Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя
(повне найменування вищого навчального закладу)
Факультет комп'ютерно-інформаційних систем і програмної інженерії
(назва факультету)
Кафедра комп'ютерних наук
(повна назва кафедри)

ПОЯСНЮВАЛЬНА ЗАПИСКА

до дипломного проекту (роботи)

Магістр
(освітній ступінь)

на тему: Сервіс для адміністрування і обліку роботи автомобільної парковки

Виконав: студент 6 курсу, групи СНм-61
спеціальності
124 «Системний аналіз»
(шифр і назва спеціальності (напряму підготовки))

(підпис) Надозірний В.Б.
(прізвище та ініціали)

Керівник (підпис) доц.Литвиненко Я.В.
(прізвище та ініціали)

Нормоконтроль (підпис) доц. Мацюк О.В.
(прізвище та ініціали)

Рецензент (підпис) проф. Лупенко С.А.
(прізвище та ініціали)

м. Тернопіль – 2019

АНОТАЦІЯ

Сервіс для адміністрування і обліку роботи автомобільної парковки // Дипломна робота освітнього рівня «Магістр» // Надозірний Володимир Богданович // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем та програмної інженерії, кафедра комп'ютерних наук, група САм–61 // Тернопіль, 2019 // С. – , рис. – *29* , табл.– *20*, слайдів – *16*, додат. – *2*, бібліогр. – *15*.

Ключові слова: XML-ФАЙЛ, АВТОМОБІЛЬ, БАЗА ДАНИХ, ПАРКОВКА, СЦЕНАРІЙ ВИКОРИСТАННЯ

Основними завданнями дипломної роботи є розробка сервісу для адміністрування і обліку роботи автомобільної парковки та його програмна реалізація.

У першому розділі дипломної роботи проведено аналіз предметної області дослідження, наведено основні характеристики, види автомобільного паркінгу та класифікація парковок. Проведено аналіз окремих представлених на ринку програмних продуктів для автомобільного паркінгу. Порівняно їх основні функціональні можливості.

У другому розділі дипломної роботи сформульовано основні функціональні можливості і завдання сервісу, проаналізовано основні бізнес-процеси паркінгу автомобілів (зокрема початок роботи в генераторі схем парко-місць, формування місць для парковки, закінчення роботи в генераторі схем). Також розроблені вимоги до сервісу: функціональні (завдання для розробника) та не функціональні (атрибути – показники якості).

У третього розділі дипломної роботи наведено процес створення проекту у MS Visual Studio. Також показано програмна реалізація частин сервісу для створення схеми паркінгу та для ведення обліку паркінгових місць. Наведено окремі лістинги для різних прогрманих методів.

У четвертому розділі дипломної роботи наведено основні можливості програмного забезпечення, яке використовується для створення сервісу, зокрема

мови програмування C#, мови розмітки документів XML та системи керування базами даних MySQL.

П’ятий розділ дипломної роботи присвячений обґрунтування економічної ефективності проведеного дослідження.

У шостому розділі висвітлені важливі питання охорони праці та безпеки в надзвичайних ситуаціях.

Сьомий розділ описує екологічні питання, які виникли при розробці.

Мета дослідження: розробка програмного продукту, який призначений для управління роботою автомобільної парковки.

Об’єкт дослідження: процес управління роботою автомобільної парковки.

Предмет дослідження: сценарії впровадження процесу адміністрування та обліку автомобільної парковки.

ANNOTATION

A service for car parking administration and operating records // Nadozirnii Volodymyr // Ternopil Ivan Pul'uj National Technical University, Faculty of Computer Information Systems and Software Engineering, Department of Computer Science // Ternopil, 2019 // P. - , Fig. - *29*, Table – *20*, Slide - *16*, References - *15* .

Keywords: XML-FILE, CAR, DATABASE, PARKING, USE CASE

Thesis deals with the development of a service for the administration and accounting of the operation of car parking and its software implementation.

In the first section of the thesis analyzes the subject area of the study, describes the main characteristics, types of car parking and classification of parking lots. The analysis of the individual software for car parking is presented. Their main functionality is compared.

The second section of the thesis the basic functionality and tasks of the service are formulated, analyzes the basic business processes of car parking (in particular, the start of work in the generator of parking spaces, formation of parking spaces, the end of work in the scheme generator). Service requirements have also been developed: functional (developer tasks) and non-functional (attributes - quality metrics).

The third section of the thesis describes the process of creating a project in MS Visual Studio. Also shown is the software implementation of parts of the service to create a parking scheme and to maintain parking spaces. Individual listings for different methods are presented.

The fourth section of the thesis describes the main features of the software used to create the service, including C # programming language, XML markup language and MySQL database management system.

Background includes the following tasks:

- the problem of car parking with possible ways of its solution is investigated;
- software for automated parking are available on the market is considered ;
- the main business processes of car parking are analyzed;
- functional and non-functional requirements for development are formulated;

– the basic functionality and tasks of the service are formulated;

– functionality has been developed to create a customer base where it will be possible to obtain a parking card, availability of seats.

Actuality of the theme. In recent years, with the increase in the number of motor vehicles, the time to find a free parking space and the time to park has increased. The problem of organizing places for permanent storage of cars and temporary accommodation (parking) in places of mass visit, especially in the central parts of the largest cities, has considerably aggravated. In large cities, motorists often have the problem of parking vehicles. Finding a parking space can take time, expensive for the driver. In Germany, for example, finding an appropriate parking space takes about 10 minutes. According to a study conducted by a European parking management company, drivers must travel up to 4.5 km in search of a place to leave a car. This, in turn, causes not only a waste of time but also money and fuel. In the parking lot, the car is placed for a relatively short period of time when its owner or passenger is at work, in a shop, in a cultural institution and in other similar places. Therefore, it is necessary to have a clear management of the process of parking and the withdrawal of motor vehicles from the parking lot, which will reduce the time spent on parking and minimize the likely risks of road accidents during parking.

