МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ВОДНОГО ГОСПОДАРСТВА ТА ПРИРОДОКОРИСТУВАННЯ
Навчально-науковий інститут кібернетики, інформаційних технологій та інженерії
Кафедра комп'ютерних наук та прикладної математики

«До захисту допущена»
Завідувач кафедри комп'ютерних наук
та прикладної математики
_____ Турбал Ю.В.
«_____» _____ 2024 р.

КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«Розробка сервісу карпулінгу на основі технологій JavaScript **»**

Виконав: Кравчук Денис Миколайович _____
(прізвище, ім'я, по батькові) (підпис)

група КНз-51

Керівник: старший викладач Харів Н.О. _____
(науковий ступінь, вчене звання, посада, прізвище та ініціали) (підпис)

Рівне – 2024

ЗМІСТ

РЕФЕРАТ

Кваліфікаційна робота: 22 с., 10 малюнків, 2 таблиці, 7 джерел.

Мета роботи: розробка інноваційного та ефективного сервісу для карпулінгу, який сприятиме зниженню транспортного навантаження, зменшенню витрат на перевезення та підвищенню екологічної стійкості.

Об'єкт досліджування – розробка сайту для карпулінгу.

Методи дослідження – технології ASP.NET MVC, Entity Framework, MS SQL, JavaScript, HTML, CSS, jQuery з дотриманням архітектури MVC.

Результатом даної кваліфікаційної роботи стало створення функціонального сервісу для карпулінгу, який успішно пройшов всі етапи розробки, тестування та пілотного запуску. Сервіс має чітку та масштабовану архітектуру, що включає компоненти фронтенду та бекенду, базу даних і API для взаємодії з іншими сервісами. Підготовлено детальну документацію з описом вимог та архітектури системи. Розроблений сайт дозволяє користувачам легко знаходити людей для спільних поїздок.

Ключові слова: **ASP.NET MVC, Entity Framework, MS SQL, JavaScript, HTML, CSS, jQuery, КАРПУЛІНГ.**

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

ASP.NET MVC – архітектурний шаблон для розробки веб-додатків на платформі .NET, що розділяє додаток на три основні компоненти: Model (Модель), View (Подання) та Controller (Контролер).

Entity Framework ORM – об’єктно-реляційний відображувач для .NET, що спрощує роботу з базами даних, дозволяючи розробникам працювати з даними у вигляді об’єктів.

MS SQL – реляційна система управління базами даних (RDBMS), розроблена Microsoft, яка підтримує зберігання та управління даними.

JavaScript – мова програмування, яка використовується для додавання інтерактивності до вебсторінок, таких як анімації, форми, обробка подій та динамічні зміни контенту.

HTML – мова розмітки, що використовується для створення та структурування контенту на вебсторінках.

CSS – Мова стилів, що використовується для опису зовнішнього вигляду HTML-документів, включаючи оформлення, кольори, шрифти та розташування елементів.

jQuery – бібліотека JavaScript, яка спрощує роботу з DOM, обробку подій, анімації та AJAX-запити, забезпечуючи кросбраузерну сумісність.

ВСТУП

В епоху, коли проблеми перенавантаження доріг, забруднення довкілля та високої вартості індивідуальних перевезень стають все актуальнішими, розробка і впровадження інноваційних рішень для покращення транспортної системи стає дуже важливим завданням. За статистикою, численні автомобілі з одним водієм, що курсують містами, спричиняють затори, марнування часу та ресурсів, а також негативно впливають на екологію.

Тривалий час традиційні методи вирішення цієї проблеми демонстрували свої обмеження та неефективність. У зв’язку з цим, виникає важлива потреба в сучасних, ефективних рішеннях, спрямованих на оптимізацію використання транспортних засобів та зменшення навантаження на дороги. Вебсервіс карпулінгу є відповіддю на це викликання, надаючи інноваційний та зручний механізм для спільних поїздок.

Метою даного проекту є створення ефективного вебсервісу, який дозволить користувачам знаходити попутників для спільних поїздок у зручний та безпечний спосіб. Застосування сучасних технологій та автоматизація процесів пошуку та взаємодії між водіями та пасажирами робить цей сервіс надзвичайно актуальним у контексті покращення транспортної ситуації в містах.

Отже, розробка вебсервісу карпулінгу відповідає сучасним викликам та прагненням створення ефективної та екологічної транспортної системи. Подальші кроки в цьому напрямку можуть значно покращити мобільність населення, зменшити витрати на пересування та сприяти зниженню негативного впливу транспорту на довкілля.

РОЗДІЛ 1.
АНАЛІЗ ПРОБЛЕМИ

1. Аналіз предметної області

У сучасному світі, де проблеми перевантаженості доріг, високої вартості транспорту та забруднення довкілля стають все гострішими, з'являється необхідність в інноваційних підходах до оптимізації транспортної системи. Таким чином, ця дипломна робота присвячена розробці та впровадженню вебсервісу карпулінгу, який дозволить користувачам ефективно та зручно знаходити попутників для спільних поїздок.

Мета даного проекту полягає у створенні зручного та інноваційного вебсервісу, який дозволить водіям та пасажирам швидко та легко знаходити один одного для здійснення спільних поїздок. Застосовуючи передові технології, ми плануємо розробити інтерактивний інтерфейс та ефективну систему управління для взаємодії між водіями та пасажирами.

Аналізуючи актуальні проблеми перевантажених доріг, високих витрат на транспорт та негативного впливу на довкілля, ми визначимо фактори, що підштовхують до розробки вебсервісу карпулінгу. Проведемо порівняльний аналіз існуючих рішень в галузі спільних поїздок, оцінимо їх переваги та недоліки [1].

Важливим аспектом буде визначення вимог до функціоналу та безпеки вебсервісу. Планується розробка інтегрованої системи управління поїздками, забезпеченням конфіденційності та якості обслуговування. Цей проект відкриває можливості покращення транспортної ситуації в містах та підвищення рівня доступності послуг карпулінгу для широкого кола користувачів.

2. Конкуренція

На сьогоднішній день ринок транспортних послуг активно розвивається, пропонуючи різноманітні рішення для пересування та спільного використання транспорту. Деякі існуючі платформи спрямовані на підтримку карпулінгу, але значна частина конкуренції зосереджена на традиційних послугах таксі та каршерінгу [2].

