму станів.

Доцільно зупинитися на діаграмі випадків використання (рис. 2.4). Вона ілюструє взаємозв’язки між акторами (користувачами або зовнішніми системами) та прецедентами (сценаріями використання), які реалізуються у системі.

Діаграма прецедентів – це тип UML-діаграми, який відображає взаємозв’язки між користувачами (акторами) та функціональністю, яку надає система (прецедентами). Така діаграма дає змогу наочно представити, які саме дії можуть ініціювати користувачі та які сценарії поведінки система має підтримувати.

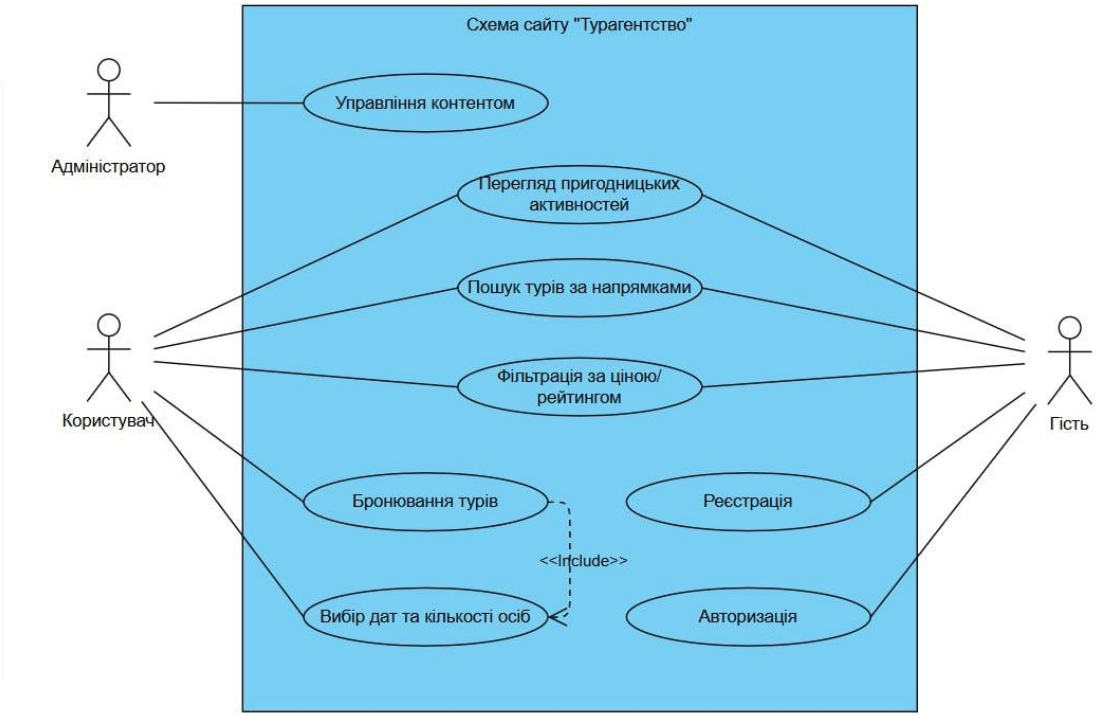

Рисунок 2.4 – UML-діаграма прецедентів

У структурі діаграми варіанти використання позначаються еліпсами, тоді як актори - умовними зображеннями людей у вигляді фігурки або палички. Усі прецеденти обмежуються рамкою, яка символізує межу системи, тобто те, що реалізується саме в її межах. Кожен прецедент пов’язується з актором лінією асоціації, яка вказує на ініціацію або взаємодію з відповідною функцією системи.

Діаграма прецедентів - це графічне зображення, що складається з акторів, прецедентів (варіантів використання), поміщених у межах системи, позначеної прямокутником. На ній відображено взаємозв'язки між акторами та прецедентами, залежності між самими варіантами використання, а також ієрархічні зв'язки між акторами. Ця діаграма візуально представляє структуру моделі прецедентів [12].

Основна мета діаграми прецедентів – показати взаємодію системи, що проектується із зовнішніми суб'єктами. Актори представляють собою користувачів або інші системи, що взаємодіють із програмним забезпеченням через визначені сценарії (варіанти використання). Кожен варіант описує певну функцію або послугу, яку система може запропонувати акторові. При цьому,

технічна реалізація такої взаємодії в моделі не розкривається – фокус зосереджено на функціональній поведінці системи [13].

Ці діаграми застосовуються на ранніх стадіях розробки як концептуальна основа, яка відображає загальний підхід до взаємодії користувачів із системою. Вони слугують фундаментом для подальшого створення більш деталізованих моделей, наприклад, логічних або фізичних схем системи [11–13].

Щоб варіант використання (прецедент) можна було включити до діаграми, він має відповідати таким вимогам:

- опис має зосереджуватись на тому, що саме виконується, без деталізації способу реалізації;

- дії мають бути подані з позиції користувача або зовнішнього актора системи;

- після виконання дій прецедент має надавати актору певний результат або повідомлення.

Діаграма діяльності (Activity Diagram) є важливим інструментом в UML, який дозволяє моделювати послідовність дій або бізнес-процесів у системі. Вона надає візуальне представлення потоку управління від однієї дії до іншої, включаючи можливі альтернативні та паралельні шляхи виконання.

Ці діаграми схожі на блок-схеми, але мають розширені можливості для моделювання складних процесів, таких як паралельне виконання та умовні переходи. Вони використовуються для опису динамічної поведінки системи, особливо коли потрібно показати, як різні дії координуються для надання певної послуги або досягнення цілі [14].

Основні елементи діаграми діяльності:

- Початковий вузол (Initial Node): позначає початок потоку дій.

- Дія (Action): представляє окрему операцію або крок у процесі.

- Потік управління (Control Flow): вказує напрямок переходу між діями.

- Умовний вузол (Decision Node): визначає розгалуження потоку на основі певної умови.

- Вузол злиття (Merge Node): об'єднує кілька альтернативних потоків в один.

- Вузол розділення (Fork Node): розділяє потік на кілька паралельних шляхів.

- Вузол з'єднання (Join Node): об'єднує паралельні потоки в один.

- Кінцевий вузол (Activity Final Node): позначає завершення всіх потоків у діаграмі.

Застосування діаграм діяльності:

- Моделювання бізнес-процесів: діаграми діяльності ефективно відображають послідовність дій у бізнес-процесах, дозволяючи виявити можливі покращення або вузькі місця.

- Аналіз сценаріїв використання: вони допомагають деталізувати сценарії використання, показуючи, як користувачі взаємодіють із системою.

- Проектування операцій класів: діаграми діяльності використовуються для візуалізації алгоритмів реалізації методів класів.

- Моделювання паралельних процесів: вони дозволяють зображати паралельне виконання дій, що є корисним при розробці багатопотокових систем [15, 16].

Діаграма класів (Class Diagram) в UML – це графічне подання структури системи, яке показує класи, їх властивості, методи та зв’язки між ними. Вона дозволяє створити модель об'єкта в єдиному, зрозумілому для всіх форматі, що забезпечує узгоджене бачення архітектури системи.

Як й інші діаграми UML, вона слугує ефективним засобом комунікації в команді розробників, а також між командою та замовником. Завдяки уніфікованому синтаксису така діаграма буде однаково зрозумілою незалежно від мови, країни чи рівня підготовки учасників проєкту.

Діаграма класів допомагає виявити загальні властивості об’єктів, згрупувати їх у класи та визначити логіку наслідування, що дозволяє уніфікувати обробку різних елементів системи. Це особливо корисно на етапі проєктування, коли потрібно закласти чітку основу для реалізації. UML-діаграми такого типу не лише описують структуру, але й стають частиною документації, яку можна підтримувати та оновлювати протягом усього життєвого циклу проєкту [14].

Наслідування: ієрархія класів для турів дозволяє уніфікувати їхню обробку. Базовий клас Tour містить спільні властивості (назва, ціна), а його спадкоємці (наприклад, AdventureTour чи FamilyTour) додають спеціалізовані функції.

Поліморфізм: інтерфейс Bookable може бути реалізований як турами, так і окремими активностями (рафтинг, дайвінг), що дозволяє їх обробляти єдиним чином у кошику.

Абстракція: використання абстрактних класів (наприклад, BaseFilter) для стандартизації фільтрації турів за ціною, рейтингом або типом активності.

Ключові класи серверної частини системи:

1. AuthController

Відповідає за безпеку та управління користувачами:

login(email: string, password: string) – автентифікація.

- register(userData: UserDto) – реєстрація нового клієнта.
- refreshToken(token: string) – оновлення сесії.

2. TourController

Надає API для роботи з туристичними пропозиціями:

- getPopularTours(limit: number) – список популярних турів.
- searchTours(keyword: string, filters: TourFilters) – пошук із фільтрацією.
- getTourDetails(id: number) – детальна інформація про тур.

3. BookingController

Керує бронюваннями:

- createBooking(bookingData: BookingRequest) – оформлення замовлення.
- cancelBooking(bookingId: number) – скасування.
- getUserBookings(userId: number) – історія бронювань клієнта.

4. AdminController

Закриті функції для адміністратора:

- addNewTour(tourData: TourDto) – додавання турів.
- updateTourAvailability(id: number, dates: DateRange) – оновлення розкладу.

Особливості реалізації:

- Поділ відповідальностей: кожен клас відповідає за окремий аспект роботи системи (наприклад, TourFilterService обробляє лише фільтрацію).

- TypeScript-інтерфейси: для опису DTO (наприклад, interface Tour { id: number; name: string; }) забезпечують узгодженість даних між клієнтом і сервером.

- Залежності (DI): класи отримують сервіси (наприклад, EmailService) через конструктори, що спрощує тестування.

UML-діаграма класів (рис. 2.5) ілюструє зв’язки між основними сутностями: користувачі, тури, бронювання та фільтри.

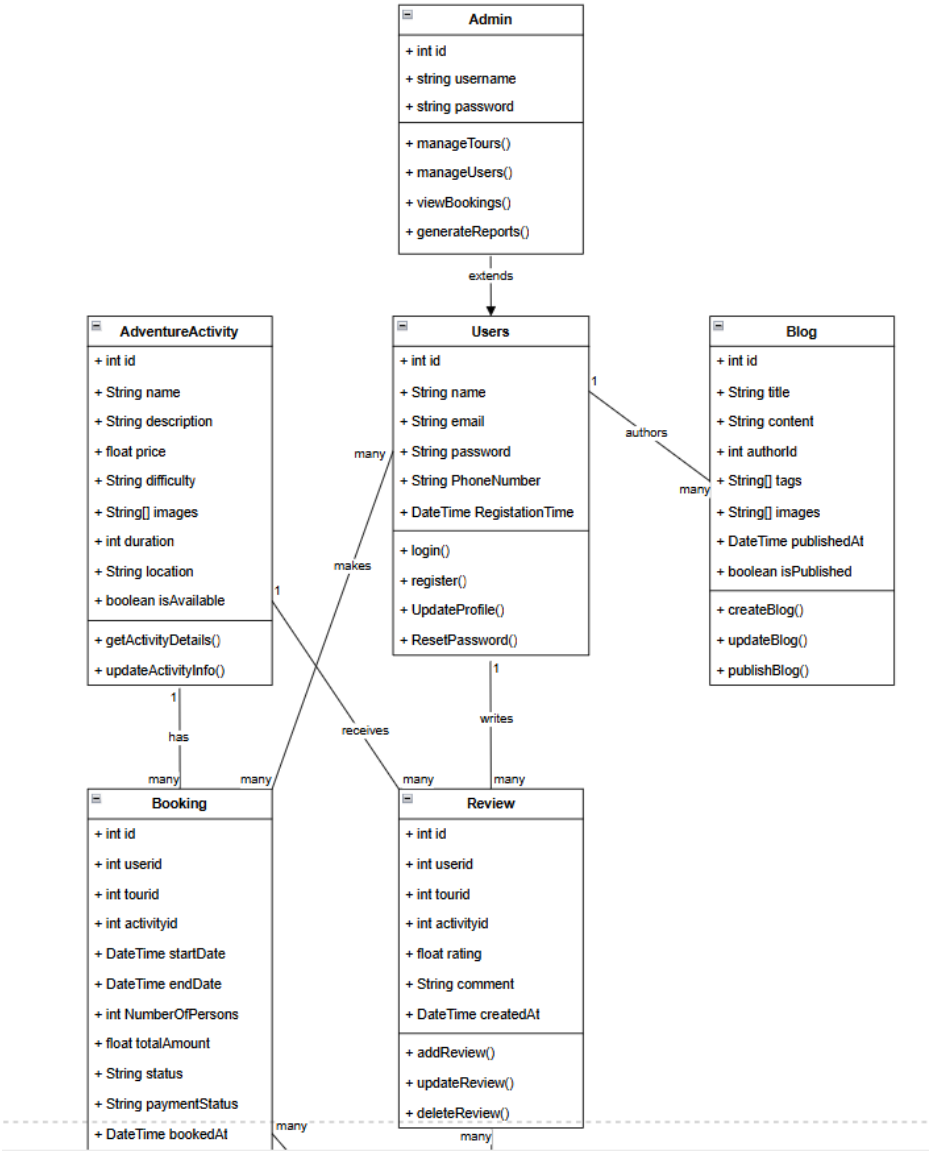

Рисунок 2.5 – UML-діаграма класів

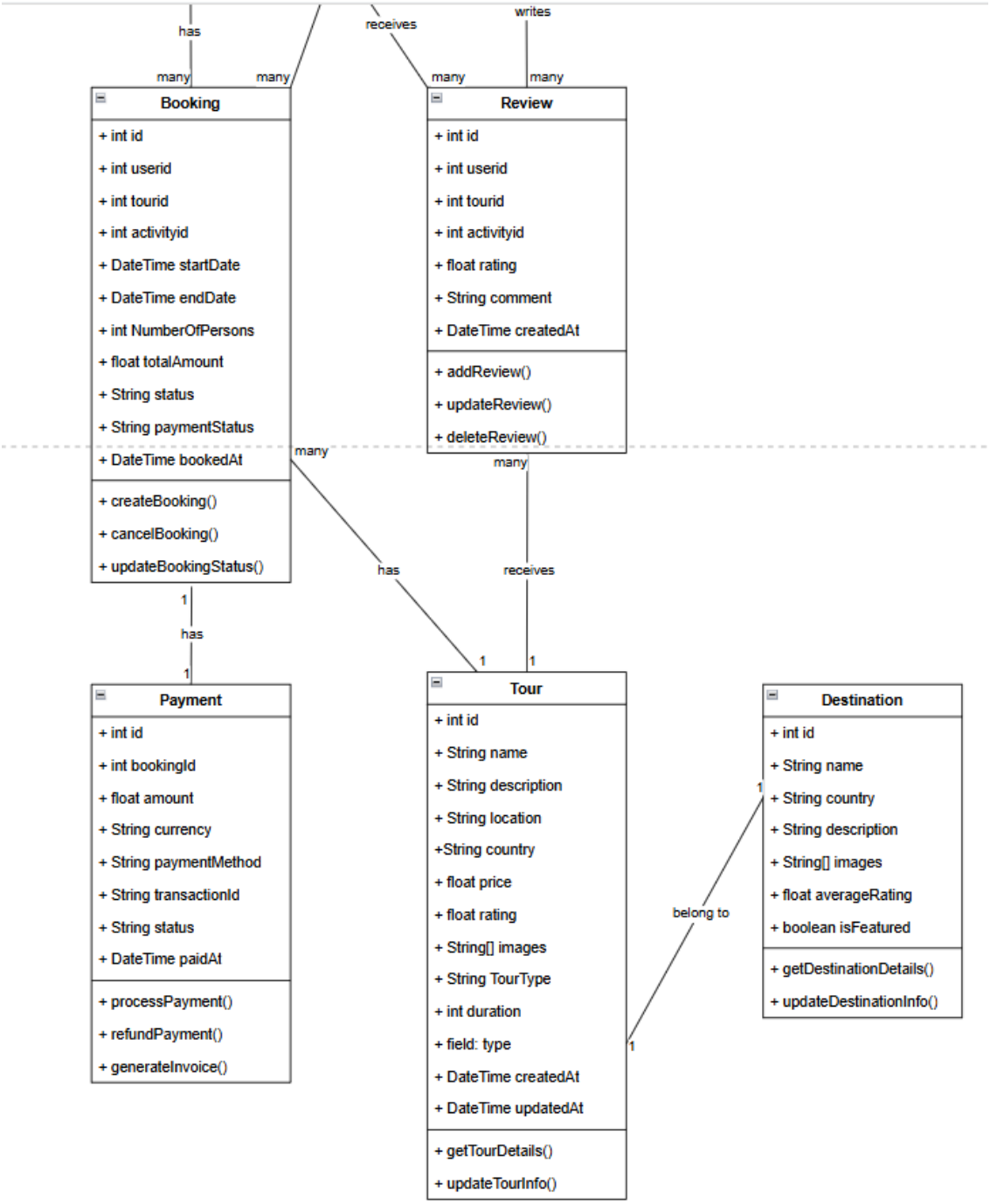

Рисунок 2.5, аркуш 2

У загальній частині програмного модуля для підбору турів є такі ключові класи: Admin, User, Tour, AdventureActivity, Booking, Payment, Review, Destination і Blog.

Клас Admin містить поля: id, username та password. Також містить в собі такі методи: manageTours(), manageUsers(), viewBooking та generateReports(). Метод manageTours дозволяє адміністратору керувати доступними турами, manageUsers

– управляти обліковими записами користувачів, viewBooking – переглядати всі бронювання, а generateReports – формувати звіти. Клас наслідує всі властивості й методи класу User.

Клас User містить такі поля: id, name, email, password, phoneNumber, registrationDate. Має методи: login(), register(), updateProfile() та resetPassword(). Метод login здійснює вхід користувача, register – реєструє нового користувача, updateProfile дозволяє редагувати профіль, а resetPassword – скидає пароль.

Клас Tour має такі поля: id, name, description, location, country, price, rating, images, tourType, duration, isActive, createdAt, updatedAt. Має лише такі методи: getTourDetails() і updateTourInfo(). Метод getTourDetails повертає детальну інформацію про тур, а updateTourInfo дозволяє оновлювати дані про тур.

Клас AdventureActivity включає в себе поля: id, name, description, price, difficulty, images, duration, location, isAvailable. Має лише такі методи: getActivityDetails() та updateActivityInfo(). Метод getActivityDetails надає дані про активність, а updateActivityInfo дозволяє редагувати інформацію про активність.

Клас Booking має такі поля: id, userId, tourId, activityId, startDate, endDate, numberOfPersons, totalAmount, status, paymentStatus та bookedAt. Такий метод як createBooking() створює нове бронювання, cancelBooking() – скасовує його, а updateBookingStatus() – оновлює його.

Клас Payment містить поля: id, bookingId, amount, currency, paymentMethod, transactionId, status і paidAt. Має такі методи: processPayment(), refundPayment() та generateInvoice(). Метод processPayment обробляє оплату, refundPayment – виконує повернення коштів, а generateInvoice – створює рахунок.

Клас Review містить поля: id, userId, tourId, activityId, rating, comment, createdAt. Має такі методи: addReview(), updateReview() та deleteReview(). Метод addReview створює новий відгук, updateReview – редагує існуючий, а deleteReview – видаляє його.

Клас Destination містить в собі такі поля: id, name, country, description, images, averageRating, isFeatured. Має лише такі методи: getDestinationDetails() та

updateDestinationInfo(). Метод getDestinationDetails надає детальну інформацію про місце призначення, а updateDestinationInfo дозволяє її оновлювати.

Клас Blog має поля: id, title, content, authorId, tags, images, publishedAt, isPublished. Має такі методи: createBlog(), updateBlog() та publishBlog(). Метод createBlog створює нову публікацію, updateBlog – редагує блог, а publishBlog – публікує його. На рисунку 2.6 подано UML-діаграма класів клієнтської частини модуля підбору турів.

