Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет центр післядипломної освіти
(повна назва)

Кафедра програмної інженерії
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

Програмна система керування автостоянкою
(тема)

Виконав:
студент 4 курсу, групи ПЗПІп-22-2

Кальє А.А.
(прізвище, ініціали)

Спеціальність 121 – Інженерія програмного забезпечення
(код і повна назва спеціальності)
Тип програми освітньо-професійна
Освітня програма Програмна інженерія
(повна назва освітньої програми)

Керівник доц. Вечур О.В.
(посада, прізвище, ініціали)

Допускається до захисту
Зав. кафедри _____ доц. Дудар З.В.
(підпис) (прізвище, ініціали)

2024 р.

Харківський національний університет радіоелектроніки

Центр післядипломної освіти
Кафедра програмної інженерії
Рівень вищої освіти перший (бакалаврський)
Спеціальність 121 – Інженерія програмного забезпечення
Тип програми Освітньо-професійна
Освітня програма Програмна Інженерія
(шифр і назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«___» _____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Кальє Андрію Адольфовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Програмна система керування автостоянкою
Затверджена наказом по університету від 17.06. 2024р. № 588 Ст
2. Термін подання студентом роботи до екзаменаційної комісії 15.07.2024
3. Вихідні дані до роботи Розробити програмний додаток з базою данних автостоянки, з мінімум двома ролями користувачів та можливістю додавати та редагувати записи, здійснювати пошук інформації
4. Перелік питань, що потрібно опрацювати в роботі
Вступ, аналіз предметної галузі, формування вимог до програмної системи, архітектура та проектування програмного забезпечення, опис прийнятих програмних рішень, тестування розробленого програмного забезпечення, висновки, додатки.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Аналіз предметної галузі	17.06.2024	*виконано*
2	Створення специфікації ПЗ	22.06.2024	*виконано*
3	Проектування ПЗ	24.06.2024	*виконано*
4	Розробка ПЗ	28.06.2024	*виконано*
5	Тестування ПЗ	30.06.2024	*виконано*
6	Оформлення пояснювальної записки	12.07.2024	*виконано*
7	Підготовка презентації та доповіді	13.07.2024	*виконано*
8	Попередній захист		
9	Нормоконтроль, рецензування		
10	Здача роботи у електронний архів		
11	Допуск до захисту у зав. кафедри		

Дата видачі завдання 17 червня 2024р.

Студент (ка)_____ Кальє А.А.
(підпис)

Керівник роботи _____ доц. Вечур О.В.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ / ABSTRACT

Пояснювальна записка до кваліфікаційної роботи бакалавра, 54 стор., 12 рис., 2 табл., 6 джерел.

БАЗА ДАНИХ, СИСТЕМА КЕРУВАННЯ, C#, SQL.

Об’єкт розробки – програмна система керування автостоянкою.

Мета розробки – створення програмної системи керування автостоянкою зі сторони адміна та користувача.

Метод рішення – середовище розробки Microsoft Visual Studio, мови програмування C# та SQL.

Explanatory note to the bachelor's qualification thesis, 53 pages, 12 figures, 2 tables, 6 sources.

DATABASE, MANAGEMENT SYSTEM, C#, SQL.

The object of development is a parking lot management software system.

The goal of the development is to create a software system for managing the parking lot from the side of the administrator and the user.

The solution method is the Microsoft Visual Studio development environment, C# and SQL programming languages.

Я, Кальє Андрій Адольфович, студент гр. ПЗПІ-22-2, здобувач вищої освіти на першому (бакалаврському) рівні кафедри «Програмна інженерія», заявляю: моя кваліфікаційна робота на тему «Програмна система керування автостоянкою», що буде представлена до екзаменаційної комісії для публічного

захисту, виконана самостійно, в ній не містяться елементи плагіату і вона може бути опублікована в електронному архіві відкритого доступу ElAr KhNURE. Усі запозичення з друкованих та електронних джерел мають відповідні посилання.

Я ознайомлений з діючим положенням «Про протидію академічному плагіату в ХНУРЕ», згідно з яким виявлення плагіату є підставою для відмови до допуску кваліфікаційної роботи до захисту та застосування дисциплінарних заходів.

ЗМІСТ

ВСТУП

Програмна система керування автостоянкою є актуальним та необхідним інструментом у сучасному урбанізованому середовищі, де ефективне управління паркувальним простором стає все більш критичним завданням. З ростом кількості автомобілів у містах та обмеженістю паркувальних площ, виникає нагальна потреба в оптимізації використання наявних ресурсів та підвищенні якості обслуговування власників транспортних засобів. Традиційні методи управління автостоянками, що базуються на паперовому документообігу та ручному обліку, вже не відповідають вимогам часу, оскільки вони є трудомісткими, схильними до помилок та неефективними в умовах високої завантаженості.

Впровадження автоматизованої системи керування автостоянкою дозволяє вирішити ряд критичних проблем, таких як неефективне використання паркувальних місць, складності з обліком та контролем оплати, а також забезпечення безпеки транспортних засобів. Крім того, така система здатна значно покращити досвід користувачів, надаючи їм зручні інструменти для пошуку вільних місць, бронювання та оплати послуг паркування.

Метою розробки програмної системи керування автостоянкою є створення комплексного рішення, яке дозволить автоматизувати та оптимізувати всі аспекти управління паркувальним простором. Це включає в себе заміну паперових журналів обліку транспортних засобів та касових книг на електронну систему, яка забезпечить точний та оперативний облік всіх операцій. Система має на меті підвищити ефективність використання паркувальних місць, спростити процес оплати та контролю доступу, а також забезпечити власників та операторів автостоянок актуальною аналітичною інформацією для прийняття обґрунтованих рішень.

Завдання, які ставляться перед програмною системою, охоплюють широкий спектр функціональностей. Вони включають створення бази даних для

зберігання інформації про паркувальні місця, автомобілі, їх власників та абонементи. Система повинна забезпечувати можливість швидкого пошуку та фільтрації даних, що дозволить операторам ефективно управляти великими обсягами інформації. Важливим завданням є реалізація функціоналу для автоматичного формування різноманітних звітів, включаючи статистику доходів за різні періоди, аналіз завантаженості стоянки та облік транспортних засобів за різними параметрами.

Окремим важливим завданням є розробка механізму автоматичного перепланування паркувальних місць у випадку проведення ремонтних робіт чи інших змін у конфігурації стоянки. Це дозволить мінімізувати незручності для клієнтів та оптимізувати використання доступного простору навіть в умовах технічних обмежень.

Реалізація цієї системи не тільки підвищить ефективність управління автостоянкою, але й сприятиме покращенню екологічної ситуації в містах шляхом оптимізації використання паркувального простору та зменшення часу, який водії витрачають на пошук місця для паркування. Крім того, вона створить основу для подальшої інтеграції з іншими міськими системами, такими як громадський транспорт чи системи управління дорожнім рухом, що в перспективі дозволить реалізувати концепцію розумного міста в сфері управління транспортною інфраструктурою.

Таким чином, розробка програмної системи керування автостоянкою є не просто технологічним вдосконаленням, а важливим кроком у напрямку створення більш ефективної, зручної та екологічно дружньої міської інфраструктури, що відповідає сучасним викликам урбанізації та зростаючим потребам населення у якісних послугах паркування.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Аналіз предметної галузі

Предметна область автоматизації автостоянки охоплює комплексний набір процесів та елементів, пов'язаних з ефективним управлінням паркувальним простором. Ця сфера включає в себе всі аспекти функціонування сучасної автостоянки, починаючи від базового обліку вільних та зайнятих місць і закінчуючи складними системами аналітики та прогнозування.

Ключовим компонентом є управління паркувальними місцями, що передбачає постійний моніторинг їх статусу та оптимізацію використання доступного простору. Це тісно пов'язано з системою контролю доступу, яка регулює в'їзд та виїзд транспортних засобів, використовуючи різноманітні технологічні рішення, такі як автоматичні шлагбауми, зчитувачі номерних знаків та смарт-картки.

Важливою складовою є облік та ідентифікація транспортних засобів, що паркуються. Система повинна фіксувати та зберігати інформацію про кожен автомобіль, включаючи час прибуття та відбуття, що є критичним для правильного нарахування оплати. Це безпосередньо пов'язано з фінансовим аспектом управління автостоянкою, який включає в себе гнучку тарифікацію, обробку різних видів платежів та генерацію фінансової звітності.

Безпека є невід'ємною частиною автоматизації автостоянки. Вона охоплює системи відеоспостереження, охоронної сигналізації та швидкого реагування на надзвичайні ситуації. Це не тільки захищає транспортні засоби клієнтів, але й забезпечує загальну безпеку об'єкта.

Обслуговування клієнтів також є важливим аспектом, який включає в себе управління базою даних клієнтів, обробку запитів та скарг, а також надання

додаткових послуг, таких як бронювання місць або інформування про наявність вільних місць.

Технічне обслуговування інфраструктури автостоянки є ще одним ключовим елементом. Сюди входить моніторинг стану обладнання, планування профілактичних робіт та швидке усунення несправностей для забезпечення безперебійної роботи.

Екологічний аспект стає все більш важливим у сучасному світі. Автоматизація автостоянки повинна враховувати потреби електромобілів, включаючи наявність зарядних станцій та спеціальних місць для екологічно чистого транспорту.

Інтеграція з міськими системами та сторонніми сервісами також є частиною предметної області. Це може включати взаємодію з муніципальними службами, навігаційними додатками та платіжними системами для створення єдиної екосистеми паркування в місті.

Аналітика та прогнозування відіграють важливу роль у сучасному управлінні автостоянкою. Ця складова дозволяє оптимізувати роботу на основі історичних даних, прогнозувати завантаженість та приймати обґрунтовані рішення щодо розвитку бізнесу.

Нарешті, гнучкість та масштабованість системи є критичними для адаптації до мінливих умов ринку, нових технологій та зростаючих потреб користувачів. Система автоматизації автостоянки повинна бути здатною еволюціонувати разом з розвитком транспортної інфраструктури та змінами в поведінці споживачів.

Основні функції системи можуть включати:

– реєстрація користувачів та автомобілів;

– пошук доступних місць для паркування;

– резервування місця на стоянці;

– оплата за користування стоянкою;

– моніторинг використання місць на стоянці;

– ведення звітності про доходи та витрати.

Реєстрація користувачів та автомобілів, ця функція передбачає можливість реєстрації користувачів, які бажають користуватися послугами платної автомобільної стоянки. При реєстрації користувач повинен ввести свої особисті дані (ім'я, прізвище, контактні дані) та дані про автомобіль (марка, модель, номерний знак тощо). Ці дані використовуються для подальшого користування системою.

Пошук доступних місць для паркування, ця функція надає можливість знайти вільне місце для паркування на платній автомобільній стоянці. Користувач може вибрати місце на стоянці згідно зі своїми потребами - біля входу, поруч зі зупинкою громадського транспорту тощо. Система повинна мати мапу стоянки та опис місць, щоб користувач міг швидко знайти підходяще місце для паркування.

Резервування місця на стоянці, ця функція дозволяє зарезервувати вільне місце на стоянці на певний період часу. Користувач може обрати період, на який він бажає зарезервувати місце - наприклад, на годину, дві або на весь день. Це дозволяє користувачеві мати гарантований доступ до вільного місця на стоянці, що сприяє його комфорту та зручності.

Оплата за користування стоянкою, ця функція передбачає можливість оплатити користування стоянкою. Система повинна мати різні способи оплати

Моніторинг використання місць на стоянці, ця функція відстежує використання кожного місця на стоянці, що дозволяє уникнути ситуації, коли стоянка переповнена, а деякі місця залишаються вільними. Система автоматично відстежує, які місця зайняті, а які - вільні, та надає цю інформацію користувачам у режимі реального часу. Це дає можливість користувачам точно знати, чи є вільні місця на стоянці, та відповідно планувати свій приїзд.

Ведення звітності про доходи та витрати, ця функція дозволяє вести детальну звітність про доходи та витрати системи «ПЛАТНА АВТОМОБІЛЬНА

СТОЯНКА». Система автоматично реєструє всі оплати, що надходять від користувачів, та зберігає цю інформацію в базі даних. Крім того, система відстежує всі витрати, пов'язані з підтримкою та обслуговуванням стоянки - зарплати працівників, оплата комунальних послуг, закупівля обладнання тощо. Звітність може бути згенерована в різних форматах - наприклад, у вигляді таблиць, графіків чи діаграм.

1.2 Виявлення та вирішення проблем

Програмні системи керування автостоянками часто стикаються з низкою проблем, які впливають на їхню ефективність та надійність.[6] Однією з ключових проблем є недостатня гнучкість системи, яка обмежує можливість адаптації до змінних умов ринку та потреб клієнтів. Це може проявлятися в неможливості швидко змінювати тарифні плани або додавати нові функції без значних змін у коді. Вирішення цієї проблеми полягає у використанні модульної архітектури та конфігураційних файлів, які дозволяють легко модифікувати поведінку системи без необхідності перепрограмування.

Інша поширена проблема – це недостатня надійність системи при високих навантаженнях, особливо в пікові години або під час масових заходів. Це може призводити до збоїв у роботі, втрати даних або неправильного обліку транспортних засобів. Для вирішення цієї проблеми необхідно впроваджувати масштабовані архітектури, такі як мікросервіси, які дозволяють розподіляти навантаження між декількома серверами. Крім того, важливо використовувати надійні системи управління базами даних з підтримкою транзакцій та резервного копіювання.

Безпека даних є ще одним критичним аспектом, який часто недостатньо враховується в існуючих системах. Це може призводити до витоку конфіденційної інформації про клієнтів або уразливості системи до кібератак.

Вирішення цієї проблеми вимагає комплексного підходу, включаючи шифрування даних, використання надійних методів аутентифікації, регулярні аудити безпеки та навчання персоналу з питань кібербезпеки.