Thus, the urgent problem is the creation of software for servicing temporary parking lots or sites that are not equipped with appropriate peripheral equipment and special systems.

The purpose of the research is development of a software product that is used for quality management of the car parking.

Object of research – the process of management of the car parking.

Subject of research – use-cases of implementation of the process of administration and accounting for car parking.

Implementation of the developed software service will allow to provide administration and accounting of temporary parking facilities not equipped with special intelligent systems.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ СКОРОЧЕНЬ І ТЕРМІНІВ

БД – база даних;

ЕОМ – електронно-обчислювальна машина;

ІЗВП – інструментальні засоби візуального програмування;

ПК – персональний комп’ютер;

СУБД – система управління базами даних;

.NET– програмна технологія, запропонована фірмою Microsoft,як платформа для створення як звичайних програм, так і веб-застосунків;

BCL (BaseClassLibrary) – стандартна бібліотека класів платформи «.NET Framework»;

CLR (Common Language Runtime) – середовище, яке підходить для різних мов програмування;

CRUD (Create Read Update Delete) – 4 базові функції управління даними в БД «створення, зчитування, зміна і видалення».

SQL (Structured query language) – декларативна мова програмування для взаємодії користувача з базами даних, що застосовується для формування запитів, оновлення і керування реляційними базами даних;

VB.NET (VisualBasic .NET) – найсучасніша BASIC-базована мова програмування;

WPF (Windows Presentation Foundation) – графічна (презентаційна) підсистема в складі .NET Framework 3.0, що має пряме відношення до XAML;

XAML (eXtensible Application Markup Language) – це декларативна мова, яка дозволяє створювати й ініціалізувати .NET об’єкти в XML;

XML (eXtensible Markup Language) – запропонований консорціумом WorldWideWeb (W3C) стандарт побудови мов розмітки ієрархічно структурованих даних для обміну між різними застосунками, зокрема, через Інтернет.

ЗМІСТ

ВСТУП

Актуальність теми. В останні роки із збільшенням кількості автомобільного транспорту збільшився час на пошук вільного парко місця та час на саме паркування. Значно загострилася проблема організації місць постійного зберігання автомобілів і тимчасового розміщення (парковки) у місцях масового відвідування, перш за все в центральних частинах найбільших міст. Пошук паркувального місця може зайняти дорогоцінний, для водія, час. Для прикладу, у центральній Європі пошук відповідного паркувального місця, в середньому, займає близько 10 хвилин. За даними дослідження, проведеного європейською компанією з управління парковкою, водії повинні їхати до 4,5 кілометра в пошуках місця, де можна залишити автомобіль. Це, відповідно, спричиняє не тільки втрату часу, а й грошей та пального На парковку автомобіль поміщається на відносно нетривалий час перебування його власника або пасажира на роботі, в магазині, в культурно-масовій установі і в інших подібних місцях. Тому необхідно забезпечити чітке керування процесом поміщення на парковку та виїзду автомобільного транспорту з парковки, що дозволить зменшити часові затрати на паркування та мінімізувати ймовірні ризики від дорожньо-транспортних пригод під час паркування.

Таким чином, актуальною є проблема створення програмного засобу обслуговування тимчасових парковок або площадок, які не обладнані відповідним периферійним устаткуванням і спеціальними системами.

Зв’язок із науковими програмами, планами, темами. Дипломна робота виконана відповідно до наукової тематики Тернопільського національного технічного університету імені Івана Пулюя, кафедри комп’ютерних наук.

Мета і задачі дослідження. Мета роботи полягає у розроці програмного продукту, який призначений для управління роботою автомобільної парковки.

Для досягнення вказаної мети, в роботі поставлено та розв’язано наступні задачі:

– досліджено проблему автомобільного паркінгу;

– проаналізовано окремі доступні на ринку програмні засоби для забезпечення автоматизованого паркування;

– виконано аналіз основних бізнес-процесів паркінгу автомобілів;

– сформульовані функціональні та нефункціональні вимоги до розробки;

– сформульовано основні функціональні можливості і завдання сервісу;

– розроблено функціонал для створення бази клієнтів, де буде можливість отримання карти парковки, наявність вільних місць.

Об’єкт дослідження: процес управління роботою автомобільної парковки.

Предмет дослідження: сценарії впровадження процесу адміністрування та обліку автомобільної парковки.

Методи дослідження. Метод теоретичного дослідження та експериментальний з використання персонального комп’ютера. Методика дослідження базується на теоретичних і прикладних результатах, досягнутих у комп’ютерних науках.

Наукова новизна отриманих результатів. Наукова новизна полягає у вирішенні науково-практичної задачі створення сервісу для адміністрування і обліку роботи автомобільної парковки, при цьому одержано наступні результати:

– розроблено вимоги до сервісу: функціональні (завдання для розробника) та нефункціональні (атрибути – показники якості);

– створено генератор схем паркувальної розмітки, який формує XML-документ з схемою парковки;

– сформульовано основні системні вимоги;

– розроблено програмний засіб.

Практичне значення одержаних результатів. Впровадження розробленого сервісу дозволить забезпечити адміністрування і обліку роботи тимчасових парковок, які не обладнані спеціальними інтелектуальними системами.