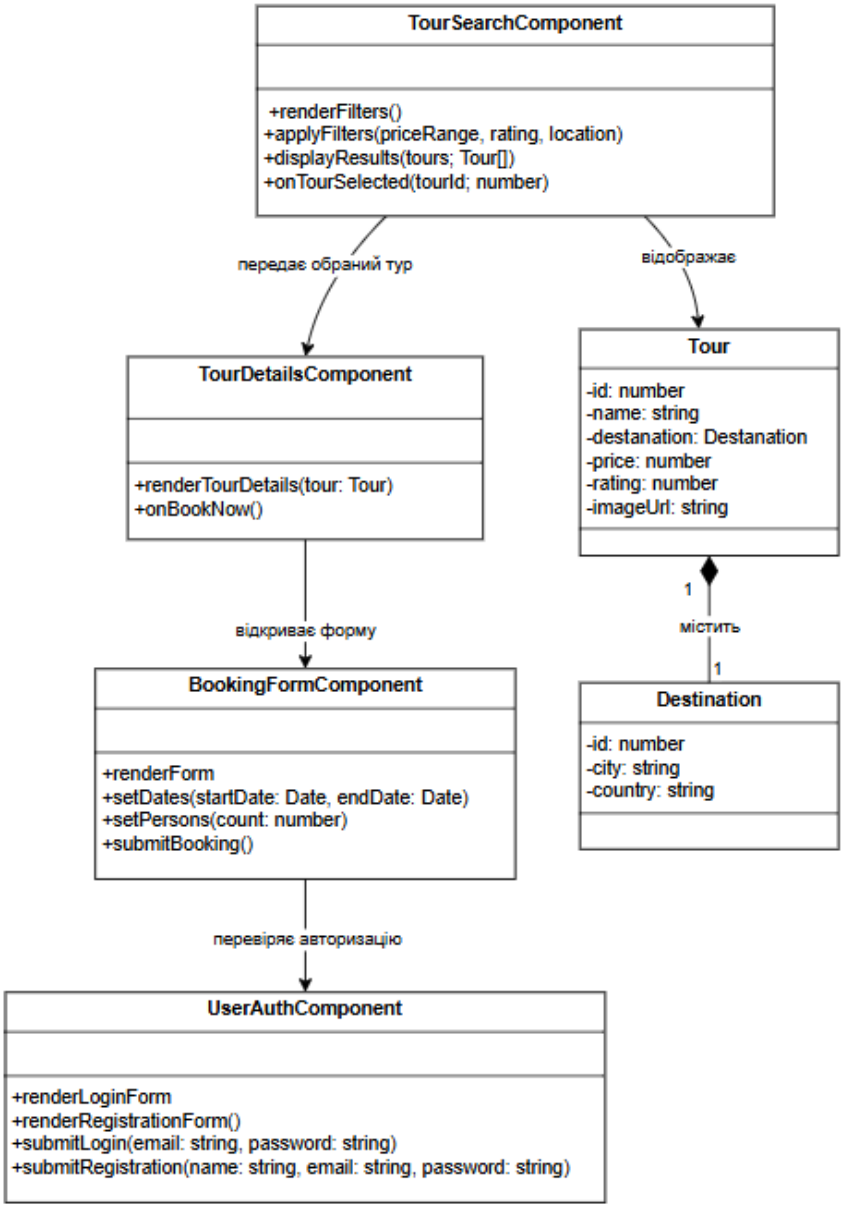

Рисунок 2.6 – UML-діаграма класів клієнтської частини модуля підбору турів

2.3 Розробка ER-моделі бази даних

Розроблена ER-модель бази даних призначена для структурованого представлення інформації, що використовується у роботі туристичного агентства. Її основна мета – забезпечити логічну організацію сутностей (наприклад, клієнти, тури, бронювання) та зв'язків між ними, щоб створити надійну, масштабовану та зрозумілу структуру даних для подальшої реалізації інформаційної системи.

ER-модель орієнтована як на потреби користувачів системи, так і на внутрішню організацію даних. Вона охоплює ключові сутності, такі як Клієнт, Тур, Бронювання, Менеджер та їх атрибути: ім'я, дата виїзду, тривалість, тип туру, ціна тощо. Модель також враховує відношення між сутностями, наприклад: клієнт може здійснювати кілька бронювань, кожне бронювання пов'язане з певним туром, менеджер додає або редагує тури.

Завдяки використанню ER-моделі забезпечується цілісність та узгодженість даних, спрощується фільтрація та пошук інформації (наприклад, вибір турів за критеріями), а також підтримується функціональність особистого кабінету клієнта та інструментів адміністратора. Це дозволяє ефективно будувати базу даних, що стане основою для повноцінного WEB-сервісу туристичного агентства.

Отже, ER-моделювання дозволяє ще на етапі проєктування системи створити чітке уявлення про дані та їхню організацію, що є критично важливим для надійної та ефективної роботи бази даних.

Для створення універсального відношення, яке охоплює сутності розроблюваної системи (користувачі, тури, активності, бронювання, платежі, відгуки, напрямки, блог-статті, адміністратори), необхідно включити основні атрибути з описаних класів. Універсальне відношення для бази даних туристичної платформи виглядатиме так: R (ім'я_користувача, пошта_користувача, пароль_користувача, номер_телефону_користувача, назва_туру, опис_туру, локація_туру, вартість_туру, рейтинг_туру, тривалість_туру, назва_активності, опис_активності, ціна_активності, рівень_складності_активності, тривалість_активності, дата_бронювання, кількість_учасників_бронювання,

статус_бронювання, сума_платежу, валюта_платежу, спосіб_оплати, статус_платежу, оцінка_відгуку, текст_відгуку, назва_напрямку, опис_напрямку, середній_рейтинг_напрямку, заголовок_блогу, текст_блогу, автор_блогу).

Ступінь відношення: 30 (оскільки враховано 30 атрибутів, що описують усі сутності системи).

У базі даних туристичної платформи визначено такі зв’язки між сутностями:

Зв’язок ОДИН-ДО-БАГАТЬОХ (1:Б) встановлено між такими сутностями:

- Користувачі-Бронювання – один користувач може створювати численні бронювання турів або активностей.
- Користувачі-Відгуки – один користувач здатен залишити кілька відгуків про тури чи активності.
- Користувачі-Блог-статті – один користувач може виступати автором багатьох статей у блозі.
- Тури-Бронювання – один тур може бути обраний для багатьох бронювань.
- Тури-Відгуки – один тур може мати численні відгуки від користувачів.
- Активності-Бронювання – одна пригодницька активність може бути включена до багатьох бронювань.
- Активності-Відгуки – одна активність може отримати багато відгуків від користувачів.
- Напрямки-Тури – один туристичний напрямок може охоплювати кілька турів.

Зв’язок БАГАТО-ДО-БАГАТЬОХ (Б:Б) використовується між сутностями:

- Бронювання-Платежі – кожне бронювання пов’язане з одним унікальним платежем.

Зв’язок БАГАТО-ДО-ОДНОГО (Б:1) використовується між сутностями:

- Тури-Напрямки - багато турів можуть належати до одного туристичного напрямку.

В таблиці 2.1 подано характеристики зв’язків між сутностями системи туристичного агентства.

Таблиця 2.1 – Характеристики зв’язків між сутностями системи туристичного агентства

Сутность 1	Сутность 2	Тип Зв’язку	Клас Належності
Користувачі	Бронювання	1:Б	Обов’язковий
Користувачі	Відгуки	1:Б	Необов’язковий
Користувачі	Блог-статті	1:Б	Необов’язковий
Тури	Бронювання	1:Б	Обов’язковий
Тури	Відгуки	1:Б	Необов’язковий
Активності	Бронювання	1:Б	Обов’язковий
Активності	Відгуки	1:Б	Необов’язковий
Напрямки	Тури	1:Б	Обов’язковий
Тури	Напрямки	Б:1	Обов’язковий
Бронювання	Платежі	1:1	Обов’язковий

Схематичне зображення ER-моделі бази даних, що використовується у програмному модулі WEB-ресурсу «Турагенство» наведено на рисунку 2.7.

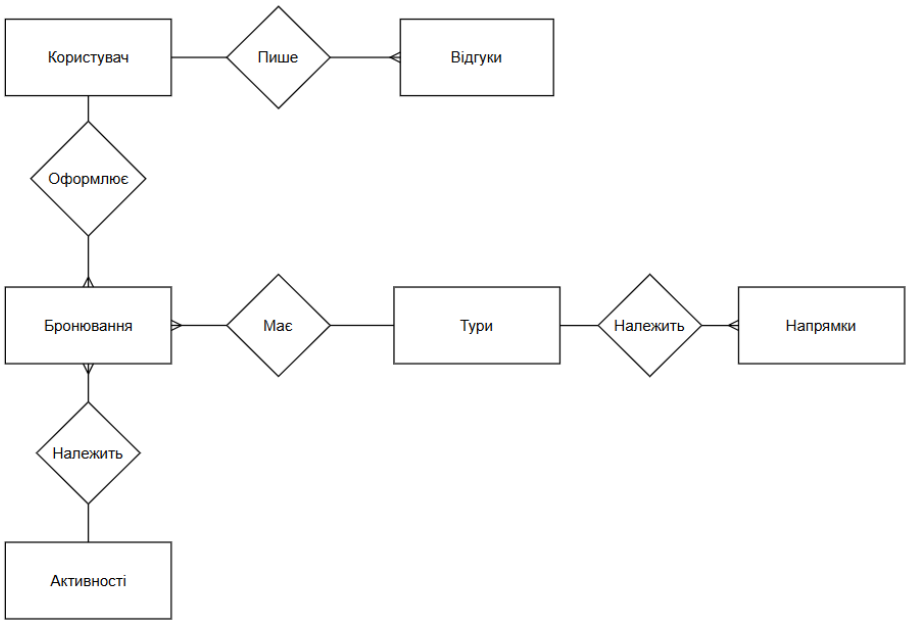

Рисунок 2.7 – ER-модель для бази даних WEB-ресурсу «Турагенство»

2.4 Розробка схеми алгоритму роботи WEB-ресурсу «Турагентство»

Розроблений WEB-ресурс призначений для забезпечення зручного онлайн-сервісу туристичного агентства. Його основна мета – надати користувачам можливість швидко та зручно знаходити туристичні пропозиції, переглядати доступні тури, здійснювати онлайн-бронювання, а також отримувати актуальну інформацію про популярні напрямки відпочинку.

WEB-ресурс орієнтований як на нових користувачів, так і на постійних клієнтів агентства. Він реалізує функціональність фільтрації турів за країною, датою виїзду, тривалістю, типом відпочинку (пляжний, екскурсійний, гірськолижний тощо) та вартістю. Також доступна реєстрація користувачів, формування персонального кабінету з історією замовлень, можливість зберігати обрані тури до «улюблених», залишати відгуки та ставити оцінки.

Адміністративна частина WEB-ресурсу дозволяє менеджерам агентства додавати нові тури, редагувати інформацію про напрямки, переглядати активність користувачів і формувати звіти. Це забезпечує ефективне керування контентом та полегшує комунікацію з клієнтами.

Завдяки використанню сучасної архітектури (наприклад, MVC або MVVM), WEB-ресурс має модульну структуру, що спрощує його підтримку, масштабування та подальший розвиток.

У цілому, створений WEB-ресурс сприяє цифровій трансформації туристичного бізнесу, забезпечуючи як клієнтам, так і співробітникам агентства комфортні та ефективні інструменти для взаємодії.

На рисунку 2.8 подана схема алгоритму роботи програмного модуля WEB-ресурсу «Турагенство».

Рисунок 2.8 – Схема алгоритму роботи програмного модуля WEB-ресурсу «Турагенство»

І випадок

Алгоритм реєстрації нового користувача складається з таких кроків (рис. 2.8):

Крок 1. Користувач відкриває головну сторінку WEB-ресурсу «Турагенство».

Крок 2. Користувач натискає кнопку «Реєстрація» у верхньому правому куті екрана.

Крок 3. Система відображає форму реєстрації з полями: ім'я, email та пароль.

Крок 4. Користувач вводить відповідні дані у поля форми реєстрації

Крок 5. Користувач натискає кнопку «Зареєструватися» для підтвердження реєстрації.

Крок 6. Система перевіряє коректність введених даних та унікальність email.

Крок 7. Якщо перевірка успішна – перехід до кроку 8, інакше – крок 9.

Крок 8. Система зберігає дані нового користувача в базу даних.

Крок 9. Система виводить повідомленні про помилку реєстрації користувача, перехід до кроку 3.

Крок 10. Система перенаправляє користувача на сторінку входу з повідомленням про успішну реєстрацію.

ІІ випадок

Алгоритм авторизації зареєстрованого користувача складається з таких кроків (рис. 2.8):

Крок 10. Користувач натискає кнопку «Авторизація» у верхньому правому куті екрана.

Крок 11. Система відображає форму авторизації з полями email та пароль.

Крок 12. Користувач вводить відповідні дані в поля форми авторизації.

Крок 13. Користувач натискає кнопку «Увійти» для підтвердження авторизації.

Крок 14. Система перевіряє відповідність email та пароля у базі даних.

Крок 15. Якщо є відповідність, то перехід до кроку 16, інакше – крок 12.

Крок 16. Система авторизує користувача.

Крок 17. Система перенаправляє авторизованого користувача в особистий кабінет.

III випадок

Алгоритм бронювання туру складається з таких кроків (рис. 2.8):

Крок 18. Користувач відкриває головну сторінку турагенства та переходить до розділу «Подорожі».

Крок 19. Користувач обирає бажаний тур зі списку доступних пропозицій і натискає кнопку «Забронювати».

Крок 20. Система відображає форму для бронювання туру з полями: кількість осіб, номер телефону, email.

Крок 21. Користувач вводить відповідні дані в поля форми бронювання туру.

Крок 22. Користувач підтверджує бронювання натисканням кнопки «Підтвердити замовлення».

Крок 23. Система перевіряє введені дані на коректність та відповідність в базі даних.

Крок 24. Якщо перевірка успішна, то перехід до кроку 25, інакше – крок 20.

Крок 25. Дані про бронювання зберігаються в базі даних.

Крок 26. Система надсилає підтвердження бронювання на email користувача.

Алгоритм всіх трьох випадків зображено на рисунку 2.8.

2.5 Висновок

У цьому розділі здійснено проєктування схеми алгоритму роботи WEB-ресурсу «Турагенство» з використанням архітектурного підходу MVC (Model–View–Controller). Обрана архітектура забезпечила чітке розділення відповідальностей між компонентами системи, що значно спрощує розробку, супровід та масштабування ресурсу.

У процесі проєктування було визначено основні функціональні елементи WEB-ресурсу «Турагенство»: форма авторизації, сторінка реєстрації користувачів, інтерфейс пошуку та перегляду турів, можливість створення спільних подорожей, а також механізм групового підбору турів. Кожен з цих компонентів було логічно розподілено між рівнями Model (обробка даних), View (відображення інформації) та Controller (керування логікою взаємодії).

З метою візуалізації структури системи було створено UML-діаграми класів для основних підсистем, які демонструють зв’язки між об’єктами, а також ключові методи, що забезпечують логіку роботи WEB-ресурсу «Турагенство». Крім того, побудована схема алгоритму роботи модуля підбору турів ілюструє послідовність виконання основних операцій з урахуванням запитів користувача та відповідей системи.

Також у рамках проєкту було створено ER-модель бази даних WEB-ресурсу «Турагентство». Вона дала змогу логічно організувати структуру збереження інформації, зокрема: турів, клієнтів, замовлень, країн та напрямків подорожей. Це значно полегшило подальше впровадження структури бази та її інтеграцію з серверною частиною.

Результати роботи дозволили не лише сформувати цілісне уявлення про структуру і функціонування розробки, але й створити надійну основу для подальшої реалізації, тестування та впровадження повноцінного інформаційного продукту у сфері туристичних послуг. Запропонований підхід сприяє ефективній організації розробки, знижує ризики помилок і забезпечує гнучкість системи для майбутніх удосконалень.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ WEB-РЕСУРСУ «ТУРАГЕНТСТВО»

3.1 Обґрунтування вибору мови та бібліотек програмування

Для створення WEB-ресурсу «Турагенство» необхідно з особливою відповідальністю підійти до вибору мов програмування та середовищ розробки. Вибір мови реалізації проєкту потребує ґрунтовного аналізу, адже від цього залежить ефективність і функціональність ресурсу. Крім того, важливо ретельно оцінити й обрати оптимальне середовище розробки, яке забезпечить зручність і продуктивність роботи.

Мова програмування – це штучно створена система символів, призначена для опису алгоритмів і структур даних. Вона слугує інструментом для формалізованого представлення алгоритмічних процесів, які може виконувати комп’ютер. Кожна мова програмування має власний набір лексичних, синтаксичних і семантичних правил, що визначають, як будуються програми та які дії виконує машина на основі вказаних інструкцій.

Перетворення програмного коду, написаного мовою програмування, на машинні команди виконує транслятор. Одну й ту ж мову програмування можуть реалізовувати різні транслятори, при цьому зберігаються її основні концепції та принципи, хоча можливі незначні відмінності в синтаксисі залежно від реалізації.

До мови програмування та середовища її використання висуваються ключові вимоги:

- Мова має підтримувати об'єктно-орієнтовану парадигму, оскільки створення великої за обсягом програми за допомогою процедурного підходу є складним і не відповідає сучасним вимогам розробки.

- Необхідно, щоб мова забезпечувала можливість використання готових модулів або компонентів, а також мала розвинену стандартну бібліотеку для спрощення реалізації функціональності.

- підтримка об'єктно-орієнтованого програмування, що забезпечує модульність, повторне використання коду та зручність масштабування;

- зручне та функціональне середовище розробки (IDE), яке підтримує автодоповнення коду, налагодження, тестування та інші інструменти для ефективної роботи програміста.

Для створення програмних модулів подібного рівня доцільно розглянути такі мови програмування, як Python, C# та JavaScript.

У сфері WEB-розробки широко використовуються спеціалізовані мови як для розмітки, так і для програмування на стороні клієнта чи сервера. Часто кілька таких мов поєднуються в рамках одного проєкту. Серед них JavaScript займає провідне місце завдяки своїй гнучкості та популярності, хоча кожна мова має власний рівень складності.

JavaScript (JS) –це легка за вагою мова програмування, яка інтерпретується або виконується з використанням JIT-компіляції (компіляції «на льоту»), та підтримує функції першого класу. Найбільш широке застосування JS знаходить як мова сценаріїв для WEB-сторінок, однак його також активно використовують в інших програмних середовищах, таких як Node.js або Apache CouchDB [17].

JavaScript є прототипно-орієнтованою, мультипарадигменною мовою з динамічною типізацією. Вона підтримує об’єктно-орієнтований, імперативний і декларативний (наприклад, функціональний) стилі програмування.

Ключові характеристики JavaScript:

Прототипне наслідування: на відміну від класичного класового наслідування, JavaScript використовує прототипи для організації спадковості. Кожен об'єкт може наслідувати властивості та методи від свого прототипу, що забезпечує гнучку систему повторного використання коду [18].

Динамічна та слабка типізація: змінні в JavaScript не мають фіксованого типу, що дозволяє змінювати їх значення на льоту. Це спрощує розробку, але вимагає уважності для уникнення помилок типів.

Автоматичне керування пам'яттю: завдяки вбудованому механізму збору сміття, JavaScript автоматично звільняє неактивну пам'ять, що зменшує ймовірність витоків пам'яті та полегшує управління ресурсами.

Функції як об'єкти першого класу: функції в JavaScript можуть бути присвоєні змінним, передані як аргументи або повернуті з інших функцій. Це дозволяє реалізовувати функціональні підходи, такі як замикання та каррінг.

Завдяки цим особливостям, JavaScript став основною мовою для розробки інтерактивних WEB-додатків, а також знаходить застосування в серверному програмуванні (наприклад, з використанням Node.js), мобільних додатках та інших сферах.

JavaScript – це потужна та універсальна мова програмування, яка відіграє ключову роль у сучасній WEB-розробці. Хоча її синтаксис є компактним, ефективне використання JavaScript вимагає глибокого розуміння його особливостей та парадигм програмування.

Переваги JavaScript:

- Універсальна підтримка в браузерах: усі сучасні WEB-браузери мають вбудовану підтримку JavaScript, що дозволяє розробникам створювати інтерактивні та динамічні WEB-сторінки без потреби в додаткових плагінах або розширеннях.
- Широке застосування та популярність: JavaScript є однією з найпопулярніших мов програмування у світі, що забезпечує велику спільноту розробників, численні ресурси для навчання та підтримку багатьох фреймворків і бібліотек.
- Підтримка багатьох парадигм програмування: мова підтримує об'єктно-орієнтоване, функціональне та імперативне програмування, що надає розробникам гнучкість у виборі стилю кодування.
- Постійний розвиток та вдосконалення: мова постійно оновлюється, з'являються нові стандарти та функціональні можливості, що дозволяє їй залишатися актуальною та відповідати сучасним вимогам розробки.

Завдяки цим перевагам, JavaScript залишається основним інструментом для створення сучасних веб-додатків та інтерактивних інтерфейсів користувача.