Інтеграція з іншими системами часто є проблематичною областю. Багато існуючих систем керування автостоянками працюють як ізольовані рішення, що ускладнює їх взаємодію з міськими інформаційними системами, платіжними шлюзами або мобільними додатками. Для вирішення цієї проблеми необхідно розробляти та впроваджувати стандартизовані API, які дозволяють легко інтегрувати систему з іншими сервісами та платформами.

Недостатня точність у визначенні наявності вільних місць є ще однією поширеною проблемою, яка може призводити до незадоволення клієнтів та неефективного використання паркувального простору. Вирішення цієї проблеми можливе за рахунок впровадження сучасних сенсорних технологій, таких як ультразвукові датчики або камери з комп'ютерним зором, які забезпечують точний моніторинг кожного паркувального місця в реальному часі.

Застарілі інтерфейси користувача, які не відповідають сучасним стандартам зручності використання, також є проблемою багатьох існуючих систем. Це може ускладнювати роботу операторів та знижувати задоволеність клієнтів. Вирішення цієї проблеми вимагає розробки інтуїтивно зрозумілих інтерфейсів з урахуванням принципів UX/UI дизайну та проведення регулярних тестувань з реальними користувачами для оптимізації взаємодії.

Недостатня аналітика та відсутність прогнозних можливостей обмежують здатність операторів автостоянок приймати обґрунтовані рішення та оптимізувати свій бізнес. Вирішення цієї проблеми передбачає інтеграцію сучасних інструментів аналізу даних та машинного навчання, які дозволяють прогнозувати завантаженість, оптимізувати тарифи та виявляти тренди у поведінці клієнтів.

Нарешті, багато існуючих систем не враховують екологічні аспекти, такі як підтримка електромобілів або оптимізація використання простору для зменшення викидів. Вирішення цієї проблеми вимагає впровадження функціоналу для управління зарядними станціями, пріоритетного розміщення екологічно чистого транспорту та інтеграції з системами громадського транспорту для заохочення альтернативних способів пересування.

Проаналізуємо сайт конкурент із Харкова https://parking.kh.city/ (див. рис. 1.1).

Перше, з чим зіткнеться користувач, це не можливість одразу перейти на головну сторінку, так як сайт має прострочений сертифікат безпеки.

Головна сторінка сайти має нормальний вигляд для комп'ютерної та мобільної версії.

Якщо користувач використовує самсунг браузер, який є браузером за замовчуванням для смартфонів самсунг, то сторінка відображається з використанням інших стилів.

Сайт має такі опції для користувачів, як:

– оплата;

– паркувальний талон;

– електронний абонемент;

– реєстрація;

– оплата через смс;

– приват 24;

– паркінг бот.

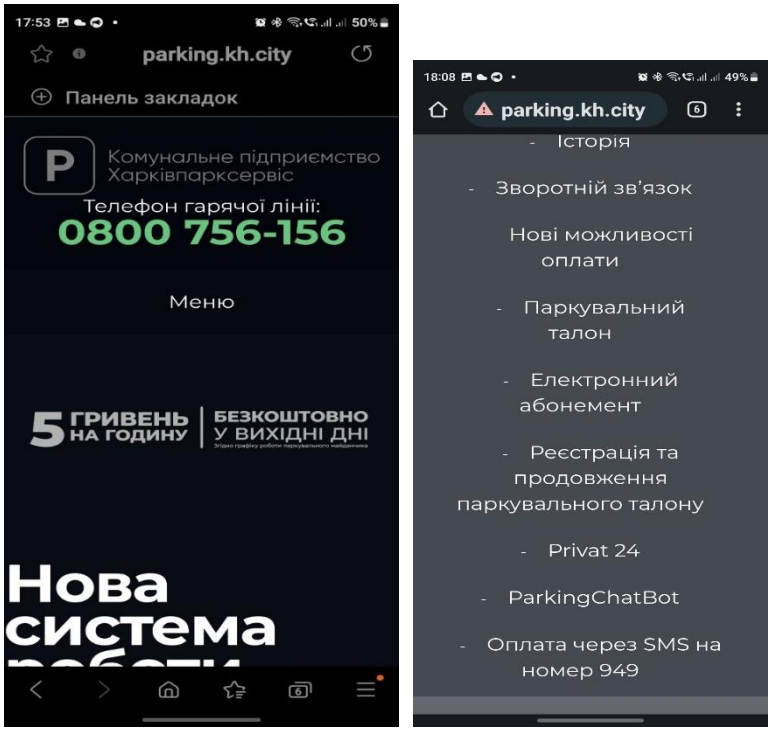


Рисунок 1.1 – Сайт конкурент із Харкова. https://parking.kh.city/

1.3 Постановка задачі

Постановка задачі на розробку додатку для керування автостоянкою передбачає створення програмного забезпечення, яке дозволить ефективно управляти паркувальними місцями та обліком автомобілів. Система повинна мати два рівні доступу: адміністратор та звичайний користувач.

Адміністратор матиме повний доступ до всіх функцій системи. Він зможе додавати, редагувати та видаляти інформацію про автомобілі, їх власників та паркувальні місця. Адміністратор матиме можливість переглядати повну статистику використання стоянки, генерувати звіти та керувати налаштуваннями системи. Також він зможе призначати ролі іншим користувачам та керувати їхніми правами доступу.

Звичайний користувач матиме обмежений доступ до функцій системи. Він зможе переглядати інформацію про вільні паркувальні місця, здійснювати пошук

за різними параметрами (наприклад, за маркою автомобіля або прізвищем власника), а також вносити дані про свій автомобіль та бронювати місце на стоянці. Користувач не матиме доступу до редагування чи видалення даних інших клієнтів та не зможе змінювати системні налаштування.

Система повинна забезпечувати зручний інтерфейс для обох типів користувачів, з можливістю швидкого пошуку та фільтрації даних. Важливою функцією є можливість відстеження стану автомобілів на стоянці, включаючи наявність пошкоджень або неповної комплектації. Це дозволить уникнути потенційних конфліктів між власниками автомобілів та адміністрацією стоянки.

Додаток повинен мати функцію автоматичного оновлення статусу паркувальних місць при в'їзді та виїзді автомобілів. Це забезпечить актуальність інформації про доступність місць у режимі реального часу. Також необхідно передбачити можливість резервування місць на певний період часу.

Безпека даних є критично важливим аспектом системи. Необхідно забезпечити надійне шифрування особистої інформації клієнтів та захист від несанкціонованого доступу. Система повинна вести журнал всіх дій користувачів для можливості аудиту та відстеження потенційних порушень.

Інтерфейс додатку має бути інтуїтивно зрозумілим та адаптивним для різних пристроїв, включаючи комп'ютери, планшети та смартфони. Це дозволить користувачам зручно працювати з системою з будь-якого місця та в будь-який час.

Система повинна мати можливість інтеграції з іншими сервісами, такими як платіжні системи для оплати послуг стоянки, а також з системами відеоспостереження для підвищення рівня безпеки.

Важливою вимогою є можливість масштабування системи для роботи з різними розмірами автостоянок – від невеликих парковок до великих багаторівневих комплексів. Це передбачає гнучку архітектуру та можливість легкого додавання нових функцій та модулів.

2 ФОРМУВАННЯ ВИМОГ ДО ПРОГРАМНОЇ СИСТЕМИ

При проектуванні системи керування автостоянкою, враховуючи надані вимоги та мету автоматизації, можна сформулювати наступні детальні функціональні вимоги:

а) управління даними:

1) створення, редагування та видалення записів про паркувальні місця;

2) створення, редагування та видалення записів про автомобілі;

3) створення, редагування та видалення записів про власників автомобілів;

4) створення, редагування та видалення записів про абонементи.

б) пошук та фільтрація:

1) пошук автомобіля за номером держ. Реєстрації;

2) пошук автомобіля за номером паркувального місця;

3) пошук власника за номером держ. реєстрації автомобіля;

4) фільтрація автомобілів за маркою та моделлю;

5) сортування списку автомобілів за маркою та моделлю.

в) управління паркуванням:

1) додавання нового автомобіля на стоянку з перевіркою наявності власника в базі даних;

2) видалення автомобіля зі стоянки зі збереженням даних про власника;

3) видалення власника з перевіркою відсутності активних абонементів;

4) додавання нових абонементів;

5) продовження існуючих абонементів;

6) ануляція прострочених абонементів.

г) формування статистики:

1) статистика кількості та доходу від проданих місячних абонементів;

2) статистика кількості проданих паркувальних місць з датою та доходом від добових абонементів;

3) облік кількості транспортних засобів у розрізі марки та моделі.

д) звітність:

1) генерація та друк місячних та добових абонементів на паркування;

2) формування звіту про дохід за місяць;

3) можливість формування довільного запиту до бази даних на мові Transact-SQL.

е) автоматичне перепланування:

1) моніторинг стану паркувальних місць;

2) визначення зон, що потребують ремонту чи обслуговування;

3) автоматичне блокування паркувальних місць, що підлягають ремонту;

4) перерозподіл автомобілів на доступні місця;

5) оповіщення власників про необхідність переміщення автомобілів;

6) оновлення схеми паркування в реальному часі.

ж) безпека та цілісність даних:

1) шифрування персональних даних власників;

2) Резервне копіювання бази даних;

3) журналювання всіх операцій в системі;

4) розмежування прав доступу для різних рівнів користувачів.

и) інтерфейс користувача:

1) інтуїтивно зрозумілий графічний інтерфейс;

2) адаптивний дизайн для різних пристроїв;

3) швидкий доступ до основних функцій системи.

к) інтеграція:

1) можливість інтеграції з платіжними системами;

2) інтеграція з системами відеоспостереження;

3) можливість експорту даних в різні формати (Excel, PDF).

л) масштабованість та продуктивність:

1) здатність обробляти великі обсяги даних;

2) можливість розширення функціоналу без перебудови системи;

3) оптимізація швидкодії при роботі з великими паркувальними комплексами.

Сформуємо таблицю функціональних вимог програмного додатку.

Таблиця 2.1 – Функціональні вимоги

ID	Функціональна вимога	Опис	Пріоритет	Складність
1.1	Створення записів про паркувальні місця	Можливість додавати нові паркувальні місця з вказанням номера, статусу, типу	Високий	Низька
1.2	Редагування записів про паркувальні місця	Зміна інформації про існуючі паркувальні місця	Середній	Низька
1.3	Видалення записів про паркувальні місця	Видалення неактуальних паркувальних місць з бази даних	Низький	Низька
2.1	Створення записів про автомобілі	Додавання нових автомобілів з вказанням марки, моделі, номера держ. реєстрації	Високий	Низька
2.2	Редагування записів про автомобілі	Оновлення інформації про існуючі автомобілі	Середній	Низька
2.3	Видалення записів про автомобілі	Видалення автомобілів з бази даних при виїзді зі стоянки	Високий	Середня
3.1	Створення записів про власників	Додавання нових власників з їх контактною інформацією	Високий	Низька
3.2	Редагування записів про власників	Оновлення контактної інформації власників	Середній	Низька
3.3	Видалення записів про власників	Видалення власників з перевіркою відсутності активних абонементів	Низький	Середня
4.1	Створення абонементів	Оформлення нових абонементів на паркування	Високий	Середня
4.2	Продовження абонементів	Можливість продовжити термін дії існуючого абонемента	Високий	Низька
4.3	Ануляція абонементів	Скасування прострочених або невикористаних абонементів	Середній	Середня

5.1	Пошук авто за номером держ. реєстрації	Швидкий пошук автомобіля в базі за його номером	Високий	Низька
5.2	Пошук авто за номером паркувального місця	Знаходження автомобіля за його розташуванням на стоянці	Високий	Низька
5.3	Пошук власника за номером авто	Швидкий доступ до інформації про власника за номером автомобіля	Високий	Низька
5.4	Фільтрація авто за маркою та моделлю	Можливість відфільтрувати список автомобілів за певними критеріями	Середній	Низька
5.5	Сортування списку авто	Впорядкування списку автомобілів за різними параметрами	Низький	Низька
6.1	Статистика місячних абонементів	Формування звіту про кількість та дохід від місячних абонементів	Високий	Середня
6.2	Статистика добових абонементів	Формування звіту про кількість та дохід від добових абонементів	Високий	Середня
6.3	Облік транспортних засобів	Статистика кількості автомобілів за марками та моделями	Середній	Середня
7.1	Генерація абонементів	Створення та друк місячних та добових абонементів	Високий	Середня
7.2	Звіт про дохід	Формування звіту про загальний дохід за вибраний період	Високий	Середня
7.3	Довільні SQL-запити	Можливість виконання користувацьких запитів до бази даних	Низький	Висока
8.1	Моніторинг стану паркувальних місць	Автоматичне відстеження стану та доступності місць	Високий	Висока
8.2	Блокування місць на ремонт	Автоматичне позначення місць, що потребують ремонту	Середній	Середня
8.3	Перерозподіл автомобілів	Автоматичне перепланування розміщення авто при ремонтних роботах	Середній	Висока
8.4	Оповіщення власників	Автоматичне інформування власників про необхідність переміщення авто	Високий	Середня
9.1	Шифрування даних	Забезпечення безпеки персональних даних власників	Високий	Висока
9.2	Резервне копіювання	Регулярне створення резервних копій бази даних	Високий	Середня

9.3	Журналювання операцій	Запис усіх дій користувачів у системний журнал	Середній	Середня
10.1	Адаптивний інтерфейс	Зручний інтерфейс, що адаптується до різних пристроїв	Високий	Висока
10.2	Швидкий доступ до функцій	Оптимізований інтерфейс для швидкого доступу до основних функцій	Високий	Середня
11.1	Інтеграція з платіжними системами	Можливість онлайн-оплати послуг паркування	Середній	Висока
11.2	Інтеграція з відеоспостереженням	Зв'язок системи з камерами спостереження на стоянці	Низький	Висока
12.1	Оптимізація продуктивності	Забезпечення швидкої роботи системи при великих обсягах даних	Високий	Висока
12.2	Масштабованість	Можливість розширення системи без суттєвої перебудови	Середній	Висока

3 АРХІТЕКТУРА ТА ПРОЄКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 UML-проектування ПЗ

Опис функціонального призначення інформаційної системи, що створюється наведено на рисунку 3.1.