Публікації. Результати дослідження апробовано на VII науково-технічній конференції «Інформаційні моделі, системи та технології» Тернопільського національного технічного університету імені Івана Пулюя (11-12 грудня 2019р.) у вигляді опублікованих тез.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Коротка історія та характеристика автомобільного паркінгу

Починаючи з 1913 року почалась повномасштабна конвеєрна збірка автомобілів Ford-T. До 27 березня 1927 року, коли з конвеєра зійшов останній Ford-T, було випущено більше п’ятнадцяти мільйонів екземплярів цих машин. Процес машинобудування йшов повним ходом, і, тому на вулицях міст США і Європи стали утворюватися перші пробки і автомобільні затори. Разом з цією проблемою з’явилася і ще одна – проблема парковки автомобіля.

На початку 1920-х найгостріше ця проблема стояла в американських містах, таких як Нью-Йорк, Чикаго, Цинцинаті і Детройт, і не дивно, що саме там і були вперше споруджені механічні паркінги. Зростання вартості нерухомості на острові Манхеттен і неможливість розмістити звичні автомобільні парковки і стоянки через відсутність вільного місця серед величезних хмарочосів змусив власників нерухомості звернутися за допомогою до інженерів. Використання електронних інтелектуальних систем паркування почалося ще в 1920-х роках, оскільки в американських містах, таких як Лос-Анджелес, Чикаго, Нью-Йорк з'явилися автоматизовані системи паркування. Окрім того, в Нью-Йорку в одній з автостоянок у арт-деко.

Важко сказати, хто перший винайшов паркувальний ліфт для паркінгів і будинків. У тому чи іншому вигляді він існував з часів Першої Світової Війни, але в 1925 році американський інженер і винахідник Макс Міллер запатентував першу в США систему механічної автопарковки, точніше сказати перший автомобільний ліфт – прообраз сучасної системи.

Система, яка поширена по всій Японії це "Ferris wheel" або "paternoster system", що була створена корпорацією Westinghouse в 1923 і реалізована в 1932 році на вулиці Монро в Чикаго.

В останні роки збільшення кількості автомобілів значно загострило проблему організації місць їх постійного зберігання і тимчасового розміщення (парковки) у місць масового відвідування, перш за все в центральних частинах найбільших міст. На парковку автомобіль поміщається на відносно нетривалий

час перебування його власника або пасажира на роботі, в магазині, в культурно-масовій установі і в інших подібних місцях, що відрізняє парковку від гаража або стоянки.

Автопарковка (англ. parking) – це будівля, споруда, конструкція або спеціальна відкрита площадка, яка призначена для тимчасового чи постійного зберігання транспортних засобів. [1]

Найпростішими є наземні, як їх ще називають, площинні, парковки, які представляють собою одно рівневі відкриті стоянки для автотранспорту. Територія під стоянку автомобілів обмежується тільки розміткою та знаками. Також є парковки, обгороджені по всьому периметру парканом, мають рознесені місця в'їзду і виїзду, охорону, засоби обліку часу та інші автоматичні системи.

Наземні майданчики для парковок займають великі території в містах, що зменшує і так невеликі острівці газонів. Для вирішення цієї проблеми створюються екопарковки за допомогою газонних решіток, які зміцнюють грунт і кореневу систему трави. В результаті виходить акуратний газон з живою трави, на який спокійно може в'їхати автомобіль, не пошкодивши рослини [2]

Підземні парковки розташовуються під бізнес-центрами, житловими комплексами і деякими торговими центрами. Вони можуть мати кілька рівнів. підземні парковки вирішують ряд екологічних проблем - таких як забруднення навколишнього середовища, шум, витіснення житлового простору мікрорайонів, не спотворює ландшафт і архітектурну цілісність міста.

У великих містах під паркінги використовують даху малоповерхових будівель. Головна відмінність таких будівель - в монолітних перекриттях останнього поверху і шару асфальту поверх гідроізоляції.

Найбільш витратна частина таких стоянок - це естакада для проїзду на верхню парковку. Естакада не повинна мати великий ухил, а також маленький радіус повороту при невеликій її ширині. Якщо ухил естакади 10-12%, відвідувач, пішовши по стрілці, може навіть здивуватися, що виявився на даху. У Німеччині ухил заїзду для легкових автомобілів не повинен перевищувати 30%. Допустимий ухил наземної парковки за американськими нормам - 8%. Це означає, що на 1 метр горизонтальної площині доводиться 8 см підйому або спуску.

Багаторівневі паркінги – єдиний ефективний спосіб вирішення проблеми

зберігання автотранспорту в великій кількості на невеликій території. Вони можуть вміщати в себе від декількох сотень до декількох тисяч машин. існує багато варіацій багаторівневих паркінгів. Вони можуть перебувати в окремій будівлі або бути прибудованими до глухих торцевих стін будівлі. Для в'їзду автомобілів в них можуть бути влаштовані прямолінійні або криволінійні рампи, напіврампами, похилі підлоги, ліфтові підйомники, механізовані і автоматизовані підйомники і маніпулятори. У них може бути передбачена електронна система оповіщення про кількість вільних місць. Для зручності і безпеки пересування автомобілів між рівнями можуть бути розділені з'їзди підйому і спуску. [2]

Останнім часом традиційні способи розміщення автомобілів не задовольняють сучасним вимогам. Машин все більше, а площі все ті ж, або навіть менше. Для того щоб зменшити втрати в площах, машини потрібно розміщувати щільніше один до одного. Але це можливо тільки в тому випадку, якщо водій не бере участі в процесі постановки автомобіля на стоянку.

Для вирішення цієї проблеми з'явилися механізовані парковки. Автомобілі здаються на зберігання, після чого в автоматичному режимі переміщаються на місце зберігання. Видача автомобілів відбувається в зворотному порядку. Механізована парковка – це багатоярусна будівля з ліфтом і комірками для машин. Система повністю автоматизована і управляється одним оператором. Від водія вимагається лише поставити машину в ліфт і вручити оператору магнітну картку. Далі ліфт підніме автомобіль на потрібний ярус, потім перемістить в комірку, відповідну коду картки.

