МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет кібербезпеки та програмної інженерії
Кафедра інженерії програмного забезпечення

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач кафедри
Катерина НЕСТЕРЕНКО
“____”_____2024 р.

КВАЛІФІКАЦІЙНА РОБОТА
(ПОЯСНЮВАЛЬНА ЗАПИСКА)

ВИПУСНИКА ОСВІТНЬОГО СТУПЕНЯ
“БАКАЛАВР”

Тема: “ Сайт утворення маршрутів перельотів пасажирів з пересадками ”

Виконавець: Муравйова Анастасія Тимурівна

Керівник: Кучеренко Володимир Миколайович

Нормоконтролер: Варнавський В’ячеслав Володимирович

Київ 2024

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет кібербезпеки та програмної інженерії
Кафедра інженерії програмного забезпечення
Освітній ступінь бакалавр
Спеціальність 121 «Інженерія програмного забезпечення»
Освітньо-професійна програма «Програмне забезпечення систем»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Катерина НЕСТЕРЕНКО
"___" _____ 2024 р.

ЗАВДАННЯ
на виконання кваліфікаційної роботи студентки
Муравйової Анастасії Тимурівни

1. Тема проекту: «Сайт утворення маршрутів перельотів пасажирів з пересадками»
затверджена наказом ректора від 8.12.2023 р. № 2483/ст

2. Термін виконання проекту: з 8.12.2023 р. до 29.02.2024 р.

3. Вихідні дані до проекту: програмний продукт розробити у вигляді сайту для утворення маршрутів перельотів пасажирів з пересадками.

4. Зміст пояснювальної записки:
1. Аналіз системи утворення маршрутів перельотів пасажирів з пересадками.
2. Вимоги до сайту утворення маршрутів перельотів пасажирів з пересадками.
3. Реалізація сайту для утворення маршрутів перельотів пасажирів з пересадками.
4. Результати роботи програми

5. Перелік обов'язкових слайдів презентації:
1. Аналіз принципу купівлі авіабілетів з пересадками
2. Огляд сервісів купівлі квитків
3. Аналіз позитивних аспектів при впровадженні системи у вигляді веб-додатку
5. Вимоги до функціоналу, інтерфейсу, безпеки і апаратна та операційна платформи
6. Результати роботи програми
7. Напрямки оптимізації роботи програми

6. Календарний план-графік

№ пор	Завдання	Термін виконання	Відмітка про виконання
1.	Складання та затвердження графіку кваліфікаційної роботи Написання 1 розділу, представлення керівнику	08.12.23 – 09.01.24	
2.	Попередній друк 1 розділу та допоміжних сторінок (чорнетка) - титульної, завдання, графіка, реферат, список скорочень, зміст, вступ, список джерел, 1-й нормо-контроль.	10.01.24 – 17.01.24	
3.	Написання 2 розділу, представлення керівнику	18.01.24 – 22.01.24	
4.	Написання 3 розділу, представлення керівнику	23.01.24 – 26.01.24	
5.	Написання 4 розділу, представлення керівнику	27.01.24 – 31.01.24	
6.	Загальне редагування та друк пояснювальної записки, графічного матеріалу	01.02.24 – 09.02.24	
7.	Проходження нормо-контролю, перевірка на антиплагіат, перепліт пояснювальної записки.	01.02.24 – 05.02.24	
8.	Розробка тексту доповіді. Оформлення графічного матеріалу для презентації	05.02.24 – 09.02.24	
9.	Отримання відгуку керівника, рецензії.	10.02.24 – 22.02.24	
10.	Підготовка матеріалів для передачі секретарю ДЕК (ПЗ, CD-R з електронними копіями ПЗ, презентації, відгук керівника, рецензія) в папці	23.02.24 – 29.02.24	

7. Дата видачі завдання 8.12.2023р.
Керівник: Володимир КУЧЕРЕНКО
Завдання прийняв до виконання: Анастасія МУРАВЙОВА
Дата

РЕФЕРАТ

Пояснювальна записка до кваліфікаційної роботи «Сайт утворення маршрутів перельотів пасажирів з пересадками»: 61 с., 21 рис., 13 інформаційних джерел.

САЙТ, УТВОРЕННЯ МАРШРУТІВ ПЕРЕЛЬОТІВ, МАРШРУТИ ПЕРЕЛЬОТІВ ПАСАЖИРІВ З ПЕРЕСАДКАМИ, КВИТКИ

Об’єкт розробки – сайт для купівлі квитків з пересадками.

Мета роботи – розробити сайт для купівлі квитків з пересадками.

ABSTRACT

Explanatory note to the thesis " The website for the formation of flight routes for passengers with transfers": 61 p. , 21 Fig. , 13 information sources.

SITE, CREATION OF FLIGHT ROUTES, PASSENGER FLIGHT ROUTES WITH CONNECTIONS, TICKETS

Property development – site for buying tickets with transfers.

Purpose – develop a site for buying tickets with transfers.

ЗМІСТ

ПЕРЕЛІК ПРИЙНЯТИХ СКОРОЧЕНЬ

БД – база даних

IDE – Integrated Development Environment, інтегроване середовище розробки

XML – EXtensible Markup Language, розширювана мова розмітки

MVC – Model-View-Controller, модель–вигляд–контролер

AJAX – Asynchronous JavaScript And XML

URL – Uniform Resource Locator, уніфікований локатор ресурсів або адреса ресурс

EF – Entity Framework

HTML – HyperText Markup Language, мова розмітки гіпертексту

CSS – Cascading Style Sheets, каскадні таблиці стилів

ВСТУП

В умовах сучасного швидкого темпу життя та постійної динаміки глобальних подорожей, актуальність створення сайту для утворення маршрутів перельотів з пересадками надзвичайно висока. Сьогодні пасажири шукають не просто спосіб пересунутися з пункту А в пункт Б, але інтелектуальний підхід до вибору оптимального маршруту, який був враховуваний під їх індивідуальні потреби і уподобання.

Зараз пасажири все більше віддають перевагу перельотам з пересадками, щоб зекономити кошти. Але цей тренд визначається не лише економічними міркуваннями, але й прагненням розширити свій досвід подорожей. Пасажири шукають не просто транспортний засіб, але й можливість взаємодії з новими культурами, відкриття цікавих локацій, а також економію часу та ресурсів. Зараз, завдяки зростанню кількості доступних маршрутів і пересадкових пунктів, пасажири можуть обирати оптимальні варіанти подорожей, що також дозволить зробити сам перельот частиною пригоди.

Саме зростання популярності пересадкових маршрутів створює потребу в інноваційних підходах до планування подорожей, де інтелектуальні алгоритми та технології допомагають максимізувати комфорт та забезпечують пасажирам можливість насолоджуватися не лише пунктом призначення, але і самим процесом подорожі. У цьому контексті, інтелектуальний сайт, який надає найоптимальніші та найзручніші маршрути з урахуванням різноманітних факторів, стає важливим інструментом для подорожуючих.

Зростання конкуренції в авіаційній галузі, стрімкий розвиток технологій та розширення мережі маршрутів вимагають нового погляду на планування подорожей. Створення цього сайту – це не просто відповідь на попит ринку, але й крок до зручної, ефективної та інтелектуальної системи, яка забезпечить найвищий рівень задоволення від подорожі для кожного пасажира.

Саме цим було інспіровано обрану тему для даної кваліфікаційної роботи, оскільки вирізняючийся попит пасажирів на пересадкові маршрути в сучасному світі подорожей створює важливий контекст для дослідження та впровадження інноваційних підходів у створенні інтелектуального сайту, спрямованого на оптимізацію та персоналізацію маршрутів, забезпечуючи пасажирам неповторний досвід та максимальну задоволеність від кожної подорожі.

РОЗДІЛ 1.
АНАЛІЗ СИСТЕМИ УТВОРЕННЯ МАРШРУТІВ ПЕРЕЛЬОТІВ ПАСАЖИРІВ З ПЕРЕСАДКАМИ

1.1. Аналіз принципу купівлі авіабілетів з пересадками

В сучасному світі, коли подорожі стають неодмінною частиною нашого життя, вибір оптимального маршруту та купівля авіабілетів стають завданням, яке потребує обдуманої стратегії.

Пересадки в авіаподорожах можуть бути як перешкодою, так і можливістю, і залежать це від індивідуальних потреб, планів та умов. У цьому контексті важливо ретельно вивчити ряд аспектів, щоб зробити інформований вибір, який враховує економічні, часові та комфортні фактори.

Розробка сайту для утворення маршрутів перельотів пасажирів з пересадками вимагає глибокого аналізу принципів на які спирається пасажир при купівлі авіабілетів з пересадками.

У цьому аналізі будуть розглянуті різні важливі аспекти при купівлі авіабілетів з пересадками, такі як вартість квитків, тривалість і час пересадки, гнучкість маршруту, а також можливості для відпочинку та економії. Також буде розглянуто й можливі ризики, пов'язані з пересадками, і як правильно обирати оптимальний варіант залежно від вподобань. Аналіз принципу на які спирається пасажир під час купівлі авіабілетів з пересадками включає в себе ряд факторів, які можуть впливати на вибір і загальний комфорт подорожі:

1. Тривалість і час пересадки

Деякі люди шукають більш тривалі маршрути, щоб вони могли витратити більше часу в аеропорту, а не літаючи. Інші віддають перевагу швидким пересадкам, щоб максимально економити час і зменшити втомленість.

2. Вартість квитків

Важливо аспектом є порівняння вартості квитків прямого рейса і рейса з пересадками для знаходження оптимальної пропозиції. Пересадки можуть здешевити квиток, особливо якщо вибрати не традиційні маршрути або авіакомпанії.

3. Зручність пересадок

Важливим буде оцінювати якість і зручність аеропортів, в яких буде пересадка. Якщо пересадка триває кілька годин, важливо бути впевненим, що буде достатньо часу для переходу між терміналами.

4. Відшкодування і відміна рейсів

Пересадкові рейси можуть збільшити ризик затримок або втрати зв'язку між рейсами, що може вплинути на план подорожі. Важливо також мати інформацію про політику авіакомпанії стосовно відшкодування у випадку непередбачуваних обставин.

5. Багаж і перевізники

Необхідно враховувати правила перевезення багажу для кожного етапу подорожі з пересадками. Тому необхідна особлива увагу на те, які авіакомпанії обслуговують кожен рейс, і як це може вплинути на багаж.

6. Лояльність авіакомпаній

Якщо пасажир учасник програми лояльності, необхідно врахувати можливість збору миль або бонусів за пересадкові рейси.

7. Економія часу і комфорт

З одного боку, прямі рейси можуть забезпечити економію часу, оскільки вони скорочують час подорожі. З іншого боку, пересадки можуть бути економічно вигідні, і деякі пасажири можуть обирати довші пересадки для відпочинку чи подорожі.

При наявності тривалих пересадок можна використати цей час для

екскурсій у місті або відпочинку у готелі тощо. Деякі пасажири шукають пересадки як можливість для короткого або більш тривалого відпочинку.

Також поєднання різних авіакомпаній може дозволити вибрати кращі сервіси, розклади і умови перельоту для кожного етапу подорожі.

Можна зробити висновок, що купівля авіабілетів з пересадками може бути вигідною для пасажирів з різними потребами і обставинами. Але треба враховувати, що пересадкові рейси можуть бути як вигідними, так і викликати певні труднощі [1, 2, 3].

Зробити правильний вибір можливо, здійснивши аналіз ключових аспектів, таких як вартість квитків, тривалість пересадок, зручність та можливості для відпочинку. Наявність потужних інструментів для пошуку, фільтрації та визначення оптимального маршруту робить процес вибору більш ефективним.

Зазначаючи можливі ризики, такі як затримки та втрата зв'язку між рейсами, необхідно надавати користувачам вичерпну інформацію та варіанти для мінімізації впливу непередбачених обставин.

З урахуванням потреб індивідуальних користувачів, аналіз принципу купівлі авіабілетів з пересадками є необхідним етапом у розробці інтелектуальних та користувацько-орієнтованих платформ для планування подорожей. Вірно підібрані та розроблені інструменти можуть допомогти максимізувати переваги пересадкових рейсів та забезпечити задоволення від подорожі для кожного пасажира. Ось кілька аспектів, які важливо врахувати під час реалізації додатку, виходячи з аналізу:

1. Пошук та фільтрація

Необхідно створити потужний пошук, який дозволяє користувачам швидко і точно знаходити оптимальні маршрути з пересадками за різними критеріями, такими як ціна, час, авіакомпанії та інші фільтри.

2. Інформаційна доступність

Забезпечення користувачів повною та зрозумілою інформацією про кожен етап подорожі, включаючи час пересадки, авіакомпанії, термінали та

інші деталі.

3. Розуміння вартості та економії

Аналіз вартості квитків, враховуючи пересадки, і відображення інформації щодо можливостей економії для користувачів.

4. Прозорі умови та політика

Включення докладної інформації щодо умов авіакомпаній, політики відшкодування та можливостей зміни маршрутів.

5. Опції для гнучкості та зупинок

Розробка системи, яка дозволяє користувачам вибирати гнучкість маршруту та включати додаткові зупинки в маршруті.

6. Ризики та інформація про затримки

Включення інформації щодо можливих ризиків затримок та як вони можуть вплинути на план подорожі.

7. Опції лояльності та бонуси

Врахування можливостей для користувачів отримувати бонуси чи вигоди від програм лояльності авіакомпаній.

1.2. Огляд сервісів купівлі квитків

Для створення власного додатку важливо бути ознайомленим з аналогічними додатками, що вже представлені на ринку, а також розуміти їх переваги та недоліки. На ринку існує багато сервісів для придбання квитків з пересадками, які допомагають пасажирам забезпечити зручний та ефективний маршрут для подорожі.

1. Skyscanner

Skyscanner є одним з найбільш відомих і широко використовуваних сервісів для пошуку квитків. Він дозволяє користувачам знаходити найзручніші та найбільш доступні варіанти подорожі з пересадками.

Переваги:

– Широкий вибір варіантів. Skyscanner співпрацює з багатьма

авіакомпаніями та агентствами, що дозволяє користувачам вибирати серед різних варіантів маршрутів та цін;

– Гнучкість в пошуку. Сервіс надає можливість введення гнучких критеріїв пошуку, таких як гнучкі дати чи «пошук за всіма аеропортами в регіоні», що може допомогти знайти оптимальний варіант;

– Цінові оповіщення. Skyscanner дозволяє встановлювати цінові оповіщення, що дозволяє користувачам отримувати повідомлення, коли ціни на певні маршрути змінюються;

– Простий інтерфейс. Інтерфейс Skyscanner є простим та легким у використанні, що робить процес пошуку квитків приємним для користувача;

– Глобальне покриття. Skyscanner працює з численними місцями призначення та авіакомпаніями по всьому світу, що робить його ідеальним вибором для міжнародних подорожей.