1.2.1. Аналіз існуючих конкурентів

Розглянемо основні аналоги нашого сервісу:

1. Таксі-сервіси: популярні таксі-платформи надають зручний спосіб пересування, але не завжди підтримують концепцію спільних поїздок та оптимізації використання транспорту. Однак вони є великим гравцем на ринку та мають велику базу користувачів.
2. Каршерінг: послуги каршерінгу також пропонують можливість самостійного керування автомобілем, але не завжди забезпечують оптимальне використання транспортних засобів через поїздки з одним водієм. Вони активно конкурують на ринку міської мобільності.
3. Мобільні додатки для пошуку транспорту: додатки, які дозволяють замовляти транспорт, стають все популярнішими. Проте, більшість з них не фокусуються на концепції карпулінгу та спільних поїздок [3].

1.2.2. Наші конкурентні переваги

Наш вебсервіс вирізняється фокусом на забезпеченні зручного та ефективного пошуку попутників для спільних поїздок. Ми надаємо користувачам можливість знаходити водіїв або пасажирів, які прямують в

одному напрямку, щоб зменшити витрати на транспорт та оптимізувати використання автомобілів.

Наш сервіс пропонує зручний інтерфейс для пошуку та організації спільних поїздок, що робить процес максимально простим для користувачів. Ми розглядаємо питання доступності для широкого кола користувачів, зокрема тих, хто шукає економічно ефективні та екологічні способи пересування.

Ми плануємо використовувати передові технології для підвищення якості наших послуг, включаючи системи геолокації для пошуку попутників, інтеграцію з картами та навігацією, а також ефективну систему управління поїздками [4].

Аналіз конкуренції допомагає нам визначити унікальні переваги та ключові особливості, що відрізняють наш вебсервіс від інших рішень на ринку транспортних послуг.

РОЗДІЛ 2.
ПОСТАНОВКА ЗАДАЧІ

2.1. Способи вирішення

У створенні вебсервісу карпулінгу існують кілька ключових способів вирішення, спрямованих на забезпечення зручності та ефективності для користувачів, адміністраторів та водіїв.

Перший спосіб передбачає створення системи, де користувачі можуть публікувати інформацію про свої заплановані поїздки та шукати попутників, які прямують в тому ж напрямку. Це надає гнучкість у плануванні спільних поїздок, але може вимагати додаткових зусиль для координації між водіями та пасажирами [5].

Другий підхід передбачає створення алгоритму автоматичного підбору попутників на основі заданих маршрутів та часу поїздки. Система аналізує запити користувачів та пропонує оптимальні варіанти спільних поїздок. Це спрощує процес пошуку попутників, але може обмежувати гнучкість у виборі конкретного водія чи пасажира.

Третій підхід передбачає створення платформи для публічного обміну послугами між водіями та пасажирами. Користувачі можуть вибирати попутників на основі рейтингу та відгуків інших користувачів, що дозволяє створювати спільноту та сприяє взаємодії між учасниками.

Обравши комбінацію другого та третього підходів, ми прагнемо створити високоефективний та зручний сервіс карпулінгу. Це дозволить користувачам не лише автоматично знаходити оптимальні варіанти спільних поїздок, але й надасть можливість вибирати конкретних попутників на основі рейтингу та відгуків. Такий підхід забезпечить ефективність пошуку та зручність взаємодії між водіями та пасажирами [6].

2.2. Технології

У процесі розробки вебсервісу карпулінгу були використані ряд технологій та інструментів, які забезпечили ефективність, безпеку та зручність сервісу. Нижче подано детальний опис використаних технологій та їхнє призначення:

Таблиця 2.1.

Використані технології

Технологія	Призначення	Роль в проекті
Microsoft SQL Server	Система управління базами даних (СУБД)	Зберігання, управління та організація даних
ASP.NET Core MVC	Фреймворк для веб-додатків на базі MVC	Побудова серверної частини, обробка HTTP-запитів, логіка роутінгу
Entity Framework Core (EF Core)	ORM-фреймворк для роботи з базою даних	Взаємодія з базою даних, використання об’єктно-орієнтованого підходу
HTML, CSS, JS та jQuery	Веб-технології для користувацького інтерфейсу	Створення інтерфейсу, реалізація взаємодії з користувачем
Visual Studio 2022	Інтегроване середовище розробки	Написання, редагування та відлагодження коду, робота з інструментами
MS SQL Server Management Studio (SSMS)	Графічний інтерфейс для управління базами даних	Адміністрування та моніторинг бази даних, виконання SQL-запитів

Використання цих технологій забезпечило створення ефективного, безпечного та зручного вебсервісу карпулінгу, який задовольняє потреби користувачів та забезпечує гладку взаємодію між різними компонентами системи [7].

РОЗДІЛ 3.
ПРАКТИЧНА ЧАСТИНА

3.1. Архітектура

Проект вебсервісу карпулінгу має чітку та організовану архітектурну структуру, яка відображає модульний підхід до розробки та дотримання принципів розділення обов’язків. Нижче подано огляд ключових компонентів та їхню роль у структурі проекту:

Таблиця 3.1.

Структура проекту

Каталоги/Файли	**Опис**	**Роль в проекті**
Controllers	Контролери для обробки HTTP-запитів та взаємодії з інтерфейсом	Управління виконанням запитів, обробка логіки відображення
Models	Моделі даних, що визначають структуру об’єктів	Визначення структури бази даних та об’єктів
Services	Сервіси для виконання операцій бізнес-логіки	Відокремлення бізнес-логіки від контролерів, забезпечення многоразового використання
ViewModels	Класи ViewModel для передачі даних між контролерами та представленнями	Передача даних між різними компонентами
Views	Файли представлень, що визначають вигляд сторінок	Візуальне відображення даних та взаємодія з користувачем
DriverDbContext.cs	Клас DriverDbContext для взаємодії з базою даних	Забезпечення спрощеного доступу до бази даних, операції CRUD
Program.cs	Файл, що містить точку входу для програми	Визначення конфігурації та запуск додатку

MVC (Model-View-Controller) – це архітектурний шаблон, який використовується для розробки програмних додатків та вебсервісів, щоб вони були легко розширюваними (рис 3.1) [8].