Механізовані парковки бувають різних видів: горизонтальні і вертикальні. Якщо будівля парковки прямокутної форми, то в ній працюють дві системи доставки транспортного кошти в бокс: система вертикального і горизонтального переміщення. Процес розміщення автомобілів в боксах і їх видача тривають досить довго. Якщо будівля виконана у вигляді багатоповерхового циліндра, то на кожному поверсі знаходиться нерухома кільцева платформа, в якій є радіально розташовані осередки для зберігання автомобілів і ліфти.

Усередині нерухомих кільцевих платформ зроблені поворотні платформи, на яких змонтовані маніпулятори, призначені для переміщення автомобілів з ліфтів і установки їх в осередки зберігання, а також у зворотньому напрямку.

Одночасно самі маніпулятори можуть використовуватися як осередки для зберігання автомобілів.

Механізовані вертикальні паркінги споруди видимі. Їх зовнішнє облицювання по металоконструкціях може бути різна: залізобетонний короб, профільний лист, сендвіч-панелі, тоноване і дзеркальне скло.

Існують також багатоповерхові стоянки, містять збірно-розбірний каркас, має кілька стелажів з осередками зберігання автомобілів. ряди стелажів утворюють на кожному ярусі коридор з протилежним розташуванням осередків. несучі конструктивні елементи каркаса утворюють шахту підйомника, що містить вантажну платформу, на якій встановлений візок з катками, що рухаються по напрямних, розташованим на вантажній платформі і вздовж коридору на кожному ярусі. У автостоянці є механізми фіксації вантажний платформи щодо ярусу і візки щодо осередків зберігання автомобілів (Боксів).

Останнім часом з'явилися багатоярусні смарт-паркінги, що працюють за принципом колеса огляду: автомобілі в'їжджають на стоянку, а потім їх піднімають нагору і мають у своєму розпорядженні в секціях один над одним. Тобто машини (а їх може бути до 12 одночасно) йдуть у висоту. Для збільшення місткості вже існуючих гаражів та автостоянок розроблені спеціальні пристрої.

У Нідерландах урядом затверджений проект зі створення не просто підземних багатоповерхових парковок, а цілих міст - паркінгів під центром Амстердама з мийками, автомагазинами і навіть спортивними залами, басейнами і кінотеатрами. Під центром міста пропонується побудувати шість підземних поверхів, що вирішить проблему нестачі вільного місця в центрі міста. На час будівництва пропонується осушувати знамениті амстердамські канали і використовувати їх як під'їзні шляхи, а по закінчення робіт знову заповнити водою. Для реалізації проекту буде потрібно 20 років і більше 10 млрд євро.

У Данії здійснено найкращий з експериментальних проектів за рішенням паркувального питання в житловій зоні. За формою споруда нагадує трибуну великого стадіону, де замість лавок розташовані ступінчасті тераси, під якими розташований паркінг. Будівля потопає в зелених насадженнях, які повинні компенсувати можливі появи вихлопних газів (в Європі заборонено розташовувати парковки поблизу житлової зони), хоча і передбачена потужна

вентиляція за останнім словом техніки і відмінна шумоізоляція.

У Німеччині інженери придумали паркувати машини прямо у себе вдома на спеціальному балконі. Причому такий варіант гаража не виключає наявності звичайного балкона, зимового саду, при бажанні власників. Завдяки підйому і спуску на спеціальному ліфті машина буде перебувати практично в квартирі.

У Римі люблять будувати багаторівневі підземні паркінги, над якими розбиті сади і дитячі майданчики.

1.2 Класифікація автопарковок

Автомобільні парковки в загальному випадку можна класифікувати за різними ознаками. [3]

За розміщення в міській забудові:

- в зоні об’єктів загальноміського значення міської забудови (громадські, спортивні, культурні, торговельні центри, вокзали, аеропорти та ін.);
- в комунальних і інших нежитлових зонах;
- в житловій зоні (в т.ч. районні, внутрішньо квартальні, дворові);
- в зоні міського транспорту (площі, вулиці, транспортні розв’язки, мости).

За тривалістю зберігання:

- постійне зберігання;
- тимчасове зберігання;
- сезонне зберігання.

За розміщення відносно об’єктів іншого призначення:

- окремо розташовані;
- прибудовані;
- вбудовані;
- комбіновані.

За розміщення відносно рівня землі

- наземні;

– підземні;
– комбіновані.

За кількістю поверховості:

– одноповерховий;
– багатоповерхові.

За способом міжповерхового переміщення:

– раптові;
– механізовані;
– автоматизовані.

За організацією зберігання:

– манежні;
– боксові;
– осередкові;
– комбіновані.

За типом огороджувальних конструкцій:

– закриті;
– відкриті;
– комбіновані.

За умовами зберігання:

– літні;
– опалювальні;
– комбіновані.

1.3 Аналіз окремих існуючих на ринку програмних продуктів автомобільних парковок

Для розробки ефективного продукту, потрібно, для початку, здійснити пошук відомих рішень, проаналізувати та дослідити переваги і недоліки уже реалізованих аналогів, і на основі отриманих результатів – визначити, яких помилок потрібно уникнути, а які переваги слід правильно використати.

У даній дипломній роботі було проаналізовано три системи-аналоги, які, за своїм функціоналом, максимально близькі з функціями досліджуваної предметної області.