C# (вимовляється «сі-шарп») – це сучасна мова програмування, розроблена компанією Microsoft для платформи .NET. Вона поєднує в собі строгість статичної

типізації та безпечне управління пам'яттю, що робить її надійним інструментом для розробки широкого спектра застосунків, від веб-сервісів до мобільних і десктопних програм [19].

Мова програмування C# із самого початку розроблялася з орієнтацією на компонентний підхід, що є однією з її ключових переваг. Такий підхід дозволяє ефективно використовувати раніше створені модулі повторно в різних проєктах. Серед інших важливих характеристик мови потрібно відзначити такі аспекти:

- Мову C# було створено одночасно з платформою .NET Framework, тому вона повністю інтегрована з її функціоналом, як із бібліотекою класів FCL (Framework Class Library), так і з середовищем виконання CLR (Common Language Runtime).

- C# є справжньою об’єктно-орієнтованою мовою, у якій навіть базові типи даних реалізовані як об'єкти відповідних класів.

- Мова підтримує ключові парадигми об’єктного програмування, включаючи спадкування, інкапсуляцію, поліморфізм і узагальнення (generics), що забезпечує гнучкість і масштабованість коду.

- Синтаксично та концептуально C# базується на мовах C та C++, переймаючи їхню виразність і продуктивність, водночас усуваючи їхні недоліки.

Крім того, завдяки інтеграції з .NET Framework, яка виступає надбудовою над операційною системою, розробники C# мають подібні переваги до тих, що пропонує мова Java через використання віртуальної машини – стабільність, кросплатформеність і керованість виконання [19].

У таблиці 3.1 наведено основні відмінності між мовами програмування C# та JavaScript.

На основі аналізу даних таблиці 3.1 встановлено, що для створення WEB-ресурсу було прийнято рішення використовувати саме JavaScript. Хоча обидві мови мають свої сильні сторони, JavaScript виявився більш придатним для WEB-розробки завдяки простоті у застосуванні, широкій підтримці браузерів та численним сучасним інструментам. Саме ці переваги дозволяють ефективніше реалізовувати функціональність системи.

Таблиця 3.1 – Порівняння мов програмування C# та JavaScript

	C#	JavaScript
Основне середовище	ASP.NET / Blazor	Браузери (Chrome, Firefox і т.д.), Node.js
Підтримка ООП	+	+
Виконання	На сервері у середовищі .NET	На клієнті в браузері або на сервері (через Node.js)
Типізація	Статична, сувора	Динамічна, слабка
Продуктивність	Висока для серверних задач	Висока на фронтенді, середня на бекенді (залежить від рушія)
Інструменти розробки	Visual Studio, Rider	VS Code, WebStorm, браузерні DevTools
Фреймворки	ASP.NET Core, Blazor, Razor Pages	React, Angular, Vue, Express
Наявність UI	Потрібні сторонні рішення (Blazor, Razor, HTML/CSS)	Прямий доступ до DOM, широкі можливості UI
Сумісність з браузером	Потрібно знати .NET, C#, архітектуру MVC	Менш крута крива навчання, багато онлайн-ресурсів
Використання на фронтенді	Через Blazor WebAssembly (ще розвивається)	Основна мова для фронтенда
Залежність від сервера	Працює на сервері або компілюється у WebAssembly	Може працювати повністю на клієнті
Спільнота/ресурси	Сильна в .NET екосистемі, але менша у вебі	Дуже велика, активна у WEB-розробці

3.2 Обґрунтування вибору мови програмування для управління організованою базою даних

Дослідимо мови, які використовуються для керування впорядкованими базами даних: Object Query Language (OQL) та Structured Query Language (SQL).

Структурована мова запитів (SQL) – це стандартна мова для керування та запитування даних у реляційних системах баз даних. Вона широко

використовується як у промисловості, так і в академічних колах для зберігання, пошуку та маніпулювання даними, і є основою більшості сучасних програм та WEB-застосунків, керованих базами даних.

У реляційних базах даних інформація організовується у вигляді таблиць, де кожен стовпець відповідає певній характеристиці (атрибуту) об'єкта, а рядки представляють окремі записи. Для взаємодії з такими базами даних широко застосовується мова SQL (Structured Query Language), яка дозволяє виконувати різноманітні операції: вибірку, оновлення, додавання та видалення даних. Крім того, SQL підтримує створення нових таблиць, баз даних, представлень (views), збережених процедур та керування правами доступу [20, 21].

Класичні системи управління базами даних будувалися на основі реляційної моделі, у якій інформація представлена у вигляді взаємопов’язаних таблиць. У таких таблицях записи пов’язуються між собою за допомогою спільних ключів, що забезпечує логічні зв’язки між різними наборами даних. Об'єктно-орієнтована мова запитів (OQL) є розширенням структурованої мови запитів (SQL), адаптованою для роботи з об'єктними базами даних. Її основна функція полягає в абстрагуванні структури даних, що дозволяє користувачеві здійснювати доступ до інформації без необхідності знання внутрішньої організації схеми. При цьому синтаксис OQL зберігає подібність до SQL, що спрощує її освоєння.

Ключова різниця між мовами програмування SQL та OQL для роботи з базами даних полягає в тому, що OQL повністю базується на виразах. Зокрема, вираз SELECT у цій мові не є окремою інструкцією, як у SQL, а саме виразом, що дозволяє використовувати його у складених конструкціях та вкладених запитах. Такий підхід значно підвищує гнучкість мови й дає змогу створювати багаторівневі вкладені запити.

На відміну від SQL, OQL підтримує комбінування виразів у довільній формі, що дозволяє ефективно оцінювати навіть складні послідовності запитів. Водночас, при використанні об'єктно-орієнтованих методів OQL необхідно дотримуватись обережності, щоб уникнути небажаних змін даних або побічних ефектів. Оскільки OQL є декларативною мовою, порядок виконання окремих частин виразу не

завжди можна передбачити, що вимагає уважного підходу до проектування запитів.

Серед ключових переваг мови запитів до об'єктів (OQL) можна виділити такі:

- можливість здійснювати вибірку будь-яких об’єктів із бази даних незалежно від їх типу чи структури;
- здатність навігації по колекціях об’єктів без потреби повного завантаження всього масиву;
- підтримка механізмів представлення та відображення даних;
- відсутність жорсткої залежності від структури схеми бази даних;
- можливість виконання операцій над даними за допомогою методів, визначених у класах.

У таблиці 3.2 представлено порівняльний аналіз основних властивостей мов OQL (Object Query Language) і SQL (Structured Query Language).

Таблиця 3.2 – Порівняння ключових характеристик мов OQL (Object Query Language) та SQL (Structured Query Language)

Функції	SQL (Structured Query Language)	OQL (Object Query Language)
Тип даних	Дані організовані у вигляді таблиць (рядки та стовпці)	Дані представлені як об’єкти з властивостями та методами
Підхід до запитів	Використовує прості команди для фільтрації та обробки даних	Дозволяє працювати зі складними структурами та посиланнями
Основні операції	Додавання, читання, оновлення та видалення записів (CRUD)	Маніпулювання об’єктами, включаючи їх зв’язки та поведінку
Типовий застосунок	Використовується в традиційних базах даних (MySQL, Oracle)	Застосовується в об’єктно-орієнтованих СКБД (MongoDB, db4o)
Обробка зв’язків	З’єднання таблиць через ключі (JOIN)	Об’єкти посилаються один на одного без явних зовнішніх ключів

Продовження таблиці 3.2

Функції	SQL (Structured Query Language)	OQL (Object Query Language)
Уникнення дублів	Потрібна нормалізація для мінімізації повторень	Автоматично уникає дублювання завдяки механізму посилань
Де використовується	WEB-сайти, фінансові системи, CRM	Ігрові рушії, графічні програми, складні моделі даних

Рішення щодо вибору між SQL та OQL залежить від вимог конкретного проекту. Для роботи з реляційними базами даних, де важлива простота та продуктивність, перевагу потрібно віддати SQL. Саме тому було обрано мову SQL для реалізації.

3.3 Обґрунтування вибору середовища розробки

Інтегроване середовище розробки (IDE – Integrated Development Environment) є програмним інструментом, що забезпечує розробників усім необхідним для створення програм. Воно об’єднує низку корисних функцій, зокрема редактор коду, засоби для виявлення та виправлення помилок, систему керування версіями, а також механізми для збірки та тестування проєкту. Завдяки комплексному підходу IDE значно полегшує щоденну роботу програміста, підвищуючи ефективність написання та підтримки програмного забезпечення [22].

Під час аналізу варіантів для вибору середовища розробки були розглянуті такі інструменти: Visual Studio Code (VS Code), WebStorm та Atom.

Visual Studio Code - це швидкий та зручний редактор програмного коду, розроблений компанією Microsoft, який функціонує на операційних системах Windows, macOS та Linux. Редактор поєднує легкість роботи з широким набором інструментів, що робить його ідеальним вибором як для початківців, так і для досвідчених розробників [23].

У VS Code вже з коробки доступна підтримка таких мов програмування, як JavaScript, TypeScript і Node.js, а також є можливість легко додати підтримку інших мов і технологій – C, C#, Python, PHP, Java, Go, .NET тощо – завдяки великій бібліотеці розширень (extensions), які можна встановити безпосередньо з вбудованого магазину.

WebStorm – це сучасне інтегроване середовище розробки (IDE), розроблене компанією JetBrains для зручної та продуктивної розробки WEB-застосунків. Воно підтримує такі технології, як JavaScript, TypeScript, HTML, CSS, а також популярні фреймворки й бібліотеки, включно з React, Angular, Vue.js та Node.js. WebStorm пропонує розробникам розумне автозаповнення коду, ефективні інструменти для налагодження та рефакторингу, а також інтеграцію з системами контролю версій і базами даних. Платформа доступна на операційних системах Windows, macOS і Linux, що дозволяє створювати сучасні WEB-застосунки з високою продуктивністю і зручністю [24].

Atom – це безкоштовний і відкритий редактор коду, створений компанією GitHub, який широко використовується для фронтенд-розробки. Він позиціонується як гнучкий «хакерський» редактор, що легко адаптується під індивідуальні потреби завдяки великій кількості плагінів і тем оформлення. Atom підтримує численні мови програмування, включаючи HTML, CSS, JavaScript, що робить його зручним інструментом для розробників інтерфейсів. Крім того, редактор має інтуїтивний інтерфейс та глибоку інтеграцію з Git і GitHub, що значно полегшує контроль версій і спільну роботу над проектами. Atom доступний на платформах Windows, macOS і Linux, забезпечуючи кросплатформену підтримку [24].

У таблиці 3.3 представлено порівняльний аналіз ключових особливостей інтегрованих середовищ розробки Visual Studio Code (VS Code), WebStorm та Atom.

Таблиця 3.3 – Порівняння функціональних можливостей Visual Studio Code (VS Code), WebStorm і Atom

	Visual Studio Code (VS Code)	WebStorm	Atom
Ліцензія	Вільний	Платний	Вільний
Швидкодія	Оптимізований для швидкої роботи, малий споживач ресурсів	Потужний, але вимагає більше системних ресурсів	Може сповільнюватись при великій кількості додатків
Інтерфейс користувача	Мінімалістичний, гнучко налаштовується	Професійний, з багатофункціональними панелями	Простий, але з обмеженими можливостями
Підтримка JavaScript	Повноцінна підтримка (ES6+, модулі, асинхронність) через вбудовані інструменти та розширення	Глибока інтеграція з підтримкою сучасних стандартів, оптимізація коду	Базова підтримка, розширювана через плагіни
Робота з TypeScript	Вбудована підтримка (запуск, налагодження)	Повний набір інструментів для розробки	Часткова підтримка через плагіни

Згідно з проведеним аналізом (табл. 3.3), Visual Studio Code демонструє найкращу підтримку JavaScript серед розглянутих середовищ розробки, що стало вирішальним фактором для його вибору.

3.4 Розробка клієнтської та серверної частин

У процесі створення WEB-ресурсу були використані такі технології, як мови програмування SQL, HTML, CSS та JavaScript. Для реалізації функціоналу надсилання електронних листів було застосовано бібліотеку PHPMailer, а для полегшення стилізації та забезпечення адаптивного дизайну інтерфейсу використовувався фреймворк Bootstrap.

Розробка проєкту розпочиналася з формування його модульної архітектури. Основним елементом системи є файл index.php, який виконує функцію стартової точки програми. Саме з нього здійснюється підключення ключових компонентів: Application, Model та Translator, що відповідають за ініціалізацію основної логіки, взаємодію з даними та мовну підтримку.

При створенні WEB-ресурсу особливу увагу було приділено модулю відображення туристичних напрямків. Його реалізація поєднує простоту, зручність та ефективність. Нижче наведено основний фрагмент коду, що відповідає за відображення інформації про курорти:

```
function display(data) {
    cont.innerHTML = "";
    for (let i = 0; i < data.length; i++) {
        let anchor = document.createElement("a");
        anchor.setAttribute("href", "./destination.html");
        anchor.addEventListener("click", function() {
            let destination = [];
            destination.push(data[i]);
            localStorage.setItem("destination",
JSON.stringify(destination));
        });
        let card = document.createElement("div");
        let image = document.createElement("img");
        image.setAttribute("src", data[i].image);
        let name = document.createElement("h5");
        name.innerText = data[i].name;
        let location = document.createElement("h4");
        location.innerText = data[i].location;
        let box = document.createElement("div");
        let rating = document.createElement("h6");
        rating.innerText = `★ ${Number(data[i].rating) / 20}`;
        let price = document.createElement("h6");
        price.innerText = `₴ ${Number(data[i].price) * 10} / Person`;
        box.append(rating, price);
```

```
        card.append(image, name, location, box);
        anchor.append(card);
        cont.append(anchor);
    }
}
```

Функція display(data) у discover.js відповідає за рендеринг карток з курортами на основі отриманих даних. Вона не використовує маршрутизацію, але схожа за логікою на передачу даних у шаблони:

```
function display(data) {
    cont.innerHTML = "";
    for (let i = 0; i < data.length; i++) {
        let anchor = document.createElement("a");
        anchor.setAttribute("href", "./destination.html");
        anchor.addEventListener("click", function() {
            localStorage.setItem("destination", JSON.stringify(data[
i]));
        });
        // ... рендеринг картки ...
    }
}
```

Клас Model виконує роль моделі даних, відповідаючи за зчитування, перевірку та збереження JSON-файлу, який містить інформацію про страви. Такий підхід дозволяє зручно оновлювати або змінювати дані без потреби у використанні повноцінної системи управління базами даних (СУБД). Під час ініціалізації класу здійснюється перевірка цілісності файлу та правильності його формату:

```
class Model {
    private static $dish_file = 'data/dishes.json';
    private static $dishes_data = [];

    public static function init() {
        if (!is_file(self::$dish_file)) {
            throw new Error('Файл з базою страв не знайдено');
        }
        $data = file_get_contents(self::$dish_file);
```

```
            self::$dishes_data = json_decode($data, true);

            if (json_last_error() !== JSON_ERROR_NONE) {
                throw new Error('Файл JSON пошкоджений');
            }
        }
        public static function getDishes() {
            return self::$dishes_data;
        }
    }
```

Цей фрагмент HTML-коду створює інтерфейс для фільтрації, сортування та пошуку об'єктів за основними критеріями: ціною, рейтингом та місцем розташування. У верхній частині інтерфейсу розміщено дві кнопки з ідентифікаторами increase та decrease, які відповідають за сортування об'єктів за ціною – відповідно за зростанням та спаданням. Поруч із ними знаходиться текстове поле з ідентифікатором searchBar, яке дозволяє користувачам здійснювати пошук за місцем розташування.

Нижче розміщений блок фільтрації, що знаходиться в елементі з ідентифікатором box. Усередині нього є контейнер filter, який містить два випадаючі списки. Перший, з ідентифікатором priceSelect, дозволяє обрати межу вартості об'єктів: нижче 500, 700 або 1000. Другий список, ratingSelect, відповідає за фільтрацію за рейтингом – користувач може вибрати об'єкти з рейтингом нижче 2 або нижче 4.

Загалом цей інтерфейс забезпечує простий і зрозумілий механізм керування відображенням об'єктів на основі важливих параметрів, що покращує зручність користування сайтом:

```
<button id="increase">Price -⬆□</button>
      <button id="decrease">Price -⬇□</button>
      <input type="text" id="searchBar" placeholder="Search for
Location" />
    </div>
    <div id="box">
```

```
<div id="filter">
  <label for="">filter by Price range</label>
  <select name="Price" id="priceSelect">
    <option value="">Select price range</option>
    <option value="500">below 500</option>
    <option value="700">below 700</option>
    <option value="1000">below 1000</option>
  </select>
  <label for="Rating">filter by Rating range</label>
  <select name="" id="ratingSelect">
    <option value="">Select Rating</option>
    <option value="2">below 2</option>
    <option value="4">below 4</option>
```

3.5 Тестування роботи програми та аналіз результатів

Тестування розробки становить важливий етап у процесі створення програмного забезпечення, оскільки безпосередньо впливає на якість, надійність та продуктивність майбутнього продукту. Систематичні перевірки дозволяють значно скоротити витрати на подальше виправлення недоліків, забезпечити високий рівень клієнтської лояльності та дотримання всіх встановлених норм і вимог.

У ході створення WEB-ресурсу «Турагенство» було здійснено комплексне тестування основних функціональних можливостей. Розроблений WEB-ресурс був ретельно протестований, із проведенням понад 50 тестових запусків. Після активації сервісу першим екраном, з яким стикається користувач, є сторінка входу в систему, де можна авторизуватися під своїм обліковим записом. Візуальне оформлення цього інтерфейсу представлено на рисунку 3.1.

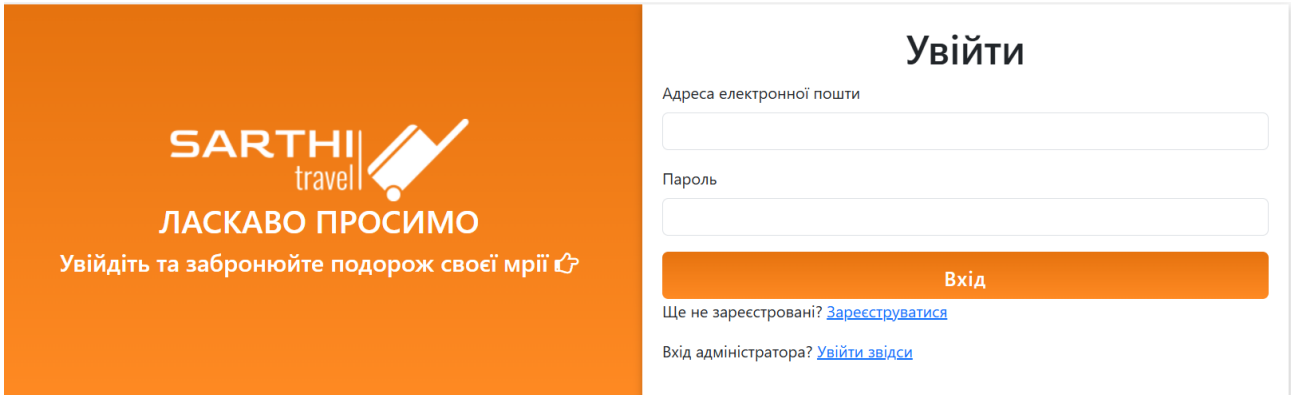

Рисунок 3.1 – Інтерфейс сторінки авторизації користувача на WEB-ресурсі «Турагенство»

Після натискання кнопки «Register» користувач переходить на сторінку реєстрації. Інтерфейс цієї сторінки показано на рисунку 3.2.