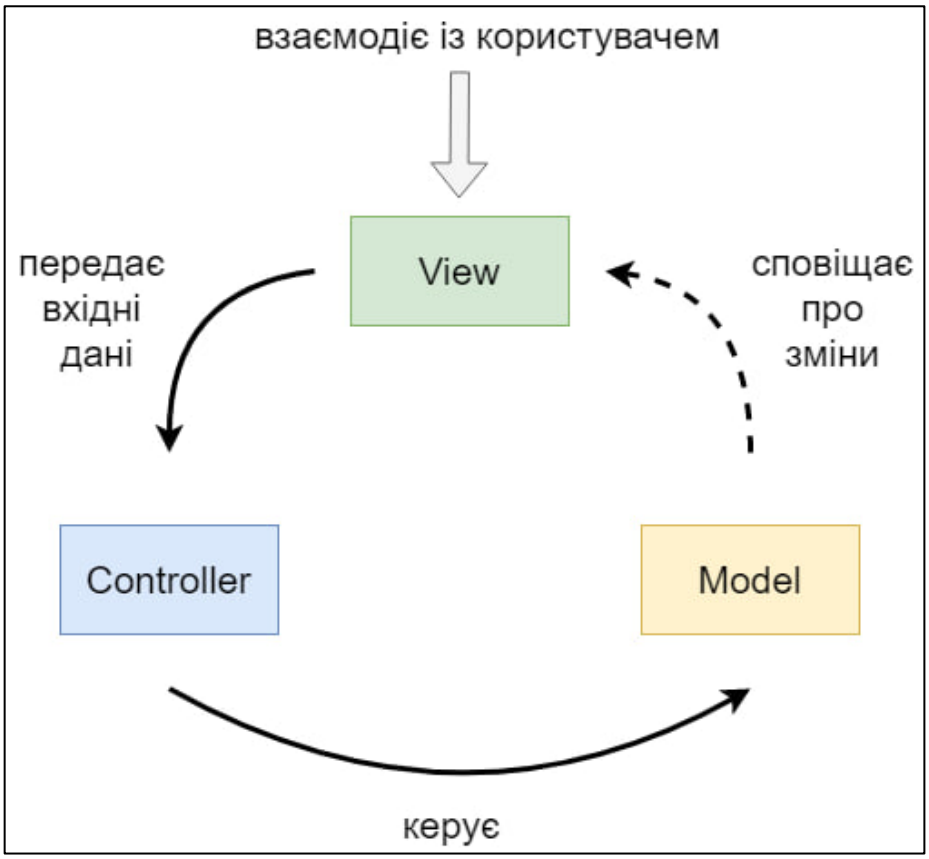

Рис. 3.1. Архітектура проєкту

Модель-Вид-Контролер розділяє програму на три основні компоненти, кожен з яких відповідає за конкретну відповідальність:

Модель (Model): Модель представляє собою даний та бізнес-логіку програми. Вона відповідає за зберігання та обробку даних, а також за визначення правил бізнес-логіки. Відповідає за забезпечення доступу до даних, їхнє зберігання та обробка, а також виконання бізнес-логіки [9].

Вид (View): Вид відповідає за відображення інформації користувачеві та взаємодію з ним. Це може бути графічний інтерфейс, текстовий вивід чи будь-який інший спосіб представлення даних.

Контролер (Controller): Контролер приймає введення від користувача та визначає, які дії потрібно виконати. Він взаємодіє з моделлю та відображенням, реагуючи на події та керуючи потоком даних. Його роль - обробка введення,

взаємодія з моделлю та вибірка потрібного вигляду для представлення результатів.

Принципи розділення обов’язків між цими компонентами дозволяють покращити легкість розробки, тестування та підтримки програми. MVC також сприяє перевикористанню коду, оскільки кожен компонент може бути розгляданий окремо [10].

У випадку вебсервісу карпулінгу:

Модель: представляє класи для роботи з даними користувачів, водіїв, поїздок та іншими об’єктами, які взаємодіють з базою даних.

Вид: включає представлення, які відповідають за відображення даних на вебсторінках та взаємодію з користувачем.

Контролер: обробляє HTTP-запити від користувачів, викликає необхідні операції в моделі та вибирає потрібні види для відображення результатів.

Така архітектура спрощує розробку та розуміння коду, а також полегшує внесення змін та підтримку проекту.

3.2. База даних

База даних була написана за допомогою Entity Framework Core з використанням підходу code first, що дозволяє використовувати міграції і розширювати та доповнювати базу протягом усього процесу розробки.

На (рис 3.2) зображено діаграму бази даних. Проект використовує базу даних з чотирма таблицями: Rides, Users, Cars та Bookings. Ось опис їх структури та призначення:

Таблиця Rides:

- Id (int, PK): Ідентифікатор поїздки.
- DepartureLocation (nvarchar(max)): Місце відправлення.
- ArrivalLocation (nvarchar(max)): Місце призначення.
- DepartureTime (datetime): Час відправлення.
- SeatsAvailable (int): Кількість доступних місць.

- Price (decimal): Ціна поїздки.
- DriverId (int, FK): Ідентифікатор водія, який створив поїздку.
- CarId (int, FK): Ідентифікатор автомобіля, який використовується для поїздки [11].

Таблиця Users:

- Id (int, PK): Ідентифікатор користувача.
- Username (nvarchar(max), NOT NULL): Ім’я користувача.
- Email (nvarchar(max), NOT NULL): Адреса електронної пошти користувача.
- Password (nvarchar(max), NOT NULL): Пароль користувача.
- Phone (nvarchar(max), NOT NULL): Номер телефону користувача.
- Role (nvarchar(max), NOT NULL): Роль користувача (адміністратор, водій, пасажир).

Таблиця Cars:

- Id (int, PK): Ідентифікатор автомобіля.
- Make (nvarchar(max)): Марка автомобіля.
- Model (nvarchar(max)): Модель автомобіля.
- Year (int): Рік випуску автомобіля.
- LicensePlate (nvarchar(max)): Номерний знак автомобіля.
- OwnerId (int, FK): Ідентифікатор власника автомобіля (зв’язок з таблицею Users)[12].

Таблиця Bookings:

- Id (int, PK): Ідентифікатор бронювання.
- RideId (int, FK): Ідентифікатор поїздки, яку забронював пасажир.
- PassengerId (int, FK): Ідентифікатор пасажира, який забронював місце.
- Seats (int): Кількість заброньованих місць.
- TotalPrice (decimal): Загальна ціна бронювання.