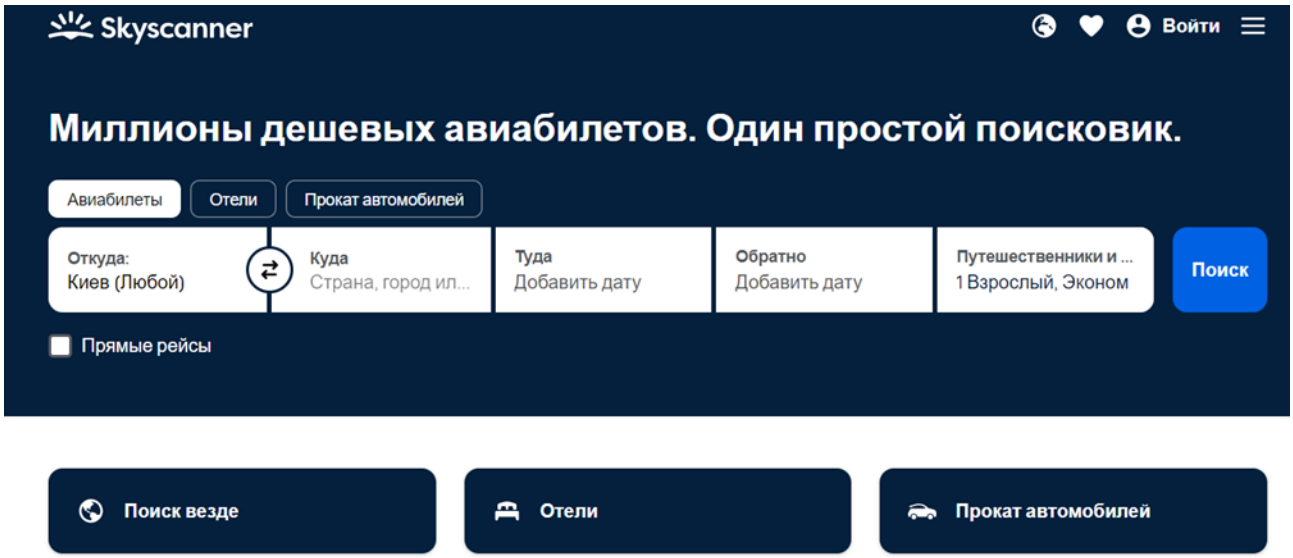

Рис. 1.1 Інтерфейс Skyscanner

Недоліки:

– Додаткові збори. Деякі користувачі скаржаться на те, що при оформленні бронювань через Skyscanner можуть виникнути додаткові збори або комісії;

– Не завжди точні ціни. Ціни на квитки можуть змінюватися, іноді вони можуть виявитися неактуальними під час оформлення бронювання;

– Можливість затримок. У рідкі випадки може виникнути затримка у відображенні актуальних результатів пошуку;

– Обмежені функції фільтрації. Інструменти фільтрації та сортування можуть бути менш розвиненими порівняно з іншими сервісами, так що користувачам може бути важче знаходити конкретні параметри;

– Не завжди найдешевші варіанти. Хоча Skyscanner часто пропонує зручні та доступні варіанти, існують інші сервіси, які можуть мати ще більш конкурентоспроможні ціни для певних маршрутів.

2. Kiwi.com

Цей сервіс спеціалізується на з'єднанні різних авіакомпаній та виділяється тим, що пропонує "складні" маршрути з пересадками, які можуть бути дешевшими або більш зручними, ніж прямі рейси.

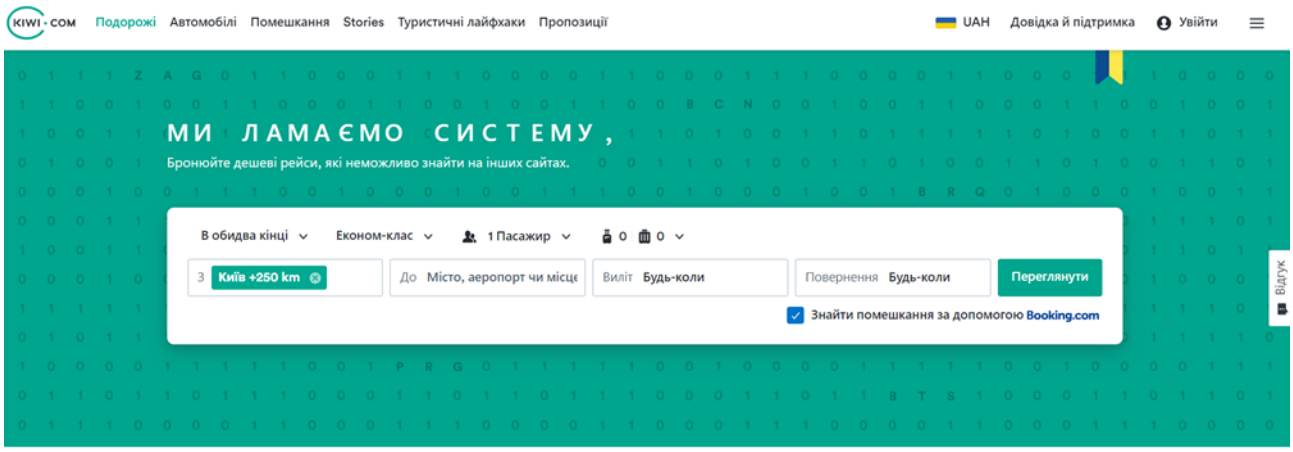

Рис. 1.2 Інтерфейс Kiwi.com

Переваги:

– Складні маршрути. Kiwi.com відомий своєю здатністю об'єднувати різні авіакомпанії та забезпечувати складні маршрути з пересадками, що може призводити до зниження вартості квитків;

– Гарантія проти затримок та втрат. Компанія пропонує «Гарантію

Kiwi.com», яка забезпечує пасажирів в разі затримок, втрати польотів або інших подій, що можуть вплинути на подорож;

– Розширений пошук з місцем призначення. Можливість вводити не конкретний аеропорт, а цілий регіон чи країну для місця призначення дозволяє знаходити більше варіантів для подорожей;

– Інтерактивна карта маршруту. Користувачі можуть переглядати свій маршрут на інтерактивній карті, щоб краще розуміти свою подорож;

– Пошук готелів і автомобілів. Kiwi.com також дозволяє користувачам забронювати готелі та автомобілі, забезпечуючи повний пакет послуг для подорожей.

Недоліки:

– Обмежені опції фільтрації. У порівнянні з деякими іншими сервісами, Kiwi.com може мати менше опцій для фільтрації результатів пошуку;

– Невизначені авіакомпанії та пересадки. Деякі користувачі вказують на те, що інформація про конкретні авіакомпанії та пересадки може бути менш деталізованою порівняно з іншими сервісами;

– Можливість зміни розкладу. З огляду на складні маршрути, Kiwi.com може змінювати розклади на деяких етапах маршруту, що може впливати на зручність подорожі;

– Не завжди найдешевші варіанти. Хоча Kiwi.com спеціалізується на складних маршрутах, іноді інші сервіси можуть пропонувати більш конкурентоспроможні ціни для простіших маршрутів;

– Оцінка пасажирських відгуків. Як і в інших подібних сервісах, які об'єднують різні авіакомпанії, якість обслуговування та інші аспекти можуть варіювати в залежності від конкретного перевізника.

3. Google Flights

Google Flights надає зручний інтерфейс для пошуку квитків, включаючи варіанти з пересадками. Сервіс також може надати корисні поради та рекомендації щодо оптимальних пересадок.

Переваги:

– Інтеграція з пошуковою системою Google. Сервіс інтегрований з пошуковою системою Google, що дозволяє швидко та зручно знаходити квитки, використовуючи звичайний пошук;

– Гнучкість в пошуку дат та місць. Google Flights пропонує користувачам можливість переглядати ціни на квитки для гнучких дат та вибору різних місць вильоту та призначення;

– Графік цін на квитки. Сервіс може надавати графіки змін цін на квитки для конкретного маршруту, що допомагає користувачам знаходити оптимальний час для покупки;

– Порівняння цін на готелі та автомобілі. Окрім квитків, Google Flights дозволяє користувачам порівнювати ціни на готелі та автомобілі, щоб забезпечити повний пакет для подорожі;

– Можливість збереження маршрутів. Користувачі можуть зберігати та відстежувати маршрути для майбутнього користування та порівняння цін.

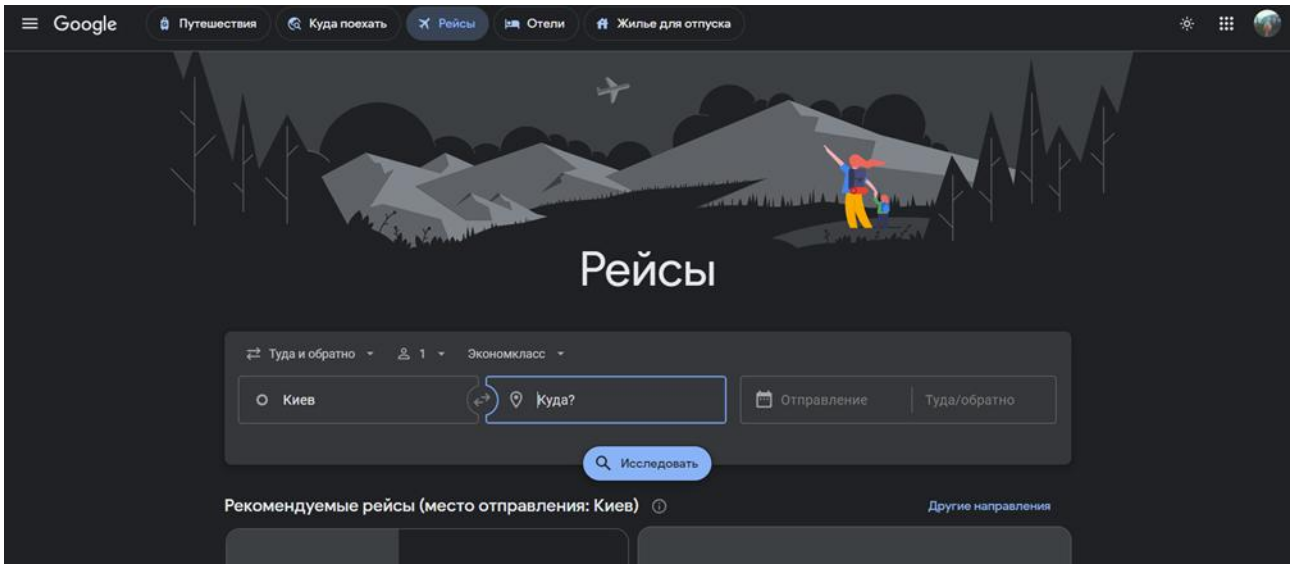

Рис. 1.3 Інтерфейс Google Flights

Недоліки:

– Відсутність багатозонового пошуку. У порівнянні з деякими конкурентами, Google Flights може бути менш гнучким, коли мова йде про

пошук багатозонових маршрутів;

– Не завжди найзручніший інтерфейс для складних маршрутів. При пошуку складних маршрутів із багатьма пересадками інтерфейс Google Flights може бути менш інтуїтивно зрозумілим порівняно з іншими сервісами;

– Обмежені можливості фільтрації. У деяких випадках може бути менше опцій для точної фільтрації результатів, порівняно з іншими спеціалізованими сервісами;

– Можливість зміни цін під час оформлення. Як і в інших сервісах, ціни можуть змінюватися під час процесу бронювання;

– Не завжди включає всі низькобюджетні авіакомпанії. Google Flights може не завжди включати всі низькобюджетні авіакомпанії чи регіональних перевізників, що може обмежити варіативність.

4. Expedia

Expedia є комплексним туристичним сервісом, який дозволяє забронювати не лише авіаквитки, але й готелі та автомобілі. Можна шукати маршрути з пересадками та порівнювати ціни.

Переваги:

– Широкий вибір послуг. Expedia пропонує не лише авіаквитки, але і готелі, автомобілі, круїзи та пакетні пропозиції, що робить його комплексним туристичним сервісом;

– Зручний інтерфейс. Інтерфейс Expedia є зрозумілим і легким у використанні, що полегшує користувачам пошук та бронювання;

– Програма лояльності. Expedia має програму лояльності, де користувачі можуть отримувати бонусні бали та знижки за кожне бронювання;

– Багатоакційні пропозиції. Сервіс часто пропонує різноманітні акції та знижки, що дозволяє зекономити кошти на подорожах;

– Можливість планування пакетів. Expedia дозволяє користувачам бронювати весь пакет послуг, включаючи авіаквитки, готелі та інші опції, за один раз.

Недоліки:

– Додаткові збори. При оформленні бронювання можуть виникати додаткові збори та комісії, які не завжди чітко вказані на початковому етапі;

– Не завжди найдешевші ціни. На деяких маршрутах і для певних послуг Expedia може не мати найкращі ціни, порівняно з іншими сервісами;

– Обмежена гнучкість в пошуку. Деякі користувачі можуть відчувати обмежену гнучкість в пошуку та фільтрації результатів, порівняно з іншими конкурентами;

– Можливість затримок в оновленні цін. Як і в інших сервісах, ціни можуть не завжди відображати найновіші зміни і можуть змінюватися під час процесу бронювання;

– Обмежена інформація про авіакомпанії та пересадки. Деякі користувачі вказують на те, що інформація про конкретні авіакомпанії та пересадки може бути менш деталізованою порівняно з іншими сервісами.

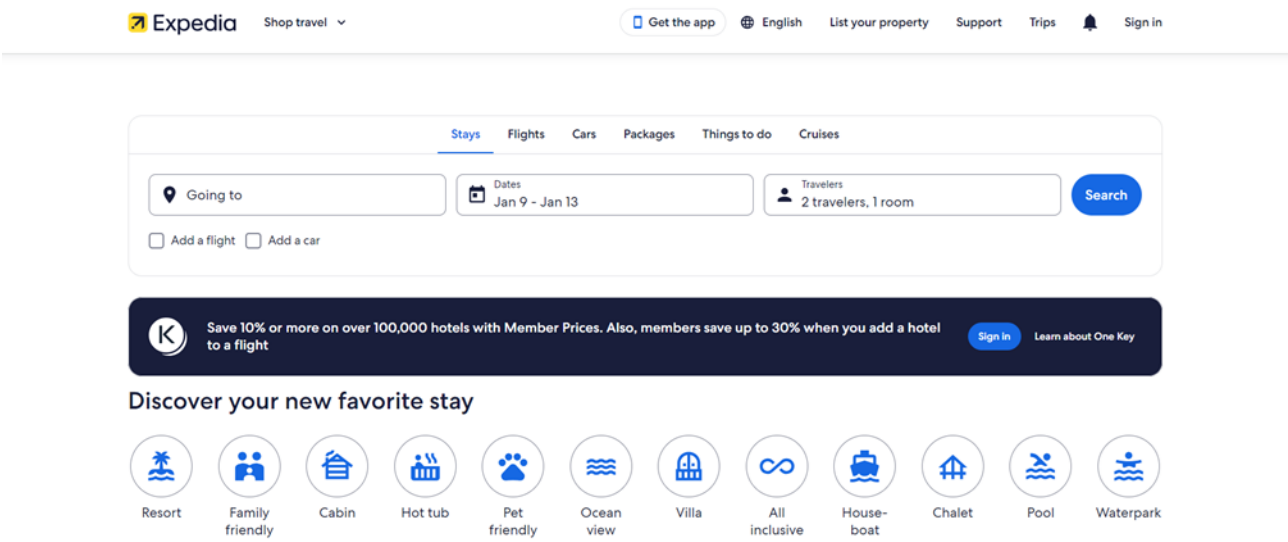

Рис. 1.4 Інтерфейс Expedia

5. Momondo

Ще один сервіс для пошуку квитків, який дозволяє користувачам знаходити найбільш вигідні варіанти подорожі. Momondo часто включає в себе варіанти з пересадками.

Переваги:

– Широкий вибір варіантів. Momondo пропонує обширний вибір

авіаквитків та варіантів маршрутів від різних авіакомпаній та агентств;

– Гнучкість в пошуку. Сервіс дозволяє користувачам вводити гнучкі критерії, такі як гнучкі дати та альтернативні аеропорти, що допомагає знаходити оптимальні варіанти;

– Порівняння цін. Momondo включає в себе ціни не тільки від авіакомпаній, але й від різних онлайн-тревел-агентств, що дозволяє користувачам знаходити найвигідніші пропозиції;

– Графік цін. Сервіс надає графік змін цін на квитки для конкретного маршруту, допомагаючи визначити оптимальний момент для покупки;

– Рейтинг авіакомпаній та комфорт. Momondo може включати рейтинги авіакомпаній та рівень комфорту, що допомагає користувачам зробити більш обдуманий вибір.