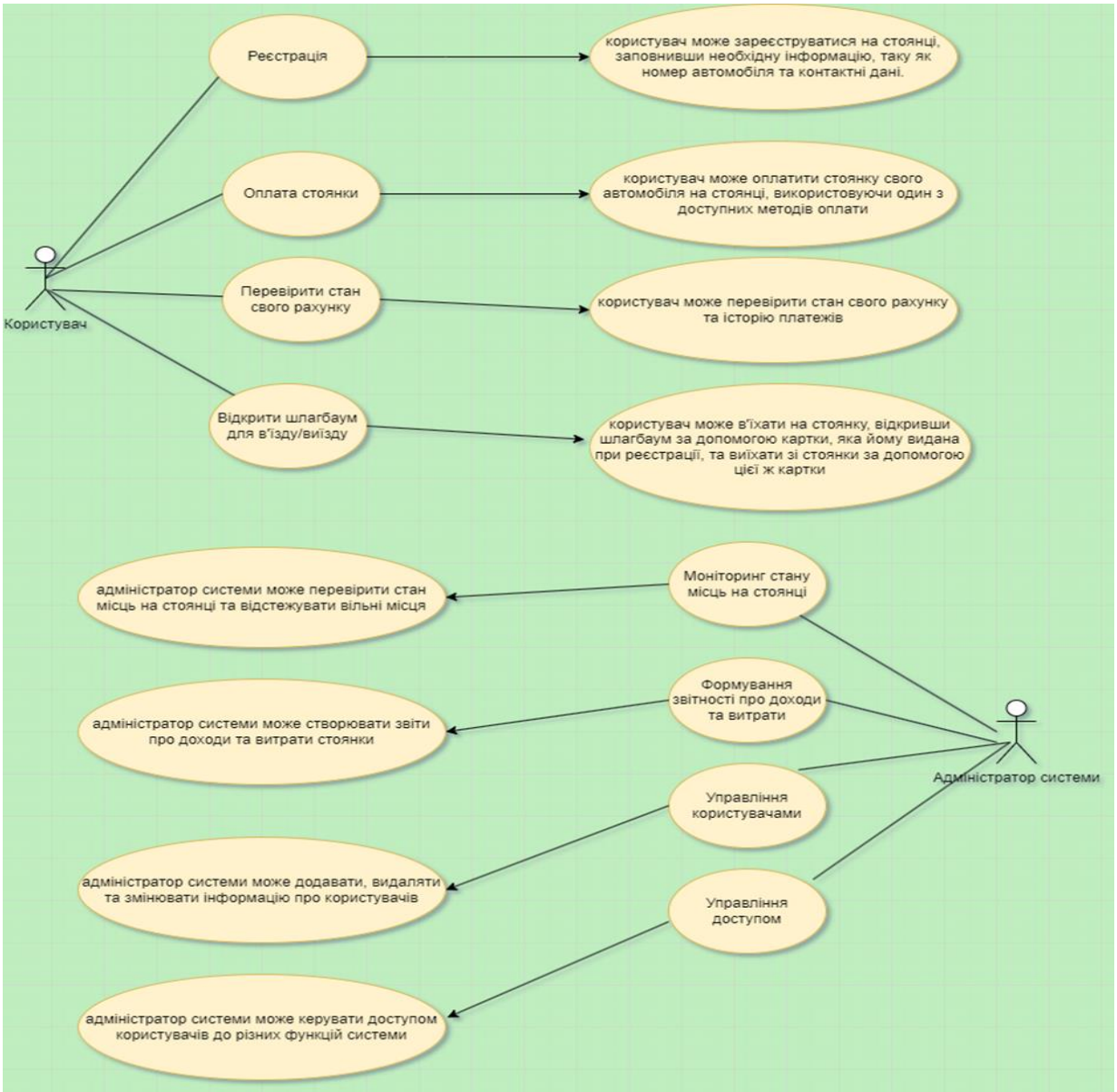


Рисунок 3.1 – Загальна USE-CASE діаграма

Основною метою інформаційної системи «ПЛАТНА АВТОМОБІЛЬНА СТОЯНКА» є забезпечення можливості користувачам оплачувати стоянку свого автомобіля на стоянці.[5] Загальна use-case діаграма для даної системи містить наступні елементи:

- зареєструватися на стоянці: користувач може зареєструватися на стоянці, заповнивши необхідну інформацію, таку як номер автомобіля та контактні дані;
- оплатити стоянку: користувач може оплатити стоянку свого автомобіля на стоянці, використовуючи один з доступних методів оплати;
- перевірити стан свого рахунку: користувач може перевірити стан свого рахунку та історію платежів;
- відкрити шлагбаум для в'їзду/виїзду: користувач може в'їхати на стоянку, відкривши шлагбаум за допомогою картки, яка йому видана при реєстрації, та виїхати зі стоянки за допомогою цієї ж картки;
- моніторинг стану місць на стоянці: адміністратор системи може перевірити стан місць на стоянці та відстежувати вільні місця;
- формування звітності про доходи та витрати: адміністратор системи може створювати звіти про доходи та витрати стоянки;
- управління користувачами: адміністратор системи може додавати, видаляти та змінювати інформацію про користувачів;
- управління доступом: адміністратор системи може керувати доступом користувачів до різних функцій системи.

Опис концептів програмної області, їх властивостей та зв'язків між ними опишемо на рисунку 3.2 у вигляді загальної діаграми класів:

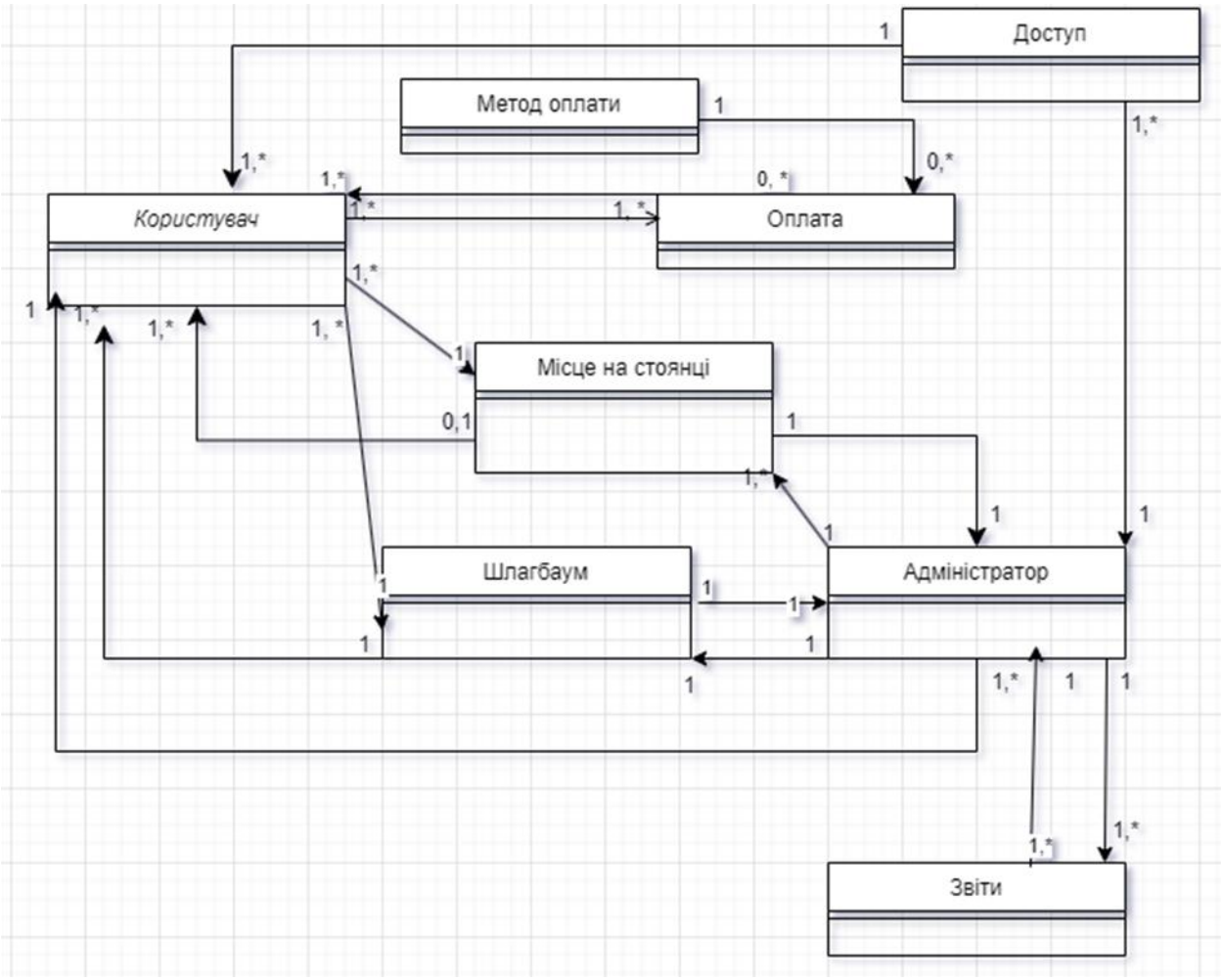


Рисунок 3.2 – Загальна діаграма класів

Клас "Користувач" – містить інформацію про користувачів системи, таку як номер автомобіля, контактні дані та історію платежів. Взаємодіє з класами "Оплата", "Місце на стоянці" та "Шлагбаум"; 2) Клас "Оплата" – містить інформацію про оплату стоянки користувачами. Взаємодіє з класом "Користувач"; 3) Клас "Місце на стоянці" - містить інформацію про стан місць на стоянці. Взаємодіє з класами "Користувач" та "Адміністратор"; 4) Клас "Шлагбаум" – містить інформацію про доступ до стоянки та стан шлагбауму. Взаємодіє з класами "Користувач" та "Адміністратор"; 5) Клас "Адміністратор" – містить інформацію про адміністраторів системи та їх повноваження. Взаємодіє з класами "Місце на стоянці", "Шлагбаум", "Звіти" та "Користувачі"; 6) Клас

"Звіти" – містить інформацію про доходи та витрати стоянки. Взаємодіє з класом "Адміністратор"; 7) Клас "Доступ" – містить інформацію про доступ користувачів до різних функцій системи. Взаємодіє з класом "Адміністратор" та "Користувач"; 8) Клас "Метод оплати" – містить інформацію про доступні методи оплати стоянки. Взаємодіє з класом "Оплата".

Схематично зобразимо інформаційні потреби робітника підприємства (див. рис. 3.3).

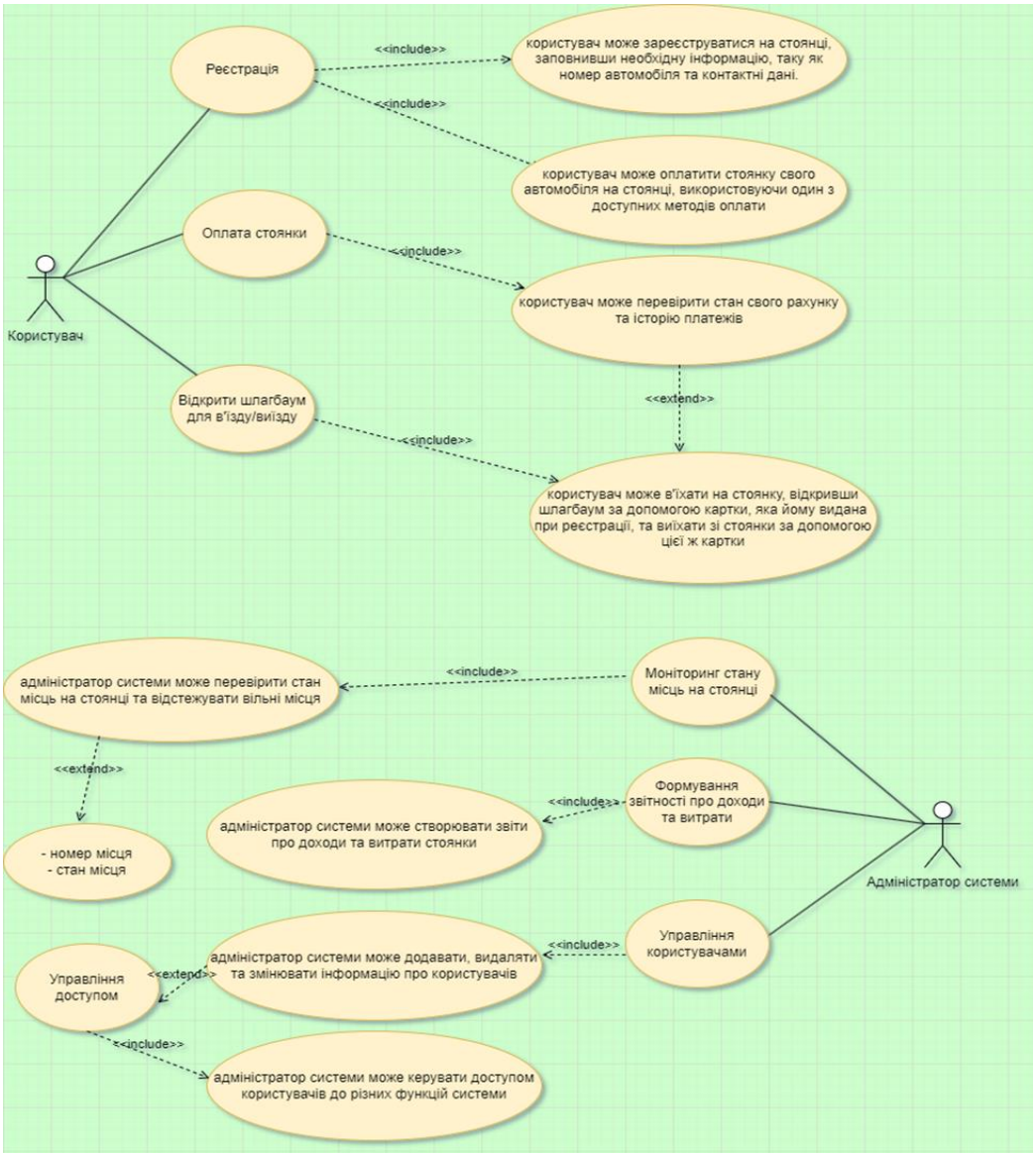


Рисунок 3.3 – Детальна USE-CASE діаграма для опису інформаційних потреб

Нижче наведено алгоритмічни залежності:

- бронювання паркувального місця: Для бронювання місця користувачеві потрібно обрати бажану дату та час, перевірити доступність місць, забронювати місце та забезпечити видачу квитка або підтвердження бронювання;

- оплата паркування: Після вибору місця користувач повинен оплатити паркування. Ця операція може включати в себе перевірку доступного балансу, обробку платежу і оновлення інформації про статус паркування;

- в'їзд і виїзд зі стоянки: Для в'їзду користувач повинен отримати доступ на стоянку, що може включати в себе автоматичне відкриття шлагбауму або введення коду доступу. Виїзд також може бути обмеженим або вільним, в залежності від оплати та часу;

- система відеоспостереження: Алгоритм для системи відеоспостереження може включати в себе виявлення руху, запис відео, збереження даних і розпізнавання номерних знаків автомобілів;

- управління запасами та місцями: Система повинна вести облік доступних паркувальних місць і оновлювати їх статус в реальному часі;

- зберігання та обробка даних користувачів: Інформаційна система повинна забезпечувати безпечне зберігання та обробку особистих даних користувачів, включаючи історію паркування і платежів;

- звіти і статистика: Система може генерувати звіти і статистичні дані щодо використання паркувальних місць, доходів, навантаження та інших аналітичних даних;

Нижче наведено опис обмежень цілісності:

- унікальність номерних знаків: Система повинна гарантувати, що номерні знаки кожного автомобіля, які зберігаються в базі даних, є унікальними. Це може вберегти від можливих конфліктів та помилок у веденні даних;

- цілісність транзакцій: Система повинна забезпечити атомарність, консистентність, ізоляцію та доларитмічність (ACID) транзакцій, щоб

гарантувати, що будь-які операції з даними виконуються правильно та без конфліктів;

- обмеження цін на паркування: Система повинна перевіряти, що встановлені ціни на паркування відповідають заданим правилам і не можуть бути недійсними або негативними;

- дотримання мінімальних та максимальних строків паркування: Система повинна гарантувати, що користувачі не можуть встановлювати неприпустимо короткі або дуже довгі строки паркування;

- заборона дублювання платежів: Система повинна перевіряти, що один і той же платіж не може бути зарахований двічі за однаковий період часу і місце паркування;

- валідація даних користувачів: Система повинна перевіряти правильність та коректність особистих даних користувачів, які реєструються або вносять зміни в свій профіль;

- підтвердження оплати перед в'їздом: Для автоматичного відкриття шлагбауму користувач повинен мати підтверджену оплату. Система має перевіряти статус оплати перед дозволом на в'їзд;

- заборона займання броньованих місць без оплати: Користувачі повинні оплачувати броньовані місця перед в'їздом на стоянку, і система має це контролювати;

Обмеження цілісності є критично важливими для забезпечення надійності, точності та безпеки інформації та операцій в ІС автостоянки. Вони допомагають запобігати помилкам, зловживанням і несанкціонованому доступу до даних.

Опис існуючого документообігу складається з наступних документів:

- створення бронювань: Користувачі можуть створювати бронювання паркувальних місць через веб-додаток або інші канали зв'язку. Під час створення бронювання створюються документи, які включають інформацію про

користувача, дату та час бронювання, місце паркування та іншу важливу інформацію;

- підтвердження бронювань: Після отримання бронювання система повинна створити підтверджувальні документи для користувача, включаючи номер бронювання, дату та час бронювання та іншу інформацію;

- оплата паркування: Після створення бронювання користувач повинен оплатити паркування. Система створює документи, що підтверджують оплату, включаючи інформацію про суму, спосіб оплати та дату оплати;

- управління доступом: Для користувачів, які мають право на в'їзд на стоянку, генеруються документи, які дозволяють в'їзд. Ці документи можуть бути в електронному або фізичному форматі (наприклад, QR-коди або RFID-мітки);

- облік в'їздів і виїздів: Система реєструє кожний в'їзд і виїзд автомобіля зі стоянки та створює документи, які фіксують ці операції;

- генерація звітів і розрахунків: ІС генерує різноманітні звіти, які містять інформацію про оплату, кількість користувачів, використання паркувальних місць та іншу статистику. Ці звіти є документами, які можуть використовуватися для аналізу та звітності;

- зберігання та архівування: Документи, пов'язані з діяльністю автостоянки, повинні бути збережені та архівовані відповідно до внутрішніх і законодавчих вимог;

- автоматизація і інтеграція: Важливою частиною ІС автостоянки є автоматизація документообігу та інтеграція з іншими підсистемами, такими як платіжні шлюзи, системи відеоспостереження та інші;

Опис документообігу для ІС платної автостоянки допомагає підтримувати порядок та дисципліну в процесі паркування автомобілів, а також забезпечує точність та надійність обліку операцій та фінансів автостоянки.

3.2 Проектування архітектури ПЗ

Архітектура програмного забезпечення, що описується наведеними SQL-командами, представляє собою систему управління паркуванням автомобілів. Ця система складається з семи взаємопов'язаних таблиць, які утворюють реляційну базу даних.

Таблиця car_owner зберігає інформацію про власників автомобілів. Вона містить такі дані, як ідентифікатор власника, прізвище, ім'я, по батькові, модель автомобіля, номер мобільного телефону та адресу (місто, вулиця, номер будинку та квартири). Ця таблиця є основною для зберігання персональних даних користувачів системи.

Таблиця parking_place описує окремі паркувальні місця. Кожне місце має свій унікальний ідентифікатор, номер, статус резервування та інформацію про необхідність ремонту. Ця таблиця пов'язана з таблицею parking через зовнішній ключ parking_id, що дозволяє визначити, до якої парковки належить кожне місце.

Таблиця cars містить інформацію про автомобілі. Вона включає такі дані, як реєстраційний номер, марка, модель, наявність пошкоджень та некомплектність. Ця таблиця має зв'язки з таблицями parking_place та car_owner, що дозволяє асоціювати кожен автомобіль з конкретним паркувальним місцем та власником.

Таблиця parking зберігає інформацію про парковки, включаючи їх адреси та кількість паркувальних місць. Вона є головною таблицею для parking_place, що дозволяє групувати місця за парковками.

Таблиця passes містить дані про пропуски на парковку. Кожен пропуск має тип, статус активності, дати початку та закінчення дії. Ця таблиця пов'язана з таблицею cars, що дозволяє асоціювати пропуски з конкретними автомобілями.

Таблиця payment зберігає інформацію про платежі. Вона містить дані про суму, дату та час оплати, тип платежу. Ця таблиця пов'язана з таблицею passes, що дозволяє відстежувати оплати за конкретні пропуски.

Таблиця car_place є проміжною таблицею, яка зв'язує пропуски (passes) з конкретними паркувальними місцями (parking_place). Це дозволяє реалізувати відношення багато-до-багатьох між пропусками та паркувальними місцями.

3.3 Проєктування структури зберігання даних

Розробка ER-діаграми для системи управління паркуванням автомобілів відбувалася шляхом детального аналізу вимог та бізнес-процесів, пов'язаних з паркуванням. Спочатку були визначені основні сутності системи: Власник, Автомобіль, Стоянка, Паркувальне місце, Абонементи та Оплата. Кожна з цих сутностей представляє ключовий аспект системи паркування.

Після визначення основних сутностей, були ідентифіковані їх атрибути. Для сутності "Власник" були включені такі атрибути як ідентифікатор, прізвище, ім'я, по батькові, контактний телефон, адреса (місто, вулиця, номер будинку, квартира). Сутність "Автомобіль" отримала атрибути: ідентифікатор, державний номер, марка, модель, наявність пошкоджень та некомплектність. Для "Стоянки" були визначені атрибути: ідентифікатор, адреса та кількість місць для паркування. "Паркувальне місце" включає ідентифікатор, номер місця, статус зайнятості та потребу в ремонті. "Абонементи" мають ідентифікатор, тип, статус дійсності, дату початку та закінчення. Нарешті, "Оплата" містить ідентифікатор, вартість, дату, час та спосіб оплати.

Наступним кроком було встановлення зв'язків між сутностями. Зв'язок між "Власником" та "Автомобілем" відображає відношення володіння. Зв'язок між "Автомобілем" та "Паркувальним місцем" показує, де припарковано автомобіль. "Стоянка" пов'язана з "Паркувальним місцем", що дозволяє групувати місця за

конкретними стоянками. "Абонементи" пов'язані з "Автомобілем", що дозволяє призначати абонементи конкретним транспортним засобам. Зв'язок між "Абонементами" та "Оплатою" відображає факт оплати за користування паркувальними послугами.

Важливим аспектом розробки діаграми було визначення кардинальності зв'язків. Наприклад, один власник може мати кілька автомобілів, але кожен автомобіль належить лише одному власнику. Одне паркувальне місце може бути зайняте лише одним автомобілем, але автомобіль може займати різні місця в різний час. Ці відношення були ретельно продумані та відображені на діаграмі.

Завершальним етапом було уточнення та оптимізація діаграми. Це включало перевірку повноти атрибутів, коректності зв'язків та загальної логіки системи. Діаграма була переглянута на предмет можливих покращень та спрощень, щоб забезпечити її ясність та ефективність у представленні структури даних системи управління паркуванням.

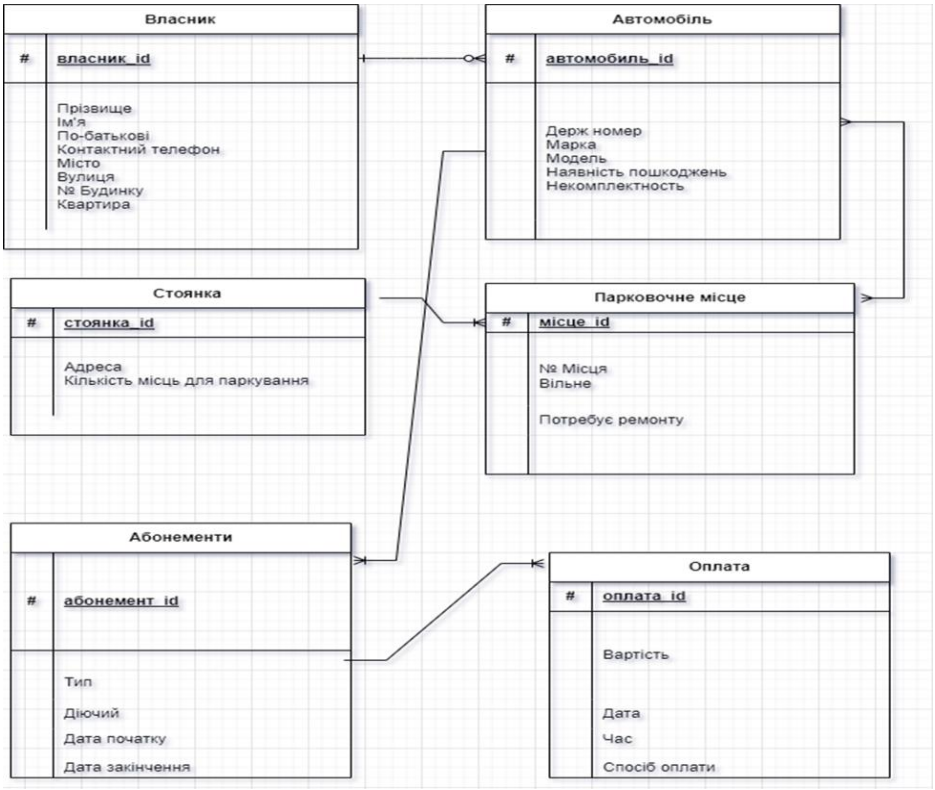


Рисунок 3.4 – ER-діаграма за нотацією Баркера

Реляційна модель представляє систему управління паркуванням автомобілів і складається з семи взаємопов'язаних таблиць:

1. car_owner: Містить інформацію про власників автомобілів, включаючи їх особисті дані та контактну інформацію.
2. cars: Зберігає дані про автомобілі, такі як реєстраційний номер, марка, модель, наявність пошкоджень чи некомплектність.
3. parking_place: Описує окремі паркувальні місця, їх номери, статус резервування та потребу в ремонті.
4. parking: Містить інформацію про парковки, їх адреси та кількість місць.
5. passes: Зберігає дані про пропуски, включаючи тип, активність, дати початку та закінчення дії.
6. payment: Містить інформацію про платежі, включаючи суму, дату, час та тип оплати.
7. car_place: Проміжна таблиця, що зв'язує пропуски з конкретними паркувальними місцями.

Зв'язки між таблицями реалізовані через зовнішні ключі:

- cars пов'язана з car_owner та parking_place;
- parking_place пов'язана з parking;
- passes пов'язана з cars;
- payment пов'язана з passes;
- car_place пов'язана з passes та parking_place.