У базі даних використовуються зовнішні ключі (FK) для встановлення зв’язків між таблицями та забезпечення цілісності даних. Зв’язки визначені таким чином:

- Rides.DriverId (FK) -> Users.Id (PK) (з видаленням водія видаляються також усі пов’язані з ним поїздки).
- Rides.CarId (FK) -> Cars.Id (PK) (з видаленням автомобіля видаляються також усі пов’язані з ним поїздки).
- Cars.OwnerId (FK) -> Users.Id (PK) (з видаленням власника автомобіля видаляються також усі пов’язані з ним автомобілі).
- Bookings.RideId (FK) -> Rides.Id (PK) (з видаленням поїздки видаляються також усі пов’язані з нею бронювання).
- Bookings.PassengerId (FK) -> Users.Id (PK) (з видаленням пасажира видаляються також усі його бронювання).

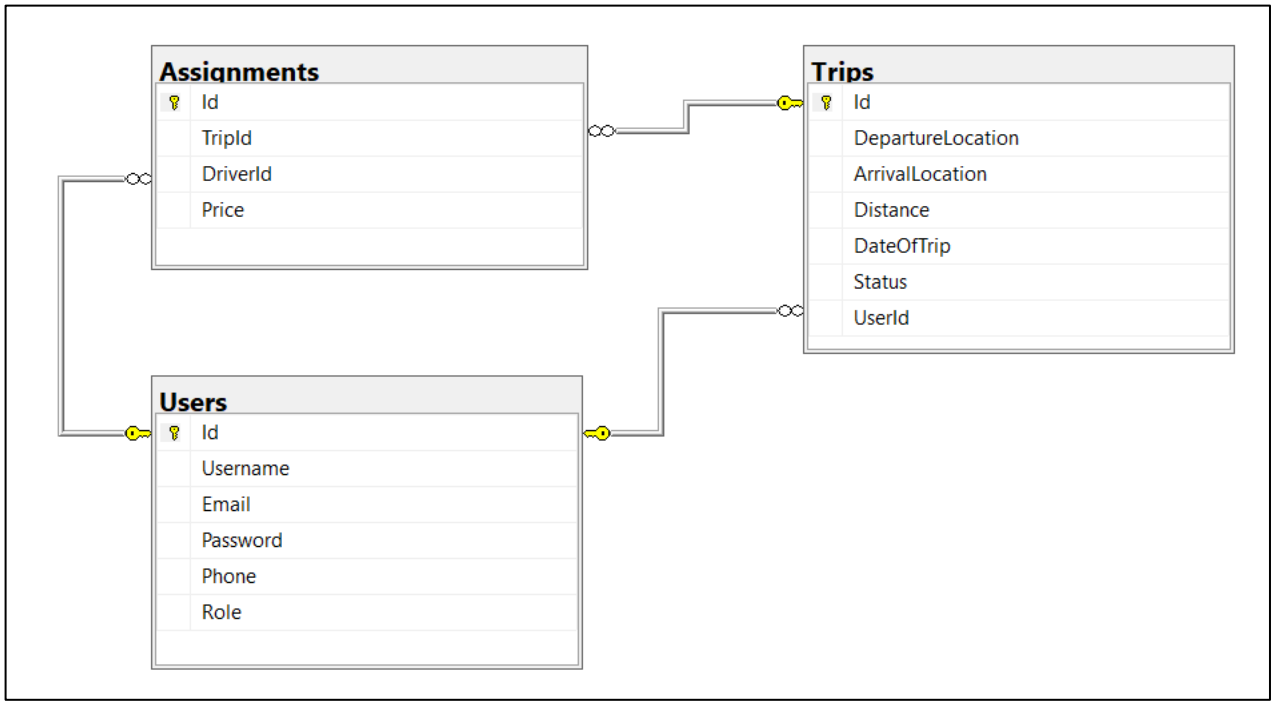

Рис. 3.2. Діаграма бази даних

3.3. Бекенд

Бекенд (частину системи, яка відповідає за обробку бізнес-логіки та взаємодію із базою даних та іншими зовнішніми сервісами) вебсайту реалізовано за допомогою технології ASP.NET Core. Нижче наведено опис ключових функціональних частин бекенду сайту.

Відповідальність за аутентифікацію та авторизацію користувачів несе клас AccountController. У ньому реалізовані методи для реєстрації (Register) та входу (Login) користувачів. Також реалізований вихід з облікового запису (Logout). Для забезпечення безпеки використовується механізм куків для аутентифікації.

Клас AdminController відповідає за керування користувачами. До його функцій входить перегляд списку користувачів (UserManagement), видалення користувача (DeleteUser), зміна ролі користувача (ChangeRole). Адміністратор має доступ до цих функцій через авторизацію з роллю "Admin" [13].

У класі RideController визначені методи для створення нової поїздки (CreateRide), перегляду існуючих поїздок (ViewRides), редагування поїздки (EditRide) та видалення поїздки (DeleteRide). Водії мають доступ до створення та керування своїми поїздками, а пасажири можуть переглядати доступні поїздки.

Клас BookingController містить методи для бронювання місць у поїздках (CreateBooking), перегляду бронювань користувача (ViewBookings) та скасування бронювання (CancelBooking). Пасажири можуть здійснювати бронювання доступних місць у поїздках.

Функції профілю користувача реалізовані в класі ProfileController. Цей контролер містить методи для перегляду та редагування профілю користувача (ViewProfile, EditProfile), включаючи зміну особистої інформації та пароля.

Взаємодія з базою даних виконується через Entity Framework Core. Клас CarpoolDbContext визначає структуру бази даних та включає таблиці для зберігання інформації про користувачів (Users), поїздки (Rides), автомобілі (Cars) та бронювання (Bookings).

Усі налаштування додатку зберігаються у файлі appsettings.json. Залежності та сервіси конфігуруються в методі ConfigureServices за допомогою вбудованого контейнера DI (Dependency Injection).

Система використовує стандартні засоби ASP.NET Core для реалізації аутентифікації та авторизації користувачів. Для цього використовуються кукі токени та ролі користувачів [14].