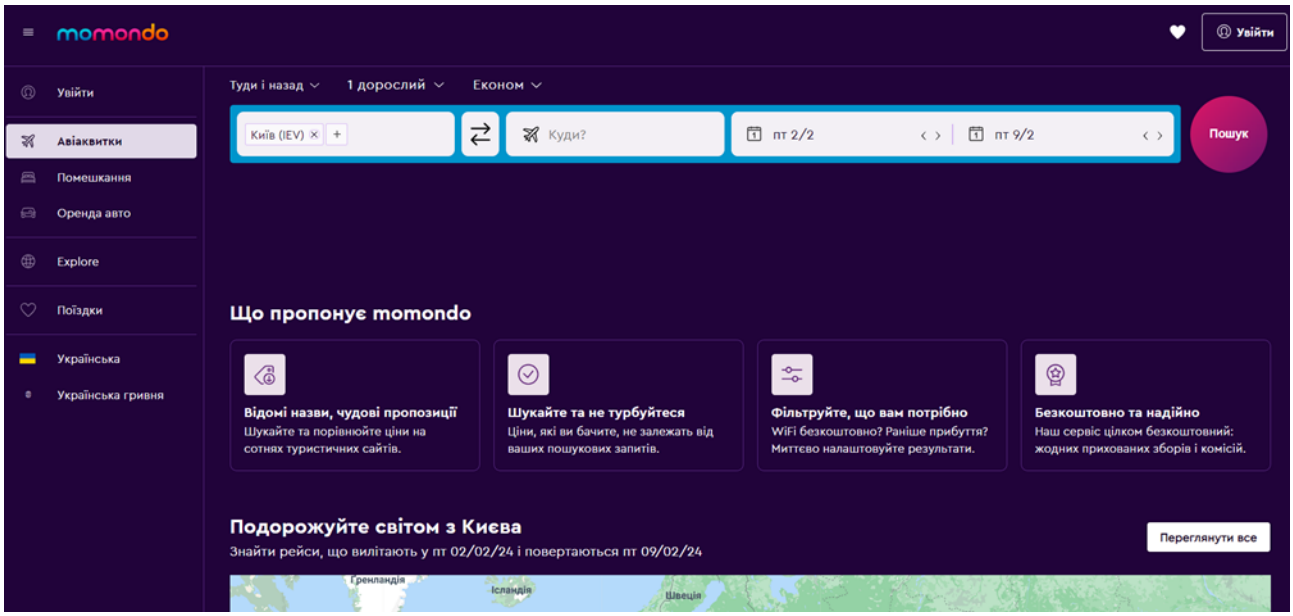

Рис. 1.5 Інтерфейс Momondo

Недоліки:

– Додаткові збори. Як і в багатьох інших сервісах, при оформленні бронювання через Momondo можуть виникнути додаткові збори та комісії;

– Можливість зміни цін під час оформлення. Ціни на квитки можуть змінюватися під час процесу бронювання, іноді виявляючи себе вищими, ніж на

етапі пошуку;

– Не завжди враховані всі авіакомпанії. Деякі низькобюджетні або регіональні авіакомпанії можуть не завжди враховуватися в результатах пошуку;

– Обмежена інформація про багаж. Інформація про правила щодо багажу може бути обмеженою, і користувачам може знадобитися перевірити деталі на веб-сайті авіакомпанії;

– Можливість затримок в оновленні результатів. Іноді результати пошуку можуть затримуватися в оновленні, особливо при пошуку складних маршрутів з численними пересадками.

6. Kayak

Kayak надає широкий вибір опцій для пошуку квитків, включаючи варіанти з пересадками. Сервіс також дозволяє встановлювати фільтри для зручності користувачів.

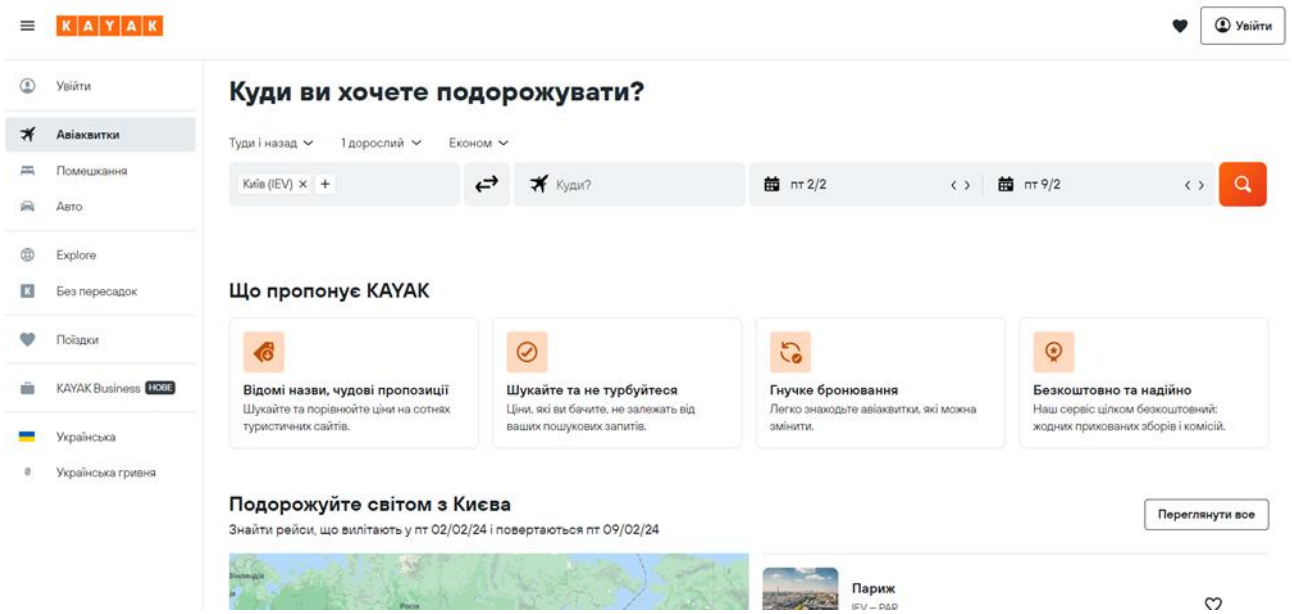

Рис. 1.6 Інтерфейс Kayak

Переваги:

– Широкий вибір варіантів. Kayak пропонує різноманітні варіанти авіаквитків, готелів, автомобілів та пакетів, що дозволяє користувачам

знаходити найбільш підходящі пропозиції;

– Зручний інтерфейс. Інтерфейс Kayak є простим та інтуїтивно зрозумілим, роблячи пошук та бронювання легкими для користувачів;

– Порівняння цін. Сервіс включає в себе ціни з різних джерел, що дозволяє користувачам порівнювати та обирати найдоступніші варіанти;

– Цінові оповіщення. Kayak дозволяє користувачам налаштовувати цінові оповіщення, отримуючи сповіщення про зміни цін на обрані напрямки;

– Пошук готелів та автомобілів. Крім авіаквитків, Kayak дозволяє користувачам шукати та бронювати готелі та автомобілі, надаючи комплексні послуги для подорожей.

Недоліки:

– Можливість зміни цін під час оформлення. Ціни на квитки та інші послуги можуть змінюватися під час процесу бронювання, що може вплинути на остаточну вартість;

– Додаткові збори та комісії. При бронюванні через Kayak можуть виникати додаткові збори та комісії, які не завжди чітко вказані на початковому етапі;

– Не завжди точні результати пошуку. Іноді результати пошуку можуть бути менш точними, особливо при пошуку складних маршрутів чи специфічних вимог;

– Обмежена інформація про багаж. Інформація про багаж та інші додаткові послуги може бути обмеженою, і користувачам може доводитися перевіряти ці деталі на веб-сайтах авіакомпаній чи інших партнерів;

– Застарілість інформації. Іноді може траплятися затримка у відображенні актуальної інформації про наявність та ціни на квитки, особливо під час великих змін у галузі авіаперевезень.

Огляд сервісів купівлі квитків вказує на різноманітність і особливості кожного з них. Перед тим як розробляти власну систему, важливо врахувати деякі ключові аспекти, щоб забезпечити ефективність та задоволення користувачів. Нижче наведено кілька висновків для розробки власної системи:

1. Широкий функціонал – система повинна підтримувати не лише купівлю авіаквитків, але й інших послуг, такі як готелі, автомобілі та пакетні пропозиції, для повного задоволення потреб користувачів.

2. Гнучкий пошук – впровадження гнучких критеріїв пошуку, таких як гнучкі дати, альтернативні аеропорти та інші параметри для зручності користувачів.

3. Цінові оповіщення, щоб користувачі могли отримувати сповіщення про зміни цін на обрані маршрути.

4. Простий та інтуїтивний інтерфейс – легкий та зрозумілий інтерфейс для користувачів, що полегшить їм використання системи та забезпечить позитивний досвід.

5. Порівняння цін від різних джерел, щоб користувачі могли знайти найвигідніші пропозиції.

6. Глобальна підтримка – можливість співпраці з різними авіакомпаніями та агентствами для глобального охоплення ринку та вибору.

7. Цінова прозорість – забезпечення користувачів точною інформацією про вартість квитків, уникнення непорозумінь та додаткових зборів під час оформлення.

8. Цінова конкурентоспроможність

9. Система лояльності для залучення постійних користувачів.

10. Інформація про багаж та інші деталі – надання користувачам повної інформації про багаж, правила авіакомпаній та інші важливі деталі.

11. Система підтримки для вирішення запитань та проблем користувачів.

1.3. Аналіз позитивних аспектів при впровадження системи у вигляді веб-додатку

Впровадження системи у вигляді веб-додатку може мати багато позитивних аспектів для різних видів діяльності, серед них можна виділити наступні:

1. Доступність

Веб-додатки можна використовувати з будь-якого пристрою, який має доступ до Інтернету та браузера. Це забезпечує зручний доступ для користувачів незалежно від їхнього місця розташування.

2. Оновлення

Веб-додатки легше оновлювати, оскільки зміни вносяться на серверному рівні. Користувачам не потрібно завантажувати та встановлювати нові версії на свої пристрої.

3. Масштабованість

Веб-додатки можуть легко масштабуватися для використання з різною кількістю користувачів. Це робить їх ідеальними для компаній та організацій будь-якого розміру.

4. Зручність в управлінні

Адміністратори можуть ефективно керувати правами доступу та іншими параметрами через веб-інтерфейс, що полегшує адміністрування системи.

5. Забезпечення безпеки

Багато веб-додатків використовують захист на рівні протоколу (наприклад, HTTPS), що робить передачу даних між користувачем і сервером безпечною. Також можливість централізованого оновлення забезпечує швидке виправлення вразливостей.

6. Інтеграція з іншими системами

Веб-додатки добре інтегруються з іншими веб-службами та API, що робить їх ідеальними для спільної роботи з іншими програмами та сервісами.

7. Зменшення витрат на обладнання

Користувачі можуть отримати доступ до веб-додатку, використовуючи будь-який пристрій, що підключений до Інтернету, що дозволяє економити витрати на обладнання та підтримку.

8. Автоматизація процесів

Веб-додатки можуть полегшити та автоматизувати різні бізнес-процеси, що дозволяє підвищити продуктивність та знизити ймовірність помилок.

Впровадження веб-додатку може значно полегшити роботу користувачів

та сприяти ефективній організації різних видів діяльності. Розглянувши переваги впровадження системи у вигляді веб-додатку можна визначити позитивні аспекти для авіакомпаній, пасажирів та інших учасників системи при впровадженні системи утворення маршрутів перельотів пасажирів з пересадками у такому вигляді:

1. Доступність – забезпечує широке коло користувачів та дозволяє їм отримувати доступ до системи з будь-якого місця у світі.

2. Зручний інтерфейс користувача. Веб-додатки мають інтуїтивно зрозумілий та легкий для користувачів інтерфейс, що полегшує їхню навігацію та використання і сприяє збільшенню попиту на додаток у користувачів.

3. Автоматичні оновлення

4. Легка інтеграція з іншими сервісами. Веб-додатки легко інтегруються з іншими веб-службами та API, що може бути важливим для підключення до інших систем, таких як бази даних авіакомпаній, метеорологічні служби та інші.

5. Можливість забезпечення високої безпеки, що є дуже важливим у додатку, де зберігається багато конфіденційної інформації користувача.

6. Масштабованість. Веб-додатки можуть легко масштабуватися для обробки великої кількості користувачів, що робить їх ефективними для великих систем, таких як система утворення маршрутів перельотів.

7. Постійний доступ до оновленої інформації. Веб-додаток може надавати користувачам актуальну інформацію про розклади, наявність місць, та інші деталі в режимі реального часу.

8. Збереження ресурсів. Користувачі можуть використовувати систему, не завантажуючи та не встановлюючи додаткові програми, що дозволяє економити ресурси пристроїв та обсяги пам'яті. Через що користувачі частіше будуть віддавати превагу саме такому типу додатку, що підвищує конкурентоспроможність серед інших схожих додатків, які не є веб.

Отже, реалізація системи у вигляді веб-додатку може значно полегшити взаємодію користувачів з нею та забезпечити більше гнучкості та зручності в її

використанні, а також має свої переваги з огляду бізнес процесів реалізації і просування.

Висновки

Були досліджені різні аспекти розробки та впровадження сайту утворення маршрутів перельотів пасажирів з пересадками. Дослідження охопило такі ключові напрямки:

Аналіз принципу купівлі авіабілетів з пересадками: були розглянуті основні принципи на які спираються пасажири при купівлі авіаквитків і додатково квитків з пересадками, а також розглянуті аспекти, які важливо врахувати під час реалізації додатку, виходячи з проведеного аналізу.

Огляд сервісів купівлі квитків: були розглянуті шість різних сервісів купівлі авіаквитків з пересадками і їх переваги і недоліки. Завдяки цьому аналізу було зроблено висновки з врахування деяких ключових аспекти, щоб забезпечити ефективність та задоволення користувачів при розробці власного додатку.

Аналіз позитивних аспектів впровадження системи у вигляді веб-додатку: були розглянуті усі позитивні аспекти використання веб-додатків для реалізації будь-яких систем і зроблені висновки з аспектами і причинами для використання саме такої моделі системи для сайту утворення маршрутів перельотів пасажирів з пересадками.

Таким чином, усі ці висновки допомагають врахувати ключові аспекти при розробці власного додатку для задоволення потреб користувачів та забезпечення ефективності системи для утворення маршрутів перельотів з пересадками.

РОЗДІЛ 2.

ВИМОГИ ДО САЙТУ УТВОРЕННЯ МАРШРУТІВ ПЕРЕЛЬОТІВ ПАСАЖИРІВ З ПЕРЕСАДКАМИ

2.1 Стек технологій для розробки

В цьому розділі буде розглянуто ключові інструменти та технології, які використовуються для розробки програмного продукта у середовищі Microsoft. Цей стек включає в себе низку потужних інструментів, що спрощують процес розробки, підвищують продуктивність та забезпечують надійність програмного забезпечення.

Microsoft Visual Studio 2022 – інтегрованого середовища розробки, яке забезпечує розширені можливості для створення, налагодження та відлагодження програмних продуктів на платформі Microsoft. Воно дозволяє розробникам створювати різноманітні додатки, включаючи веб-сайти, мобільні додатки, настільні програми та хмарні сервіси. Visual Studio 2022 підтримує мови програмування такі як C#, Visual Basic, C++, Python та інші. Серед його функцій є налагоджування, автодоповнення коду, візуальне проектування інтерфейсів, інструменти для роботи з версіями коду, а також інтегровані інструменти тестування та аналізу коду. Візуальне середовище Visual Studio сприяє підвищенню продуктивності розробників та полегшує процес створення високоякісного програмного забезпечення [4].

Під час розробки програмних рішень, база даних грає критичну роль, і для її управління можна використати Microsoft SQL Server 2022, що забезпечує високу продуктивність, масштабованість та безпеку.

Microsoft SQL Server 2022 – це реляційна система керування баз даних, розроблена Microsoft. Вона пропонує надійні, масштабовані та продуктивні рішення для зберігання, управління та аналізу даних. SQL Server 2022 має широкий спектр функцій, включаючи підтримку транзакцій, резервне копіювання та відновлення, розподілені бази даних, інтегровану безпеку та ролеву автентифікацію, аналітичні можливості, такі як векторна обробка, підтримку різних типів даних, включаючи текстові, географічні та XML дані, а також інструменти для розробки додатків, які використовують бази даних, такі як IDE та різноманітні додаткові компоненти та бібліотеки. SQL Server 2022 підтримує як локальні, так і хмарні розгортання, що робить його відмінним вибором для широкого спектра застосувань [5].