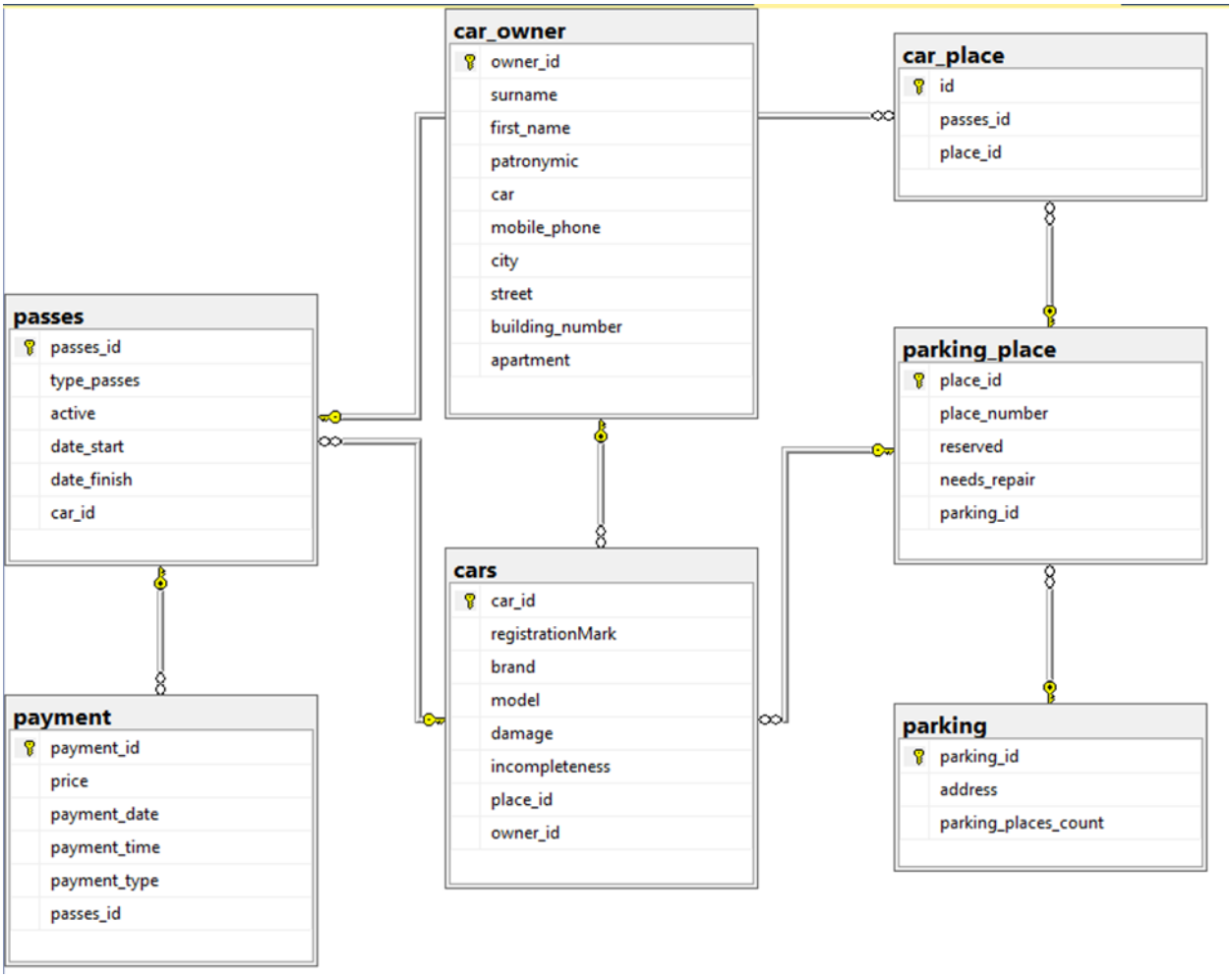


Рисунок 3.5 – Схема реляційної бази даних

Зв’язки у базі даних є цілісні і коректні для подальшого її кодування в системі.

4 ОПИС ПРИЙНЯТИХ ПРОГРАМНИХ РІШЕНЬ

C# (C-Sharp) є сучасною, об'єктно-орієнтованою та типобезпечною мовою програмування, розробленою компанією Microsoft. Створена на початку 2000-х років Андерсом Хейлсбергом та його командою, ця мова була частиною ініціативи .NET Framework. C# поєднує в собі простоту та ефективність C++ і потужність Java, надаючи розробникам всі необхідні інструменти для створення різноманітних додатків.[1] Важливими аспектами C# є підтримка об'єктно-орієнтованих концепцій, таких як класи, об'єкти, спадкування, інкапсуляція та поліморфізм. Мова забезпечує строгу перевірку типів під час компіляції, що допомагає уникнути помилок, пов'язаних з несумісністю типів. Завдяки вбудованому збирачу сміття, C# автоматично управляє виділенням і звільненням пам'яті, знижуючи ймовірність витоків пам'яті. Крім того, мова включає потужні засоби для роботи з даними через LINQ (Language Integrated Query), що дозволяє виконувати запити до різних джерел даних безпосередньо в коді. Хоча C# була розроблена Microsoft для Windows, сучасні технології, такі як .NET Core та Xamarin, дозволяють використовувати її для розробки додатків під різні платформи, включаючи macOS, Linux, iOS та Android. Мова забезпечує різні рівні безпеки, включаючи безпеку доступу до коду та безпечне виконання коду, і поставляється з обширною стандартною бібліотекою, яка включає широкий спектр функціональності від базових операцій з вводу-виводу до складних мережевих протоколів. C# використовується в широкому спектрі застосувань: від простих консольних програм до великих веб-додатків, від розробки ігор (особливо на платформі Unity) до розробки мобільних додатків через Xamarin.

Microsoft SQL Server є системою управління базами даних (СУБД), розробленою компанією Microsoft.[3] Як реляційна база даних, вона підтримує структурований запит мовою SQL (Structured Query Language) і широко використовується для зберігання та управління великим обсягом інформації.

Основні особливості та можливості Microsoft SQL Server включають роботу на основі реляційної моделі даних, що означає, що всі дані зберігаються у формі таблиць, які можуть бути зв'язані між собою за допомогою відношень. Система використовує стандартну мову запитів SQL для створення, оновлення, керування та маніпулювання даними. Вона гарантує надійність та цілісність даних шляхом підтримки транзакцій, дозволяючи групувати декілька операцій з даними в одну єдину одиницю роботи, яка або виконується повністю, або не виконується взагалі. SQL Server включає потужні механізми індексації та оптимізації запитів для покращення швидкості обробки даних, підтримує різні рівні безпеки, включаючи аутентифікацію, авторизацію, шифрування даних та аудит. Тісна інтеграція з іншими продуктами Microsoft, такими як Microsoft Office, SharePoint та Dynamics, забезпечує додаткові можливості для користувачів. Підтримка створення звітів (Reporting Services) та аналітики даних (Analysis Services) робить SQL Server потужним інструментом для бізнес-аналізу. Інструмент для інтеграції даних SQL Server Integration Services (SSIS) дозволяє виконувати завдання екстракції, перетворення та завантаження (ETL). Підтримка хмарних розробок дає змогу розгортати SQL Server у хмарі (наприклад, на Microsoft Azure), що забезпечує гнучкість та масштабованість.

ADO.NET (ActiveX Data Objects for .NET) є ключовою частиною платформи Microsoft .NET Framework, яка надає інтерфейс для доступу до даних і їх маніпуляції.[2] Це набір класів, що дозволяють взаємодіяти з джерелами даних, зазвичай реляційними базами даних, такими як Microsoft SQL Server, Oracle, MySQL та інші. ADO.NET може взаємодіяти з різними джерелами даних через .NET Data Providers, які включають спеціалізовані компоненти для доступу до SQL Server (SqlClient), Oracle, OLE DB та ODBC. Однією з ключових особливостей ADO.NET є підтримка "від'єднаної архітектури", що означає, що додаток може працювати з даними локально, а потім взаємодіяти з базою даних для оновлення. Це зменшує навантаження на мережу та сервер баз даних.

ADO.NET використовує об'єкти DataSet та DataTable для представлення даних, де DataSet може містити одну або кілька DataTable об'єктів, які можуть взаємодіяти між собою за допомогою відношень.[2] Інтерфейс дозволяє виконувати запити SQL та використовувати збережені процедури для роботи з даними, підтримує управління транзакціями, що дозволяє забезпечити цілісність даних під час виконання кількох операцій з базою даних. Основні компоненти, такі як SqlConnection, SqlCommand, SqlDataReader, SqlDataAdapter, використовуються для встановлення з'єднання з базою даних, виконання команд, читання даних та адаптації даних для DataSet. ADO.NET забезпечує безпечний і ефективний доступ до даних, включаючи шифрування даних і оптимізацію запитів, що робить його популярним вибором для багатьох розробників.

Опишемо декотрі вікна програмної системи. "MainForm" містить меню з пунктами File, Database, View, EditForm та Statistics. У верхній частині вікна розташовані поля пошуку за брендом та прізвищем, а також кнопки Search, Filter Apply та Filter Clear.[4] Нижче знаходиться таблиця "Cars" з колонками: place_id, car_id, registrationMark, brand, model, damage, incompleteness та owner_id (рис.4.1).

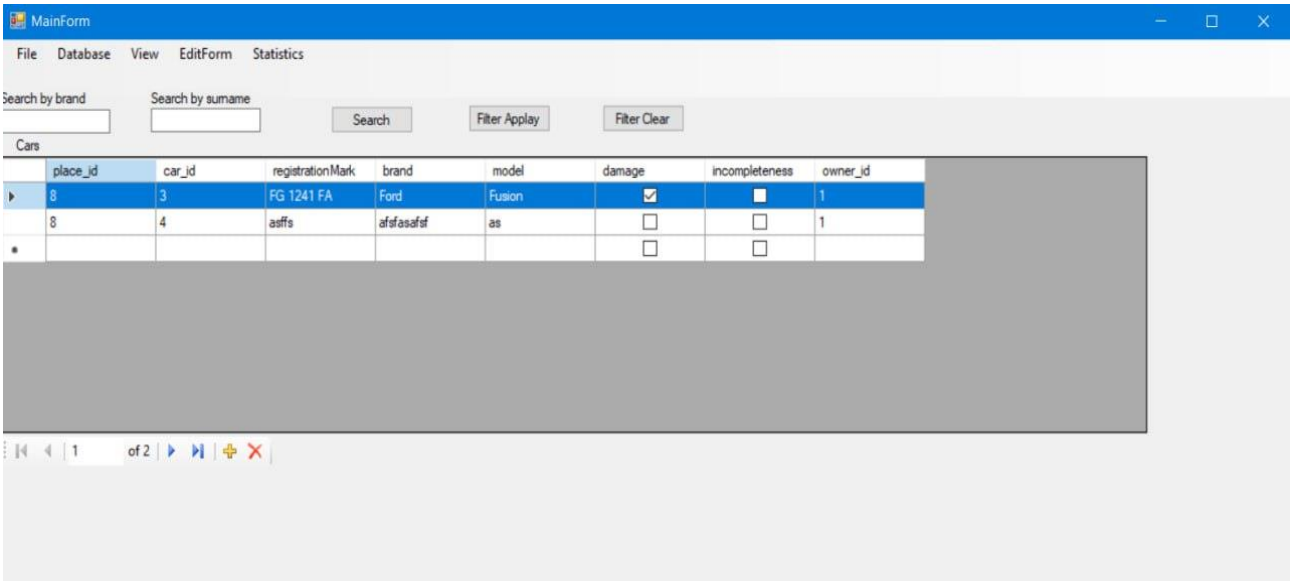


Рисунок 4.1 – Основне вікно та опції системи

Наступний екран також відображає головне вікно "MainForm", але з іншою таблицею – "Car owner". Колонки включають: owner_id, surname, first_name, patronymic, car, mobile_phone, city, street, building_number та apartment (рис.4.2).

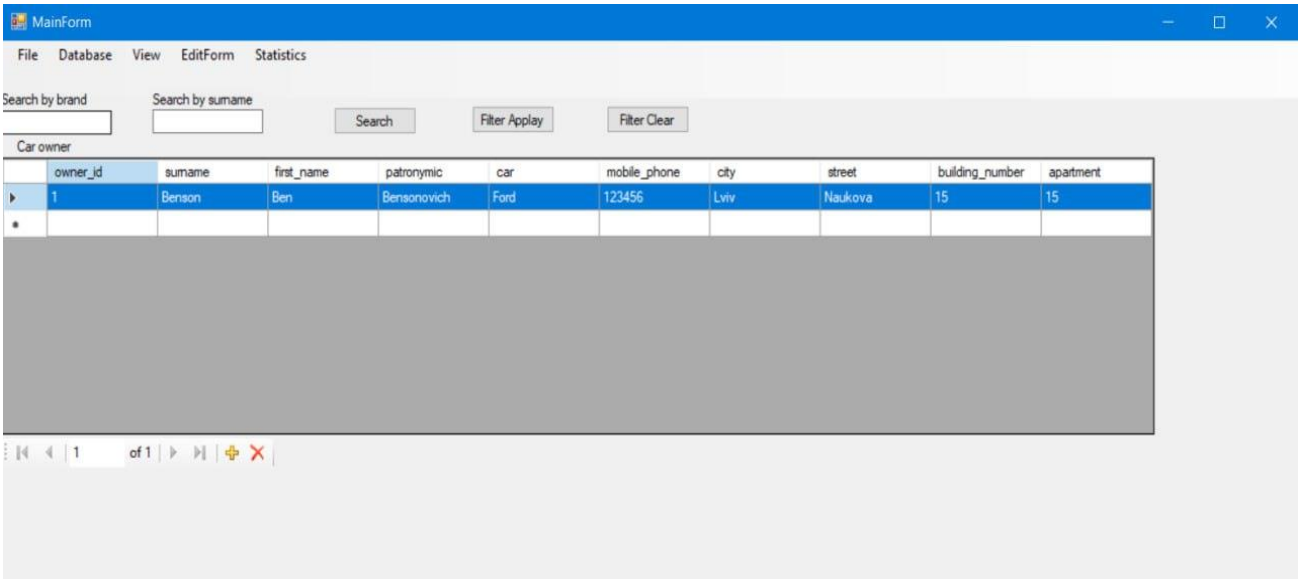


Рисунок 4.2 – Таблиця власників авто

Розглянемо вікно з розширеним набором даних у таблиці "Cars". Видно 7 записів про різні автомобілі. У правому верхньому куті з'явилася інформація про роль та ім'я користувача (Role: Admin, Name: Admin) (рис.4.3.).