HTTP-запити обробляються через ASP.NET Core Middleware, яке включає в себе різні компоненти для обробки запитів, включаючи маршрутизацію, авторизацію та обробку виключень.

Додаток взаємодіє з зовнішніми сервісами, такими як Google Maps для отримання геолокації та розрахунку відстаней, та використовує HTTP-клієнт для здійснення HTTP-запитів.

Це загальний огляд бекенду вебсайту. Важливим елементом є також конфігурація сервісів, робота з базою даних та обробка HTTP-запитів через контролери.

3.4. Фронтендна частина

Перейшовши на сайт карпулінгу, користувач потрапляє на сторінку входу (рис 3.3), де він може ввести свою електронну пошту та пароль, або перейти на сторінку реєстрації. Важливо відмітити, що в бекенді пароль хешується за допомогою алгоритму PBKDF2, що забезпечує безпеку та конфіденційність даних користувача.

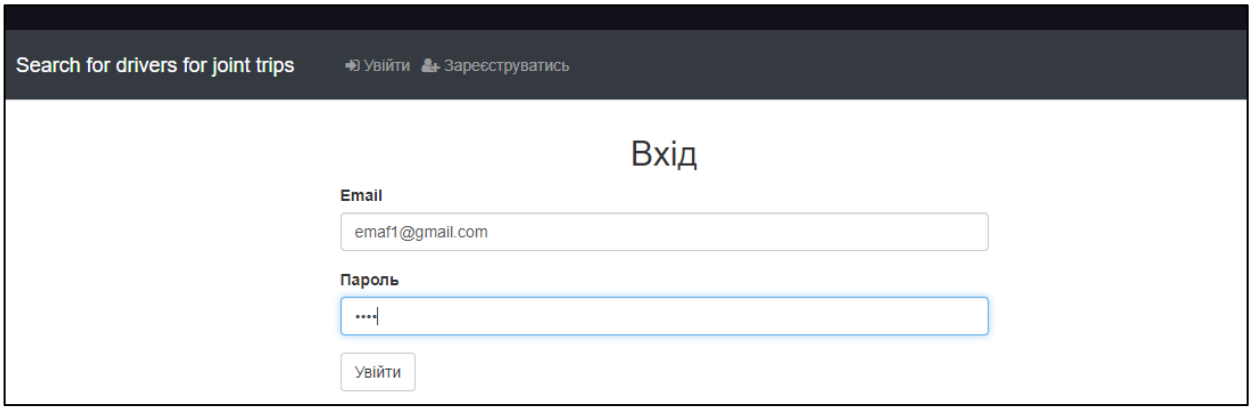

Рис. 3.3. Сторінка входу в систему

Якщо користувач вперше відвідує сайт і бажає зареєструватися, він може натиснути на посилання "Зареєструватися" на сторінці входу. Це перенаправить його на сторінку реєстрації (рис 3.4), де він зможе заповнити необхідні поля для створення облікового запису, такі як ім’я, електронна пошта, пароль та роль (водій або пасажир) [15].

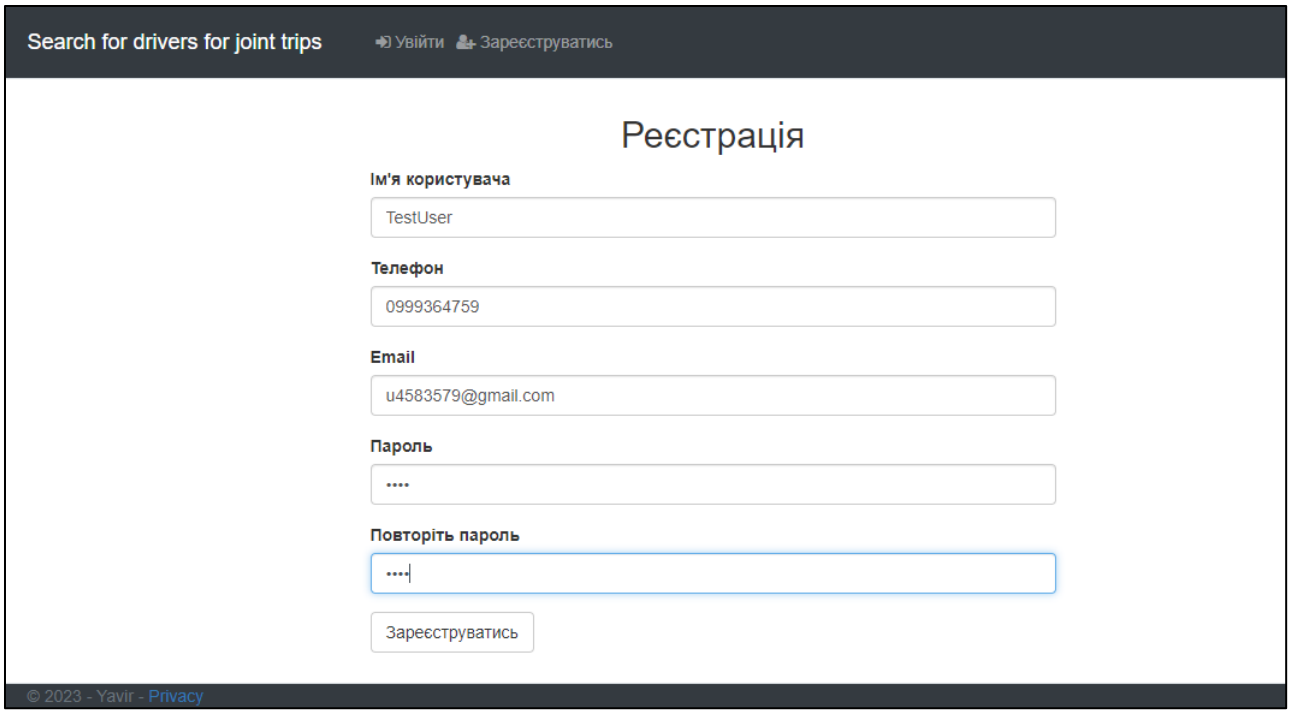

Рис. 3.4. Сторінка реєстрації

Після успішної авторизації користувач потрапляє на головну сторінку (рис. 3.5), де відображається список доступних поїздок. Користувач може переглядати деталі поїздок, такі як місце відправлення, місце призначення, дата і час, кількість доступних місць та ціна. Він також може здійснювати пошук поїздок за заданими критеріями [16].