При розробці веб-застосунків використовуємо C# разом з ASP.NET MVC Framework, що дозволяє швидко створювати потужні та масштабовані веб-додатки. Для доступу до даних використовуємо Entity Framework, який надає зручний механізм роботи з базою даних.

C# – це сучасна, об'єктно-орієнтована мова програмування, розроблена компанією Microsoft. Вона поєднує в собі потужність та простоту використання, що робить її популярним вибором для розробки різних типів програм, включаючи веб-додатки, мобільні додатки, настільні програми, ігри та багато іншого. C# є частиною платформи .NET, що дозволяє розробникам використовувати широкий спектр інструментів та бібліотек для швидкої та ефективної розробки програмного забезпечення. Вона має ряд переваг, включаючи безпеку типів, автоматичне керування пам'яттю, розширену підтримку багатопотоковості та асинхронного програмування, а також багато інших функцій, що сприяють швидкому та надійному розробленню програм [6].

ASP.NET MVC Framework – це фреймворк для розробки веб-додатків на платформі .NET. MVC розділяє аспекти веб-додатка на дані, представлення та логіку програми, що сприяє розділенню обов'язків та полегшує підтримку та розширення коду. ASP.NET MVC надає широкий набір інструментів для

створення динамічних веб-сторінок та додатків, включаючи вбудовану підтримку AJAX, маршрутизації URL, перевірки даних на стороні клієнта та сервера, аутентифікації та авторизації, а також інтеграцію з іншими компонентами .NET та сторонніми бібліотеками. Цей фреймворк забезпечує швидку розробку, підтримку тестування та можливість легкого внесення змін у веб-додатки [7].

Entity Framework – це технологія доступу до даних в середовищі .NET, що дозволяє розробникам працювати з даними в базі даних за допомогою об'єктно-орієнтованого підходу. EF дозволяє визначати моделі даних в коді як класи та взаємозв'язки між ними, а фреймворк автоматично генерує відповідні структури бази даних. Це спрощує розробку та підтримку додатків, оскільки розробникам не потрібно писати складні запити SQL або вручну мапувати дані між об'єктами та таблицями бази даних. За допомогою Entity Framework можна виконувати операції з базою даних, такі як додавання, оновлення, видалення та запити, використовуючи Language Integrated Query або SQL. EF підтримує різні типи баз даних, включаючи SQL Server, MySQL, PostgreSQL та інші, що робить його універсальним інструментом для доступу до даних у .NET додатках [9].

Забезпечуючи аутентифікацію та авторизацію користувачів у веб-додатках, використовуємо Identity ASP.NET Core, що дозволяє легко і безпечно керувати доступом до ресурсів.

Identity ASP.NET Core – це система аутентифікації та авторизації для веб-додатків на платформі .NET Core. Вона надає готові компоненти для реалізації основних функцій, таких як реєстрація користувачів, управління ролями та дозволами, підтримка зберігання паролів у зашифрованому вигляді, двофакторна аутентифікація та інші механізми безпеки. Завдяки Identity ASP.NET Core розробники можуть легко і швидко інтегрувати систему аутентифікації та авторизації у свої веб-додатки, що дозволяє зосередитися на функціональності додатка, замість написання складного коду для забезпечення безпеки [8].

Для створення користувацького інтерфейсу використовуємо технології, такі як cshtml для розробки сторінок, JavaScript для динамічного взаємодії з користувачем та CSS для оформлення та стилізації веб-сторінок.

cshtml – це розширення файлів, що використовується у веб-розробці на платформі .NET, зокрема в ASP.NET Core. Файли cshtml поєднують HTML-код з кодом серверного боку, який може бути написаний на мові програмування C# або інших мовах .NET. Це дозволяє розробникам створювати динамічні веб-сторінки, які можуть взаємодіяти з базою даних або іншими сервісами. Файли cshtml дозволяють використовувати різні вбудовані конструкції, такі як цикли, умовні вирази та вбудовувати серверний код прямо в HTML, що робить розробку веб-додатків на платформі .NET більш зручною та ефективною [10].

JavaScript – це мова програмування, що використовується для розробки веб-додатків. Вона широко використовується для створення динамічних інтерактивних елементів на веб-сторінках, таких як анімація, валідація форм, взаємодія з користувачем, асинхронні запити на сервер і багато іншого. JavaScript може працювати у веб-браузері користувача, а також на серверній стороні за допомогою платформи Node.js. Вона є важливим компонентом веб-розробки і дозволяє створювати багатофункціональні та інтерактивні веб-додатки [11].

CSS – це мова опису стилів, яка використовується для оформлення веб-сторінок і контенту на них. За допомогою CSS можна задавати різноманітні стилі, такі як кольори, шрифти, розміри, відступи, рамки та інші властивості елементів HTML. Це дозволяє розробникам контролювати вигляд та розташування елементів на веб-сторінці, щоб створювати привабливий та зручний дизайн для користувача. CSS також дозволяє робити сторінки адаптивними до різних пристроїв і розмірів екранів, щоб забезпечити оптимальний вигляд на будь-яких пристроях [12].

В цьому розділ було детально розглянуто кожну з цих технологій та їх використання в процесі розробки програмного продукту.

2.2 Апаратна та операційна платформа

Вимоги до пристрою користувача для роботи з сайтом утворення маршрутів перельотів пасажирів з пересадками включають наступне:

1. Інтернет-підключення. Сайт потребує доступу до Інтернету. Швидкість Інтернет-підключення може впливати на продуктивність та швидкість завантаження сторінок.

2. Веб-браузер. Користувач повинен мати веб-браузер, сумісний з веб-застосунком. Популярні браузери, такі як Google Chrome, Mozilla Firefox, Safari або Microsoft Edge, підтримують сайт.

3. Операційна система. Веб-застосунок можє працювати на різних операційних системах, таких як Windows, MacOS, Linux або мобільні операційні системи, таких як iOS або Android.

4. Роздільна здатність екрану. Сайт оптимізований для мобільних пристроїв і для великих екранів комп'ютерів.

5. Можливість взаємодії зі специфічними елементами керування. Веб-застосунок може використовувати специфічні елементи керування, такі як тачскрін, миша або клавіатура. Відповідно, пристрій користувача повинен мати можливість взаємодіяти з такими елементами.

2.3 Вимоги до функціоналу, інтерфейсу та безпеки

Створення маршрутів перельотів з пересадками вимагає від веб-застосунку, який надає такі послуги, високого рівня функціональності, зручного інтерфейсу та надійної системи безпеки. Даний розділ ставить за мету визначення вимог до цих аспектів для успішної реалізації сайту утворення маршрутів перельотів пасажирів з пересадками.

Вимоги до функціоналу:

– Пошук та вибір оптимальних маршрутів. Система повинна забезпечувати користувачу можливість швидкого та зручного пошуку

маршрутів з пересадками з урахуванням різних критеріїв, таких як місто або конкретний аеропорт відправки і прибуття і дата подорожі.

– Відображення детальної інформації. Після вибору маршруту користувач повинен мати можливість отримати детальну інформацію про кожен перельот, включаючи розклад, тривалість пересадок тощо.

– Бронювання та оплата. Функціонал сайту повинен дозволяти користувачам зручно та безпечно здійснювати бронювання та оплату обраних маршрутів.

Вимоги до інтерфейсу:

– Інтуїтивно зрозумілий дизайн. Сайт повинен мати зрозумілий та легкий у використанні інтерфейс, що дозволить користувачам швидко зорієнтуватися та здійснити необхідні дії без додаткових пояснень.

– Мобільна сумісність. З урахуванням поширеності мобільних пристроїв, інтерфейс повинен бути адаптований для коректного відображення на різних типах екранів та пристроях.

Вимоги до безпеки:

– Захист персональних даних. Система повинна забезпечувати надійний захист персональних даних користувачів, включаючи дані про оплату та особисту інформацію.

– Шифрування транзакцій. Усі фінансові транзакції, пов'язані з оплатою бронювання перельотів, повинні бути шифровані для захисту від несанкціонованого доступу до платіжних даних.

– Захист від кібератак. Система повинна мати вбудовані заходи захисту від різних видів кібератак, таких як DDoS атаки, SQL ін'єкції тощо.

Враховуючи ці вимоги, можна створити високоякісний та безпечний веб-застосунок для утворення маршрутів перельотів з пересадками, що задовольнить потреби користувачів та забезпечить їхню безпеку під час використання.

Висновки

У цьому розділі були розглянуті вимоги до сайту утворення маршрутів перельотів з пересадками з точки зору технологій, апаратної та операційної платформи, функціоналу, інтерфейсу та безпеки.

Стек технологій для розробки визначено як Microsoft Visual Studio 2022, Microsoft SQL Server 2022, C#, ASP.NET MVC Framework, Identity ASP.NET Core, Entity Framework, cshtml, JavaScript та css. Використання такого стеку технологій забезпечить швидку та ефективну розробку веб-застосунку з врахуванням сучасних стандартів розробки.

Апаратна та операційна платформа мають відповідати вимогам підтримки обраного стеку технологій, забезпечуючи оптимальну продуктивність та надійність роботи веб-застосунку на різних пристроях та операційних системах.

Вимоги до функціоналу, інтерфейсу та безпеки включають в себе ряд ключових аспектів, таких як можливість швидкого пошуку та вибору маршрутів, зручний та інтуїтивно зрозумілий інтерфейс користувача, а також надійну захисту персональних даних та фінансових транзакцій. Дотримання цих вимог дозволить створити веб-застосунок, який буде задовольняти потреби користувачів та забезпечувати їхню безпеку та задоволення від використання.

Реалізація вказаних вимог дозволить створити високоякісний та конкурентоздатний веб-застосунок для утворення маршрутів перельотів пасажирів з пересадками, що відповідає сучасним стандартам розробки та вимогам користувачів.

РОЗДІЛ 3.
РЕАЛІЗАЦІЯ САЙТУ ДЛЯ УТВОРЕННЯ МАРШРУТІВ ПЕРЕЛЬОТІВ ПАСАЖИРІВ З ПЕРЕСАДКАМИ

3.1 Модель бази даних

База даних є одним із ключових компонентів будь-якої програми, що працює зі значною кількістю інформації. Для програми, що відповідає за утворення маршрутів перельотів пасажирів з пересадками, база даних відіграє важливу роль у збереженні та організації інформації. У цьому розділі детально розглянемо структуру бази даних для програми. Опишемо таблиці, поля, відносини між ними, а також важливі атрибути, які необхідно врахувати для забезпечення ефективної та надійної роботи програми. Розуміння структури та функціоналу бази даних допомагає розробнику ефективно реалізувати програму та забезпечити її надійність та швидкодію [13].

База даних має наступну структуру:

– Таблиця «Локація»: Первинний ключ (location_id): унікальний ідентифікатор локації, Місто (city): назва міста, Країна (country): назва країни.

– Таблиця «Аеропорт»: Первинний ключ (airport_id): унікальний ідентифікатор аеропорту, Повна назва (fullTitle): повна назва аеропорту, Коротка назва (shortTitle): коротка назва аеропорту, Зовнішній ключ (location_id): зв'язок з таблицею «Локація».

– Таблиця «Авіокомпанія»: Первинний ключ (airline_id): унікальний ідентифікатор авіокомпанії, Назва (title): назва авіокомпанії.

– Таблиця «Літак»: Первинний ключ (airplane_id): унікальний ідентифікатор літака, Назва (title): назва літака, Кількість місць економ (number_economy): кількість місць в економ-класі, Кількість місць бізнес (number_business): кількість місць в бізнес-класі, Кількість місць першого (number_first): кількість місць в першому-класі, Кількість місць преміум (number_ premium): кількість місць в преміум-класі, Зовнішній ключ (airline_id): зв'язок з таблицею «Авіокомпанія».

– Таблиця «Політ»: Первинний ключ (flight_id): унікальний ідентифікатор політу, Дата і час відправки (departure_datetime): дата та час відправлення, Дата і час прибуття (arrival_datetime): дата та час прибуття, Ціна (price): вартість квитка, Зовнішній ключ (airplane_id): зв'язок з таблицею «Літак», Зовнішній ключ (departureAirport_id): зв'язок з таблицею «Аеропорт» для визначення аеропорту відправлення, Зовнішній ключ (arrivalAirport_id): зв'язок з таблицею «Аеропорт» для визначення аеропорту прибуття.

У цій базі даних існує кілька зв'язків між сутностями:

– Зв'язок між таблицею «Аеропорт» і «Локація». У таблиці «Аеропорт» існує зовнішній ключ (location_id), який посилається на первинний ключ таблиці "Локація" (location_id). Цей зв'язок визначає місце розташування аеропорту у відповідному місті та країні.

– Зв'язок між таблицею «Літак» і «Авіокомпанія». У таблиці «Літак» є зовнішній ключ (airline_id), який посилається на первинний ключ таблиці «Авіокомпанія» (airline_id). Цей зв'язок вказує на те, які авіакомпанії володіють або експлуатують конкретні літаки.

– Зв'язок між таблицею «Політ» і «Літак». У таблиці «Політ» є зовнішній ключ (airplane_id), який посилається на первинний ключ таблиці «Літак» (airplane_id). Цей зв'язок вказує на те, який літак здійснює конкретний рейс.

– Зв'язок між таблицею «Політ» і «Аеропорт» для аеропортів відправлення та прибуття. У таблиці «Політ» є два зовнішніх ключі (departureAirport_id і arrivalAirport_id), які посилаються на первинний ключ

таблиці «Аеропорт» (airport_id). Перший ключ вказує на аеропорт відправлення, а другий – на аеропорт прибуття.

Ці зв'язки допомагають у відображенні залежностей між різними сутностями в базі даних і спрощують взаємодію між ними при здійсненні операцій, таких як пошук, оновлення та видалення даних.

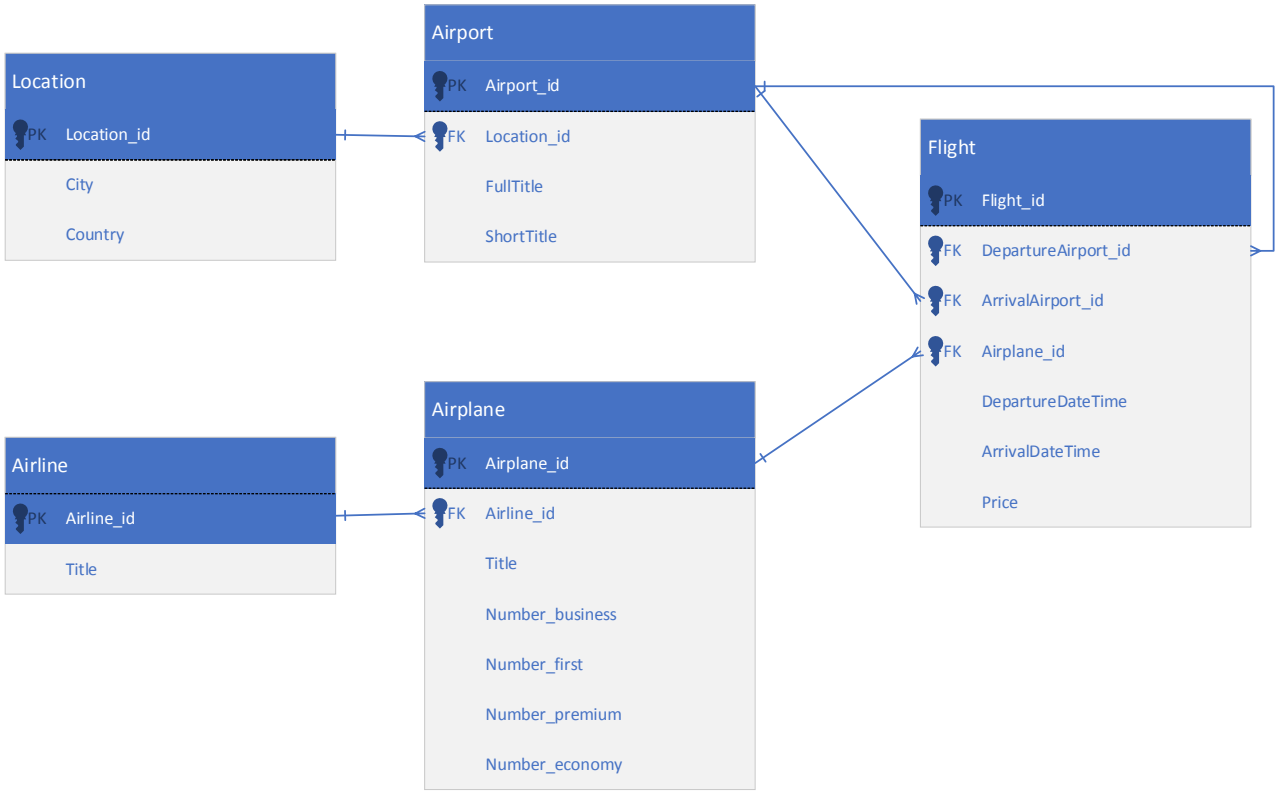

Рис. 3.1 Діаграма бази даних

3.2 Діаграма класів програми

У цьому розділі буде розглянуто структуру програми за допомогою діаграми класів, яка відображає взаємозв'язки між класами та їхні атрибути та методи. Діаграма класів також відображає взаємозв'язки між класами. Кожен клас має свої атрибути та методи, які визначають його поведінку та властивості.