Перший аналог представлений такою системою: InrixParkMe – система представлена світовою компанією Inrix, призначена для управління автостоянкою в реальному часі; Крім того, проаналізовано систему AS101 ProPark розроблена компанією “МИККОМ”. Це рішення слугує для автоматизації парковок і являється інформаційно-навігаційною системою. Третій аналог представлений системою ModusPark, яка є продуктом компанії UniPark. Система призначена для регулювання автомобільного потоку на парковці.

1.3.1 InrixParkMe

У цьому розширенні існує можливість перегляду автопарковок, як у веб-застосунку так і через мобільний пристрій (рисунок 1.1)

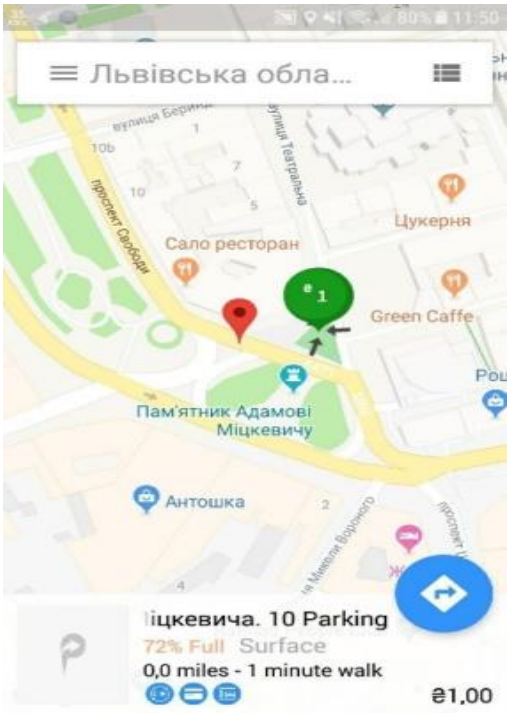

Рисунок 1.1 – Представлення системи InrixParkMe через мобільну платформу

Розташування та площу парковки можна побачити на GoogleMaps. Також користувачу надається додаткова інформація про тарифи, графік роботи, доступні зручності та примітки, що стосуються автопарковки, що добре проілюстровано на

рисунку 1.2.

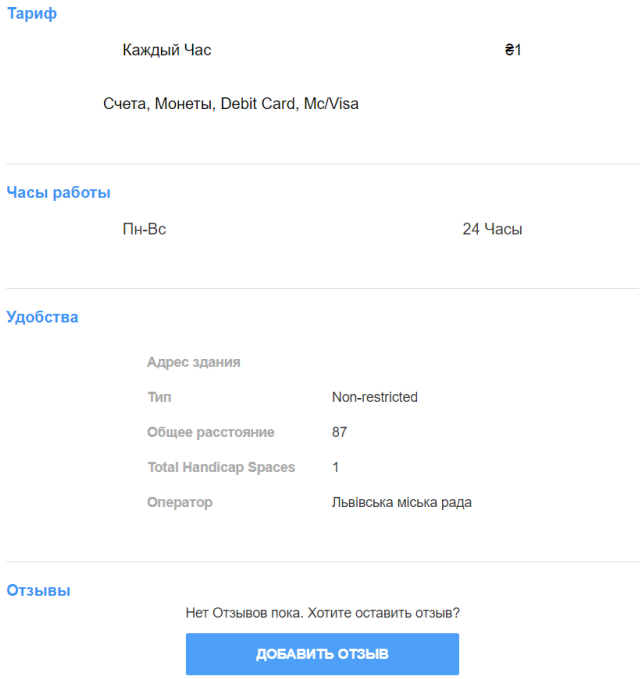

Рисунок 1.2 – Інформація про автостоянку

Змінювати, обновляти і зберігати інформацію про стан парковочних місць можна у режимі реального часу. Також адміністратор парковки володіє можливістю відслідковування кількості клієнтів, які відвідували вибрану локацію та отриманий дохід від оренди автопарковочних місць (рисунок 1.3).

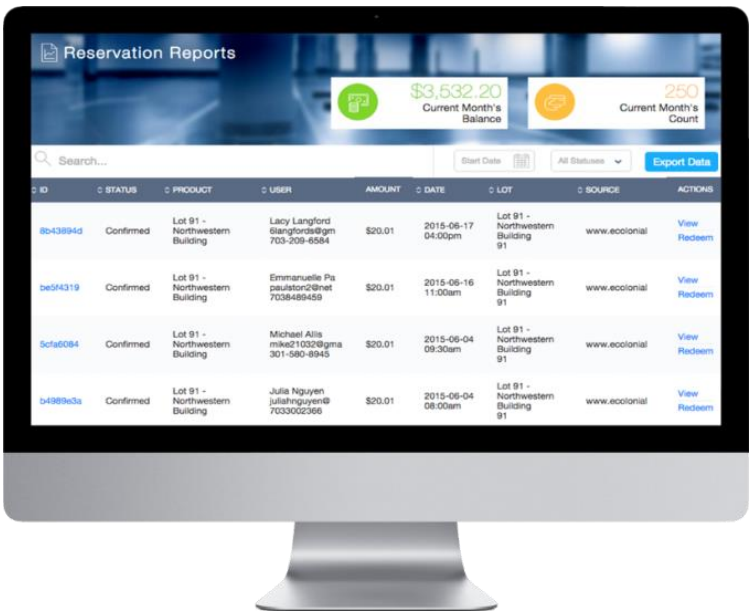

Рисунок 1.3 – Відслідковування кількості клієнтів

1.3.2 AS101 ProPark

Ця автоматизована інформаційно-навігаційна система призначена для

точного визначення і підрахунку вільних місць на парковці. Рівень зайнятості стоянки можна побачити на рисунку 1.4.

Рисунок 1.4 – Карта автостоянки

Також дана система містить функцію оповіщення водіїв і персоналу. Принцип роботи системи заключається у точному визначені зайнятих і вільних місць за допомогою підрахунку машин, які виїхали і заїхали.