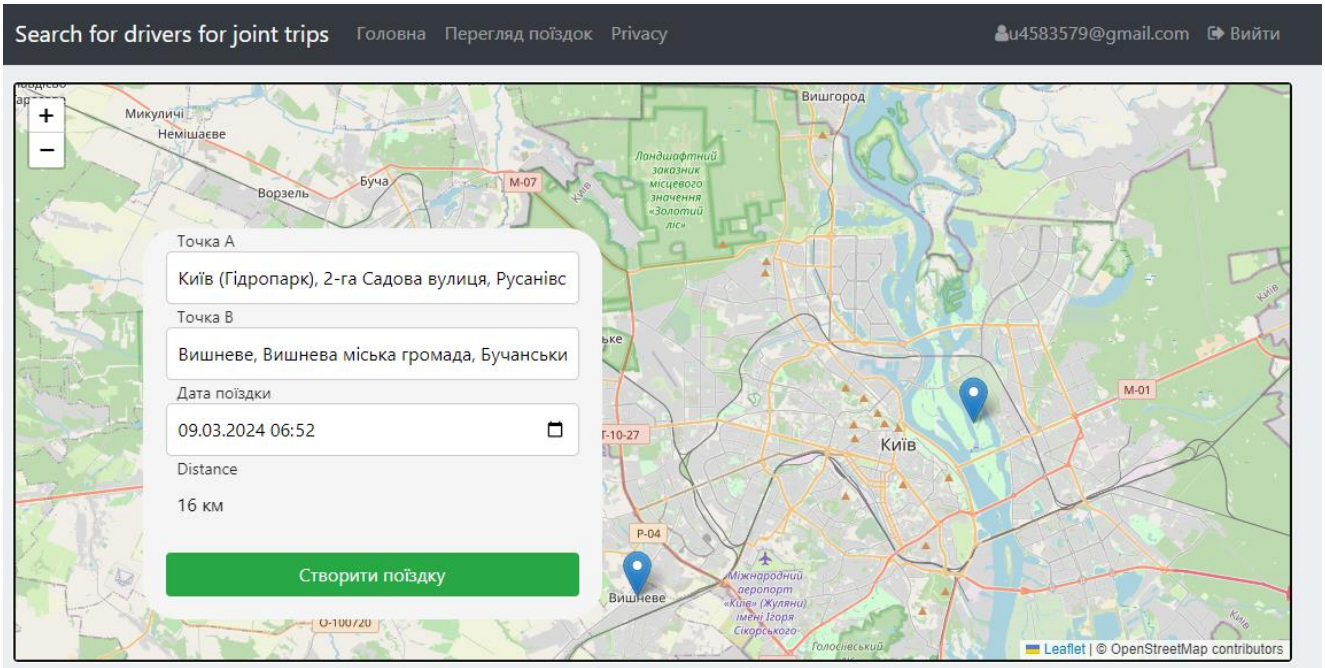

Рис. 3.5. Головна сторінка зі списком поїздок

Якщо користувач авторизований як водій, він має можливість створювати нові поїздки (рис. 3.6). Для цього йому потрібно заповнити форму з деталями поїздки, такими як місце відправлення, місце призначення, дата і час, кількість доступних місць та ціна. Після створення поїздки вона стає доступною для бронювання пасажирами.

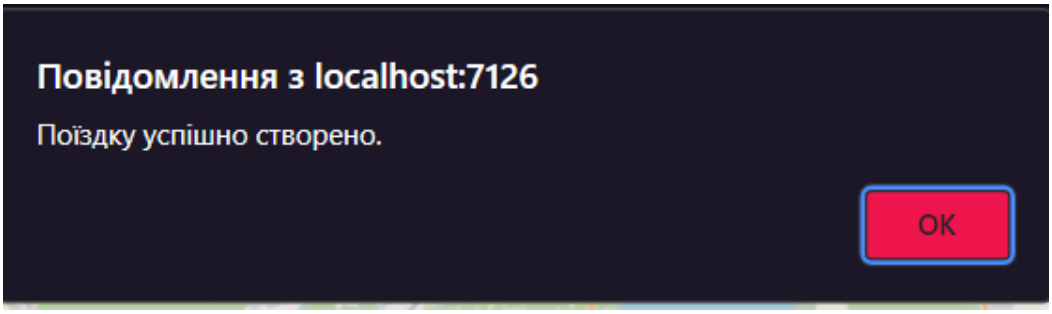

Рис. 3.6. Сторінка створення поїздки

Пасажири можуть забронювати місця у доступних поїздках (рис. 3.7). Для цього вони вибирають потрібну поїздку зі списку, вказують кількість місць, які хочуть забронювати, та підтверджують бронювання. Після успішного бронювання місць, пасажир отримує підтвердження на свою електронну пошту.