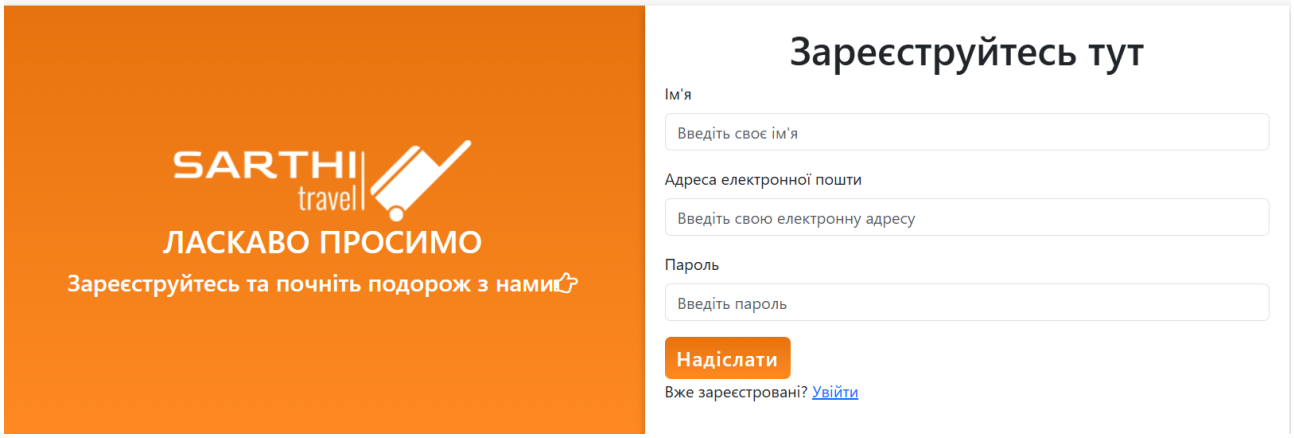

Рисунок 3.2 – Інтерфейс сторінки реєстрації користувача на WEB-ресурсі«Турагенство»

Після проходження процедури входу або створення облікового запису, користувач автоматично потрапляє на головну сторінку, де відображаються актуальні туристичні тренди на поточний день, про нас, види туризма, блоги а також найпопулярніші напрямки за останній тиждень (рис. 3.3, 3.4, 3.5, 3.6).

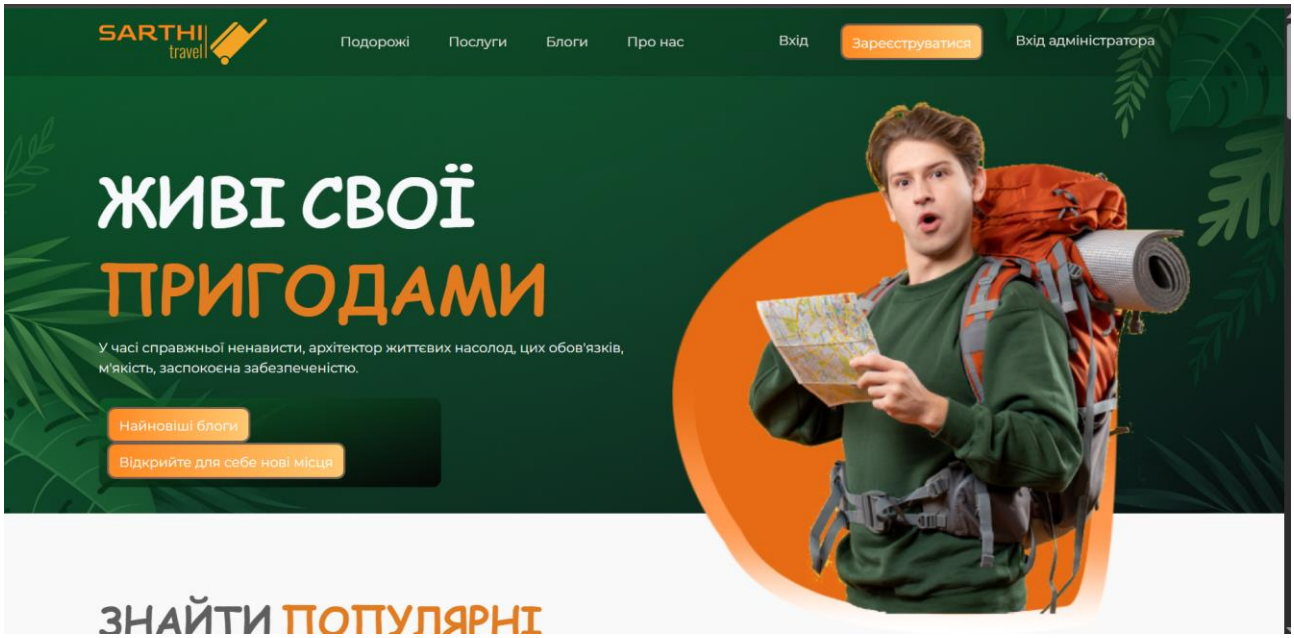

Рисунок 3.3 – Інтерфейс головної сторінки на WEB-ресурсі «Турагенство»

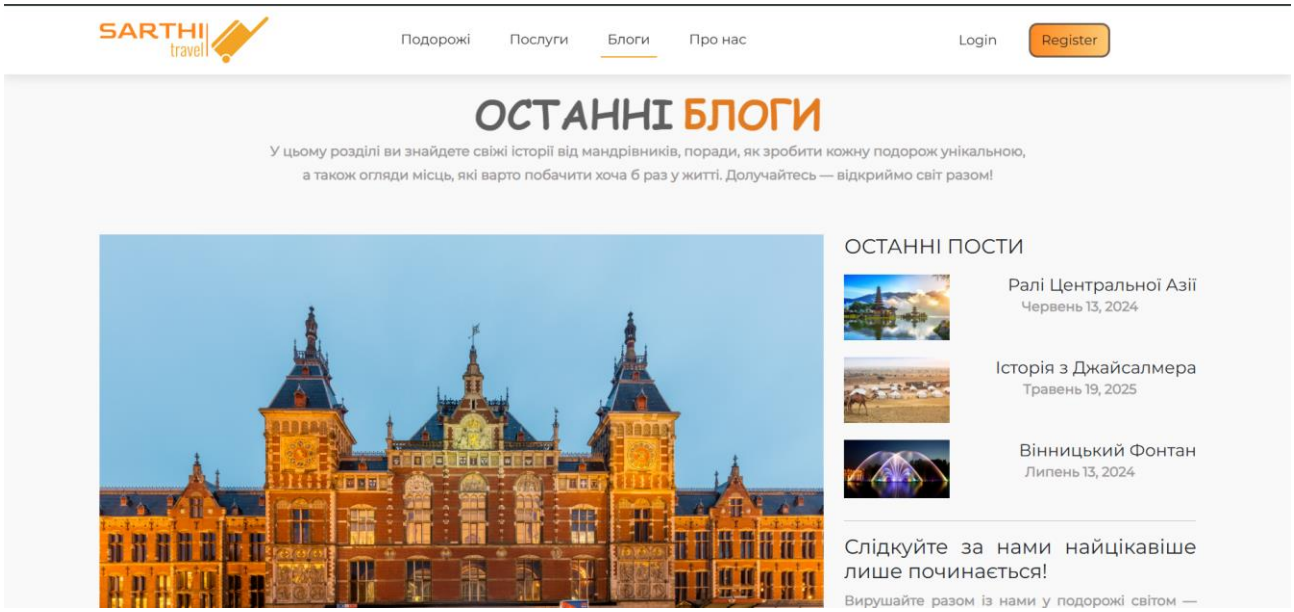

Рисунок 3.4 – Інтерфейс сторінки блоги на WEB-ресурсі «Турагенство»

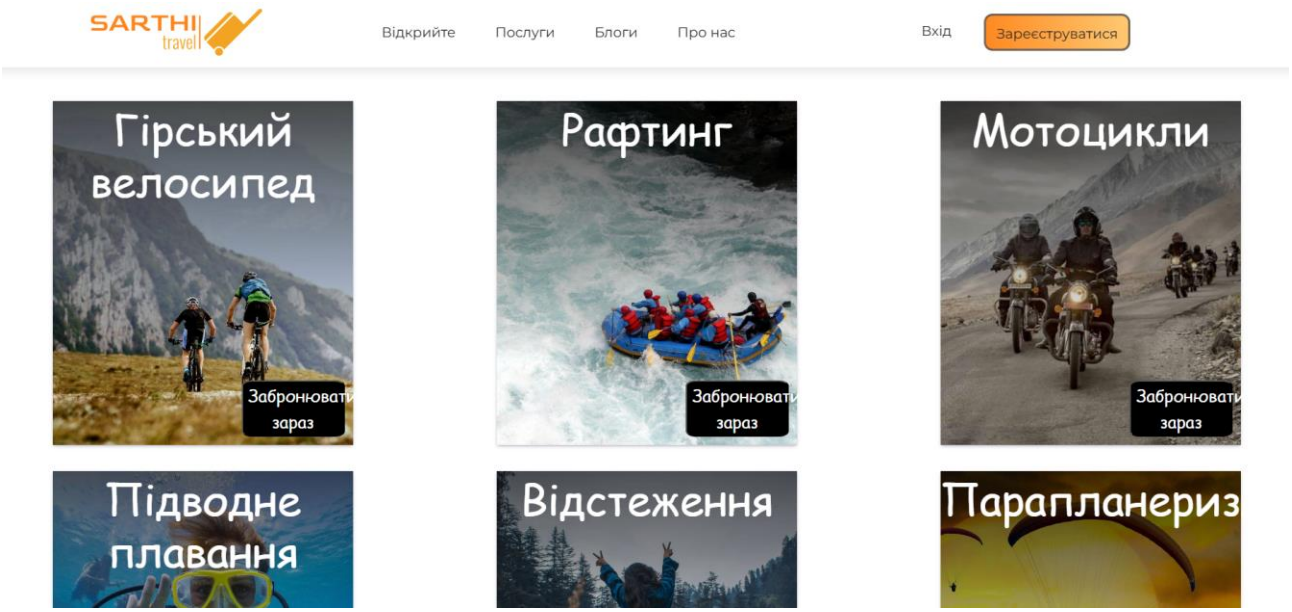

Рисунок 3.5 – Інтерфейс сторінки види туристичних активностей на WEB-ресурсі «Турагенство»

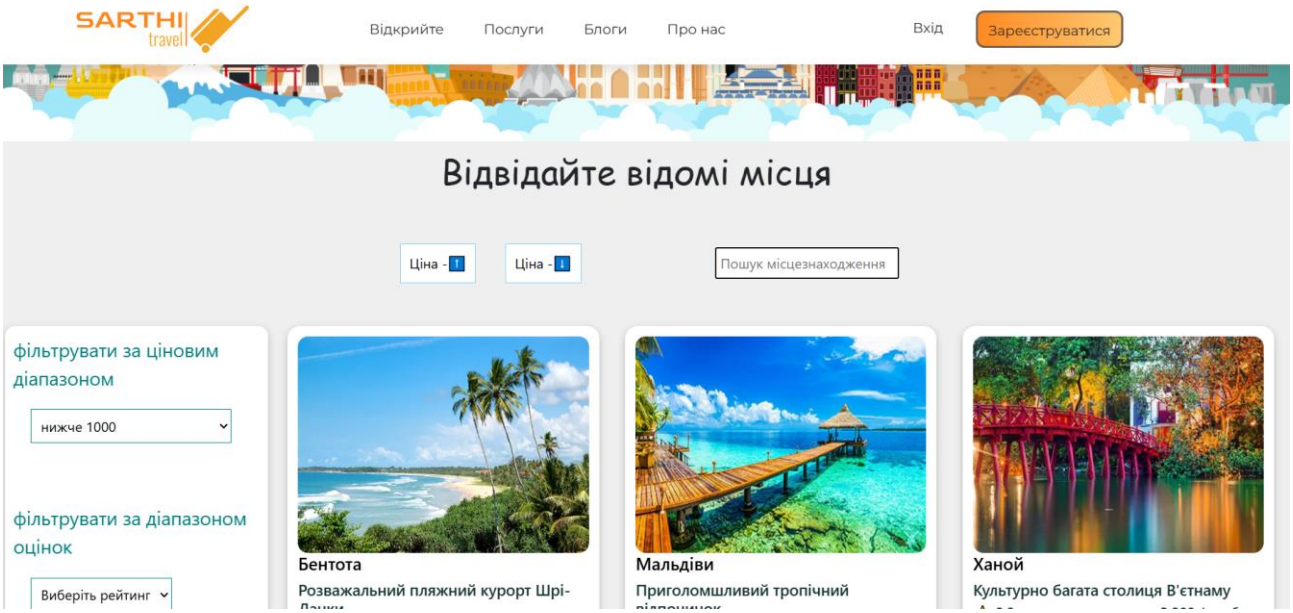

Рисунок 3.6 – Інтерфейс сторінки запропоновані тури на WEB-ресурсі «Турагенство»

Користувач може ініціювати створення нової туристичної групи для подорожі. Для цього потрібно ввести назву туру та додати інших користувачів, які братимуть участь у бронюванні. Візуальне представлення цих сторінок WEB-ресурсу«Турагенство» показано на рисунку 3.7.

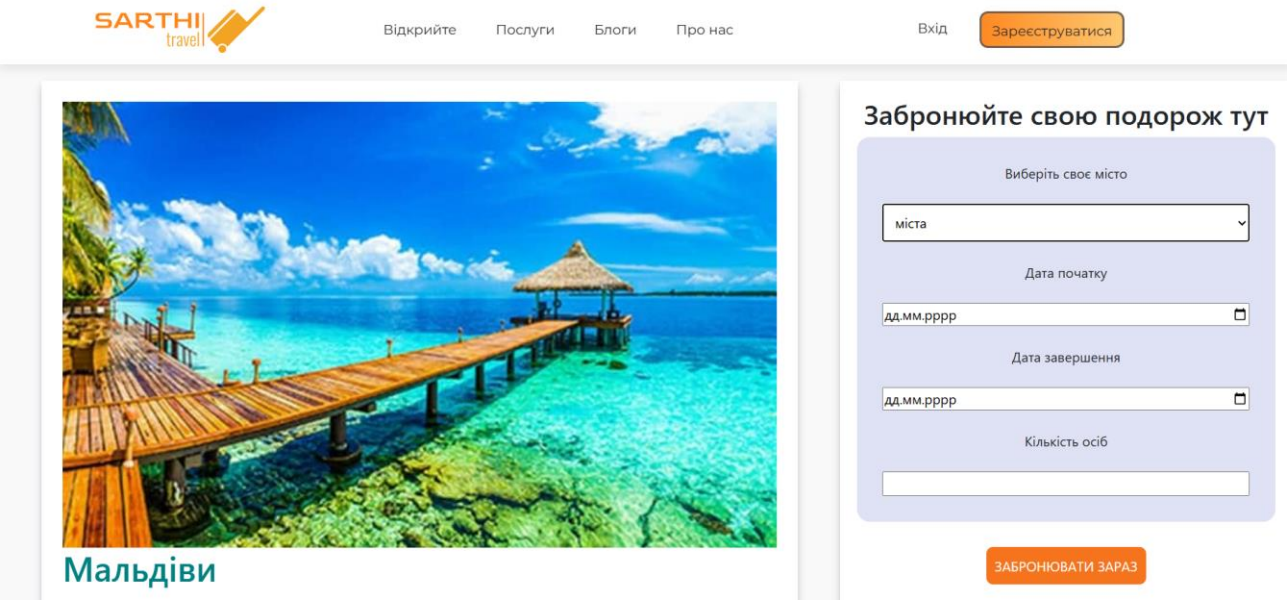

Рисунок 3.7 – Інтерфейс сторінки для бронювання поїздки

Адміністратор має можливість перейти до адмін-панелі, де відображаються всі створені ним тури. У цьому розділі можна редагувати інформацію про тур, видалити його або переглянути деталі для подальшого керування бронюваннями. Інтерфейс сторінки з управління турами продемонстровано на рисунках 3.8 та 3.9.

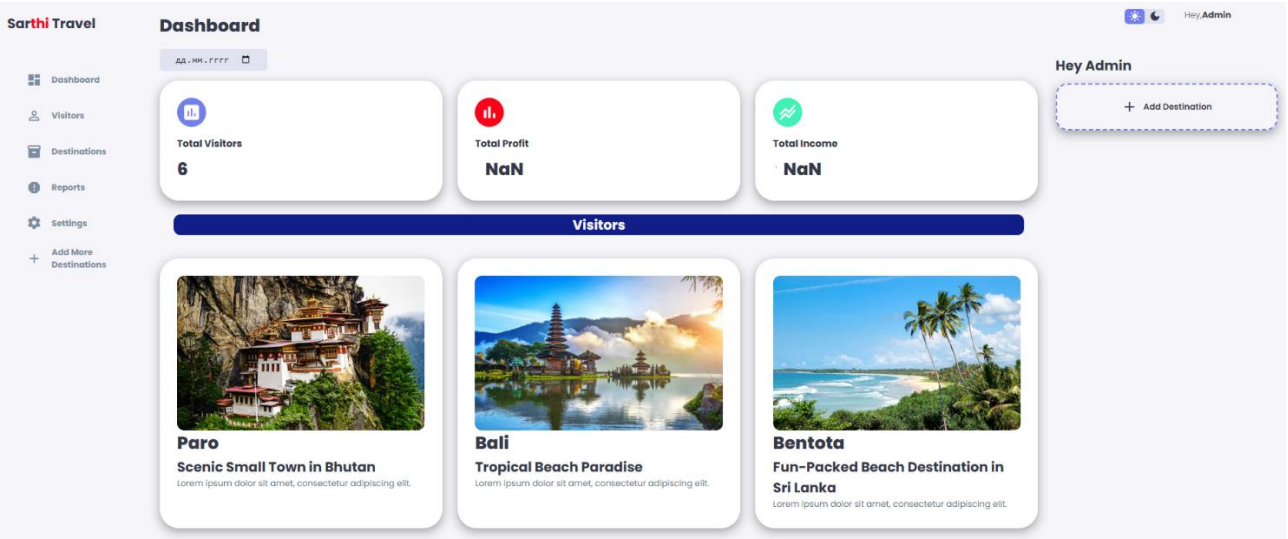

Рисунок 3.8 – Інтерфейс адмінської сторінки управління сайтом

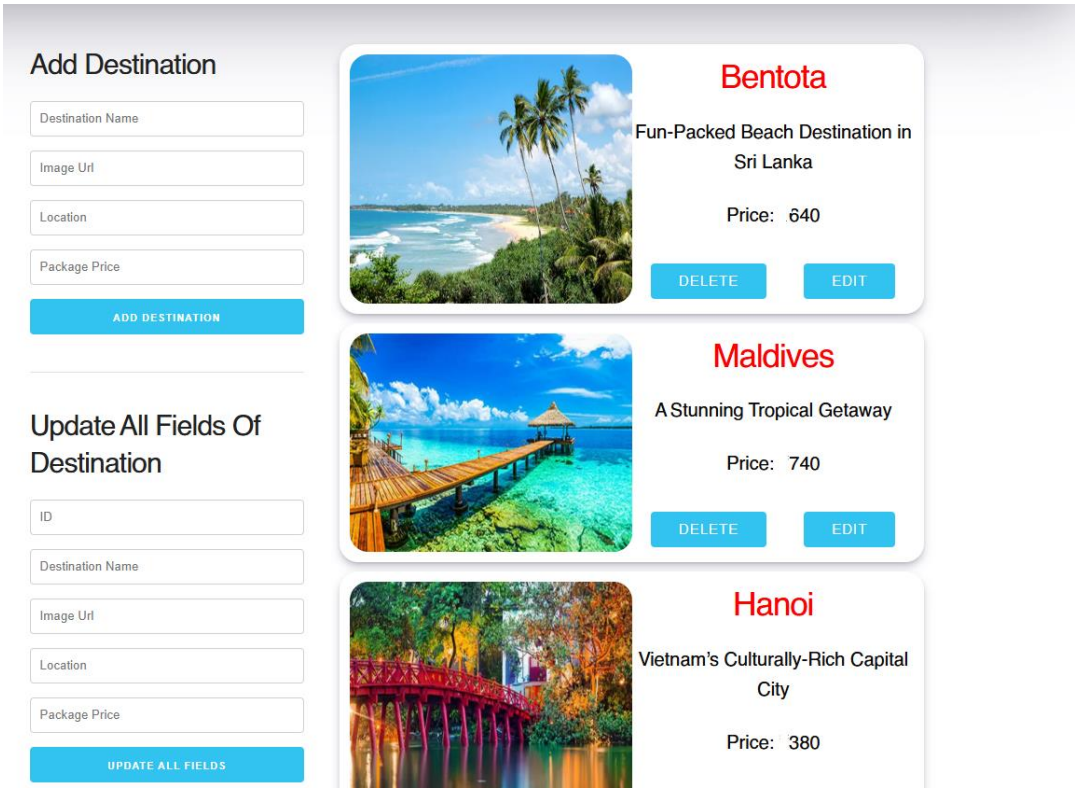

Рисунок 3.9 – Інтерфейс сторінки додавання турів в адмін-панелі

Розроблений WEB-ресурс для туристичного агентства був успішно протестований та повністю відповідає всім заданим вимогам. У процесі перевірки було підтверджено стабільну роботу ключових компонентів: відображення турів, їх додавання та редагування через адміністративну панель, а також можливість перегляду та бронювання пропозицій користувачами. Інтерфейс ресурсу інтуїтивно зрозумілий, що забезпечує зручну взаємодію як для клієнтів, так і для адміністраторів.