За допомогою світлової індикації кожного парковочного місця і інформаційного повідомлення, водієві представляються вільні місця і оптимальний маршрут до цілі, як показано на рисунку 1.5.

Рисунок 1.5 – Приклад системи навігації водіїв у AS101 ProPark

Система виконана на базі комплексу AS101 Pro і володіє високою надійністю, зручністю і простотою експлуатації.

1.3.3 ModusPark

Система в’їзду на стоянку ModusPark, яка є зрозумілою і простою: жодних складнощів з парковкою, безперебійна робота всіх елементів автоматичного паркінгу.

Система пропонує ефективні і сучасні рішення по управлінню транспортом з можливістю самостійного контролю автомобільного потоку. Грамотне управління необхідно для роботи всіх систем без збоїв. Контроль кожної стоянки здійснюється з допомогою сервера. Але, попри це, автоматизація парковки, в значній мірі, здійснюється співробітниками парковки.

На рисунку 1.6 представлено можливу модель паркінгу під керування системи ModusPark.

Рисунок 1.6 – Модульний паркінг ModusPark

Технічна підтримка та сервісне обслуговування автомобільних стоянок. Автомобільні парковки також обладнані, як на грунті, так і на твердій поверхні, компанія забезпечує всі необхідні дорожні елементи для автоматизації паркування, виконує горизонтальну і вертикальну розмітку, здійснює технічну підтримку і сервісне обслуговування паркінгів.

Технічна підтримка та повна автоматизація паркінгів. ModusPark надає можливість скористатися такими технологічними розробками (веб-сторінка самообслуговування клієнтів, замовлення послуг паркування через Інтернет, що

можна побачити на рисунку 1.7, мобільні додатки), службою допомоги, що працює 24 години на добу.

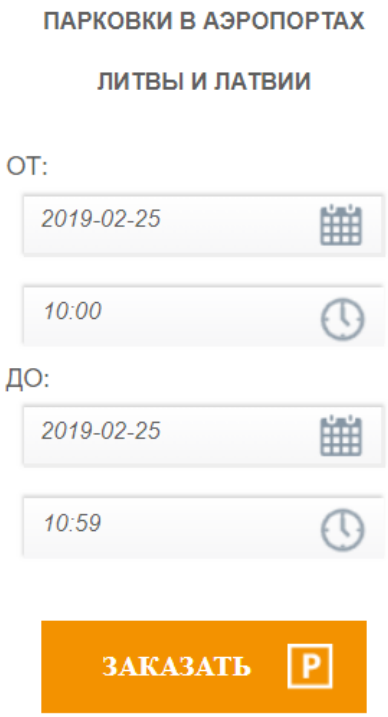

Рисунок 1.7 – Замовлення парковки ModusPark

Сервісне обслуговування стоянок ModusPark включає в себе:

– безперервну цілодобову технічну підтримку стоянок, щоб оперативно вирішувати будь-які питання;

– дистанційний контроль всіх систем паркування, щоб оперативно усувати збої і помилки;

– своєчасне оновлення програмного забезпечення для підтримки високого рівня ефективності;

– постійний моніторинг і сервісне обслуговування обладнання стоянок (контроль витрати чекової паперу, ремонт і налагодження всіх систем, інкасація);

– швидку віддалену коригування тарифів, розбивку цін за часом і дням тижня;

– відстеження наявності вільних місць, видачу карт постійного клієнта для цільових користувачів;

– контроль безпеки парковки за допомогою систем

відеоспостереження.

1.3.4 Порівняння основних можливостей розглянутих аналогів

Проведений короткий огляд існуючих програмних засобів для організації автомобільної парковки дозволив звести дані для порівняння їх можливостей у таблицю 1.1.

Таблиця 1.1 – Порівняльна характеристика програмних продуктів

Назва програмного продукту	InrixParkMe	AS101 ProPark	ModusPark
Версія продукту	2.0	1.0	1.0
Функціональність	- Профіль користувача - Маркер парковки - Карта парковки - Додавання улюблених паркінгів	- Допомога водіям - Система оповіщення	- Замовлення парковки через мережу -Дистанційний контроль всіх систем паркування
Інтерфейс користувача	WEB-застосунок/Смарт-застосунок	Desktop-система	WEB-інтрефйейс
Допомога користувачеві	Присутня	Присутня	Присутня
Читабельність	Присутня	Присутня	Присутня

Після детального аналізу систем «InrixParkMe», «AS101ProPark», «ModusPark», можна, зробити висновок про те, яким критеріям повинна відповідати розроблюваний сервіс у межах цієї дипломної роботи та, які функціональні та нефункціональні вимоги повинен виконувати.

Спираючись на згадану в п.1.2 класифікацію парковок, потрібно створити таке ПЗ, яке б дало можливість генерування та розмітки автомобільної парковки, в залежності від існуючої локації. Це дозволить створювати універсальну розмітку парковки, максимально використовувати наданий простір, і тим самим, економити дорогоцінне місце стоянки.

Будь-яке суспільство складається з 10%, які ніколи не будуть порушувати правила, ще 10% будуть порушувати їх завжди і 80% будуть вести себе за обставинам. Але поки 90% людей змушені порушувати правила паркування, порядок навести не вдасться.

Тому, для фільтрування та наведення порядку слід зробити автостоянку платною, оскільки, більшість водіїв готові паркуватися за правилами, але у них немає такої можливості, тому всі і стають порушниками. Платна автопарковка забезпечить водія місцем і усуне необхідність порушувати правила, що, в свою чергу зекономить автовласникам певні кошти.