Діаграма класів програми відображає основні класи, які використовуються для реалізації функціональності веб-додатку утворення маршрутів перельотів пасажирів з пересадками.

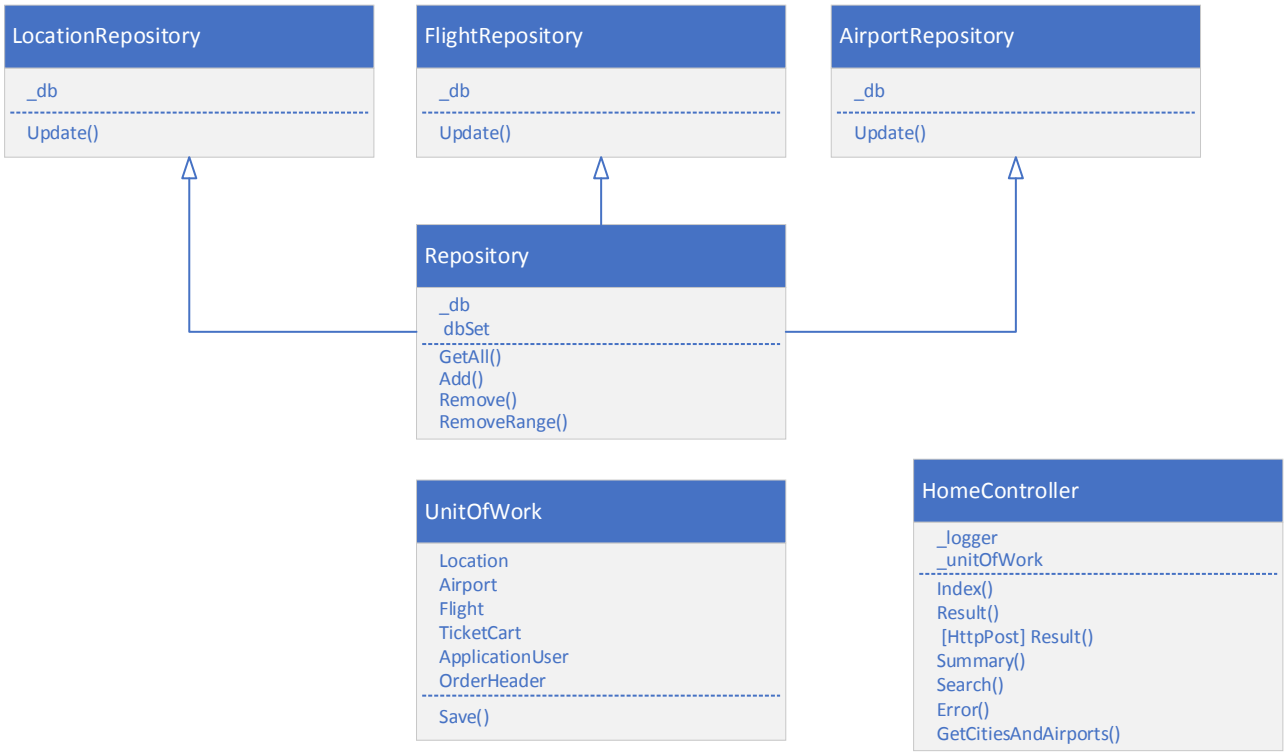

Рис. 3.2 Діаграма класів

Діаграма класів програми дозволяє краще розібратися у структурі програми та взаємозв'язках між її складовими частинами. Вона служить важливим інструментом для розробників під час розробки та підтримки програмного забезпечення.

3.3 Проектування інтерфейсу користувача

У цьому розділі розглянемо проектування інтерфейсу користувача для веб-додатку утворення маршрутів перельотів пасажирів з пересадками.

Загальна концепція дизайну є наступною: дизайн інтерфейсу користувача має бути зручним, інтуїтивно зрозумілим та привабливим для користувачів. Основні принципи, які були використані при проектуванні інтерфейсу, включають:

– Простота: мінімізація складності та зайвих елементів для забезпечення зрозумілості інтерфейсу.

– Зручність: легка навігація та швидкий доступ до основних функцій додатку.

– Контрастність: використання контрастних кольорів для підвищення видимості важливих елементів інтерфейсу.

– Адаптивність: забезпечення коректного відображення на різних пристроях та розмірах екранів.

Структура інтерфейсу розділена на різні розділи та компоненти для забезпечення логічності та зручності використання. Основні компоненти інтерфейсу включають:

– Головне меню: доступ до основних функцій додатку, таких як пошук рейсів, перегляд маршрутів, реєстрація та вхід в особистий кабінет тощо.

– Форма пошуку рейсів: введення критеріїв пошуку, таких як місце відправлення, місце прибуття, дата тощо.

– Список результатів пошуку: відображення доступних рейсів згідно введених критеріїв.

– Особистий кабінет: перегляд інформації про користувача.

– Форма оформлення броні: оформлення замовлення на обраний користувачем квиток.

– Посилання на додатки для оформлення оренди житла та транспорту.

При проектуванні інтерфейсу використовувалися різноманітні елементи для покращення візуальної привабливості та зручності використання, такі як:

– Кнопки та посилання: для виклику різних функцій та переходу між сторінками.

– Форми введення даних: для введення різних параметрів пошуку та реєстрації.

– Списки: для відображення результатів пошуку та іншої структурованої інформації.

Проектування інтерфейсу користувача було проведено з урахуванням потреб та очікувань користувачів, що дозволило створити зручний та

ефективний інтерфейс для веб-додатку утворення маршрутів перельотів пасажирів з пересадками.

3.4 Модуль автодоповнення

Модуль автодоповнення є важливою частиною інтерфейсу користувача, яка спрощує процес введення даних та забезпечує швидкий та точний пошук необхідної інформації. У цьому розділі буде розглянуто проектування та реалізація модуля автодоповнення для веб-додатку утворення маршрутів перельотів пасажирів з пересадками.

Модуль автодоповнення призначений для автоматичного заповнення поля введення на основі вже існуючої інформації в системі. Основні функції модуля включають автодоповнення місць відправлення та прибуття, поля введення місць відправлення та прибуття можуть автоматично доповнюватися на існуючих маршрутів в системі.

Модуль автодоповнення реалізований з використанням технологій JavaScript та AJAX. При введенні даних у відповідному полі, відбувається асинхронний запит до сервера, який повертає список можливих варіантів автодоповнення. На основі цього списку, відображається випадаючий список або підказки для вибору користувачем.

```
[HttpGet]
        public IActionResult GetCitiesAndAirports(string obj)
        {
            if (obj != null)
            {
                var citiesAndAirports = _unitOfWork.Location.GetAll()
                    .Where(city => city.City.ToLower().Contains(obj.ToLower()))
                    .Select(city => new
                    {
                        id = $"city_{city.Id}",
                        originalId = city.Id,
                        text = city.City,
                        country = city.Country,
```

```
                                children    =    _unitOfWork.Airport.GetAll(airport    =>
airport.LocationId == city.Id)
                                    .Select(airport => new
                                    {
                                        id = $"airport_{airport.Id}",
                                        originalId = airport.Id,
                                        text = airport.FullTitle
                                    }).ToList()
                            }).ToList<object>();

                        return Json(citiesAndAirports);
                    }
                    return null;
                }
        function loadToSelect(selector) {
            $(selector).select2({
                allowClear: true,
                placeholder: "",
                maximumSelectionLength: 1,
                ajax: {
                    url: '/customer/home/GetCitiesAndAirports',
                    type: 'GET',
                    dataType: 'json',
                    delay: 250,
                    cache: true,
                    data: function (params) {
                        return {
                            obj: params.term
                        };
                    },
                    processResults: function (data) {
                        var results = data.flatMap(function (city) {
                            return [{
                                id: city.id,
                                originalId: city.originalId,
                                text: city.text,
                                country: city.country,
                                isCity: true
                            }].concat(city.children.map(function (airport) {
                                return {
                                    id: airport.id,
                                    originalId: airport.originalId,
```

```
                            text: airport.text,
                            isCity: false
                        };
                    }));
                });
                return {
                    results: results
                };
            }
        },
        templateResult: function (data) {
            if (data.isCity) {
                return $('<i class="bi bi-geo-alt ms-2"></i><span><b> ' + data.text + ", " + data.country + ' </b></span>');
            } else {
                return $('<i class="bi bi-airplane ms-4"></i><span> ' + data.text + ' </span>');
            }
        }
    });
}
```

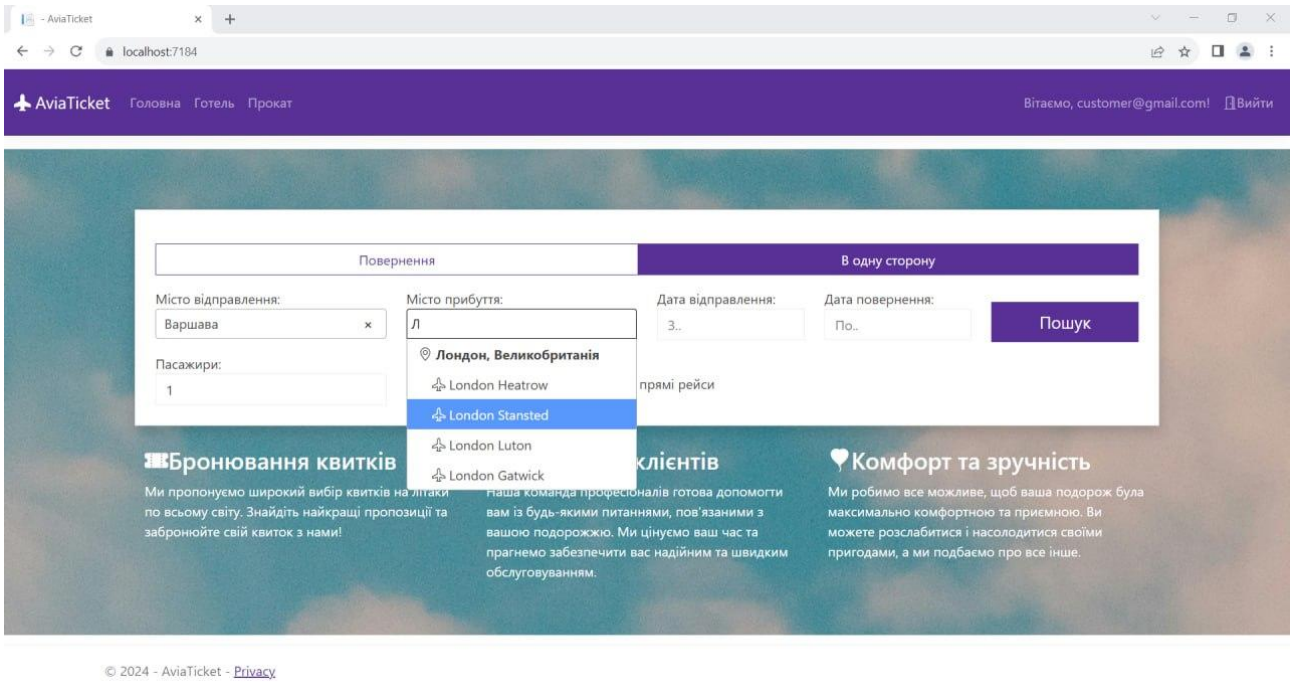

Рис. 3.3 Результат реалізації модуля автодоповнення

Модуль автодоповнення був протестований з використанням різних наборів тестових даних для перевірки правильності та швидкості автодоповнення. Під час тестування перевірялася коректність відображення підказок, швидкість реакції системи та загальна зручність використання(див. рис. 3.3).

Модуль автодоповнення є важливою складовою частиною інтерфейсу користувача, яка спрощує та прискорює процес введення даних. Його правильне проектування та реалізація дозволяє полегшити використання веб-додатку та покращити досвід користувача.

3.5 Модуль пошуку рейсу

Модуль пошуку рейсу є ключовою складовою частиною веб-додатку для утворення маршрутів перельотів пасажирів з пересадками. У цьому розділі розглянемо проектування та функціональні можливості цього модуля.

Модуль пошуку рейсу надає користувачеві можливість здійснювати пошук доступних рейсів з урахуванням різних критеріїв. Основні функції цього модуля включають: пошук за місцем відправлення та прибуття, користувач вказує місце відправлення та місце прибуття для пошуку рейсів між цими пунктами; пошук за датою та часом, користувач вказує дату та час вильоту для пошуку рейсів, що відповідають його потребам. Користувач може переглянути докладну інформацію про знайдені рейси, включаючи час вильоту, час прибуття, тривалість перельоту, наявність пересадок та інші деталі.