В таблиці 3.4 подано результати тестування розробленого програмного продукту та аналогів.

На основі даних таблиці 3.4 можна зробити висновок, що розроблений WEB-ресурс має 4 ключові переваги порівняно з аналогами, зокрема:

- персоналізований вибір турів, це дає змогу користувачам швидше знаходити тури, що відповідають їхнім уподобанням;

- можливість створення списків улюблених турів, що дозволяє зберігати вподобані варіанти для подальшого аналізу чи поділу з іншими;

- українська локалізація, що робить його зручним для користувачів з України;

- зручність інтерфейсу, це дозволяє користувачам швидше орієнтуватися та ефективніше використовувати функціонал ресурсу.

Таблиця 3.4 – Результати тестування розробленого програмного продукту та аналогів

Функціональна можливість	Розроблений ресурс	Expedia	KAYAK	Анекс Тур
Персоналізований вибір турів	+	+	-	-
Детальна інформація про тури	Відсутня	Присутня	Присутня	Відсутня
Створення списків улюблених турів	+	-	-	-
Українська локалізація	Присутня	Відсутня	Відсутня	Присутня
Зручність інтерфейсу	Простий в використанню	Присутня реклама	Не локалізований	Інтуїтивний і простий
Точність цін	Присутня	Присутня	Присутня	Присутня

Таким чином, розроблений WEB-ресурс «Турагентство» повністю відповідає поставленим технічним вимогам, забезпечує зручний користувацький досвід та може бути використаний для реального впровадження в туристичному бізнесі.

3.6 Висновок

У цьому розділі було проведено аналіз мов програмування, які можна використати для створення WEB-ресурсу «Турагенство». Зокрема, розглянуто JavaScript та C# для реалізації клієнтської частини. В результаті порівняння особливостей обох мов, їхніх переваг та сфер застосування було зроблено вибір на

користь JavaScript. Ця мова є лідером у сфері WEB-розробки, має велику спільноту, підтримку сучасних фреймворків, а також дозволяє ефективно працювати з інтерфейсом користувача.

Крім того, було виконано порівняльний аналіз мов запитів до баз даних – SQL та OQL. У результаті аналізу перевага була надана SQL, оскільки вона є універсальним і найпоширенішим інструментом для роботи з реляційними базами даних, що є доцільним у контексті управління даними про тури, клієнтів, бронювання та інші об’єкти.

Для реалізації та тестування розробки обрано середовище розробки Visual Studio Code. Це рішення обґрунтовано його зручністю, продуктивністю, широкими можливостями розширення, підтримкою Git, вбудованим терміналом та інтелектуальним автодоповненням. Додатковою перевагою стала кросплатформеність і відкритий вихідний код.

Готовий WEB-ресурс пройшов етап тестування: перевірялися основні сценарії користувача, такі як перегляд турів, авторизація, фільтрація напрямків, а також робота адміністративної частини сайту – додавання нових турів, редагування наявних записів тощо. Тестування підтвердило працездатність ключових функцій. Розроблений WEB-ресурс має розширений функціонал на 4 можливостей порівняно з аналогами, зокрема: можливість створення списків улюблених турів, персоналізований вибір турів, українська локалізація, зручність інтерфейсу.

ВИСНОВКИ

Всі задачі, поставлені для реалізації WEB-ресурсу «Турагенство» були виконані в повному обсязі, а саме: обґрунтувана доцільність розробки WEB-ресурсу «Турагентство», враховуючи сучасні тенденції розвитку ринку туризму; спроектовано WEB-ресурс «Турагентство»; програмно реалізувано WEB-ресурс «Турагентство»; виконано тестування WEB-ресурсу «Турагентство» та проведено аналіз отриманих результатів; розроблено інструкцію користувача.

У результаті виконання бакалаврської кваліфікаційної роботи було розроблено WEB-ресурс «Турагентство», який дозволяє автоматизувати основні процеси бронювання турів, перегляду пропозицій та керування інформацією про клієнтів.

Під час розробки було проведено аналіз сучасних мов програмування, зокрема JavaScript та C#, для реалізації клієнтської частини ресурсу, а також SQL та OQL – для побудови логіки взаємодії з базою даних. На основі порівняльного аналізу було обрано JavaScript для фронтенду та SQL для управління реляційною структурою даних.

Як середовище розробки було використано Visual Studio Code, що забезпечило зручність розробки завдяки широкому набору розширень, підтримці автозаповнення, налагодження, інтеграції з Git та високій швидкодії.

Побудована ER-модель бази даних дала змогу чітко визначити основні сутності системи та їх взаємозв’язки, що спростило подальшу реалізацію функціоналу.

Розроблений програмний продукт успішно пройшов тестування коректності роботи, а порівняльний аналіз із відомими аналогами показав розширений функціонал на 4 можливості.

Мета роботи, яка полягала в розширенні функціональних можливостей WEB-ресурсу «Турагенство», була успішно досягнута завдяки впровадженню 4 нових функцій, які суттєво підвищують зручність користування, доступність сервісу та ефективність адміністрування. Зокрема, було реалізовано:

україномовний інтерфейс, що робить ресурс зрозумілим і доступним для широкого кола користувачів з України; персоналізований вибір турів дає змогу користувачам швидше знаходити пропозиції, які максимально відповідають їхнім уподобанням; можливість створення списків улюблених турів дозволяє зберігати вподобані варіанти для подальшого аналізу або обміну з іншими; зручність інтерфейсу допомагає швидше орієнтуватися та ефективно використовувати весь функціонал ресурсу.

Робота виконана у повному обсязі відповідно до поставлених цілей, вимог та методичних рекомендацій щодо підготовки бакалаврських кваліфікаційних робіт. Усі етапи були реалізовані належним чином, що підтверджує якість, завершеність і практичну цінність розробленого рішення [25].

Таким чином, розроблений WEB-ресурс «Турагенство» повністю задовольняє вимоги до сучасної системи онлайн-бронювання турів, є зручним у використанні як для клієнтів, так і для працівників турагентства, а також має потенціал для подальшого розширення та інтеграції з іншими сервісами.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Кравчук А. О., Крилик Л. В. Аналіз передумов розробки WEB-ресурсу «Турагентство». *LIV Всеукраїнська науково-технічна конференція факультету інтелектуальних інформаційних технологій та автоматизації*. URL: https://conferences.vntu.edu.ua/index.php/all-fksa/all-fksa-2025/paper/view/23326/19294 (дата звернення:17.05.2025).

2. Туроператор і Турагент: Поняття, Різниця, Обов'язки. URL: https://tourkazka.com/turoperator-i-turahent-ponyattya-obovyazky/ (дата звернення: 17.05.2025).

3. Що таке роздрібна туристична агенція? Ось як вони діють. URL: https://mize.tech/blog/what-is-a-retail-travel-agency-this-is-how-they-operate/ (дата звернення: 17.05.2025).

4. Expedia. URL: https://www.expedia.com/ (дата звернення: 17.05.2025).

5. KAYAK. URL: https://www.ua.kayak.com/ (дата звернення: 17.05.2025).

6. Annex Tours. URL: https://anex-tour.com.ua/uk/ (дата звернення: 17.05.2025).

7. HIGHLOAD. URL: https://highload.tech/uk/blogs/shho-take-mvc-ta-mvp-patterni/ (дата звернення: 17.05.2025).

8. Make use of. URL: https://www.makeuseof.com/mvc-mvp-mvvm-which-choose/#:~:text=The%20three%20most%20popular%20design,%2C%20view%2C%20and%20view%20model (дата звернення: 17.05.2025).

9. Geeksforgeeks. URL: https://www.geeksforgeeks.org/difference-between-mvc-mvp-and-mvvm-architecture-pattern-in-android/ (дата звернення: 17.05.2025).

10. Maxym Zosym. URL: https://www.maxzosim.com/unifikovana-mova-modeluvannia/ (дата звернення: 17.05.2025).

11. Основи UML. URL: https://docs.kde.org/trunk5/uk/umbrello/umbrello/uml-basics.html (дата звернення: 17.05.2025).

12. DOU. URL: https://dou.ua/forums/topic/40575/ (дата звернення: 17.05.2025).

13. Geeksforgeeks. URL: https://www.geeksforgeeks.org/use-case-diagram/ (дата звернення: 17.05.2025).

14. Evergreens. URL: https://evergreens.com.ua/ua/articles/uml-diagrams.html (дата звернення: 17.05.2025).

15. GURU99. URL: https://www.guru99.com/uk/uml-activity-diagram.html (дата звернення: 17.05.2025).

16. Елементи UML.
URL: https://docs.kde.org/trunk5/uk/umbrello/umbrello/uml-elements.html (дата звернення: 27.05.2024).

17. Biz. URL: http://ni.biz.ua/6/6_3/6_33177_diagramma-deyatelnosti-activity-diagram.html (дата звернення: 27.05.2025).

18. Javascript info. URL: https://uk.javascript.info/prototype-inheritance (дата звернення: 27.05.2025).

19. Beetroot academy. URL: https://beetroot.academy/blog/shcho-take-c-chi-pidhodit-meni-cya-mova-programuvannya-chomu-vona-kruta (дата звернення: 17.03.2025).

20. SQL Tutorial. URL: https://www.geeksforgeeks.org/sql-tutorial/ (дата звернення: 27.05.2025).

21. Medium. URL: https://medium.com/academy-team/the-easiest-way-to-create-queries-in-sql-44124a64e2a5 (дата звернення: 27.05.2025).

22. W3schoolsUA. URL: https://w3schoolsua.github.io/hyperskill/ide.html#gsc.tab=0 (дата звернення: 27.05.2025).

23. TopKeis. URL: https://top-keis.com.ua/shho-take-visual-studio-code/ (дата звернення: 27.05.2025).

24. Jetbrains. URL: https://www.jetbrains.com/webstorm/ (дата звернення: 27.05.2025).

25. Методичні вказівки до виконання бакалаврських кваліфікаційних робіт для студентів спеціальності 122 «Комп’ютерні науки» [Електронний ресурс] / уклад.: А. А. Яровий, О. В. Сілагін, І. В. Богач. Вінниця : ВНТУ, 2023. (PDF, 54 с.).

ДОДАТКИ

ДОДАТОК А (обов'язковий)
Протокол перевірки кваліфікаційної роботи

Назва роботи: Розробка WEB-ресурсу «Турагентство»

Тип роботи: бакалаврська кваліфікаційна робота
(бакалаврська кваліфікаційна робота / магістерська кваліфікаційна робота)

Підрозділ кафедра комп'ютерних наук
(кафедра, факультет, навчальна група)

Коефіцієнт подібності текстових запозичень, виявлених у роботі системою StrikePlagiarism 6,81 %

Висновок щодо перевірки кваліфікаційної роботи (відмітити потрібне)

✓ Запозичення, виявлені у роботі, є законними і не містять ознак плагіату, фабрикації, фальсифікації. Роботу прийняти до захисту

☐ У роботі не виявлено ознак плагіату, фабрикації, фальсифікації, але надмірна кількість текстових запозичень та/або наявність типових розрахунків не дозволяють прийняти рішення про оригінальність та самостійність її виконання. Роботу направити на доопрацювання.

☐ У роботі виявлено ознаки плагіату та/або текстових маніпуляцій як спроб укриття плагіату, фабрикації, фальсифікації, що суперечить вимогам законодавства та нормам академічної доброчесності. Робота до захисту не приймається.

Експертна комісія:

Яровий А. А., зав. каф. КН _____
(прізвище, ініціали, посада) (підпис)

Колесницький О. К., проф. каф КН _____
(прізвище, ініціали, посада) (підпис)

Особа, відповідальна за перевірку _____ Озеранський В. С.
(підпис) (прізвище, ініціали)

З висновком експертної комісії ознайомлений(-на)

Керівник _____ Крилик Л. В.
(підпис) (прізвище, ініціали, посада)

Здобувач _____ Кравчук А. О.
(підпис) (прізвище, ініціали)

ДОДАТОК Б (обов'язковий)
Лістинг програми

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>SARTHI TRAVEL</title>
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha3/dist/css/bootstrap.min.css" rel="stylesheet">
  <link rel="icon" type="image/x-icon" href="image/Sarthi-favicon.png">
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/animate.css/4.1.1/animate.min.css" />
  <link rel="preconnect" href="https://fonts.googleapis.com">
  <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
  <link href="https://fonts.googleapis.com/css2?family=Montserrat&family=Righteous&family=Tilt+Prism&display=swap" rel="stylesheet">
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css">
  <link rel="stylesheet" href="css/owl.carousel.min.css">
  <link rel="stylesheet" href="css/owl.theme.default.min.css">
  <script src="IndexScript/jquery.min.js"></script>
  <script src="IndexScript/owl.carousel.js"></script>
  <link rel="stylesheet" href="css/index.css">
</head>
<body>
  <!--****************************** navbar ******************-->
  <nav class="navbar navbar-expand-lg fixed-top topColor" id="navBar">
    <div class="container">
      <a class="navbar-brand" href="index.html">
        <img src="image/logo-no-background.png" alt="" class="logo">
      </a>
      <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation">
        <span class="navbar-toggler-icon"></span>
      </button>
      <div class="collapse navbar-collapse float-xl-end" id="navbarSupportedContent">
        <ul class="navbar-nav mx-auto mb-2 mb-lg-0">
          <li class="nav-item">
            <a class="nav-link" aria-current="page" href="./discover.html">Подорожі</a>

          </li>
          <li class="nav-item">
            <a class="nav-link" href="service.html">Послуги</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="blog.html">Блоги</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="about.html">Про нас</a>
          </li>
        </ul>
        <ul class="navbar-nav mx-auto mb-2 mb-lg-0">
```

```
            <li class="nav-item">
              <a class="nav-link" href="userLoginReg/login.html" id="logdiv">Логін</a>
              <!-- <a class="nav-link" href="userLoginReg/login.html">Logout</a> -->
            </li>
            <li class="nav-item">
              <a class="btn btn-primary reg-btn" href="userLoginReg/registration.html" id="registerDiv"></a>
            </li>
            <li class="nav-item">
              <a class="nav-link" href="login/adminlogin.html" id="logdiv" target="_blank">Адмін логін</a>
              <!-- <a class="nav-link" href="userLoginReg/login.html">Logout</a> -->
            </li>
          </ul>
        </div>
      </div>
    </nav>
    <!--****************************** navbarEnd ******************-->
    <div class="d-flex heroSection">
      <div class="container">
        <div class="row">
          <div class="align-self-center text-left text-light col-md-6">
            <div class="lc-block mb-4">
              <div editable="rich">
                <h1 class="display-1 fw-bolder wow animate__animated animate__fadeInDown">ЖИВИ СВОЇМИ</h1>
                <h1 class="display-1 fw-bolder second-h1 wow animate__animated animate__fadeInDown">ПРИГОДАМИ</h1>
                <p class="paraBox">У часи справжньої ненависті, архітектор життєвих насолод, цих обов'язків, м'якість, заспокоєна необхідністю. </p>
              </div>
            </div>
            <div class="searchBox">
              <!-- <i class="fa fa-map-marker" aria-hidden="true"></i>
              <select name="" id="">
                <option value="">Location</option>
                <option value=""></option>
              </select>   
              <i class="fa fa-calendar" aria-hidden="true"></i>
              <input type="date" name="" id="">
              <button class="btn btn-primary reg-btn">Search</button> -->
              <div class="indexBtn">
                <a href="blog.html" class="btn btn-primary reg-btn">Найновіші блоги</a>
                <a href="discover.html" class="btn btn-primary reg-btn">Відкрийте для себе нові місця</a>
              </div>
            </div>
          </div>
          <div class="col-md-6 d-none d-sm-block">
            <img src="image/Boy-Picture.png" alt="" class="wow animate__animated animate__fadeInUp heroImg">
          </div>
        </div>
      </div>
    </div>
    <!--************************* popular-destination *****************-->
    <section class="popularDest">
```

```
    <div class="container">
      <section id="demos">
        <h1 class="fw-bolder wow animate__animated animate__fadeInDown">ЗНАЙТИ
<span>ПОПУЛЯРНІ</span></h1>
        <h1 class="fw-bolder wow animate__animated animate__fadeInDown">ПРИЗНАЧЕННЯ</h1>
        <div class="owl-carousel owl-theme wow animate__animated animate__fadeIn" id="desti">
          <div class="item card popCard ">
            <div class="">
              <img src="image/popular-img/Amsterdam.jpg" alt="">
              <h3>Amsterdam</h3>
              <p>Netherlands</p>

              <div class="price-box">
                <p><span>₴ 21000</span>/людина</p>
                <a href="discover.html" class="btn btn-primary bookBtn">Переглянути більше</a>
              </div>
            </div>
          </div>
          <div class="item card popCard">
            <img src="image/popular-img/bali.jpg" alt="">
            <h3>Bali</h3>
            <p>Tropical Beach Paradise</p>
            <div class="price-box">
              <p><span>₴ 12000</span>/людина</p>
              <a href="discover.html" class="btn btn-primary bookBtn">Переглянути більше</a>
            </div>
          </div>
          <div class="item card popCard">
            <img src="image/popular-img/darjeeling.jpg" alt="">
            <h3>Darjeeling</h3>
            <p>West Bengal, India</p>

            <div class="price-box">
              <p><span>₴ 15000</span>/людина</p>
              <a href="discover.html" class="btn btn-primary bookBtn">Переглянути більше</a>
            </div>
          </div>
          <div class="item card popCard">
            <img src="image/popular-img/Jaisalmer.jpg" alt="">
            <h3>Jaisalmer</h3>
            <p>Rajasthan, India</p>
            <div class="price-box">
              <p><span>₴ 12500</span>/людина</p>
              <a href="discover.html" class="btn btn-primary bookBtn">Переглянути більше</a>
            </div>
          </div>
          <div class="item card popCard">
            <img src="image/popular-img/maldives.jpg" alt="">
            <h3>Maldives</h3>
            <p>A Stunning Tropical Getaway</p>
            <div class="price-box">
              <p><span>₴ 15400</span>/людина</p>
              <a href="discover.html" class="btn btn-primary bookBtn">Переглянути більше</a>
            </div>
          </div>
          <div class="item card popCard">
```

```
                <img src="image/popular-img/paris.jpg" alt="">
                <h3>Paris</h3>
                <p>France</p>
                <div class="price-box">
                  <p><span>₴ 25000</span>/людина</p>
                  <a href="discover.html" class="btn btn-primary bookBtn">Переглянути більше</a>
                </div>
              </div>
              <div class="item card popCard">
                <img src="image/popular-img/pattaya.jpg" alt="">
                <h3>Pattaya</h3>
                <p>the Gulf of Thailand</p>
                <div class="price-box">
                  <p><span>₴ 18800</span>/людина</p>
                  <a href="discover.html" class="btn btn-primary bookBtn">Переглянути більше</a>
                </div>
              </div>
            </div>
          </section>
        </div>
      </section>
      <!--************************** popular-destination *****************-->
      <!--************************** stories *****************-->
      <section class="storiesSection">
        <div class="container">
          <div class="row">
            <div class="col-md-6">
              <img src="image/Boy-Picture-2.png" alt="" class="wow animate__animated animate__fadeInLeft">
            </div>
            <div class="col-md-6">
              <h1 class="fw-bolder wow animate__animated animate__zoomIn">НАШІ <span>ІСТОРІЇ</span> 3</h1>
              <h1 class="fw-bolder wow animate__animated animate__zoomIn">ПРИГОДАМИ</h1>
              <p class="wow animate__animated animate__zoomIn">Ми маємо досвід у пригодах, щоб розпочати їхню подорож; всі місця для відпочинку на природі у світі – наша спеціалізація</p>
              <p class="wow animate__animated animate__zoomIn">Тож не вагайтеся розпочати свою пригоду просто зараз; природа вже покликала тебе!</p>
              <div class="numberBox">
              </div>
            </div>
          </div>
        </div>
      </section>
      <!--************************** stories *****************-->
      <section class="mapSection">
        <div class="container">
          <div class="row">
            <div class="col-md-12">
              <h1 class="fw-bolder wow animate__animated animate__zoomIn">РОЗПОЧНІТЬ СВОЇ НОВІ <span>ПРИГОДИ</span> </h1>
              <h1 class="fw-bolder wow animate__animated animate__zoomIn">ПО ВСЬОМУ СВІТУ</h1>
              <p>Є ще багато дивовижних місць</p>
              <p> розкидані по всьому світу, вам слід</p>
              <p> спробувати відвідати їх усі</p>
              <img src="image/map-img.jpeg" alt="">
```