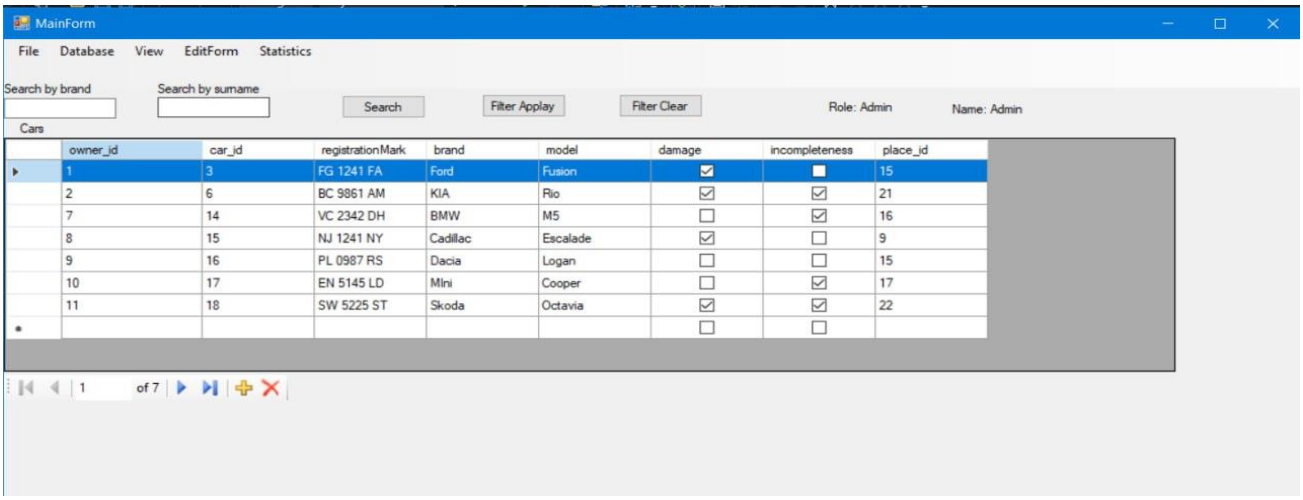


Рисунок 4.3 – Вікно з розширеними даними

Наступний екран показує головне вікно "MainForm" разом з додатковим вікном "EditForm". "EditForm" дозволяє редагувати дані про конкретний автомобіль, включаючи реєстраційний номер, бренд, модель, наявність пошкоджень, неповноту інформації, place_id та owner_id (рис.4.4).

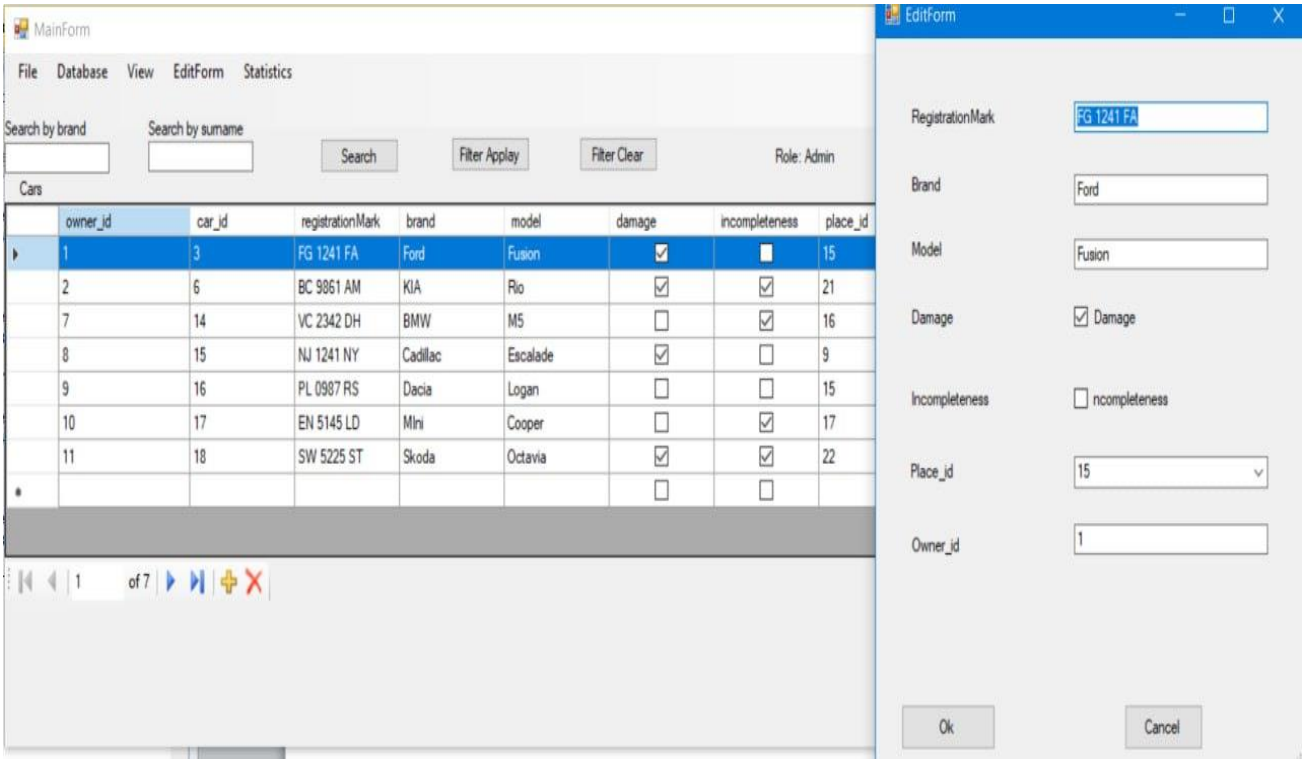


Рисунок 4.4 – Редагування даних

Наступне вікно демонструє результат пошуку в головному вікні. У поле "Search by brand" введено "Ford", і таблиця показує відфільтрований результат - один автомобіль марки Ford. Також видно додаткові колонки owner_id1 та surname, що вказує на об'єднання даних з таблиці власників (рис.4.5).

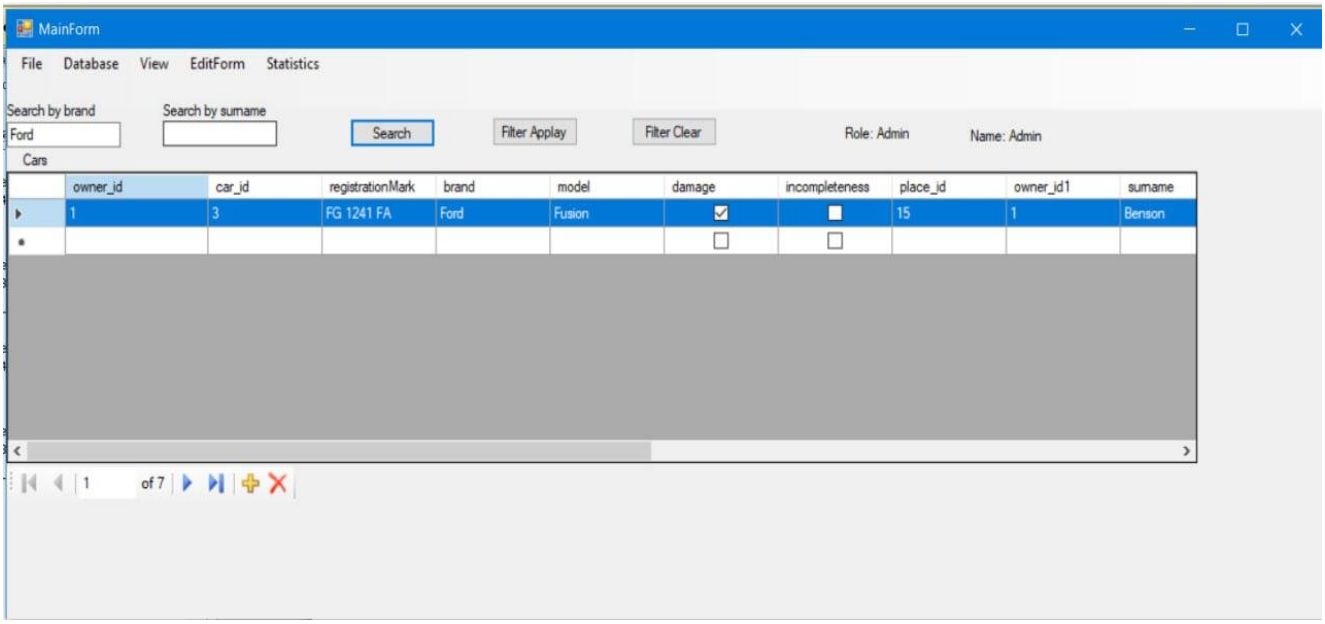


Рисунок 4.5 – Фільтрація даних

Статистичний модуль додатку має наступний вигляд (рис.4.6).

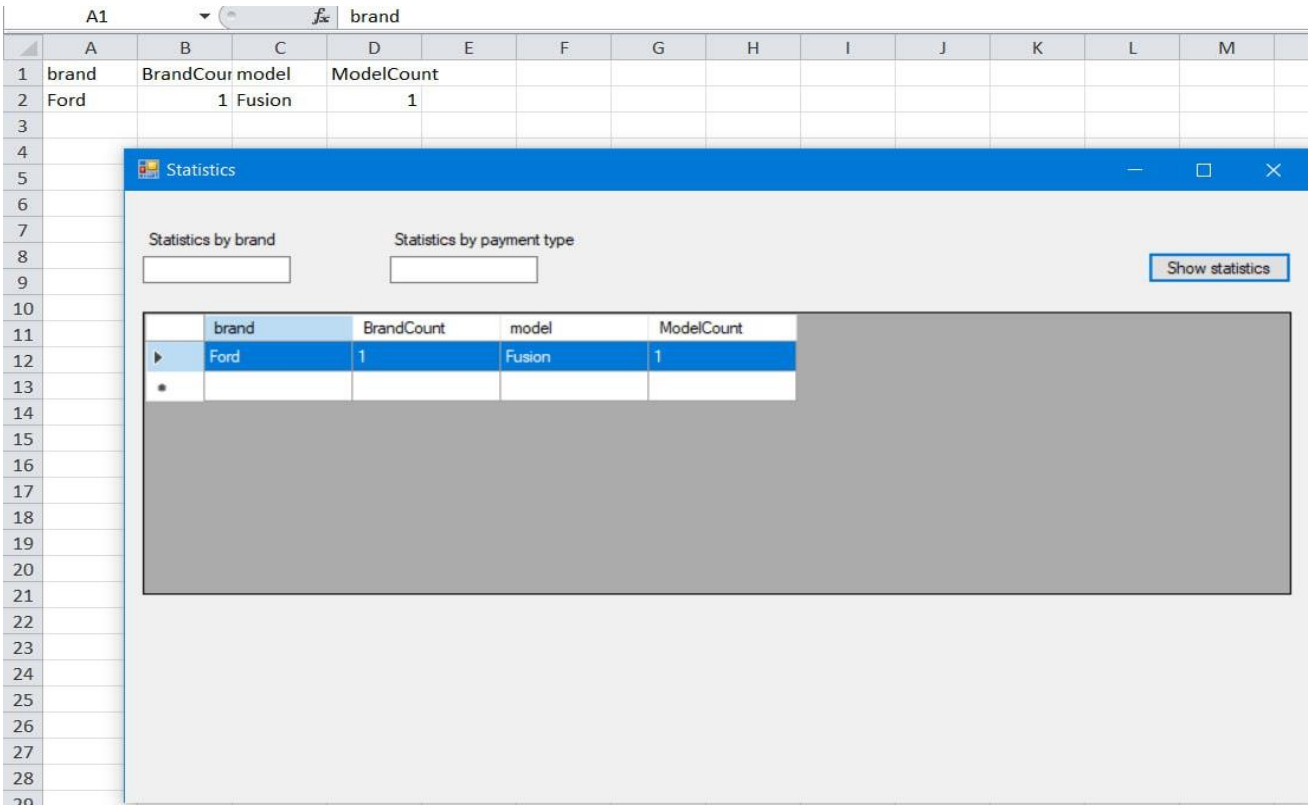


Рисунок 4.6 – Статистичний модуль додатку

5 ТЕСТУВАННЯ РОЗРОБЛЕНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Тестування розробленого ПЗ для управління автопарковкою є критичним етапом забезпечення якості, надійності та відповідності вимогам користувача. Основними компонентами даного ПЗ є використання мови програмування C#, MS SQL Server для управління базою даних, Windows Forms для графічного інтерфейсу користувача та технології доступу до даних ADO.NET. Тестування охоплює функціональне, інтеграційне, системне, продуктивне, регресійне та тестування безпеки.

Функціональне тестування спрямоване на перевірку основних функцій додатку. Це включає перевірку правильного завантаження даних з бази даних у таблиці cars, car_owner та parking_place при завантаженні форми, роботу кнопок для додавання, оновлення та видалення записів, пошуку та фільтрації даних за критеріями марки автомобіля та прізвища власника.