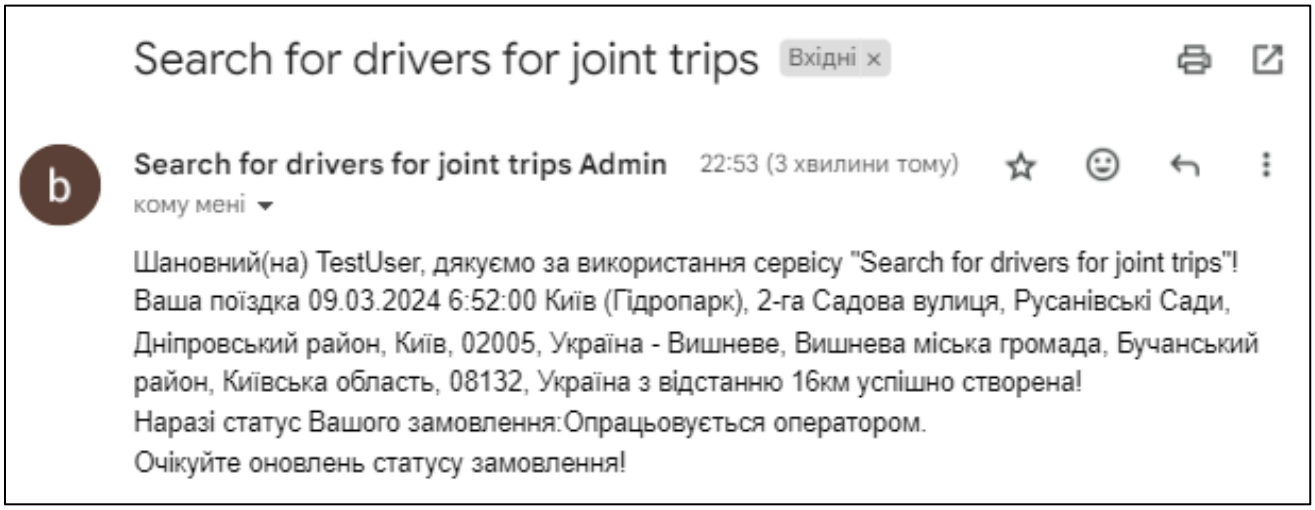

Рис. 3.7. Сторінка бронювання місць у поїздці

У розділі "Профіль" (рис 3.8) користувач може переглядати та редагувати свою особисту інформацію, таку як ім’я, електронна пошта, телефон та пароль. Водії також мають можливість додавати та керувати своїми автомобілями.

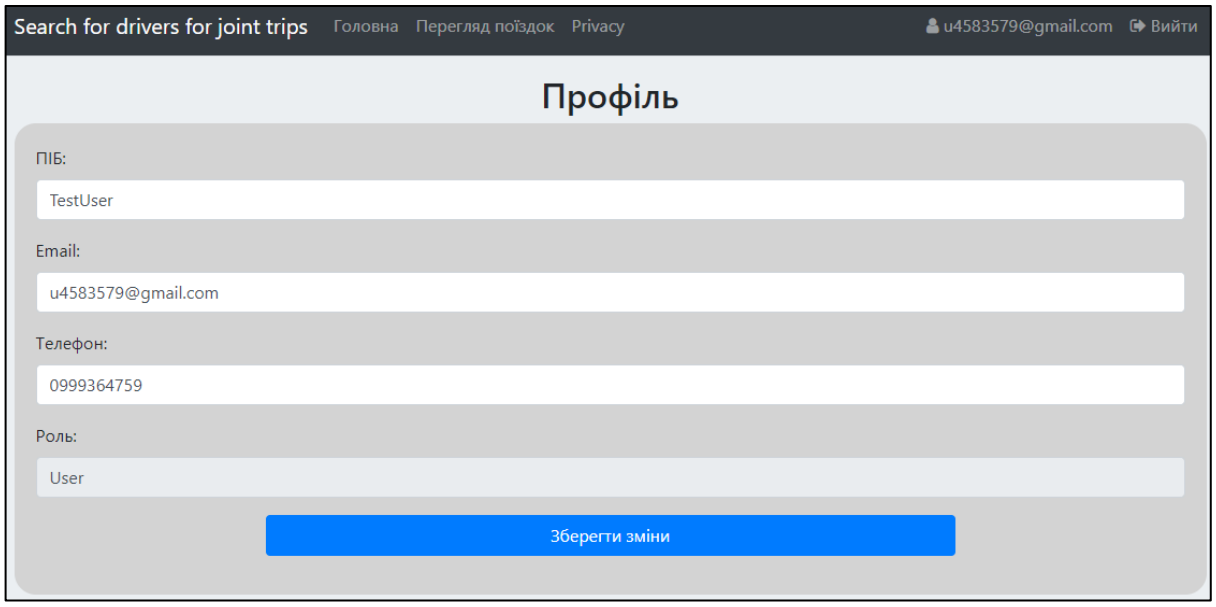

Рис. 3.8. Сторінка профілю користувача

Користувачі можуть переглядати історію своїх поїздок та бронювань (рис. 3.9). Для водіїв відображаються створені ними поїздки, а для пасажирів – заброньовані ними місця у поїздках. Це дозволяє користувачам відстежувати свою активність та керувати своїми поїздками [17].

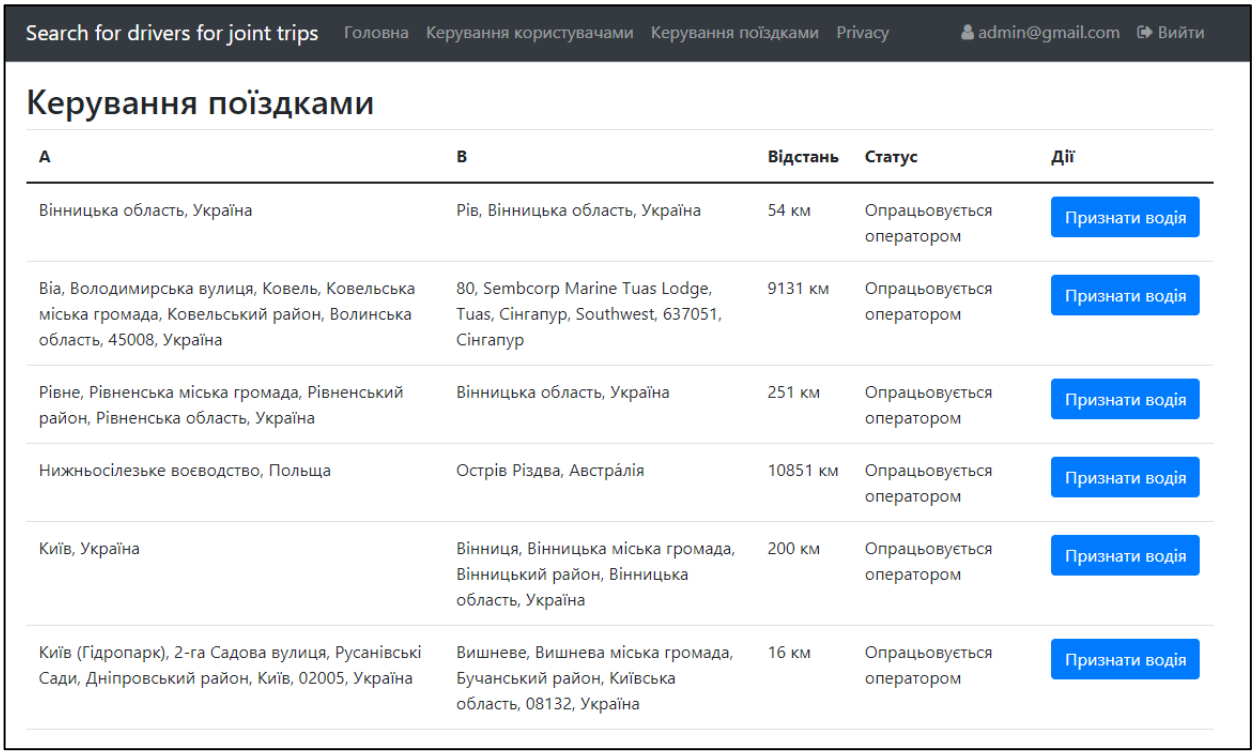

Рис. 3.9. Сторінка історії поїздок та бронювань

Адміністратори мають доступ до панелі керування (рис. 3.10), де вони можуть переглядати та керувати користувачами, поїздками та бронюваннями. Вони можуть видаляти користувачів, змінювати їхні ролі, а також модерувати поїздки та бронювання.