```
            </div>
          </div>
        </div>
    </section>
    <!--========================== world-map ============================-->
    <!--========================== world-map-End ============================-->
    <!--======================= adventure-about-us ====================-->
    <section class="storiesSection">
      <div class="container">
        <div class="row">
          <div class="col-md-6">
            <h1 class="fw-bolder wow animate__animated animate__zoomIn"> ЯКІ ШУКАЧІ ПРИГОД</h1>
            <h1 class="fw-bolder wow animate__animated animate__zoomIn">СКАЗАЛИ <span>ПРО НАС</span></h1>
            <p class="wow animate__animated animate__zoomIn">Це було справжнє відкриття! Дуже приємно, що команда завжди враховує наш досвід і думку. Подорож з вами залишила лише найкращі враження!</p>
            <p class="wow animate__animated animate__zoomIn">Вирішив не зволікати - і не пошкодував! Природа справді надихнула, а подорож подарувала масу позитиву.!</p>
            <section id="demos">
 <div class="owl-carousel owl-theme wow animate__animated animate__fadeInDown" id="testimonial">
                <div class="item card popCard">
                  <div class="">
                    <p>Ця поїздка стала для мене справжнім відкриттям! Кожен момент був наповнений емоціями, красою природи та теплом людей. Я ніби знайшов себе заново - без зайвого шуму, без фальші. Щиро дякую за цей неймовірний досвід!</p>
                    <i class="fa fa-star" aria-hidden="true"></i>
                    <i class="fa fa-star" aria-hidden="true"></i>
                    <i class="fa fa-star" aria-hidden="true"></i>
                    <i class="fa fa-star" aria-hidden="true"></i>
                    <i class="fa fa-star" aria-hidden="true"></i>
                    <h5>Андрій Кравчук</h5>
                    <p>Студент</p>
                  </div>
                </div>
                <div class="item card popCard">
                  <div class="">
                    <p>Подорож залишила незабутні враження! Все було щиро, красиво і по-справжньому. Хочеться ще!</p>
                    <i class="fa fa-star" aria-hidden="true"></i>
                    <i class="fa fa-star" aria-hidden="true"></i>
                    <i class="fa fa-star" aria-hidden="true"></i>
                    <i class="fa fa-star" aria-hidden="true"></i>
                    <i class="fa fa-star" aria-hidden="true"></i>
                    <h5>Діана Демюк</h5>
                    <p>Програміст</p>
                  </div>
                </div>
                <div class="item card popCard">
                  <div class="">
                    <p>Подорож у Ростов видалась не з тих, що плануєш через турфірму. Але, як то кажуть, коли життя підкидає лимони - пакуй золотий батон і ближче до кордону. Унітаз, на жаль, не вліз у багаж, але духовно я з ним досі пов’язаний. Ростов прийняв тепло - майже як Межигір’я, тільки без фонтанів і фазанов. Хоча, фазани тут теж бувають... просто в костюмах.</p>
                    <i class="fa fa-star" aria-hidden="true"></i>
```

```
                        <i class="fa fa-star" aria-hidden="true"></i>
                        <i class="fa fa-star" aria-hidden="true"></i>
                        <i class="fa fa-star" aria-hidden="true"></i>
                        <i class="fa fa-star" aria-hidden="true"></i>
                        <h5>Віктор Янукович</h5>
                        <p>Легітимний Президент України</p>
                      </div>
                    </div>









                  </div>
                </section>
              </div>
              <div class="col-md-6">
                <img src="image/girl.png" alt="" class="wow animate__animated animate__fadeInLeft">
              </div>
            </div>
          </div>
        </section>
        <!--======================== adventure-about-usend ======================-->
        <!--================ get-started-section ====================-->
        <section class="getSection owl-theme wow animate__animated animate__fadeIn">
          <div class="container">
            <div class="row">
              <div class="col-md-12">

                <div class="getstarted">
                  <h1 class="fw-bolder owl-theme wow animate__animated animate__fadeInDown">ПОЧАТИ З</h1>
                  <h1 class="fw-bolder owl-theme wow animate__animated animate__fadeInDown"><span>ПОДОРОЖІ</span></h1>
                  <br>
                  <p class="owl-theme wow animate__animated animate__fadeInDown">Підпишіться та знайдіть суперпривабливі цінові пропозиції</p>
                  <p class="owl-theme wow animate__animated animate__fadeInDown">від нас, ми чекаємо на вас у найкращому місці</p>
                  <div class="btn btn-primary start-btn owl-theme wow animate__animated animate__bounceIn">Почати</div>
                </div>
              </div>
            </div>
          </div>
        </section>
        <!--================ get-started-section end====================-->
        <footer class="py-5 wow animate__animated animate__fadeIn">
          <div class="container">
            <div class="row">
              <div class="col-md-4  mb-3">
                <img src="image/logo-no-background.png" alt="" class="logo">
                <p>Нехай Sarthi Travel відкриває для себе нові враження та комфорт, які дарує кожна подорож - адже справжні відкриття заслуговують на це сповна.</p>
              </div>
```

```
            <div class="col-6 col-md-2 mb-3">
              <h5>ПРО</h5>
              <ul class="nav flex-column">
                <li class=" mb-2"><a href="#" class="nav-link p-0 text-muted">Про нас</a></li>
                <li class=" mb-2"><a href="#" class="nav-link p-0 text-muted">Особливості</a></li>
                <li class=" mb-2"><a href="#" class="nav-link p-0 text-muted">Новини та блоги</a></li>
              </ul>
            </div>
            <div class="col-6 col-md-2 mb-3">
              <h5>РУХ</h5>
              <ul class="nav flex-column">
                <li class=" mb-2"><a href="#" class="nav-link p-0 text-muted">Що SARTHI</a></li>
                <li class=" mb-2"><a href="#" class="nav-link p-0 text-muted">Підтримайте нас</a></li>
              </ul>
            </div>
            <div class="col-6 col-md-2 mb-3">
              <h5>КОМПАНІЯ</h5>
              <ul class="nav flex-column">
                <li class=" mb-2"><a href="#" class="nav-link p-0 text-muted">Що SARTHI</a></li>
                <li class=" mb-2"><a href="#" class="nav-link p-0 text-muted">Капітал</a></li>
                <li class=" mb-2"><a href="#" class="nav-link p-0 text-muted">Безпека</a></li>
              </ul>
            </div>
            <div class="col-6 col-md-2 mb-3">
              <h5>ПІДТРИМКА</h5>
              <ul class="nav flex-column">
                <li class=" mb-2"><a href="#" class="nav-link p-0 text-muted">FAQ</a></li>
                <li class=" mb-2"><a href="#" class="nav-link p-0 text-muted">Центр підтримки</a></li>
                <li class=" mb-2"><a href="#" class="nav-link p-0 text-muted">Зв'яжіться з нами</a></li>
              </ul>
            </div>
          </div>
          <div class="row">

            <div class="col-md-12 ">
            </div>
          </div>
        </div>
    </footer>
    <!--********************** script ****************-->
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha3/dist/js/bootstrap.bundle.min.js">
    </script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/wow/1.1.2/wow.min.js"></script>
    <script>
      new WOW().init();
    </script>
    <script>
      $(document).ready(function() {
        $('#desti').owlCarousel({
          loop: true,
          margin: 10,
          responsiveClass: true,
          responsive: {
            0: {
              items: 2,
              nav: true
```

```
            },
            600: {
                items: 2,
                nav: false
            },
            1000: {
                items: 3,
                nav: true,
                loop: false,
                margin: 30
            }
        }
    })
})
</script>


<script>
    $(document).ready(function() {
        $('#testimonial').owlCarousel({
            loop: true,
            margin: 10,
            responsiveClass: true,
            responsive: {
                0: {
                    items: 1,
                    nav: true
                },
                600: {
                    items: 1,
                    nav: false
                },
                1000: {
                    items: 1,
                    nav: true,
                    loop: false,
                    margin: 30
                }
            }
        })
    })
</script>
<script>
    function navbarColour() {
        let navbar = document.getElementById("navBar")
        let scrollValue = window.scrollY

        if (scrollValue < 500) {
            navbar.classList.remove("scroll")
        } else {
            navbar.classList.add("scroll")
        }
    }
    window.addEventListener("scroll", navbarColour)
</script>
<script src="userLoginReg/logstatus.js"></script>
```

```
</body>
</html>

<!DOCTYPE html>
<html lang="uk">
  <head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <title>SARTHI TRAVEL | Вхід</title>

    <link
      href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha3/dist/css/bootstrap.min.css"
      rel="stylesheet"
      integrity="sha384-
KK94CHFLLe+nY2dmCWGMq91rCGa5gtU4mk92HdvYe+M/SXH301p5ILy+dN9+nJOZ"
      crossorigin="anonymous"
    />
    <link
      rel="stylesheet"
      href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css"
    />
    <link rel="stylesheet" href="userRegCss.css" />
  </head>

  <body>
    <div class="registrationSec">
      <div class="container">
        <div class="row justify-content-md-center">
          <div class="col-md-10 regisBox">
            <div class="leftBox">
              <a href="../index.html">
                <img src="../image/white-logo.png" alt="" class="logo" />
              </a>

              <h2>Ласкаво просимо</h2>
              <h4>
                Увійдіть та забронюйте свою мрію
                <i class="fa fa-hand-o-right" aria-hidden="true"></i>
              </h4>
            </div>
            <form id="logForm">
              <center>
                <h1>Вхід</h1>
              </center>
              <div class="mb-3">
                <label for="" class="form-label">Електронна пошта</label>
                <input type="email" class="form-control" id="logemail" aria-describedby="emailHelp"/>
              </div>
              <div class="mb-3">
                <label for="" class="form-label">Пароль</label>
                <input type="logpassword" class="form-control" id="password" />
              </div>
              <div class="d-grid gap-2">
                <button class="btn btn-primary registerBTn" id="logbtn">
                  Увійти
                </button>
```

```
            </div>
            <p>
              Ще не зареєстровані? <a href="registration.html">Реєстрація</a>
            </p>
            <p>
              Вхід для адміністратора?
              <a href="../login/adminlogin.html">Увійти тут</a>
            </p>
          </form>
        </div>
        <center>
          <br /><br />
          <h5>
            <p>© 2025 SARTHI TRAVEL.</p>
          </h5>
        </center>
      </div>
    </div>
  </div>
  <script
    src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha3/dist/js/bootstrap.bundle.min.js"
    integrity="sha384-
ENjdO4Dr2bkBIFxQpeoTz1HIcje39Wm4jDKdf19U8gI4ddQ3GYNS7NTKfAdVQSZe"
    crossorigin="anonymous"
  ></script>
  <script src="login.js"></script>
 </body>
</html>
<!doctype html>
<html lang="uk">
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>SARTHI TRAVEL | Реєстрація</title>
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha3/dist/css/bootstrap.min.css" rel="stylesheet"
integrity="sha384-
KK94CHFLLe+nY2dmCWGMq91rCGa5gtU4mk92HdvYe+M/SXH301p5ILy+dN9+nJOZ"
crossorigin="anonymous">
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-
awesome.min.css" />
  <link rel="stylesheet" href="userRegCss.css">
</head>
<body>
  <div class="registrationSec">
    <div class="container">
      <div class="row justify-content-md-center">
        <div class="col-md-10 regisBox">
          <div class="leftBox">
            <a href="../index.html">
              <img src="../image/white-logo.png" alt="" class="logo">
            </a>
            <h2>ЛАСКАВО ПРОСИМО</h2>
            <h4>Зареєструйтесь та розпочніть подорож з нами <i class="fa fa-hand-o-right" aria-
hidden="true"></i> </h4>
          </div>
          <form id="regForm">
```

```
                <center>
                  <h1>Реєстрація</h1>
                </center>
                <div class="mb-3">
                  <label for="" class="form-label">Ім'я</label>
                  <input type="text" class="form-control" id="name" required placeholder="Введіть ваше ім'я">
                </div>
                <div class="mb-3">
                  <label for="" class="form-label">Електронна пошта</label>
                  <input type="email" class="form-control" id="email" required placeholder="Введіть вашу електронну пошту">
                </div>
                <div class="mb-3">
                  <label for="" class="form-label">Пароль</label>
                  <input type="password" class="form-control" id="password" required placeholder="Введіть пароль">
                </div>
                <div class="d-grid gap-2">
                  <input type="submit" value="Зареєструватись" class="btn btn-primary registerBTn">
                </div>
                <p>Вже зареєстровані? <a href="login.html">Увійти</a></p>
              </form>
            </div>
            <center>
              <br><br>
              <h5>
                <p>© 2025 SARTHI TRAVEL.</p>
              </h5>
            </center>
          </div>
        </div>
      </div>


<!DOCTYPE html>
<html lang="uk">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="icon" type="image/x-icon" href="image/Sarthi-favicon.png">
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha3/dist/css/bootstrap.min.css" rel="stylesheet">
  <link rel="preconnect" href="https://fonts.googleapis.com">
  <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
  <link href="https://fonts.googleapis.com/css2?family=Montserrat&family=Righteous&family=Tilt+Prism&display=swap" rel="stylesheet">
  <link href="https://fonts.googleapis.com/css2?family=Darumadrop+One&display=swap" rel="stylesheet">
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css">
  <link rel="stylesheet" href="css/index.css">
  <title>SARTHI TRAVEL | Послуги</title>
  <style>
    body {
      background-color: rgb(255, 255, 255);
```

```
        width: 100%;
    }
    .box1 {
        display: grid;
        grid-template-columns: repeat(3, 1fr);
        gap: 30px;
        place-items: center;
        margin-top: 120px;
        width: 100%;
    }
    .div1 {
        height: 400px;
        width: 350px;
        margin-left: 30px;
        margin-right: 30px;
        background-color: aqua;
        column-gap: 5px;
        row-gap: 80px;
        opacity: 1;
        box-shadow: rgba(50, 50, 93, 0.25) 0px 2px 5px -1px, rgba(0, 0, 0, 0.3) 0px 1px 3px -1px;
        position: relative;
    }
    button {
        background-color: transparent;
        color: white;
        background-color: black;
        width: 120px;
        font-size: 20px;
        border-radius: 10%;
        font-family: 'Darumadrop One', cursive;
    }
    .div1 button {
        position: absolute;
        bottom: 10px;
        right: 10px;
    }
    button:hover {
        cursor: pointer;
        background-color: red;
        width: 130px;
        font-size: 25px;
        transition: 400ms;
    }
    .div1:hover {
        opacity: 0.8;
    }
    #imgdiv1 {
        background-image:          url("https://internationalyouthclub.org/wp-content/uploads/2021/10/Untitled-
design-1.jpg");
        background-size: cover;
        background-position: center;
    }
    #imgdiv2 {
        background-image: url("https://internationalyouthclub.org/wp-content/uploads/2021/10/dedwf.jpg");
        background-size: cover;
        background-position: center;
```

```
    }
    #imgdiv3 {
        background-image: url("https://internationalyouthclub.org/wp-content/uploads/2021/10/fadasfwef.jpg");
        background-size: cover;
        background-position: center;
    }
    #imgdiv4 {
        background-image: url("https://internationalyouthclub.org/wp-content/uploads/2021/10/f4ffwf.jpg");
        background-size: cover;
        background-position: center;
    }

    #imgdiv5 {
        background-image: url("https://internationalyouthclub.org/wp-content/uploads/2021/10/fsdvbcszce.jpg");
        background-size: cover;
        background-position: center;
    }

    #imgdiv6 {
        background-image: url(" https://internationalyouthclub.org/wp-content/uploads/2021/10/wefsfregrge.jpg");
        background-size: cover;
        background-position: center;
    }
    #imgdiv7 {
        background-image: url("https://internationalyouthclub.org/wp-content/uploads/2021/10/7.jpg");
        background-size: cover;
        background-position: center;
    }
    #imgdiv8 {
        background-image: url("https://internationalyouthclub.org/wp-content/uploads/2021/10/8.jpg");
        background-size: cover;
        background-position: center;
    }
    #imgdiv9 {
        background-image: url(" https://internationalyouthclub.org/wp-content/uploads/2021/10/Untitled-designyju6yhtgf.jpg");
        background-size: cover;
        background-position: center;
    }
    h2 {
        color: white;
        font-size: 50px;
        text-align: center;
        font-family: 'Darumadrop One', cursive;
    }

    .box2 {
        display: flex;
        justify-content: space-around;
        margin-top: 100px;
        width: 100%;
    }
    .logo>img {
```

```
    width: 300px;
}
.logo>p {
    font-size: 20px;
    color: black;
}
.destination {
    font-size: 20px;
}
li>a {
    text-decoration: none;
    color: black;
}
.pay>img {
    width: 200px;
    margin-right: 20px;
}
.pay>p {
    padding-bottom: 30px;
    font-size: 20px;
}
.last {
    padding-bottom: 20px;
    text-align: center;
    font-size: 20px;
}
.topColor {
    background-color: #fff;
    color: #000;
    box-shadow: rgba(0, 0, 0, 0.1) 0px 4px 12px;
}
.topColor a {
    color: #292828;
}
.nav-item:hover {
    border-bottom: 2px solid #f4a424;
}
.nav-link:focus,
.nav-link:hover {
    color: #242222;
}
body {
    margin: 0;
    padding: 0;
    font-family: Arial, sans-serif;
}
#content {
    margin: 50px;
    text-align: center;
}
.form-container {
    position: fixed;
    top: 50%;
    left: 50%;
    transform: translate(-50%, -50%);
    width: 60%;
```