Модуль пошуку рейсу реалізований з використанням технологій JavaScript для взаємодії зі сторінкою користувача та AJAX для виконання асинхронних запитів до сервера. Після отримання результатів від сервера, вони відображаються на сторінці у відповідному форматі.

```
    public IActionResult Result(int departureId, bool departureIsCity, int arrivalId, bool arrivalIsCity, string selectedDate)
    {
        List<IEnumerable<Flight>> flightList = new List<IEnumerable<Flight>> { };
```

```
            IEnumerable<Flight>    flights1,    flights2,    connectingFlightsDeparture,
connectingFlightsArrival;
            if (departureIsCity && arrivalIsCity)
            {
                flights1 = _unitOfWork.Flight.GetAll(f => f.DepartureAirport.LocationId
== departureId
                    && f.ArrivalAirport.LocationId == arrivalId &&
                    f.DepartureDateTime.Date    ==    DateTime.Parse(selectedDate).Date,
includeProperties: "ArrivalAirport,DepartureAirport");
                connectingFlightsDeparture     =     _unitOfWork.Flight.GetAll(f     =>
f.DepartureAirport.LocationId == departureId
                    &&  f.DepartureDateTime.Date  ==  DateTime.Parse(selectedDate).Date,
includeProperties: "ArrivalAirport,DepartureAirport");
                connectingFlightsArrival      =      _unitOfWork.Flight.GetAll(f      =>
f.ArrivalAirport.LocationId == arrivalId
                    &&  f.DepartureDateTime.Date  ==  DateTime.Parse(selectedDate).Date,
includeProperties: "ArrivalAirport,DepartureAirport");
                flights2 = connectingFlightsDeparture
                    .SelectMany(departureFlight =>
                        connectingFlightsArrival.Where(arrivalFlight =>
                            departureFlight.ArrivalAirport.LocationId                   ==
arrivalFlight.DepartureAirport.LocationId
                            &&          departureFlight.DepartureDateTime.Date          ==
arrivalFlight.DepartureDateTime.Date)
                        .SelectMany(arrivalFlight    =>    new[]    {    departureFlight,
arrivalFlight })
                    );
            }
            else if (!departureIsCity && !arrivalIsCity)
            {
                flights1  =  _unitOfWork.Flight.GetAll(f  =>  f.DepartureAirportId  ==
departureId
                    && f.ArrivalAirportId == arrivalId &&
                    f.DepartureDateTime.Date    ==    DateTime.Parse(selectedDate).Date,
includeProperties: "ArrivalAirport,DepartureAirport");
                connectingFlightsDeparture     =     _unitOfWork.Flight.GetAll(f     =>
f.DepartureAirportId == departureId
                    &&  f.DepartureDateTime.Date  ==  DateTime.Parse(selectedDate).Date,
includeProperties: "ArrivalAirport,DepartureAirport");
                connectingFlightsArrival      =      _unitOfWork.Flight.GetAll(f      =>
f.ArrivalAirportId == arrivalId
```

```
                && f.DepartureDateTime.Date == DateTime.Parse(selectedDate).Date,
includeProperties: "ArrivalAirport,DepartureAirport");
            flights2 = connectingFlightsDeparture
                .SelectMany(departureFlight =>
                    connectingFlightsArrival.Where(arrivalFlight =>
                        departureFlight.ArrivalAirport.LocationId ==
arrivalFlight.DepartureAirport.LocationId
                        && departureFlight.DepartureDateTime.Date ==
arrivalFlight.DepartureDateTime.Date)
                    .SelectMany(arrivalFlight => new[] { departureFlight,
arrivalFlight })
                );
        }
        else if (departureIsCity && !arrivalIsCity)
        {
            flights1 = _unitOfWork.Flight.GetAll(f => f.DepartureAirport.LocationId
== departureId
                && f.ArrivalAirportId == arrivalId &&
                f.DepartureDateTime.Date == DateTime.Parse(selectedDate).Date,
includeProperties: "ArrivalAirport,DepartureAirport");
            connectingFlightsDeparture = _unitOfWork.Flight.GetAll(f =>
f.DepartureAirport.LocationId == departureId
                && f.DepartureDateTime.Date == DateTime.Parse(selectedDate).Date,
includeProperties: "ArrivalAirport,DepartureAirport");
            connectingFlightsArrival = _unitOfWork.Flight.GetAll(f =>
f.ArrivalAirportId == arrivalId
                && f.DepartureDateTime.Date == DateTime.Parse(selectedDate).Date,
includeProperties: "ArrivalAirport,DepartureAirport");
      flights2 = connectingFlightsDeparture
                .SelectMany(departureFlight =>
                    connectingFlightsArrival.Where(arrivalFlight =>
                        departureFlight.ArrivalAirport.LocationId ==
arrivalFlight.DepartureAirport.LocationId
                        && departureFlight.DepartureDateTime.Date ==
arrivalFlight.DepartureDateTime.Date)
                    .SelectMany(arrivalFlight => new[] { departureFlight,
arrivalFlight })
                );
        }
        else
        {
```

```
            flights1 = _unitOfWork.Flight.GetAll(f => f.DepartureAirportId ==
departureId
                && f.ArrivalAirport.LocationId == arrivalId &&
                f.DepartureDateTime.Date == DateTime.Parse(selectedDate).Date,
includeProperties: "ArrivalAirport,DepartureAirport");
            connectingFlightsDeparture = _unitOfWork.Flight.GetAll(f =>
f.DepartureAirportId == departureId
                && f.DepartureDateTime.Date == DateTime.Parse(selectedDate).Date,
includeProperties: "ArrivalAirport,DepartureAirport");
            connectingFlightsArrival = _unitOfWork.Flight.GetAll(f =>
f.ArrivalAirport.LocationId == arrivalId
                && f.DepartureDateTime.Date == DateTime.Parse(selectedDate).Date,
includeProperties: "ArrivalAirport,DepartureAirport");
            flights2 = connectingFlightsDeparture
                .SelectMany(departureFlight =>
                    connectingFlightsArrival.Where(arrivalFlight =>
                        departureFlight.ArrivalAirport.LocationId ==
arrivalFlight.DepartureAirport.LocationId
                        && departureFlight.DepartureDateTime.Date ==
arrivalFlight.DepartureDateTime.Date)
                    .SelectMany(arrivalFlight => new[] { departureFlight,
arrivalFlight })
                );
        }
        flightList = new List<IEnumerable<Flight>>();
        if (flights1 != null)
        {
            foreach (var flight in flights1)
            {
                List<Flight> flightEnum = new List<Flight> { flight };
                flightList.Add(flightEnum);
            }
        }
        if (flights2 != null)
        {
            flightList.Add(flights2);
        }
        return View(flightList);
    }
```

Модуль пошуку рейсу був підданий ретельному тестуванню для перевірки правильності та швидкості пошуку рейсів. Тестування включало в себе введення різних комбінацій критеріїв пошуку та перевірку результатів на відповідність введеним параметрам (див. рис. 3.4).

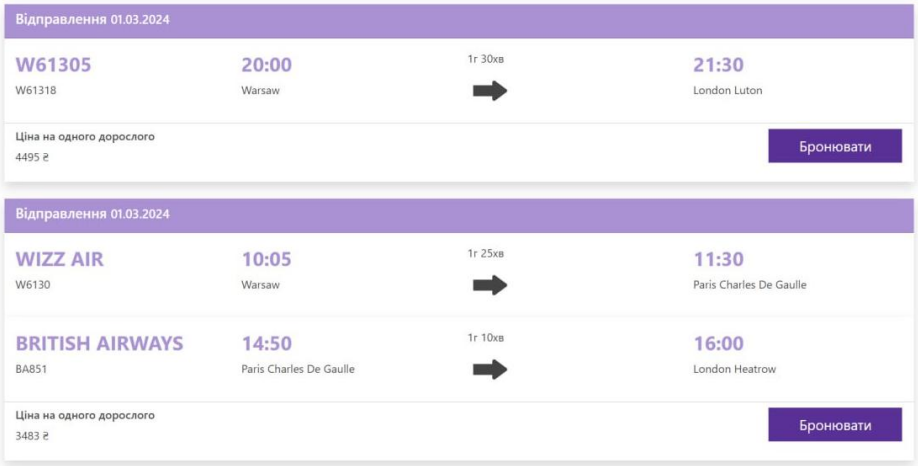

Рис. 3.4 Результат реалізації модуля пошуку рейсів

Модуль пошуку рейсу є важливою функціональною частиною веб-додатку, яка дозволяє користувачам швидко та зручно знаходити необхідні рейси для подорожей. Правильне проектування та реалізація цього модуля дозволяє забезпечити ефективність та задоволення користувачів.

3.6 Модуль реєстрації

Модуль реєстрації з використанням Entity Framework є важливою частиною веб-додатку, яка відповідає за збереження та управління даними користувачів у базі даних. У цьому розділі ми розглянемо, реалізацію функціоналу реєстрації користувачів.

Для реалізації модуля реєстрації, необхідно визначити модель даних для користувачів. У цьому випадку, модель користувача містить такі поля, як ім'я, прізвище, адреса електронної пошти та пароль тощо.

При реалізації модуля реєстрації також потрібно враховувати аспекти безпеки. Наприклад, паролі користувачів повинні бути шифровані перед

збереженням у базі даних. Для цього можна використовувати хеш-функції, наприклад, SHA256.

```
      public async Task<IActionResult> OnPostAsync(string returnUrl = null)
              {
                  returnUrl ??= Url.Content("~/");
                  ExternalLogins = (await
_signInManager.GetExternalAuthenticationSchemesAsync()).ToList();
                  if (ModelState.IsValid)
                  {
                      var user = CreateUser();

                      await _userStore.SetUserNameAsync(user, Input.Email,
CancellationToken.None);
                      await _emailStore.SetEmailAsync(user, Input.Email,
CancellationToken.None);
                      user.Name = Input.Name;
                      user.StreetAddress = Input.StreetAddress;
                      user.City = Input.City;
                      user.PostalCode = Input.PostalCode;
                      user.State = Input.State;
                      user.PhoneNumber = Input.PhoneNumber;
                      var result = await _userManager.CreateAsync(user,Input.Password);
                      if (result.Succeeded)
                      {
                          _logger.LogInformation("User created a new account with
password.");
                          await _userManager.AddToRoleAsync(user, SD.Role_Customer);
                          var userId = await _userManager.GetUserIdAsync(user);
                          var code = await
_userManager.GenerateEmailConfirmationTokenAsync(user);
                          code =
WebEncoders.Base64UrlEncode(Encoding.UTF8.GetBytes(code));
                          var callbackUrl = Url.Page(
                              "/Account/ConfirmEmail",
                              pageHandler: null,
                              values: new { area = "Identity", userId = userId, code =
code, returnUrl = returnUrl },
                              protocol: Request.Scheme);
                          await _emailSender.SendEmailAsync(Input.Email, "Confirm your
email",
```

```
                        $"Please confirm your account by <a
href='{HtmlEncoder.Default.Encode(callbackUrl)}'>clicking here</a>.");

                    if (_userManager.Options.SignIn.RequireConfirmedAccount)
                    {
                        return RedirectToPage("RegisterConfirmation", new { email
= Input.Email, returnUrl = returnUrl });
                    }
                    else
                    {
                        await _signInManager.SignInAsync(user, isPersistent:
false);
                        return LocalRedirect(returnUrl);
                    }
                }
                foreach (var error in result.Errors)
                {
                    ModelState.AddModelError(string.Empty, error.Description);
                }
            }
            return Page();
        }
```

Після реалізації модулю реєстрації важливо провести тестування для перевірки правильності його функціонування. Це було зроблено шляхом створення тестових сценаріїв, що включають реєстрацію нового користувача та перевірку наявності цього користувача у базі даних (див. рис. 3.5).

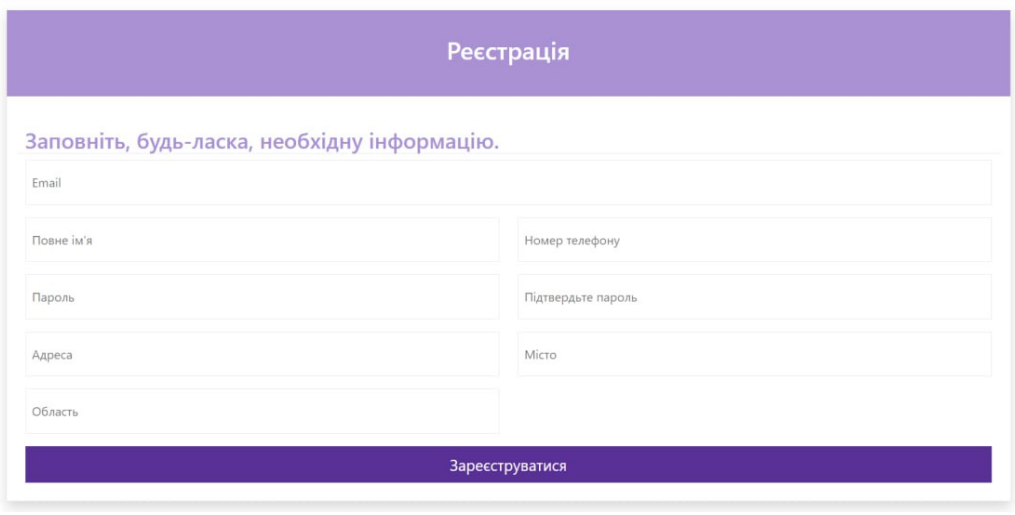

Рис. 3.5 Результат реалізації модуля реєстрації

Модуль реєстрації з використанням Entity Framework є важливою складовою частиною веб-додатку, яка дозволяє зберігати та управляти даними користувачів у базі даних. Правильна реалізація цього модуля дозволяє забезпечити безпеку та ефективність реєстраційного процесу користувачів.

Висновки

У даному розділі було розглянуто процес реалізації сайту утворення маршрутів перельотів пасажирів з пересадками.

Розроблено та реалізовано модель бази даних, яка забезпечує збереження та організацію необхідної інформації про рейси, літаки, аеропорти та інші важливі дані. Модель бази даних була структурована таким чином, щоб забезпечити ефективне та оптимізоване зберігання даних.

Була створена діаграма класів програми, що відображає структуру та взаємозв'язки між класами програми. Ця діаграма була використана для кращого розуміння архітектури програми та планування подальшого розвитку.

Був розроблений інтерфейс користувача, який включає в себе різноманітні елементи та функціональність для зручного та інтуїтивного використання веб-додатку. Проектування інтерфейсу було здійснено з урахуванням потреб та вимог користувачів.

Був реалізований модуль автодоповнення, який допомагає користувачам з швидкістю та точністю введення необхідних даних. Цей модуль сприяє полегшенню процесу заповнення форм та зменшує ймовірність помилок.

Реалізовано модуль пошуку рейсу, який дозволяє користувачам швидко та зручно знаходити необхідний рейс за різними критеріями, такими як дата вильоту, місто, аеропорт та інші.

Був створений модуль реєстрації, який дозволяє користувачам створювати облікові записи та отримувати доступ до особистого кабінету на веб-сайті. Цей модуль забезпечує зручний та безпечний спосіб реєстрації користувачів.

У результаті реалізації цих аспектів веб-додатку було досягнуто успішної інтеграції функціональності та забезпечено зручний та ефективний інтерфейс для користувачів.

РОЗДІЛ 4.
РЕЗУЛЬТАТИ РОБОТИ ПРОГРАМИ

4.1. Тестування програми

Для тестування програми виконаємо дії з функціоналом веб-додатку.

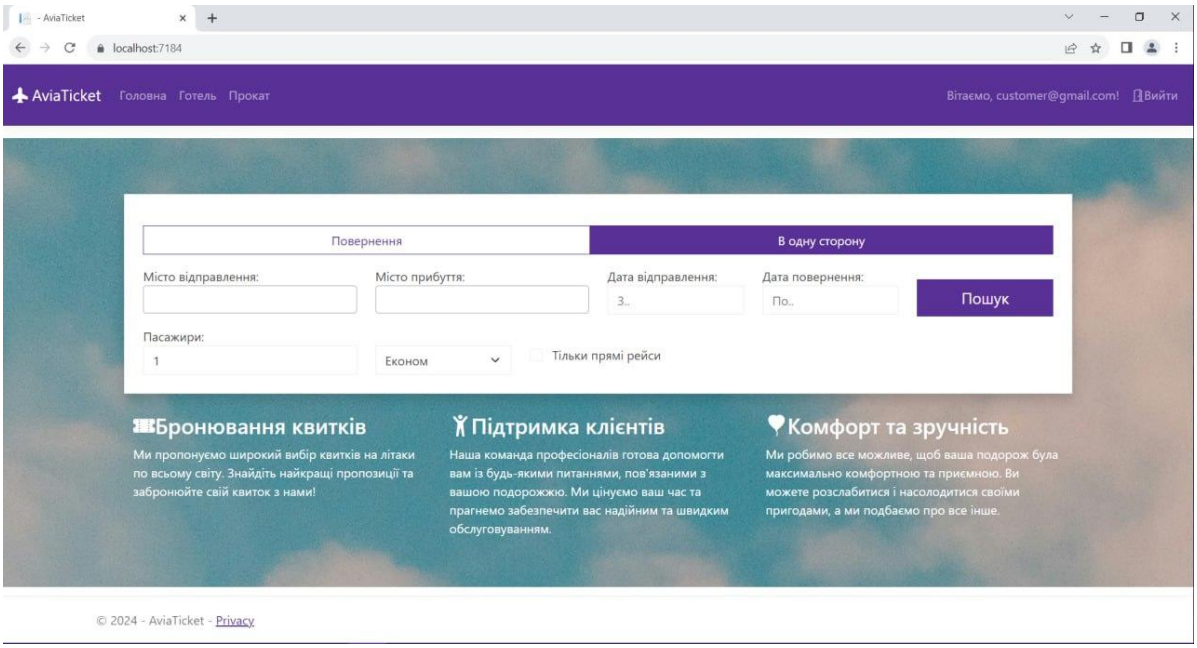

Рис. 4.1 Головна сторінка сайту

На головній формі можна виконати пошук рейса (див. рис. 4.2) або перейти на сайти бронювання авто (див. рис. 4.3) або житла (див. рис. 4.4), а також перейти в особистий кабінет (див. рис. 4.5), на форму реєстрації (див. рис. 4.6) або авторизації (див. рис. 4.7).