Інтеграційне тестування перевіряє взаємодію між різними модулями додатку. Це включає перевірку, чи коректно працює зв’язок між формою Windows Forms та базою даних через ADO.NET, чи правильно оновлюються дані в базі після виконання CRUD-операцій, та чи працюють взаємодії між різними формами додатку, такими як MainForm, RSForm, QueryEdit, EditForm та Statistics.[1]

Системне тестування охоплює перевірку всієї системи в цілому. Це включає перевірку роботи ПЗ на різних конфігураціях обладнання та операційних систем, взаємодію з іншими програмними продуктами, перевірку роботи мережевих з’єднань, якщо додаток працює в мережевому середовищі, та загальну стабільність системи при різних сценаріях використання.

Тестування продуктивності спрямоване на оцінку ефективності роботи додатку під різними навантаженнями. Це включає перевірку швидкості завантаження даних, часу відгуку при виконанні CRUD-операцій, ефективність

пошуку та фільтрації даних, та здатність додатку працювати з великою кількістю записів у базі даних.

Регресійне тестування перевіряє, чи не порушуються існуючі функції додатку після внесення змін або виправлення помилок. Це включає перевірку всіх основних функцій додатку після кожного оновлення або виправлення, щоб впевнитися, що нові зміни не призвели до появи нових помилок або порушень в роботі існуючих функцій.

Тестування безпеки охоплює перевірку захисту додатку від несанкціонованого доступу, забезпечення конфіденційності та цілісності даних. Це включає перевірку коректної роботи механізмів аутентифікації та авторизації, захисту даних під час їх передачі та зберігання, та вразливостей до атак типу SQL Injection.

Таблиця 5.1 – Основні тест-кейси

Тест-кейс	**Очікуваний результат**
Перевірка завантаження даних з бази даних	Дані з таблиць cars, car_owner та parking_place коректно завантажуються в DataGridView.
Додавання нового запису	Новий запис коректно додається до відповідної таблиці та відображається в DataGridView.
Оновлення існуючого запису	Існуючий запис коректно оновлюється в базі даних та відображається в DataGridView.
Видалення запису	Вибраний запис коректно видаляється з бази даних та зникає з DataGridView.
Пошук за маркою автомобіля та прізвищем власника	Дані коректно фільтруються та відображаються в DataGridView відповідно до введених критеріїв.

Сортування даних у DataGridView	Дані коректно сортуються у відповідності до вибраного стовпця та напряму сортування.
Робота з різними формами	Дані коректно передаються та оновлюються між формами MainForm, RSForm, QueryEdit, EditForm, Statistics.
Перевірка продуктивності при великій кількості записів	Додаток працює стабільно та швидко при роботі з великою кількістю записів у базі даних.
Перевірка захисту від SQL Injection	Додаток захищений від SQL Injection та інші вразливості до атак.
Перевірка коректної роботи після внесення змін	Існуючі функції продовжують працювати коректно після внесення змін або виправлення помилок.

Програмне забезпечення успішно пройшло всі види тестування, включаючи функціональне, інтеграційне, системне, продуктивне, регресійне та тестування безпеки. Було перевірено основні функції додатку, такі як завантаження даних з бази даних у таблиці cars, car_owner та parking_place, робота кнопок для додавання, оновлення та видалення записів, а також функціональність пошуку та фільтрації даних.

У результаті функціонального тестування було підтверджено, що всі основні функції додатку працюють відповідно до специфікацій. Дані коректно завантажуються з бази даних і відображаються в DataGridView, нові записи успішно додаються, існуючі записи коректно оновлюються, а непотрібні записи успішно видаляються з бази даних. Пошук та фільтрація даних за критеріями марки автомобіля та прізвища власника працюють належним чином.

Інтеграційне тестування підтвердило, що всі модулі додатку взаємодіють коректно. Дані передаються між формами MainForm, RSForm, QueryEdit,

EditForm та Statistics без помилок, а всі CRUD-операції виконуються належним чином.

Системне тестування показало, що додаток працює стабільно на різних конфігураціях обладнання та операційних систем, а також успішно взаємодіє з іншими програмними продуктами. Тестування продуктивності показало, що додаток ефективно працює під різними навантаженнями, швидко завантажує дані та має прийнятний час відгуку при виконанні CRUD-операцій, пошуку та фільтрації даних.

Регресійне тестування підтвердило, що внесення змін або виправлення помилок не призвело до порушення роботи існуючих функцій додатку. Всі основні функції продовжують працювати коректно після кожного оновлення або виправлення.

Тестування безпеки показало, що додаток захищений від несанкціонованого доступу, забезпечує конфіденційність та цілісність даних, а також є стійким до атак типу SQL Injection та інших вразливостей.

6 ВПРОВАДЖЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Впровадження програмного забезпечення для управління автопарковкою розпочинається з ретельного планування. Спочатку визначаються цілі та завдання впровадження, складається детальний план дій, визначаються відповідальні особи та ресурси, а також оцінюються потенційні ризики та розробляються стратегії їх мінімізації.

Підготовка інфраструктури є важливим кроком, який включає налаштування серверного обладнання, встановлення необхідних програмних компонентів, таких як MS SQL Server, та налаштування мережевих параметрів. Це забезпечить стабільну роботу додатку та надійне зберігання даних.

Наступним етапом є встановлення програмного забезпечення. Встановлюється сама програма, а також налаштовуються всі необхідні компоненти, включаючи бази даних та інтерфейси користувача. Після цього здійснюється первинне завантаження даних, яке включає імпорт існуючих даних про автомобілі, власників та місця паркування в систему. Це дозволяє одразу почати роботу з актуальною інформацією.

Далі проводиться навчання користувачів. Персонал, який буде працювати з новим програмним забезпеченням, проходить навчання для ознайомлення з його функціями та можливостями. Це може включати як індивідуальні заняття, так і групові тренінги, а також створення навчальних матеріалів та інструкцій.

Після навчання користувачів проводиться тестування у робочому середовищі. Це дозволяє переконатися, що програма працює належним чином в реальних умовах, та виявити можливі проблеми чи недоліки. Тестування включає перевірку всіх основних функцій, таких як додавання, редагування та видалення записів, а також пошук та фільтрація даних.

Після успішного тестування у робочому середовищі здійснюється безпосереднє впровадження програмного забезпечення. Програма запускається у

штатному режимі, і користувачі починають використовувати її у своїй повсякденній роботі. У цей період важливо забезпечити підтримку користувачів, оперативно реагувати на їхні запитання та вирішувати можливі проблеми.

Під час впровадження також проводиться моніторинг роботи системи для оцінки її ефективності та виявлення можливих проблем. Це включає аналіз продуктивності, стабільності роботи та задоволеності користувачів.

Завершальним етапом впровадження є постійна підтримка та оновлення програмного забезпечення. Забезпечується регулярне оновлення системи, включаючи виправлення помилок, покращення функціональності та безпеки, а також навчання нових користувачів та забезпечення підтримки для існуючих.

ВИСНОВКИ

Програмна система керування автостоянкою є критично важливим інструментом у сучасному урбанізованому середовищі. Її розробка та впровадження спрямовані на вирішення проблем, пов'язаних із зростаючою кількістю автомобілів та обмеженістю паркувальних площ. Традиційні методи управління, що базуються на паперовому документообігу та ручному обліку, вже не відповідають вимогам часу, оскільки вони є трудомісткими, схильними до помилок та неефективними. Впровадження автоматизованої системи дозволяє вирішити проблеми неефективного використання паркувальних місць, складності з обліком та контролем оплати, а також забезпечення безпеки транспортних засобів.

Розробка програмної системи керування автостоянкою має на меті створення комплексного рішення, яке автоматизує та оптимізує всі аспекти управління паркувальним простором. Це включає в себе заміну паперових журналів обліку на електронну систему, яка забезпечує точний та оперативний облік всіх операцій. Система підвищує ефективність використання паркувальних місць, спрощує процес оплати та контролю доступу, а також надає актуальну аналітичну інформацію для прийняття обґрунтованих рішень.

Система включає створення бази даних для зберігання інформації про паркувальні місця, автомобілі, їх власників та абонементи. Вона забезпечує можливість швидкого пошуку та фільтрації даних, що дозволяє операторам ефективно управляти великими обсягами інформації. Важливим завданням є реалізація функціоналу для автоматичного формування різноманітних звітів, включаючи статистику доходів за різні періоди, аналіз завантаженості стоянки та облік транспортних засобів за різними параметрами.

Окремим важливим завданням є розробка механізму автоматичного перепланування паркувальних місць у випадку проведення ремонтних робіт чи

інших змін у конфігурації стоянки. Це дозволяє мінімізувати незручності для клієнтів та оптимізувати використання доступного простору навіть в умовах технічних обмежень.

Реалізація цієї системи не тільки підвищує ефективність управління автостоянкою, але й сприяє покращенню екологічної ситуації в містах шляхом оптимізації використання паркувального простору та зменшення часу, який водії витрачають на пошук місця для паркування. Крім того, вона створює основу для подальшої інтеграції з іншими міськими системами, такими як громадський транспорт чи системи управління дорожнім рухом, що в перспективі дозволяє реалізувати концепцію розумного міста в сфері управління транспортною інфраструктурою.

Розробка програмної системи керування автостоянкою є важливим кроком у напрямку створення більш ефективної, зручної та екологічно дружньої міської інфраструктури, що відповідає сучасним викликам урбанізації та зростаючим потребам населення у якісних послугах паркування.

Система була розроблена з урахуванням сучасних стандартів програмування та безпеки. Вона складається з кількох взаємопов'язаних модулів, які забезпечують повний цикл обслуговування автостоянки: від реєстрації автомобіля до здійснення оплати та формування звітності. Особливу увагу було приділено забезпеченню зручності та інтуїтивності інтерфейсу користувача, що дозволяє мінімізувати час на навчання та впровадження системи в експлуатацію.

Програмне забезпечення було успішно протестоване на відповідність вимогам функціональності, надійності та безпеки. В процесі тестування було виявлено та усунуто всі критичні помилки, що гарантує стабільну роботу системи в умовах реальної експлуатації. Система була інтегрована з існуючими інфраструктурними рішеннями та протестована в умовах високого навантаження, що підтвердило її здатність ефективно працювати в реальних умовах.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Офіційна документація Microsoft: ADO.NET. URL: https://learn.microsoft.com/ru-ru/dotnet/framework/data/adonet/ (дата звернення: 10.12.2023).

2. Pro C# 7: With .NET and .NET Core Авторы: Andrew Troelsen, Philip Japikse URL: https://books.google.com.ua/books?id=Jus_DwAAQBAJ&printsec=frontcover&redir_esc=y#v=onepage&q&f=false (дата звернення: 14.12.2023).

3. SQL Server technical documentation URL: https://learn.microsoft.com/en-us/sql/sql-server/?view=sql-server-ver16 (дата звернення: 15.12.2023).

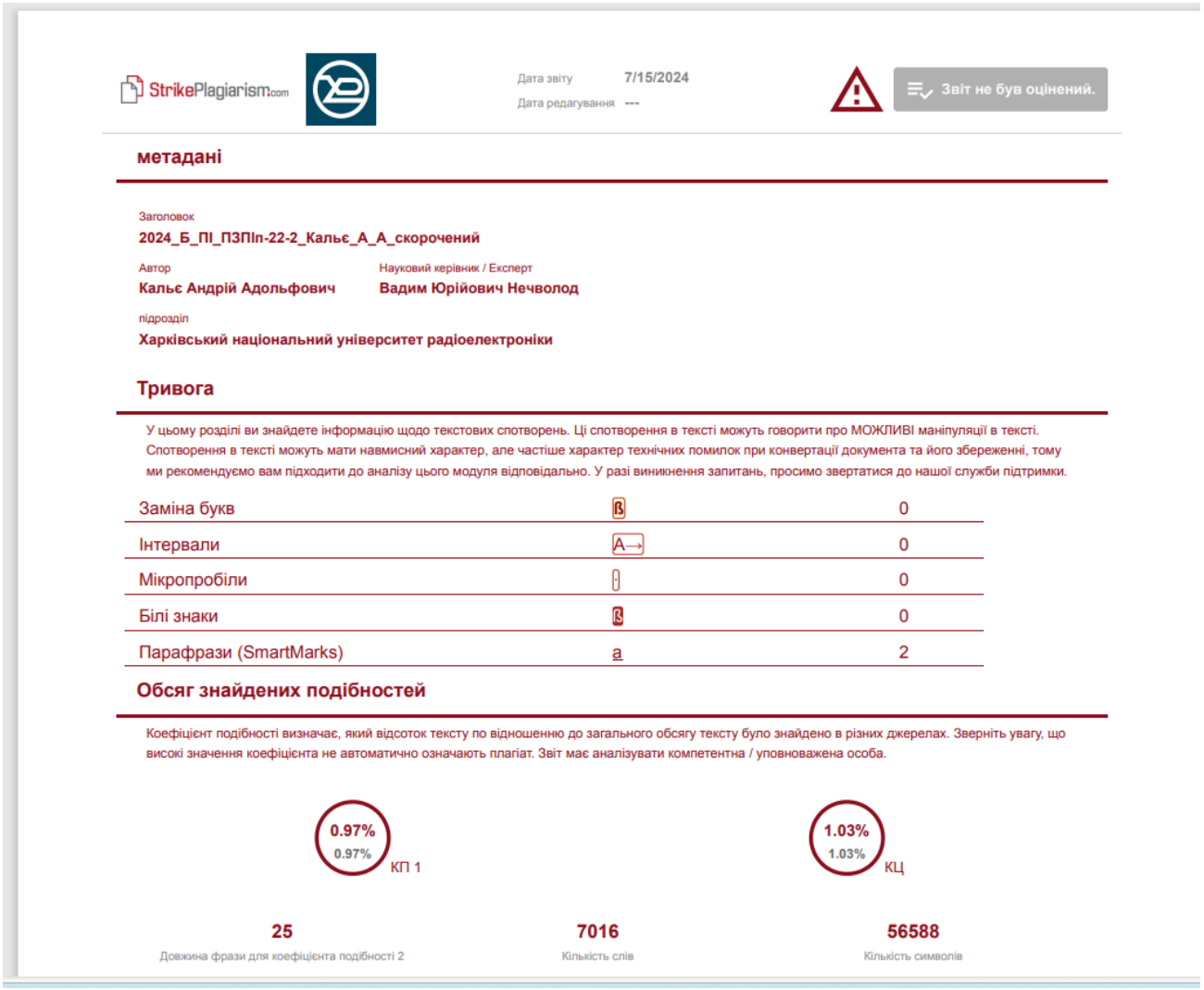
4. Desktop Guide (Windows Forms .NET) URL: https://learn.microsoft.com/enus/dotnet/desktop/winforms/overview/?view=netdesktop-8.0 (дата звернення: 10.12.2023).