```
    height: 70%;
    padding: 20px;
    background-color: #f4a424;
    z-index: 9999;
    display: none;
    opacity: 0;
    transition: opacity 2s ease-in-out;
}
.form-container.show {
    display: block;
    opacity: 1;
}
.form-container h2 {
    margin-top: 0;
}
.form-container p {
    text-align: center;
    color: white;
}
.form-container label {
    display: block;
    margin-bottom: 8px;
    font-weight: bold;
}
.form-container input[type=text] {
    width: 100%;
    padding: 12px 20px;
    margin-bottom: 12px;
    box-sizing: border-box;
}
.form-container input[type=number] {
    width: 100%;
    padding: 12px 20px;
    margin-bottom: 12px;
    box-sizing: border-box;
}
.form-container input[type=email] {
    width: 100%;
    padding: 12px 20px;
    margin-bottom: 12px;
    box-sizing: border-box;
}
.form-container input[type=submit] {
    background-color: #4CAF50;
    color: white;
    padding: 12px 20px;
    border: none;
    cursor: pointer;
    width: 100%;
    font-size: 16px;
    border-radius: 4px;
}
.form-container .close-button {
    position: absolute;
    top: 10px;
```

ДОДАТОК В (обов'язковий)

ГРАФІЧНА ЧАСТИНА

«РОЗРОБКА WEB-РЕСУРСУ «ТУРАГЕНТСТВО»

Виконав: студент 4 курсу, групи 2КН-21б
спеціальності 122 – Комп'ютерні науки
(шифр і назва напряму підготовки, спеціальності)
Кравчук А. О.
(прізвище та ініціали)

Керівник: к.т.н., доцент каф. КН
Крилик Л. В.
(прізвище та ініціали)
«03» червня 2025 р.

Вінниця ВНТУ - 2025 рік

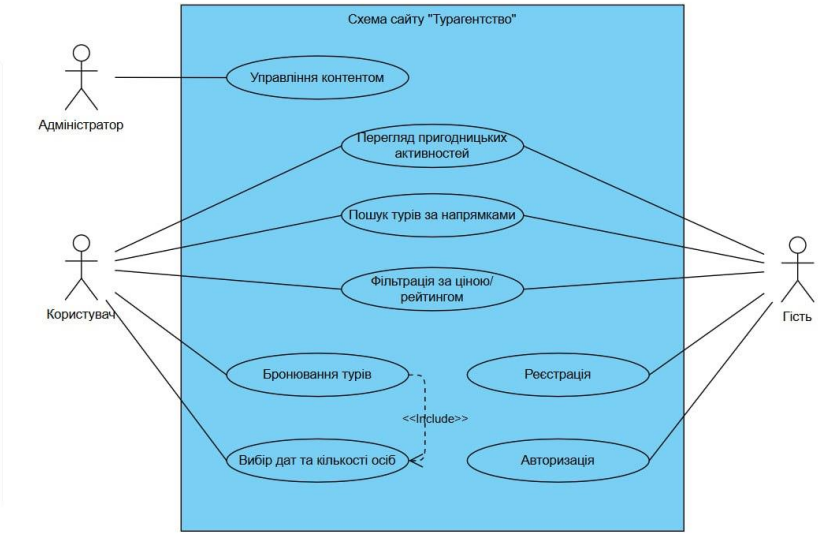

Рисунок В.1 – UML-діаграма прецедентів

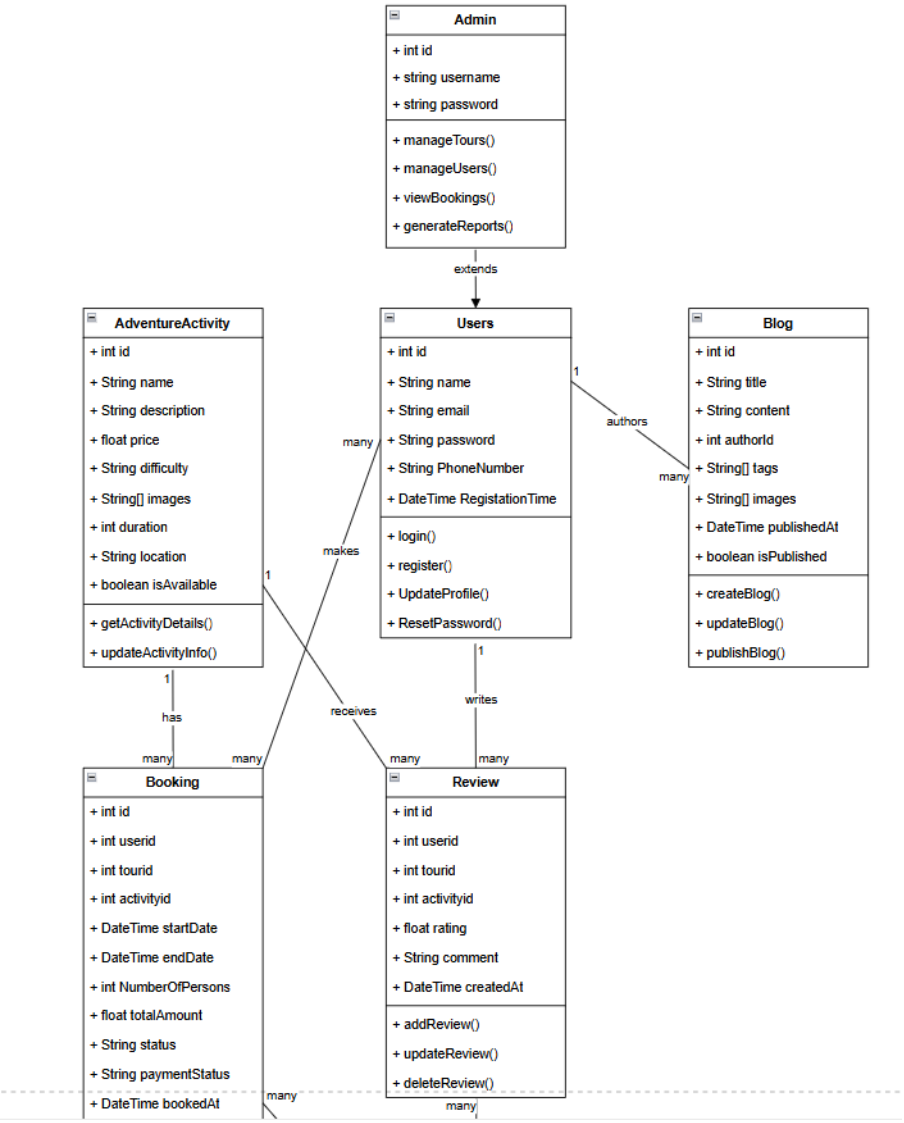

Рисунок В.2 – UML-діаграма класів

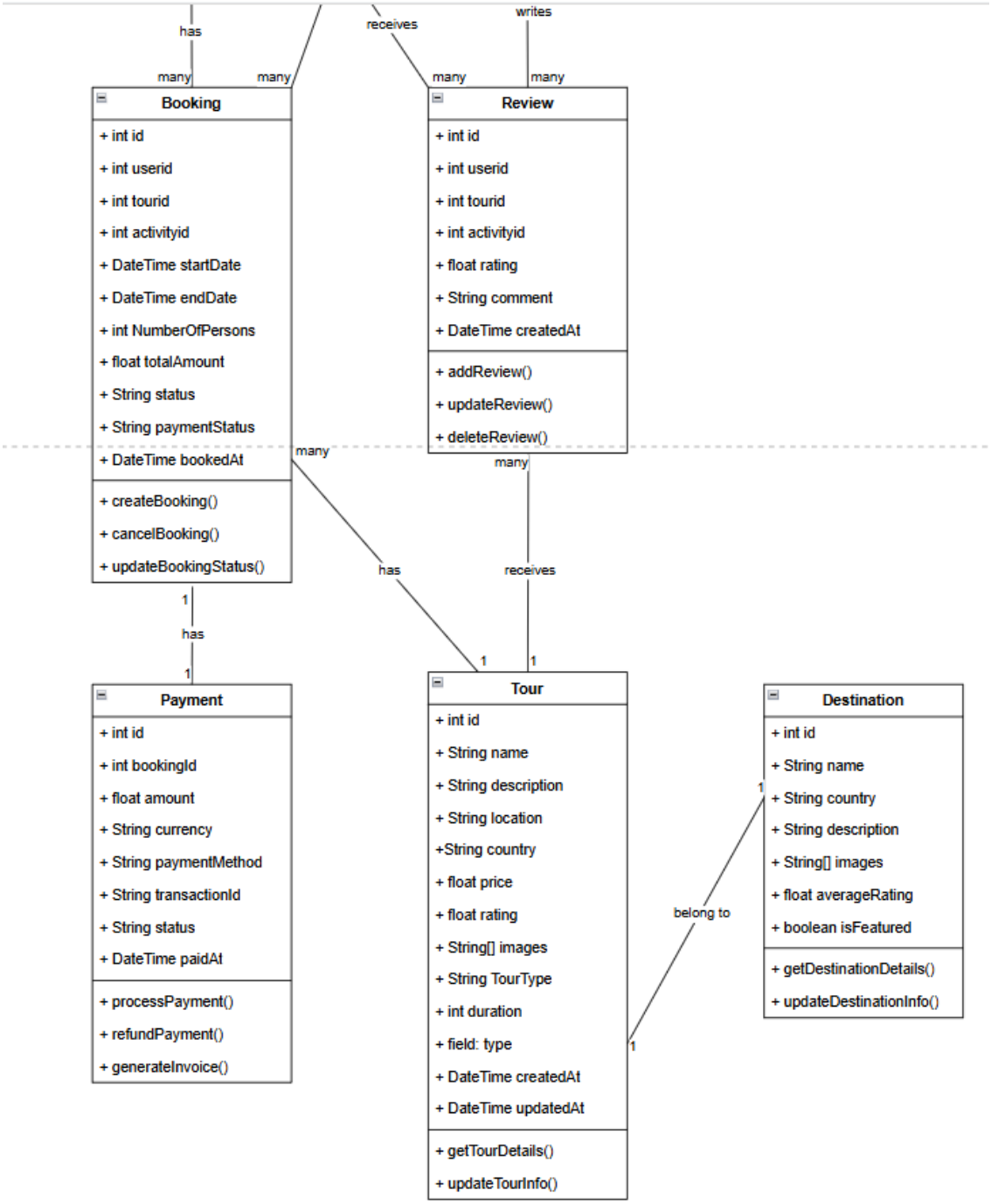

Рисунок В.2, аркуш 2

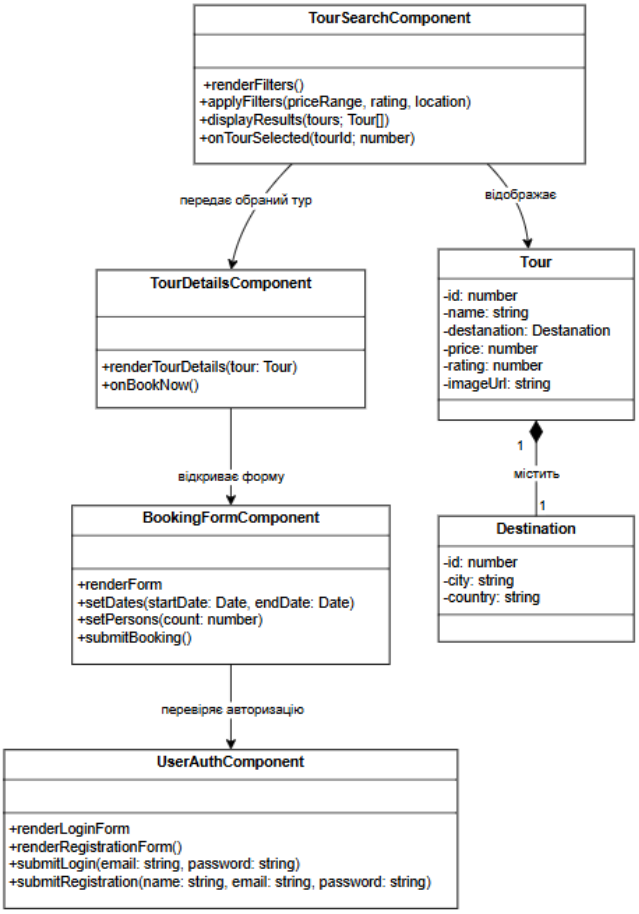

Рисунок В.3 – UML-діаграма класів клієнтської частини модуля підбору турів

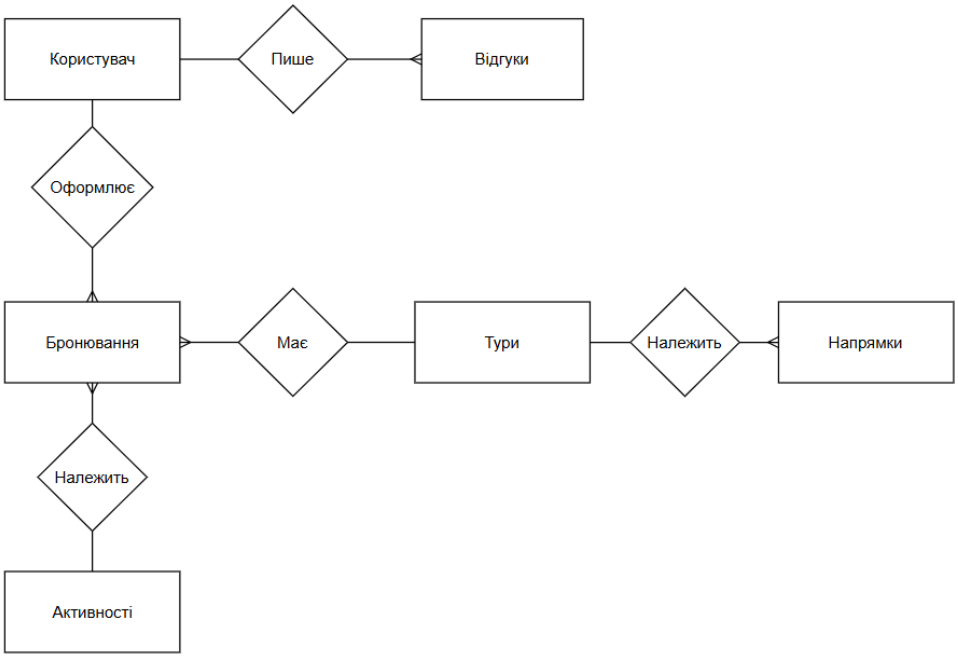

Рисунок В.4 – ER-модель для бази даних WEB-ресурсу «Турагенство»

Рисунок В.5 – Схема алгоритму роботи програмного модуля WEB-ресурсу «Турагенство»

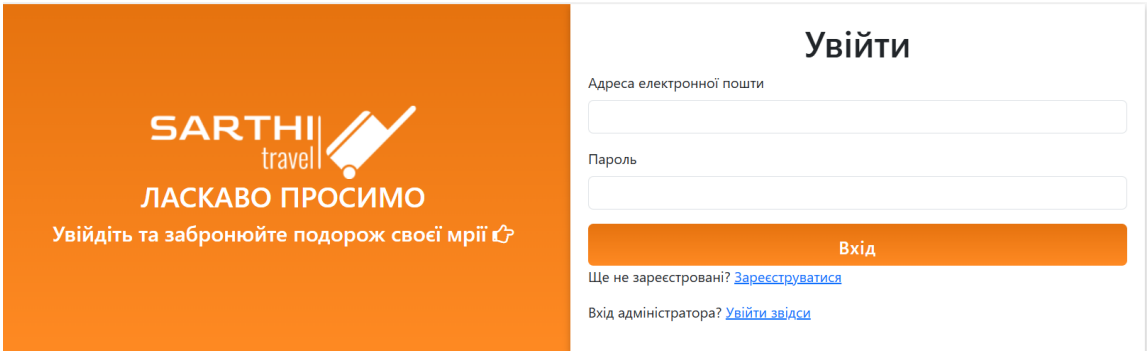

Рисунок В.6 – Приклад роботи сторінки авторизація користувача на WEB-ресурсі «Турагенство»

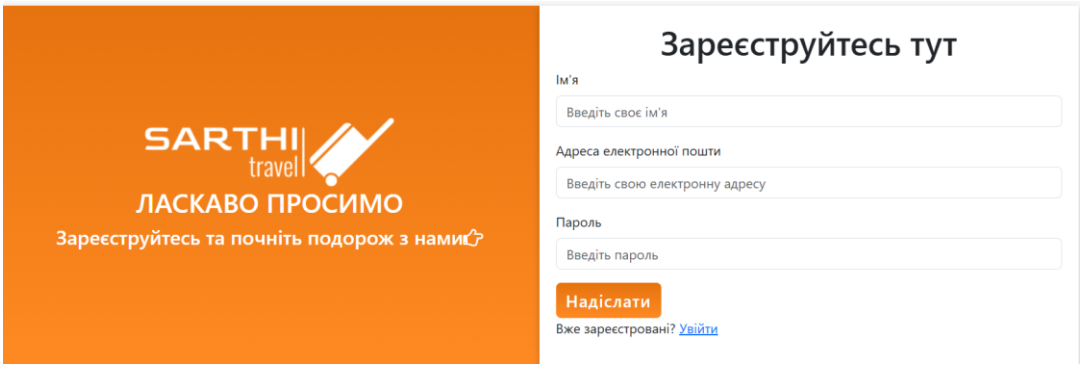

Рисунок В.7 – Приклад роботи сторінки реєстрації користувача на WEB-ресурсі «Турагенство»

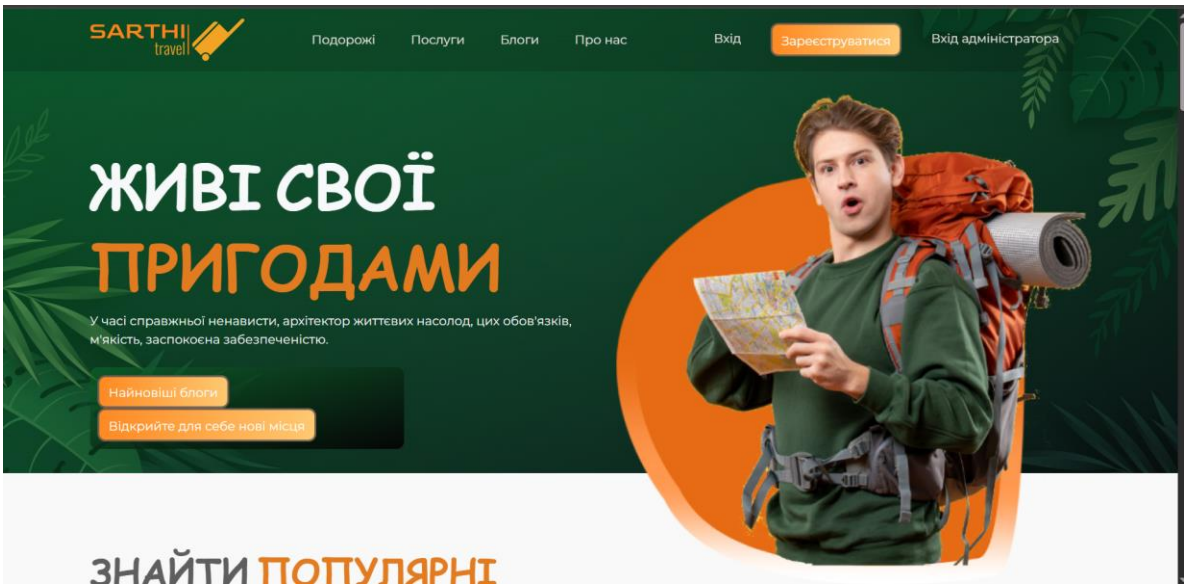

Рисунок В.8 – Приклад роботи головної сторінки на WEB-ресурсі «Турагенство»

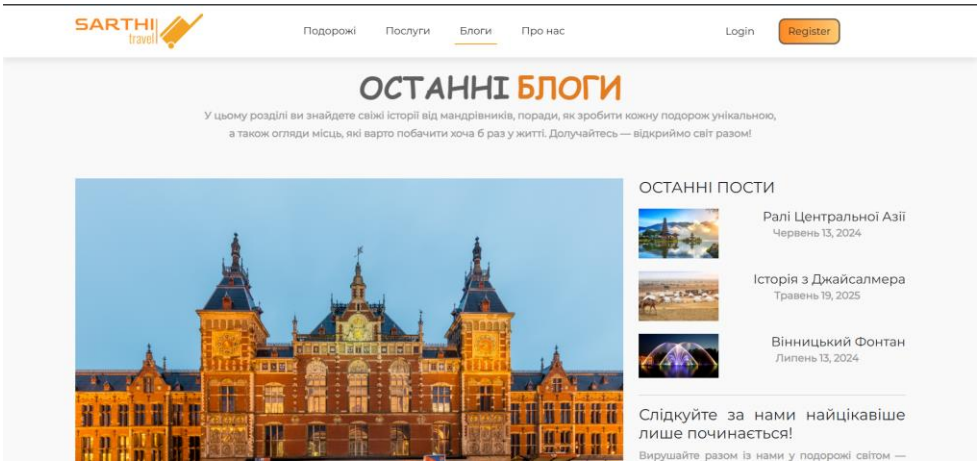