У формі пошуку користувачу надається можливість здійснити пошук потрібного рейсу відповідно до заданих характеристик, а саме місто або конкретний аеропорт відправлення і прибуття, дат відправлення і повернення, кількості пасажирів, класу білетів і обрати можливість тільки прямих рейсів або з врахуванням рейсів з пересадками.

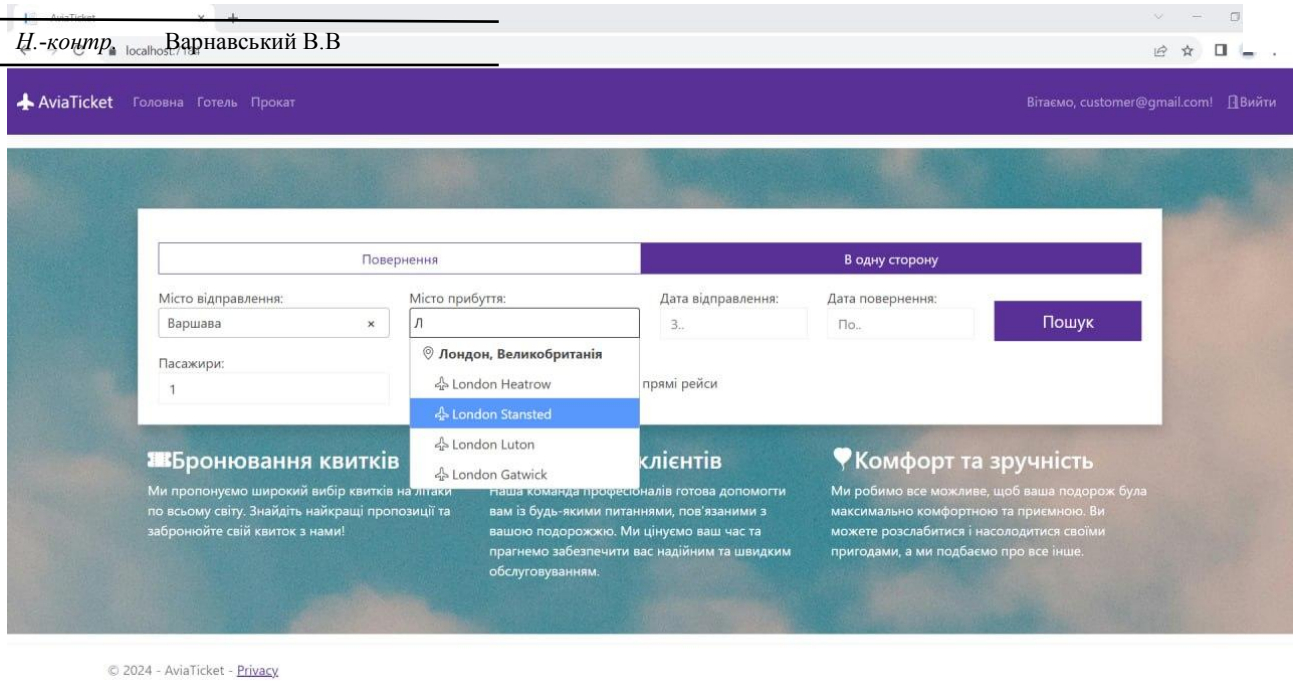

Рис. 4.2 Заповнення форми пошуку

Сайти оренди житла і авто не є частиною сервісу, а є сайтами партнерами доступ на які надається через реферальне посилання.

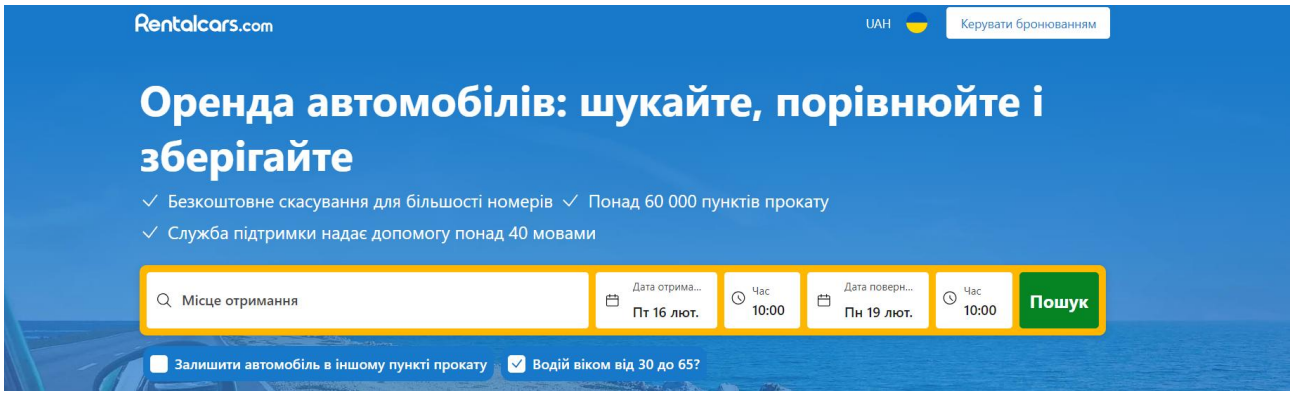

Рис. 4.3 Сайт оренди авто

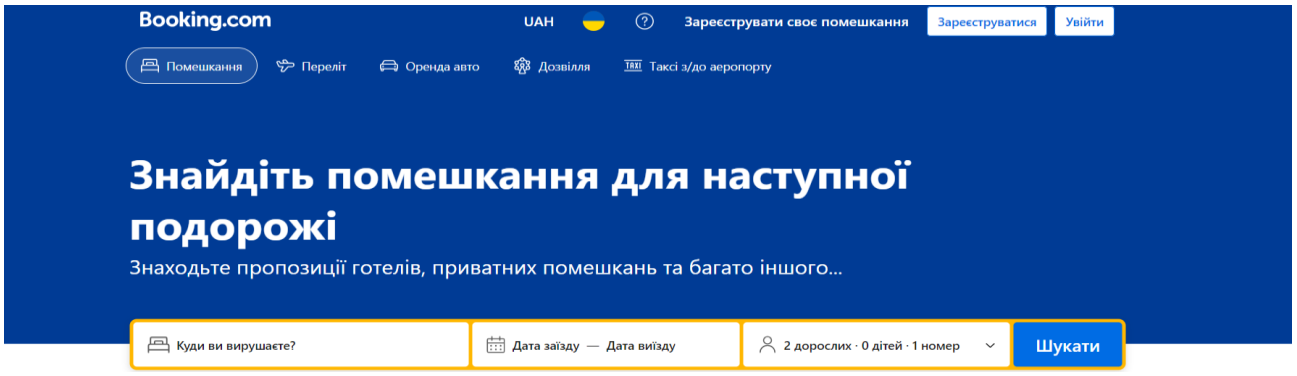

Рис. 4.4 Сайт оренди житла

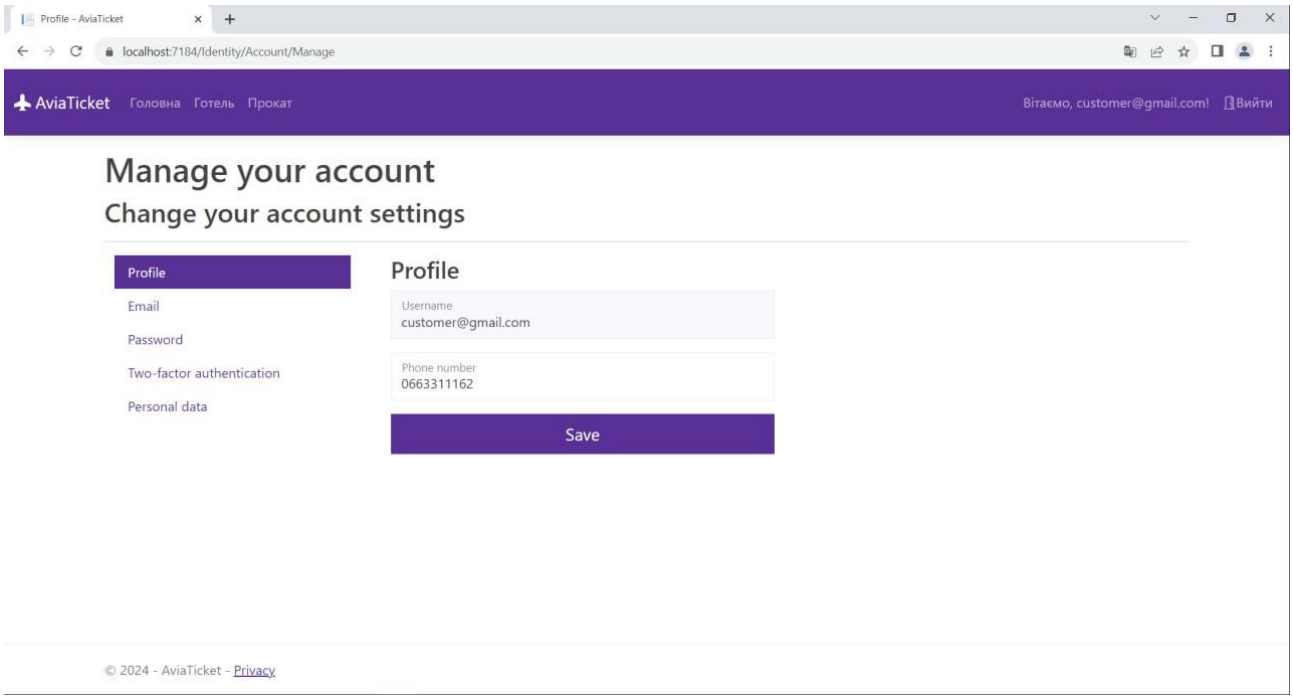

Рис. 4.5 Особистий кабінет користувача

В особистому кабінеті користувачу надається основна інформація по його обліковому запису, також надається можливість редагування цих даних. Він необхідний для верифікації користувача і здійснення ним бронювання квитків.

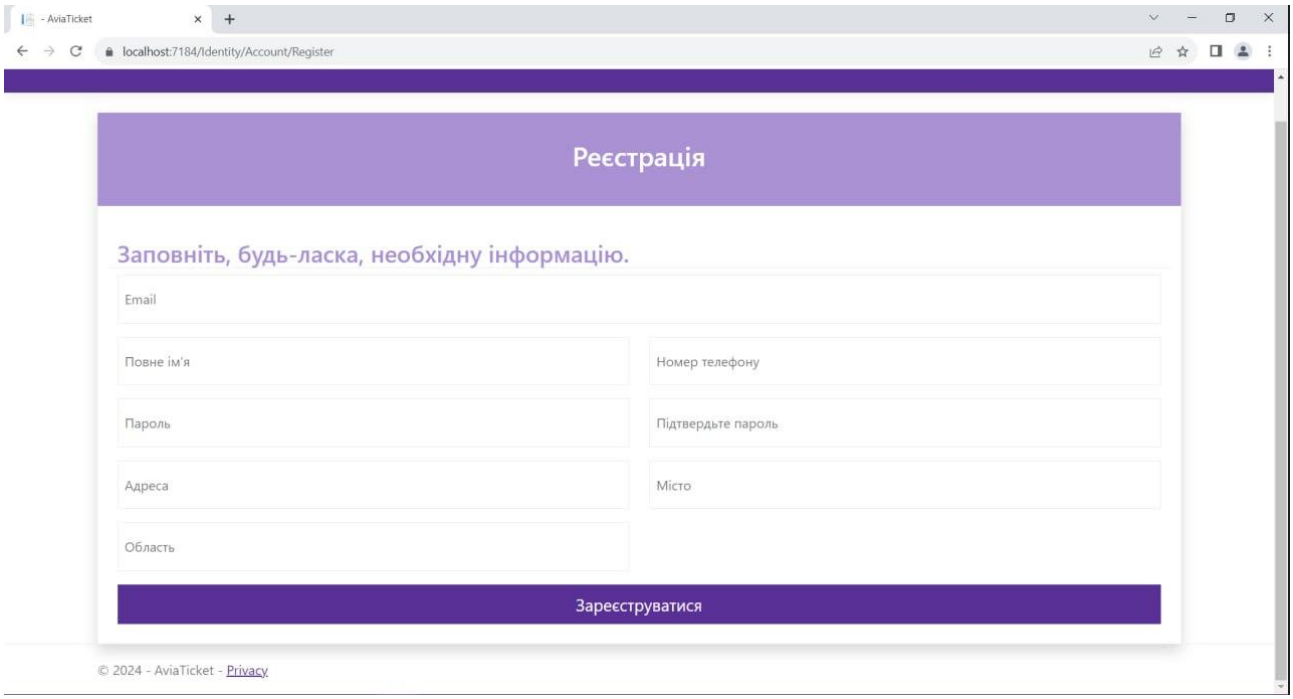

Рис. 4.6 Форма реєстрації

На формі реєстрації користувач надає усю необхідну інформацію для створення особистого облікового запису і подальшого використання сервісу.

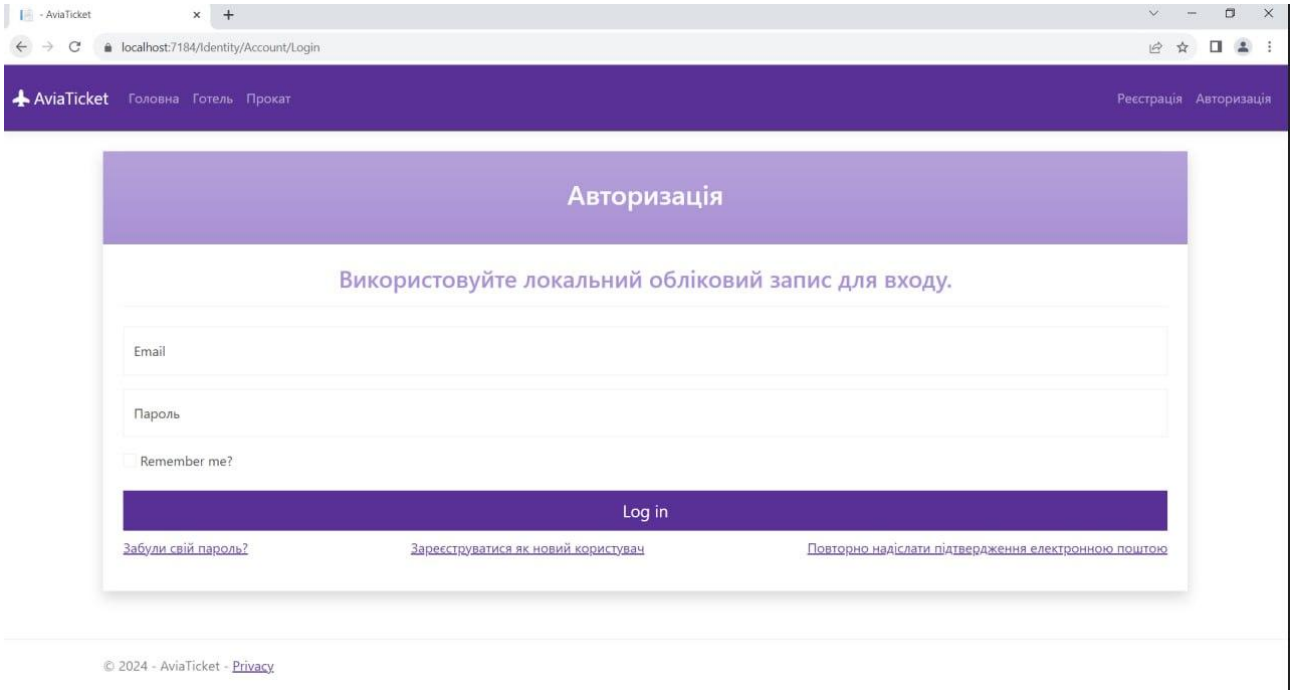

Рис. 4.7 Форма авторизації

Форма авторизації призначена для входу користувачем в свій обліковий запис використовуючи електронну пошту і пароль для входу. Також надає можливість відновлення пароля, переходу на форму реєстрації тощо.