5. UML Diagrams in Software Engineering Research URL: https://www.mdpi.com/2504-3900/74/1/13. (дата звернення: 14.12.2023).

6. Методичні вказівки до курсового проектування з дисципліни «Бази даних» для студентів усіх форм навчання спеціальності 121 – «Інженерія програмного забезпечення» (освітньо-професійна програма «Програмна інженерія») / Упоряд. О.О. Мазурова, М.С. Широкопетлєва, Ю.Ю.Черепанова. Харків: ХНУРЕ, 2022. 61 с.

ДОДАТОК А

Звіт результатів перевірки на унікальність тексту в базі ХНУРЕ



StrikePlagiarism.com

Дата звіту 7/15/2024
Дата редагування ---

Звіт не був оцінений.

метадані

Заголовок
2024_Б_ПІ_ПЗПІп-22-2_Кальє_А_А_скорочений

Автор
Кальє Андрій Адольфович

Науковий керівник / Експерт
Вадим Юрійович Нечволод

підрозділ
Харківський національний університет радіоелектроніки

Тривога

У цьому розділі ви знайдете інформацію щодо текстових спотворень. Ці спотворення в тексті можуть говорити про МОЖЛИВІ маніпуляції в тексті. Спотворення в тексті можуть мати навмисний характер, але частіше характер технічних помилок при конвертації документа та його збереженні, тому ми рекомендуємо вам підходити до аналізу цього модуля відповідально. У разі виникнення запитань, просимо звертатися до нашої служби підтримки.

Заміна букв	ß	0
Інтервали	A→	0
Мікропробіли		0
Білі знаки	ß	0
Парафрази (SmartMarks)	a	2

Обсяг знайдених подібностей

Коефіцієнт подібності визначає, який відсоток тексту по відношенню до загального обсягу тексту було знайдено в різних джерелах. Зверніть увагу, що високі значення коефіцієнта не автоматично означають плагіат. Звіт має аналізувати компетентна / уповноважена особа.

0.97% 0.97% КП 1

1.03% 1.03% КЦ

25
Довжина фрази для коефіцієнта подібності 2

7016
Кількість слів

56588
Кількість символів

ДОДАТОК Б

Слайди презентації

Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Кваліфікаційна робота бакалавра

Програмна система керування автостоянкою

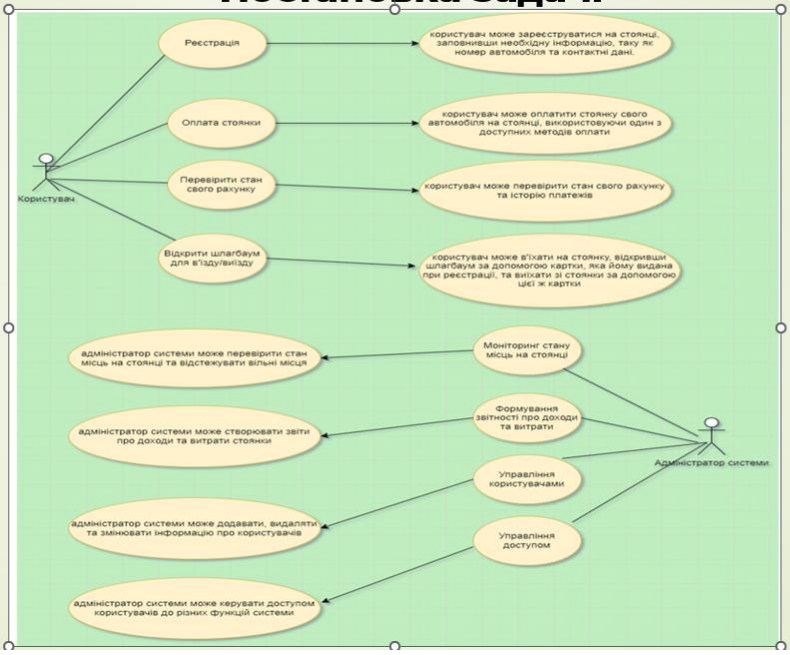
1

Виконав:
студент гр. ПЗПІп-22-2
Кальє А.А.

Науковий керівник:
Вечур О.В.

2

Постановка задачі



3

Аналіз предметної галузі

Ключовим компонентом є управління паркувальними місцями, що передбачає постійний моніторинг їх статусу та оптимізацію використання доступного простору. Це тісно пов'язано з системою контролю доступу, яка регулює в'їзд та виїзд транспортних засобів, використовуючи різноманітні технологічні рішення, такі як автоматичні шлагбауми, зчитувачі номерних знаків та смарт-картки.

Важливою складовою є облік та ідентифікація транспортних засобів, що паркуються. Система повинна фіксувати та зберігати інформацію про кожен автомобіль, включаючи час прибуття та відбуття, що є критичним для правильного нарахування оплати. Це безпосередньо пов'язано з фінансовим аспектом управління автостоянкою, який включає в себе гнучку тарифікацію, обробку різних видів платежів та генерацію фінансової звітності.

4

Приклад аналогу-конкуренту розробки

5

Фрагмент таблиці функціональних вимог

ID	Функціональна вимога	Опис	Пріоритет	Складність
1.1	Створення записів про паркувальні місця	Можливість додавати нові паркувальні місця з вказанням номера, статусу, типу	Високий	Низька
1.2	Редагування записів про паркувальні місця	Зміна інформації про існуючі паркувальні місця	Середній	Низька
1.3	Видалення записів про паркувальні місця	Видалення неактуальних паркувальних місць з бази даних	Низький	Низька
2.1	Створення записів про автомобілі	Додавання нових автомобілів з вказанням марки, моделі, номера держ. реєстрації	Високий	Низька
2.2	Редагування записів про автомобілі	Оновлення інформації про існуючі автомобілі	Середній	Низька
2.3	Видалення записів про автомобілі	Видалення автомобілів з бази даних при виїзді зі стоянки	Високий	Середня
3.1	Створення записів про власників	Додавання нових власників з їх контактною інформацією	Високий	Низька
3.2	Редагування записів про власників	Оновлення контактної інформації власників	Середній	Низька
3.3	Видалення записів про власників	Видалення власників з перевіркою відсутності активних абонементів	Низький	Середня

6

Використані технології

C# (C-Sharp) є сучасною, об'єктно-орієнтованою та типобезпечною мовою програмування, розробленою компанією Microsoft.

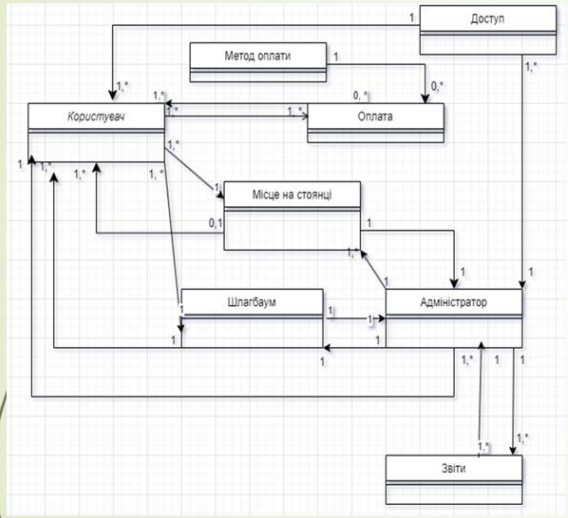
Microsoft SQL Server є системою управління базами даних (СУБД), розробленою компанією Microsoft.

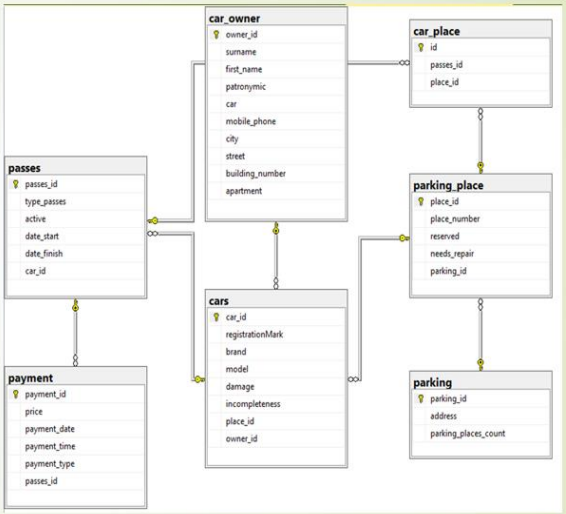
ADO.NET (ActiveX Data Objects for .NET) є ключовою частиною платформи Microsoft .NET Framework, яка надає інтерфейс для доступу до даних і їх маніпуляції.

Windows Forms для разробки графічного інтерфейсу користувача.

7

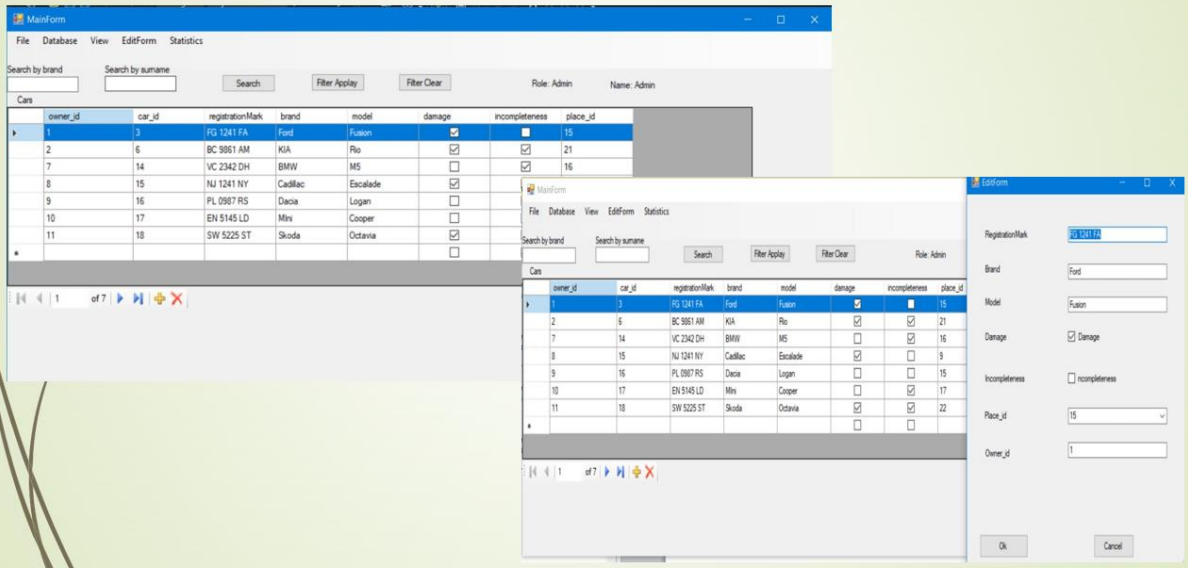
Діаграма класів та схема реляційної БД





8

Інтерфейс користувача



Декотрі базові тест кейси

Тест-кейс	Очікуваний результат
Перевірка завантаження даних з бази даних	Дані з таблиць cars, car_owner та parking_place коректно завантажуються в DataGridView.
Додавання нового запису	Новий запис коректно додається до відповідної таблиці та відображається в DataGridView.
Оновлення існуючого запису	Існуючий запис коректно оновлюється в базі даних та відображається в DataGridView.
Видалення запису	Вибраний запис коректно видаляється з бази даних та зникає з DataGridView.
Пошук за маркою автомобіля та прізвищем власника	Дані коректно фільтруються та відображаються в DataGridView відповідно до введених критеріїв.
Сортування даних у DataGridView	Дані коректно сортуються у відповідності до вибраного стовпця та напряму сортування.
Робота з різними формами	Дані коректно передаються та оновлюються між формами MainForm, RSForm, QueryEdit, EditForm, Statistics.
Перевірка продуктивності при великій кількості записів	Додаток працює стабільно та швидко при роботі з великою кількістю записів у базі даних.
Перевірка захисту від SQL Injection	Додаток захищений від SQL Injection та інші вразливості до атак.
Перевірка коректної роботи після внесення змін	Існуючі функції продовжують працювати коректно після внесення змін або виправлення помилок.

10

Висновки

Було розроблено програмну систему з урахуванням сучасних стандартів програмування та безпеки. Вона складається з кількох взаємопов'язаних модулів, які забезпечують повний цикл обслуговування автостоянки: від реєстрації автомобіля до здійснення оплати та формування звітності. Особливу увагу було приділено забезпеченню зручності та інтуїтивності інтерфейсу користувача, що дозволяє мінімізувати час на навчання та впровадження системи в експлуатацію.

Програмне забезпечення було успішно протестоване на відповідність вимогам функціональності, надійності та безпеки. В процесі тестування було виявлено та усунуто всі критичні помилки, що гарантує стабільну роботу системи в умовах реальної експлуатації. Система була інтегрована з існуючими інфраструктурними рішеннями та протестована в умовах високого навантаження, що підтвердило її здатність ефективно працювати в реальних умовах.