Для коректної та точної розмітки і адміністрування автомобільної парковки необхідно використати ПК, тому, що:

– комп’ютер володіє більшою кількістю периферійних пристроїв, які дозволять точно та максимально розмітити доступний простір;

– для адміністрування баз даних ПК володіє об’ємним ресурсом оперативної та постійної пам’яті;

– для операційних систем комп’ютерів існує значна кількість технологій для ефективної роботи з базами даних.

1.5 Висновки до першого розділу

У першому розділі дипломної роботи проведено аналіз предметної області дослідження, наведено основні характеристики, види автомобільного паркінгу та класифікація парковок. Проведено аналіз окремих представлених на ринку програмних продуктів для автомобільного паркінгу. Порівняно їх основні функціональні можливості.

2 СИСТЕМНИЙ АНАЛІЗ СЕРВІСУ ДЛЯ АДМІНІСТРУВАННЯ І ОБЛІКУ РОБОТИ АВТОМОБІЛЬНОЇ ПАРКОВКИ

2.1 Основні функціональні можливості і завдання сервісу

Розробка сервісу керування паркінгом включає в себе декілька проблем, які необхідно було вирішити і в подальшому правильно створити архітектуру програми.

Так як кожний паркінг не може бути подібною на іншу і має свої особливості, необхідно було створити таку частину системи, яка б відповідала за створення схеми паркінгу, яка б відображала його усі архітектурі дані.

Наступне завдання полягало у створенні частини, яка б відповідала за ведення обліку паркінгових місць, використовуючи дані зі створеної схеми.

Відповідно до цих завдань, було вирішено створити 2 програми, кожна з яких би вирішувала поставлене завдання.

В загальному, система повинна виконувати наступні функції:

– можливість створення схеми паркінгу, шляхом реалізації розмітки XML-файлу;

– можливість розміщення елементів паркінгу на робочій області;

– можливість завантаження створеної схеми в основну програму;

– можливість запису інформації про клієнтів паркінгу в базу даних;

– реалізація CRUD для роботи з базою даних;

– створення інформаційного вікна для відображення інформації про клієнтів паркінгу.

2.2 Основні бізнес-процеси паркінгу автомобілів

Оскільки, сервіс складається з декількох самостійних програм, то для правильної роботи генератора схем парковки повинні відбуватися такі бізнес-процеси: [4]

1) старт роботи генератора схем:

– завантаження попередньої схеми;

– побудова нової схеми;

2) створення парко-місць:

– додавання місць;

– додавання вказівних знаків;

– додавання додаткових знаків;

– видалення позначок;

3) кінець роботи генератора схем:

– збереження сформованої схеми;

– запис назви і адреси схеми;

– створення XML-файлу.

Генератор схем призначений для розмітки площі або деякого периметру, які в подальшому будуть слугувати в якості місць для парковки.

Користувачу буде запропоновано варіанти розміщення паркувального місця за напрямком в'їзду.

Також для навігації водіїв присутня можливість додавання на схему парковки вказівних знаків. Додатковою функцією є розміщення перешкод у формі рослинності.

На початку роботи користувач робить вибір: завантажити уже попередньо створену схему або створити нову. На рисунку 2.1 проілюстровано схему бізнес-процесу «Старт роботи генератора схем».

Рисунок 2.1 –Діаграма бізнес-процесу «Старт роботи генератора схем»

Характеристика бізнес-процесу «Старт роботи генератора схем» представлена у таблиці 2.1.

Таблиця 2.1 – Старт роботи генератора схем

Назва характеристики	Значення характеристики
Ім’я бізнес-процесу	Старт роботи генератора схем
Основні учасники	Адміністратор
Вхідна подія	Запуск додатку
Вхідні документи	Розмітка, вказівні знаки
Вихідна подія	Модель схеми парковки
Вихідні документи	Схема парковки
Клієнт-бізнес	Адміністратор

У процесі формування схеми парковки користувач має можливість додавати місця, вказівні та додаткові знаки.

Діаграма бізнес-процесу «Створення парко-місць» (рисунок 2.2) показує основну роботу у генераторі схем.

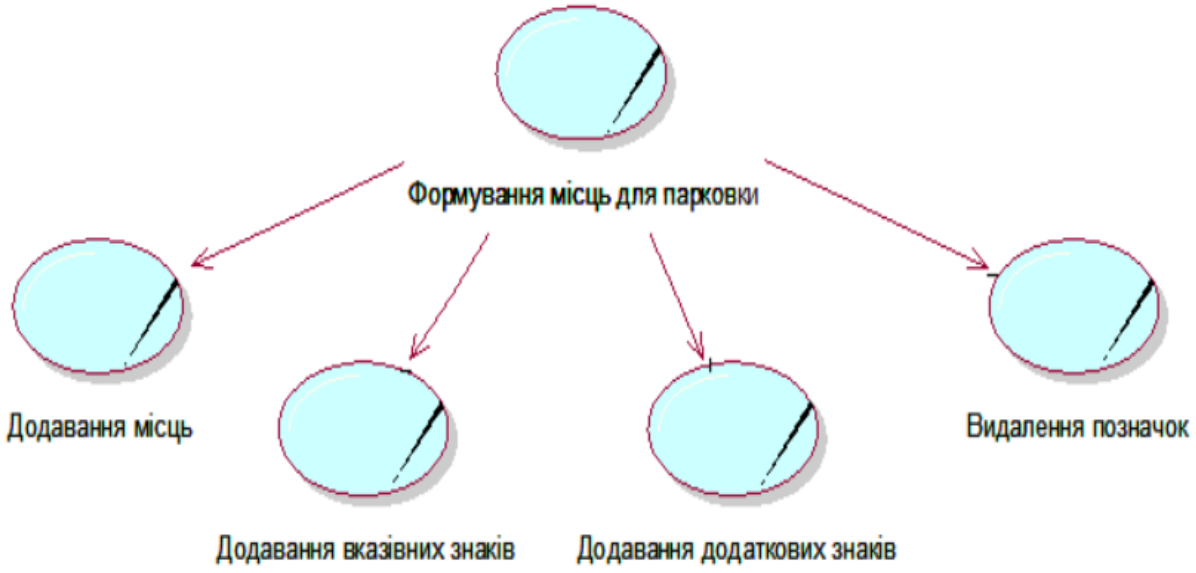

Рисунок 2.2 – Діаграма бізнес-процесу «Створення парко-місць»

Для повного розуміння бізнес-процесу «Створення парко-місць» представлена таблиця 2.2.