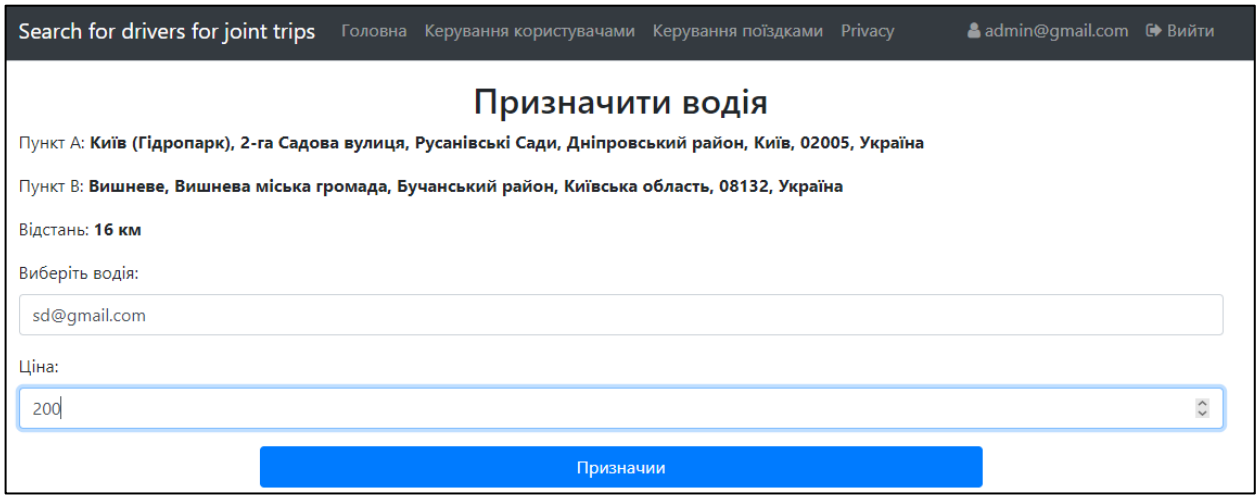

Рис. 3.10. Панель керування адміністратора

Вебсервіс карпулінгу має зручний та інтуїтивно зрозумілий інтерфейс користувача, який забезпечує легку навігацію та взаємодію з системою. Він надає користувачам можливість ефективно знаходити та організовувати спільні поїздки, що сприяє зменшенню витрат на транспорт та зниженню негативного впливу на довкілля [15].

ВИСНОВКИ

У даній дипломній роботі був розглянутий та реалізований вебсервіс карпулінгу, спрямований на підвищення зручності пересування, зменшення трафіку на дорогах та зниження витрат на транспорт. Результати дослідження та розробки свідчать про актуальність та необхідність використання сучасних технологій для оптимізації транспортної системи та сприяння спільному використанню автомобілів.

Розроблений вебсервіс включає в себе функціональність для реєстрації користувачів, створення та пошуку поїздок, бронювання місць у поїздках, а також адміністративні можливості для керування користувачами та модерації контенту. Інтерфейс користувача максимально зручний та інтуїтивно зрозумілий, що забезпечує ефективну взаємодію з системою.

Однією з ключових переваг розробленого вебсервісу є спрощення процесу пошуку та організації спільних поїздок. Користувачі можуть легко знаходити попутників, які прямують в одному напрямку, що дозволяє зменшити витрати на транспорт та оптимізувати використання автомобілів.

Також важливою перевагою є використання технологій автентифікації та авторизації, що забезпечує безпеку та конфіденційність особистих даних користувачів. Застосування алгоритму хешування паролів додає додатковий рівень захисту та підвищує довіру користувачів до системи.

Вебсервіс карпулінгу сприяє зменшенню кількості автомобілів на дорогах, що призводить до зниження трафіку та зменшення викидів шкідливих речовин в атмосферу. Це позитивно впливає на екологічну ситуацію та якість життя в містах.

Розроблений вебсервіс має потенціал для подальшого розвитку та вдосконалення. Можливі напрямки включають інтеграцію з системами громадського транспорту, додавання функцій рейтингу та відгуків для користувачів, а також розширення географії послуг на інші міста та регіони.

Отже, вебсервіс карпулінгу є ефективним та інноваційним рішенням для оптимізації транспортної системи та сприяння спільному використанню автомобілів. Він надає користувачам зручний та економічно вигідний спосіб пересування, а також сприяє зменшенню негативного впливу транспорту на довкілля. Подальший розвиток та популяризація подібних сервісів можуть зробити значний внесок у створення більш стійкої та ефективної транспортної інфраструктури в містах.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Документація ASP.NET Core. (n.d.). URL: https://docs.microsoft.com/en-us/aspnet/core/?view=aspnetcore-5.0 (дата звернення 05.06.2024).
2. Документація Entity Framework Core. (n.d.). URL:https://docs.microsoft.com/en-us/ef/core/ (дата звернення 05.06.2024).
3. PBKDF2 алгоритм хешування паролів. (2017). URL: https://www.ietf.org/rfc/rfc8018.txt (дата звернення 05.06.2024).
4. Google Maps API Documentation. (n.d.). URL: https://developers.google.com/maps/documentation (дата звернення 05.06.2024).
5. Хешування паролю. (2023). URL: https://auth0.com/blog/hashing-passwords-one-way-road-to-security/ (дата звернення 05.06.2024).
6. Ms Sql документація. URL: https://www.w3schools.com/sql/ (дата звернення 05.06.2024).