Після виконання користувачем авторизації і заповнення форми пошуку рейсів на екран виводиться список рейсів за заданими критеріями, де можна переглянути, як прямі рейси (див. рис. 4.8) і рейси з пересадками (див. рис. 4.9).

Після вибору користувачем бажаного рейсу він може здійснити бронювання квитка натиснувши відповідну кнопку біля квитка. На формі бронювання (див. рис. 4.10) одразу зазначені дані з облікового запису користувача, які можна відредагувати для конкретного квитка, і інформація про обраний квиток.

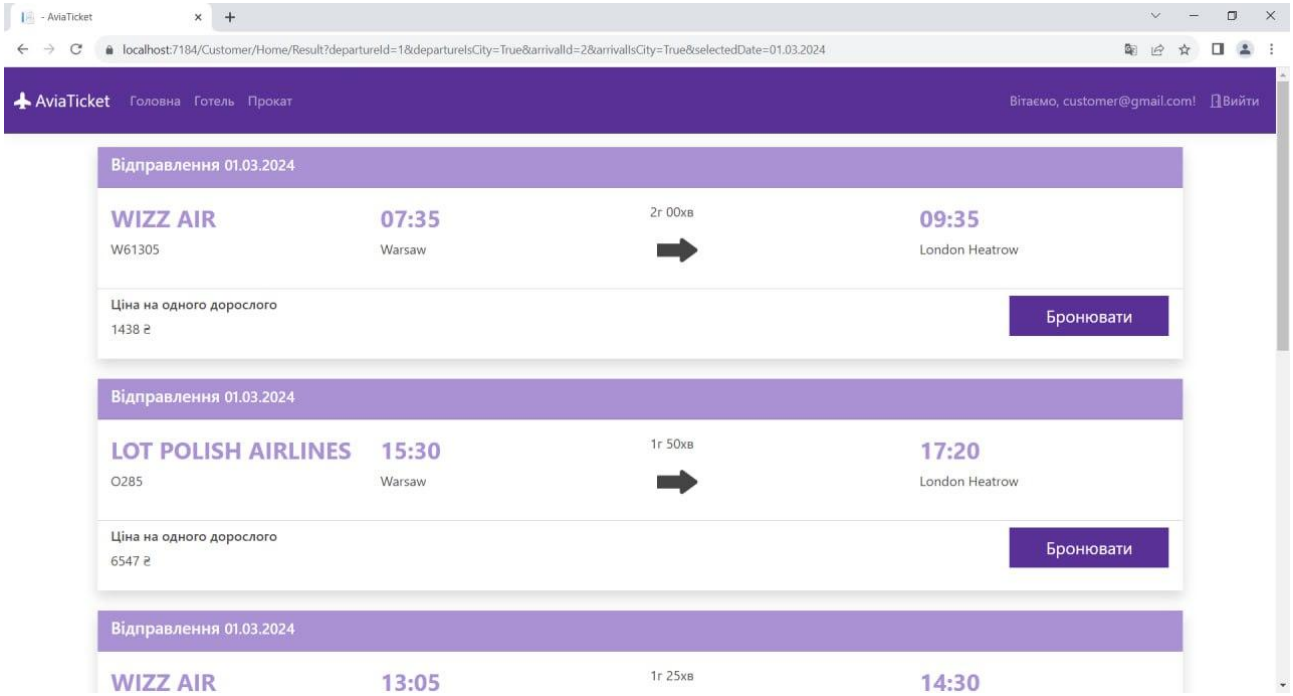

Рис. 4.8 Результат пошуку прямих рейсів

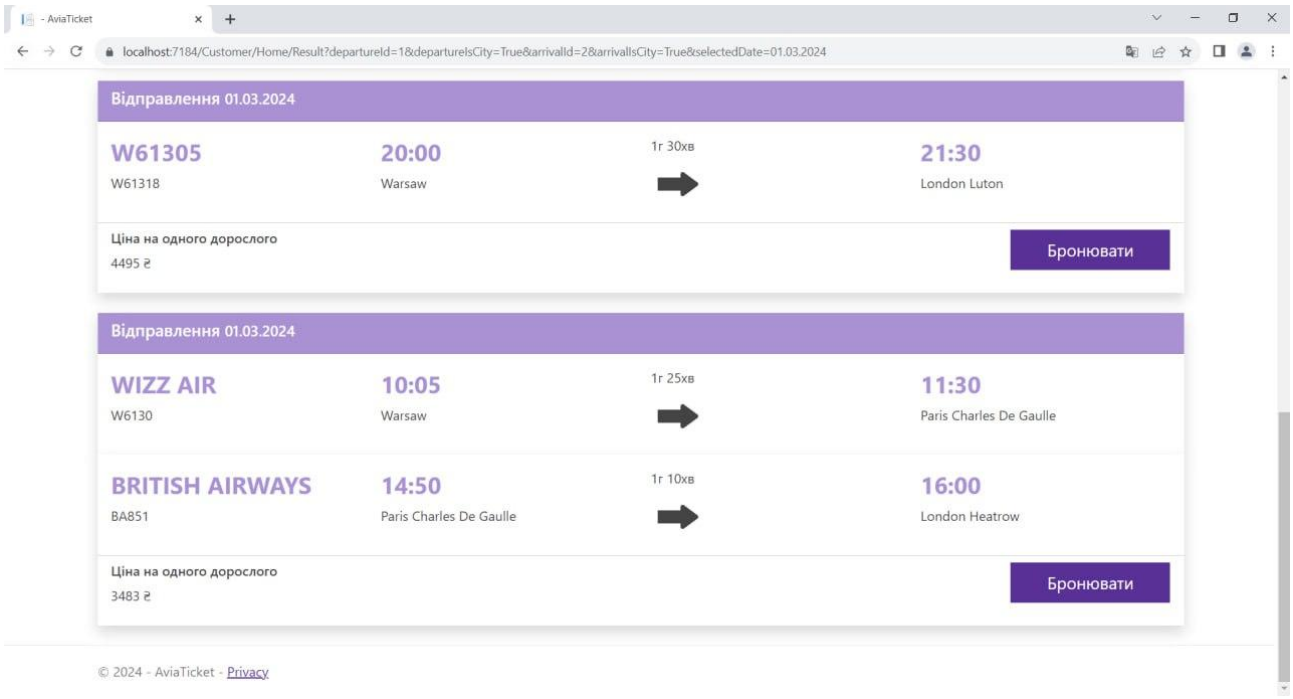

Рис. 4.9 Результат пошуку рейсів з пересадками

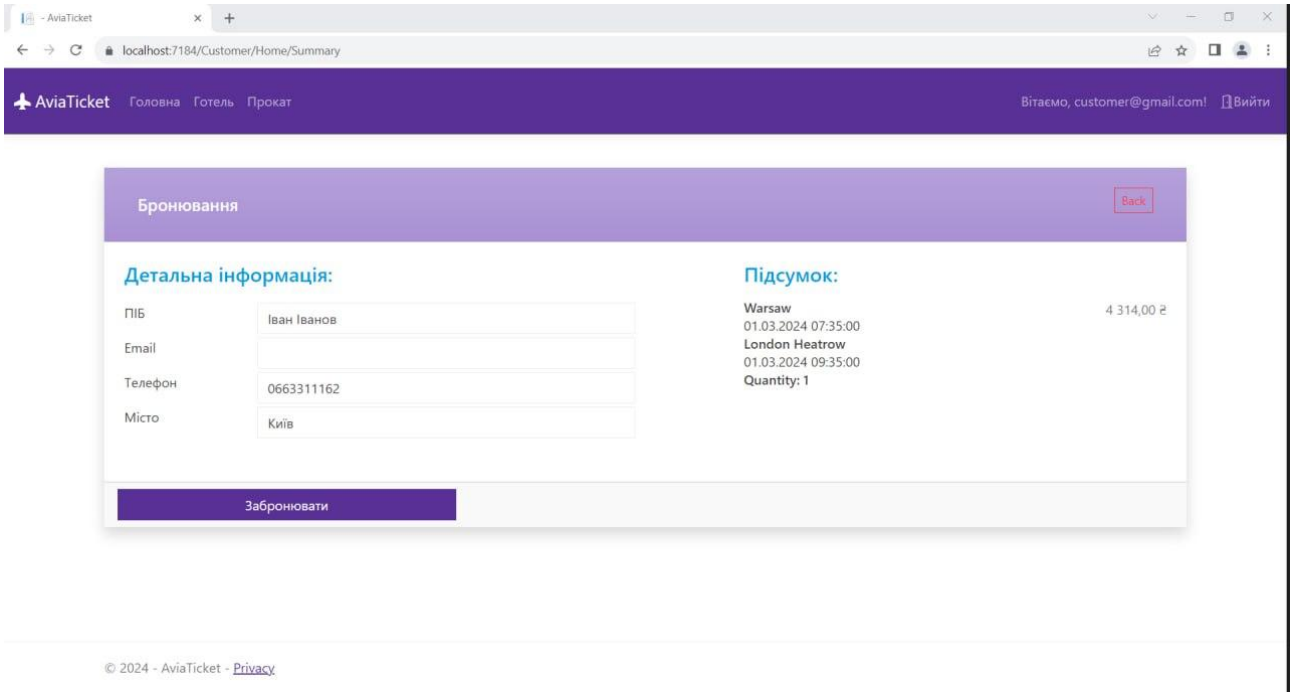

Рис. 4.10 Форма оформлення бронювання

4.2 Напрямки оптимізації роботи програми

Розглянемо можливі напрямки оптимізації роботи програми з метою підвищення ефективності та зручності використання.

1. Покращення алгоритмів пошуку рейсів

Одним з головних аспектів, що впливає на швидкодію програми, є ефективність алгоритмів пошуку рейсів. Розробка та впровадження більш оптимальних алгоритмів може значно зменшити час, потрібний для отримання результатів, та підвищити загальну продуктивність програми.

2. Підвищення швидкості завантаження сторінок

Швидкість завантаження сторінок є ключовим фактором в забезпеченні задоволення користувача. Оптимізація процесу завантаження, використання кешування та мінімізація обсягу передачі даних можуть значно покращити швидкість роботи веб-додатку.

3. Удосконалення інтерфейсу користувача

Вдосконалення інтерфейсу користувача спрямоване на забезпечення зручності та зрозумілості використання програми. Це може включати в себе

впровадження інтуїтивно зрозумілих елементів управління, оптимізацію розташування елементів на сторінці та підвищення візуальної привабливості.

4. Оптимізація використання ресурсів сервера

Ефективне використання ресурсів сервера є важливим аспектом для забезпечення стабільної роботи веб-додатку. Це включає в себе оптимізацію запитів до бази даних, раціональне використання пам'яті та обчислювальних ресурсів, а також моніторинг навантаження для попередження можливих перевантажень.

5. Підвищення безпеки програми

Забезпечення високого рівня безпеки є критично важливим для веб-додатків, особливо в області обробки особистої інформації користувачів та платіжних даних. Оптимізація заходів безпеки, таких як використання шифрування даних, валідація введених даних та захист від атак, допоможе забезпечити безпеку та конфіденційність інформації користувачів.

Впровадження цих напрямків оптимізації допоможе покращити роботу програми, забезпечити її більшу продуктивність та зручність використання для користувачів.

Висновки

В результаті проведення тестування програми виявило, що веб-додаток працює стабільно та відповідає вимогам, встановленим у попередніх розділах. Виявлені незначні недоліки та помилки були виправлені, що забезпечило коректну роботу програми під час використання.

Аналіз результатів роботи програми дозволив виявити деякі можливості для оптимізації та покращення функціональності. Зокрема, ідентифіковані напрямки оптимізації включають в себе вдосконалення алгоритмів пошуку рейсів, підвищення швидкості завантаження сторінок, та удосконалення інтерфейсу користувача для забезпечення кращої зручності та зрозумілості використання.

Отже, результати роботи програми свідчать про успішність її розробки та функціональність, але також вказують на потенційні можливості для подальшого вдосконалення та оптимізації з метою забезпечення ще кращої ефективності та задоволення потреб користувачів.

ВИСНОВКИ

Метою даної роботи була розробка сайта утворення маршрутів перельотів пасажирів з пересадками з проведенням попереднього аналізу і підготовки підґрунтя для розробки.

Було проведено аналіз системи утворення маршрутів перельотів пасажирів з пересадками. Виконано аналіз принципу купівлі авіабілетів з пересадками, а саме огляд принципів та особливостей купівлі авіабілетів з пересадками, що дозволило виявити ключові фактори та виклики у цьому процесі. Проведено також детальний огляд існуючих сервісів купівлі авіабілетів з метою зрозуміння їхньої функціональності та можливостей. Визначено переваги впровадження системи утворення маршрутів перельотів пасажирів з пересадками у формі веб-додатку.

Для розробки якісного веб-додатку було обґрунтовано вибір необхідних технологій для розробки і визначено вимоги до апаратної та операційної платформи для ефективної роботи веб-сайту. Сформульовано також вимоги до функціональності, дизайну інтерфейсу та заходів безпеки для веб-додатку.

Проведено моделювання бази даних для зберігання інформації про перельоти та пасажирів. Представлено діаграму класів програми для зрозуміння структури коду. Розроблено дизайн інтерфейсу, що відповідає вимогам та забезпечує зручність використання.

Розглянуто реалізація основних модулів програми, а саме модуля автодоповнення для полегшення введення даних користувачем, модуля пошуку рейсу з урахуванням різних параметрів та варіантів та модуля реєстрації користувачів для доступу до особистого кабінету.

У підсумку було проведено тестування веб-додатку для перевірки його працездатності та відповідності вимогам. А також виявлено можливі напрямки оптимізації роботи веб-додатку для покращення його ефективності та зручності використання.

В цілому, робота над системою утворення маршрутів перельотів пасажирів з пересадками дозволила детально розглянути всі аспекти цього

процесу та розробити ефективний та зручний веб-додаток для користувачів, а також додатково розглянути подальші дії для покращення в наступних версіях.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. TripAdvisor [Електронний ресурс] – Режим доступу до ресурсу: https://www.tripadvisor.com/
2. International Air Transport Association (IATA) [Електронний ресурс] – Режим доступу до ресурсу: https://www.iata.org/
3. Bureau of Transportation Statistics (BTS) [Електронний ресурс] – Режим доступу до ресурсу: https://www.bts.gov/
4. Visual Studio Community [Електронний ресурс] – Режим доступу до ресурсу: https://visualstudio.microsoft.com
5. SQL Server 2022 [Електронний ресурс] – Режим доступу до ресурсу: https://www.microsoft.com
6. Документація за C# [Електронний ресурс] – Режим доступу до ресурсу: https://learn.microsoft.com
7. Посібник з ASP.NET MVC [Електронний ресурс] – Режим доступу до ресурсу: https://learn.microsoft.com
8. ASP.NET Identity та системи аутентифікації [Електронний ресурс] – Режим доступу до ресурсу: https://metanit.com
9. Центр документації Entity Framework [Електронний ресурс] – Режим доступу до ресурсу: https://learn.microsoft.com
10. CSHTML [Електронний ресурс] – Режим доступу до ресурсу: https://docs.fileformat.com
11. JavaScript [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org
12. CSS: каскадні таблиці стилів [Електронний ресурс] – Режим доступу до ресурсу: https://developer.mozilla.org
13. Типи баз даних: особливості, відмінності та приклади [Електронний ресурс] – Режим доступу до ресурсу: https://dou.ua