Рисунок В.9 – Приклад роботи сторінки блоги на WEB-ресурсі «Турагенство»

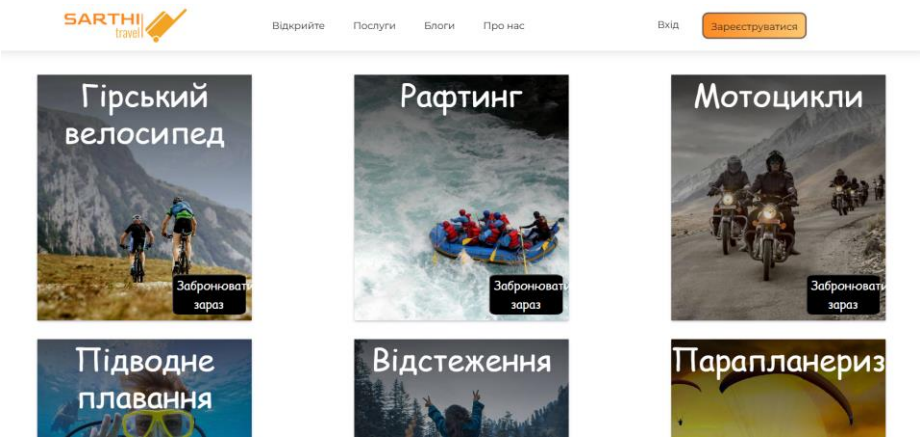

Рисунок В.10 – Приклад роботи сторінки види туристичних активносте на WEB-ресурсі«Турагенство»

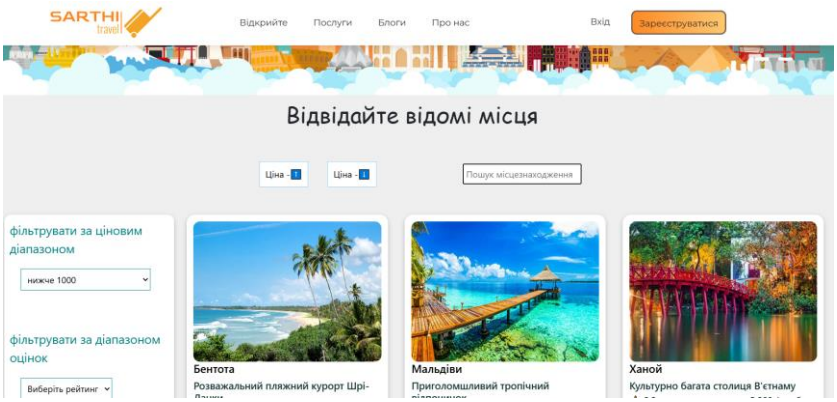

Рисунок В.11– Приклад роботи сторінки запропоновані тури на WEB-ресурсі «Турагенство»

ДОДАТОК Г (довідниковий)

Інструкція користувача

Щоб розпочати користування сервісом, необхідно пройти процедуру реєстрації або авторизуватись у вже створеному обліковому записі (рис. Г.1, Г.2). На сторінці реєстрації користувач вводить своє ім’я, електронну пошту та пароль, після чого натискає кнопку «Зареєструватись». Якщо обліковий запис уже існує, можна перейти за посиланням «Увійти» для авторизаці.

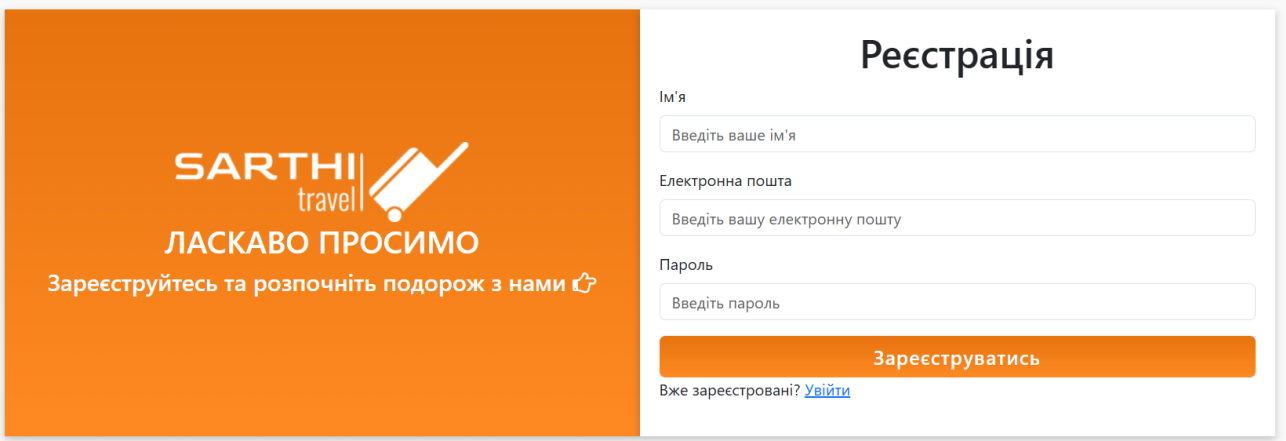

Рисунок Г.1 – Інтерфейс сторінки реєстрації користувача

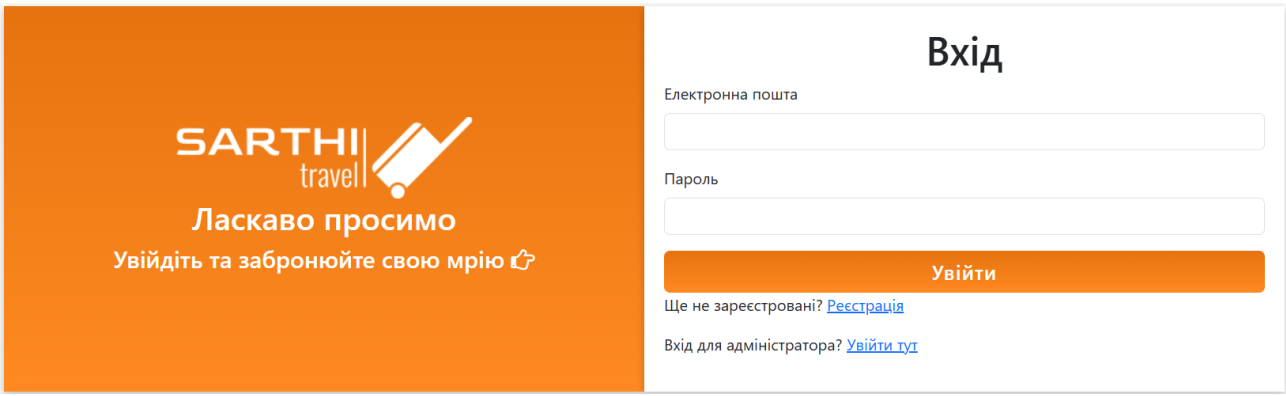

Рисунок Г.2 – Інтерфейс сторінки авторизації користувача

Після успішної реєстрації або авторизації користувач автоматично перенаправляється на головну сторінку (рис. Г.3). Тут у меню є такі посилання: «Подорожі» (рис. Г.4), «Послуги» (рис. Г.5), «Блоги» (рис. Г.6) та «Про нас»

(рис. Г.7). У WEB-ресурсі ви маєте змогу створити нове бронювання, переглянути список вже існуючих подорожей або вибрати тури, або вийти зі свого облікового запису. Також ви можете переглянути блоги про тури та інформація про нас.

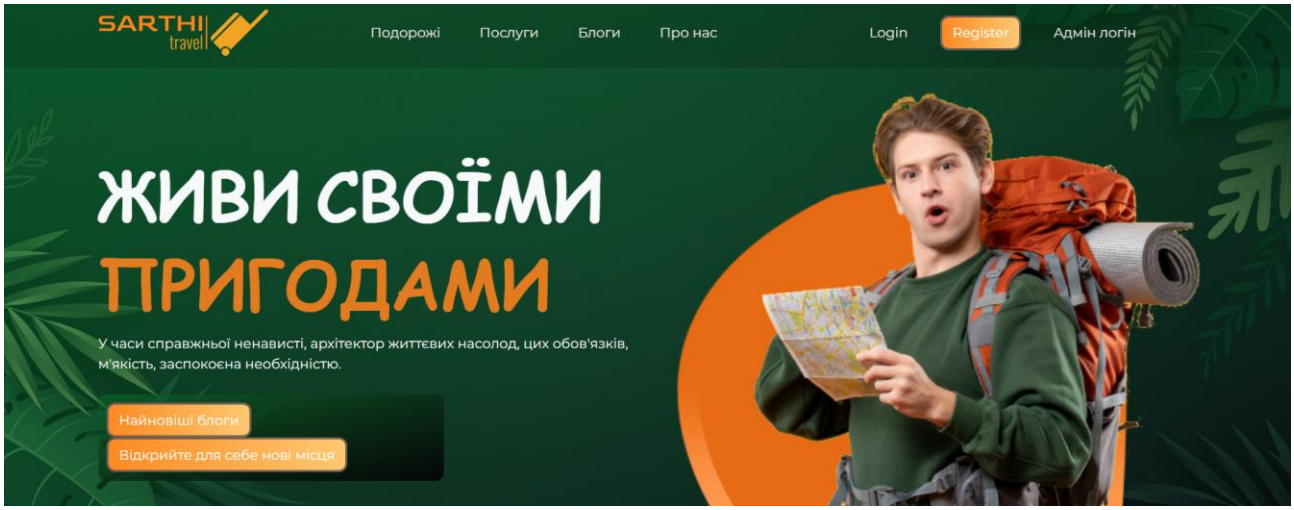

Рисунок Г.3 – Інтерфейс головної сторінки WEB-ресурсу «Турагенство»

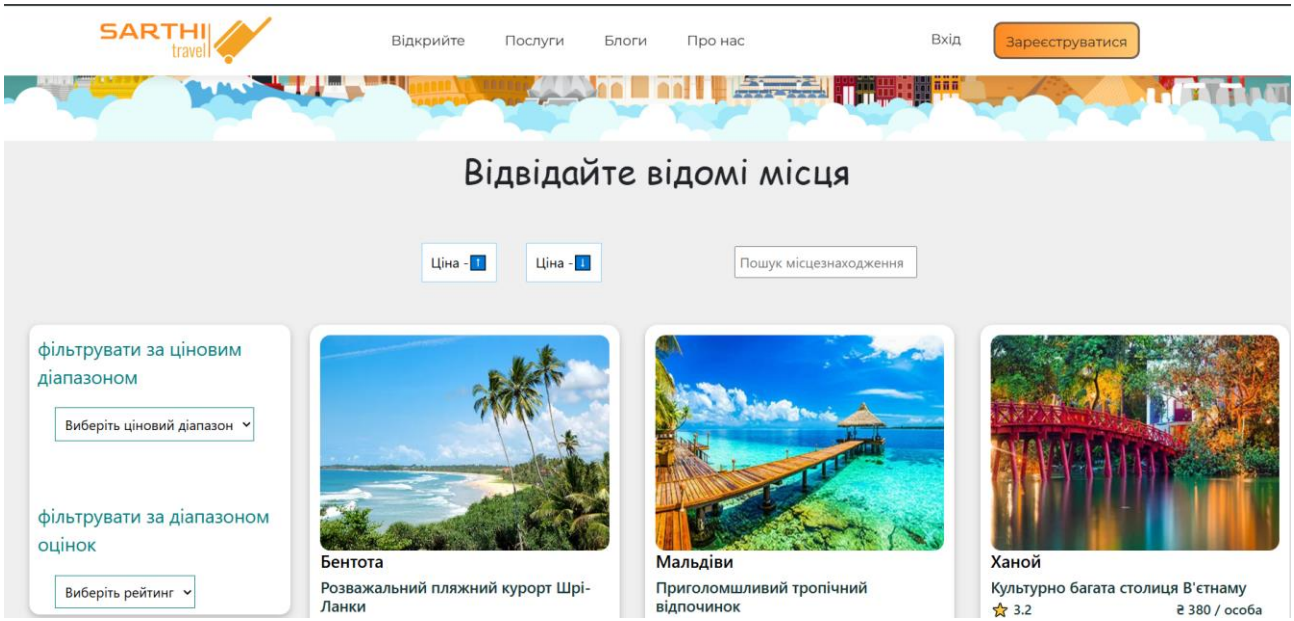

Рисунок Г.4 – Інтерфейс сторінки для пошуку турів по цінам на WEB-ресурсі «Турагенство»

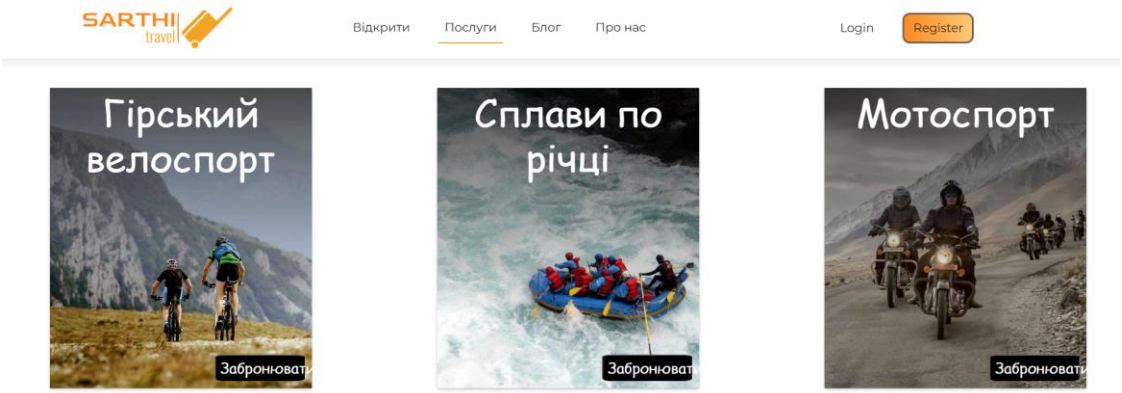

Рисунок Г.5 – Інтерфейс сторінки бронювання турів на WEB-ресурсі «Турагенство»

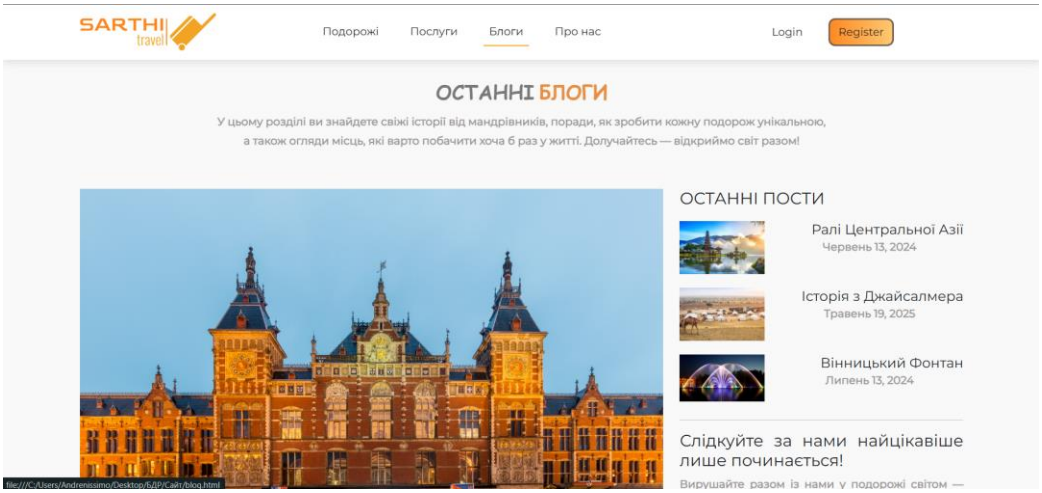

Рисунок Г.6 – Інтерфейс сторінки блоги після входу в систему

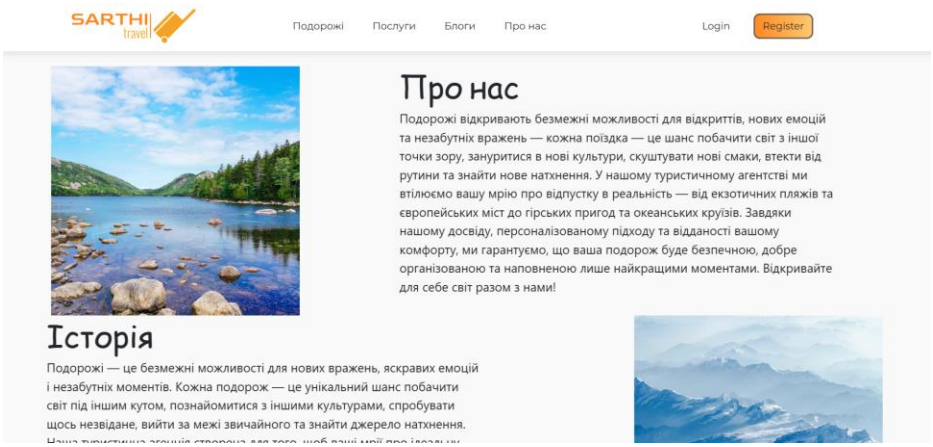

Рисунок Г.7 – Інтерфейс сторінки «Про нас» після входу в систему

Щоб забронювати одну з активностей, достатньо обрати бажану категорію (наприклад, «Гірський велоспорт», «Сплави по річці» або «Мотоспорт») та натиснути кнопку «Забронювати». Після цього відкриється форма, де потрібно заповнити контактні дані або увійти до системи, якщо ви вже зареєстровані (рис. Г.8, Г.9).

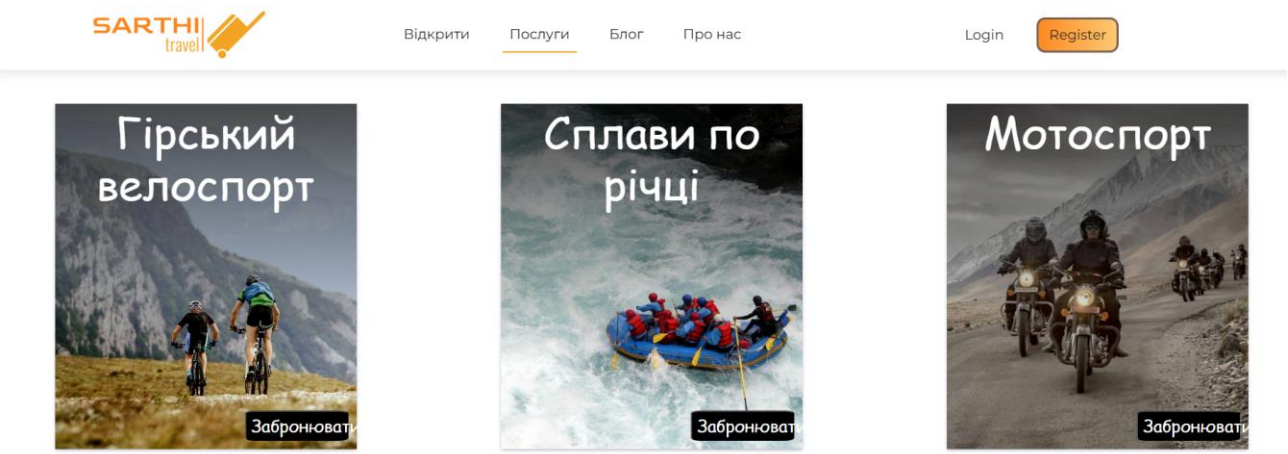

Рисунок Г.8 – Інтерфейс сторінки бронювання турів на WEB-ресурсі «Турагенство»

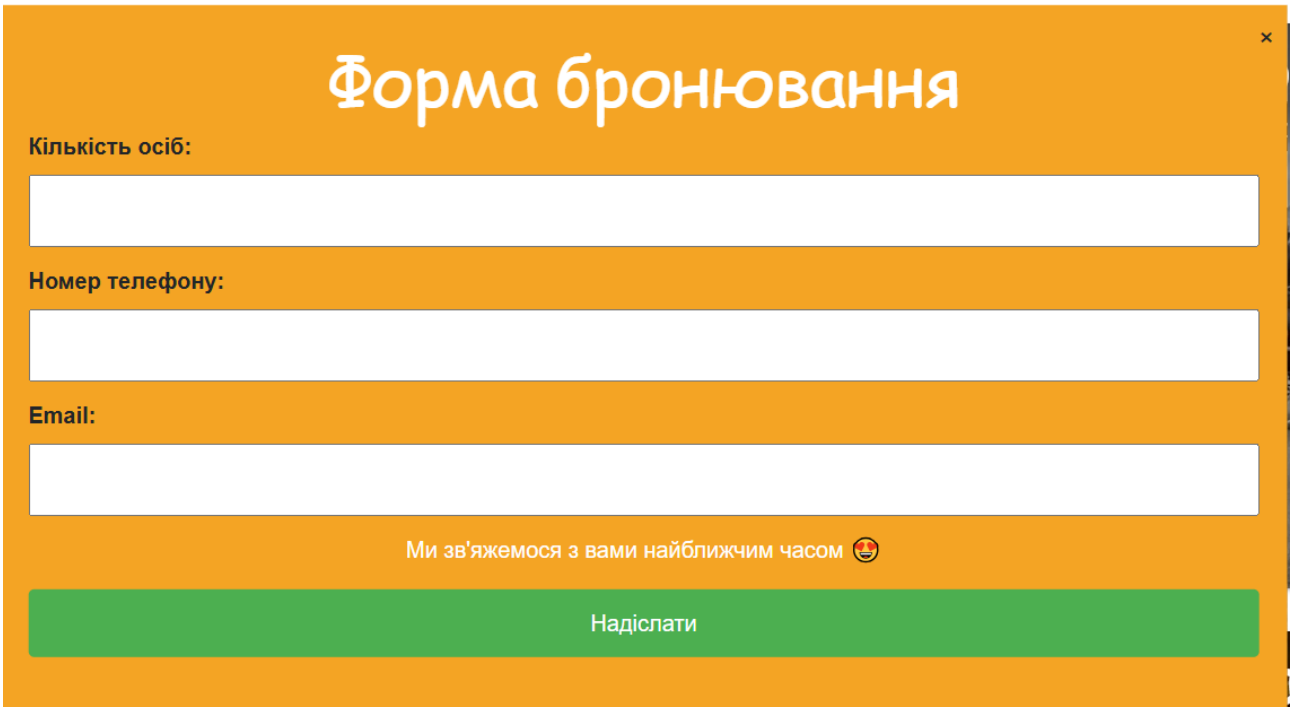

Рисунок Г.9 – Форма бронювання на сторінці «Послуги»