Таблиця 2.2 – Характеристика бізнес-процесу «Створення парко-місць»

Характеристика	Її значення
Ім’я бізнес-процесу	Створення парко-місць
Основні учасники	Адміністратор
Вхідна подія	Завантаження, створення схеми
Вхідні документи	XML-файл, проект нової схеми
Вихідна подія	Додавання місць для парковки, вказівних знаків
Вихідні документи	Модель схеми парковки
Клієнт бізнес	Адміністратор

У лістингу 2.1 подано методи для переміщення паркінгових місць на робочу область.

Лістинг 2.1 – Методи для переміщення паркінгових місць на робочу область

```
 private void pitureBox_DragDrop(object sender,DraEvents e)
{
 pictureBox.Image = (Image)e.Data.GetData(DataFormat.Bitmap);
 matrixEvent[1] = primeriTeg;
}
private void pitureBox_DragEnter(object sender,DraEvent e)
    {
 e.Efect = DragDropEfects.Copy;
    }
 private void pitureBox_DragDrop(object sender,DraEvennt e)
    {
    pitureBox.Image = (Image)e.Data.GetData(DataFormat.Bitmap);
    matrixEvent[2] = primeriTeg;
    }
    private void pitureBox_DragEnter(object sender,DraEvnt e)
    {  e.Efect = DragDropEfects.Copy;     }
```

Після формування схеми автомобільної парковки необхідно зберегти внесенні зміни, як можна побачити на діаграмі бізнес-процесу «Кінець роботи генератора схем» (рисунок 2.3).

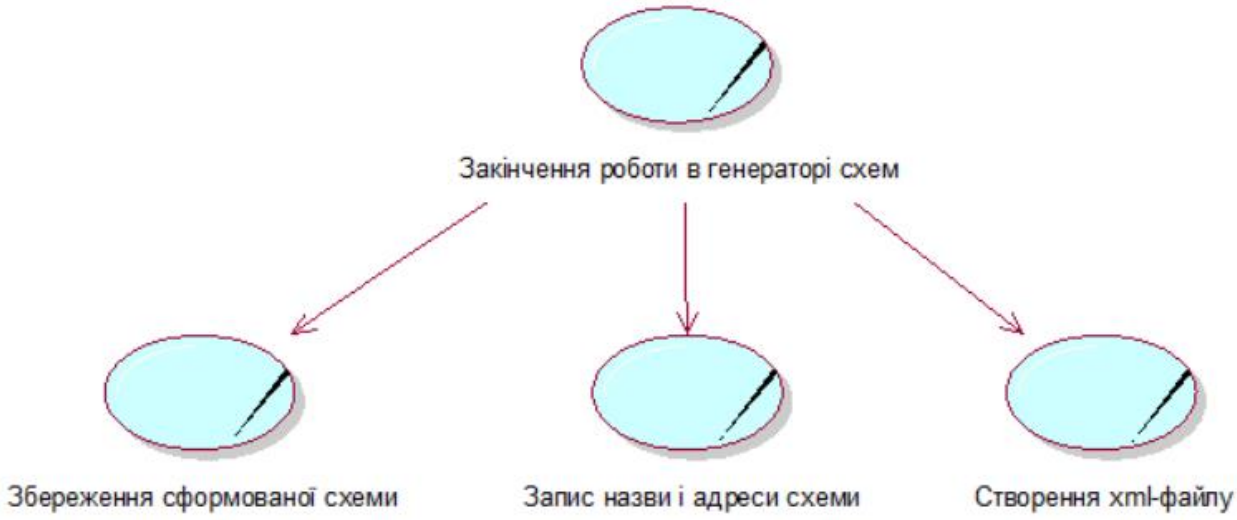

Рисунок 2.3 – Діаграма бізнес-процесу «Кінець роботи генератора схем»

Для бізнес-процесу «Кінець роботи генератора схем» представлена характеристика у таблиці 2.3.

Таблиця 2.3– Характеристика бізнес-процесу «Кінець роботи генератора схем»

Назва характеристики	Значення характеристики
Ім’я бізнес-процесу	Кінець роботи генератора схем
Основні учасники	Адміністратор
Вхідна подія	Формування схеми
Вхідні документи	Модель схеми парковки
Вихідна подія	Збереження змін
Вихідні документи	XML-файл
Клієнт бізнес	Адміністратор

Для роботи системи обліку місць на паркінгу необхідні такі бізнес-процеси: [4]

1) завантаження схеми парковки:

– пошук XML-файлу за допомогою діалогового вікна;

– завантаження XML-файлу;

2) перегляд схеми парковки:

– візуальне представлення схеми;

– підрахунок вільних місць, засобів навігації;

3) робота з БД:

– заповнення полів БД;

– відображення вільних зайнятих місць;

– відображення записів у БД.

Програма-адміністратор візуалізує схему парковки, показує де є вільні місця. Також присутня можливість формування бази клієнтів, по якій буде здійснюватися видача карт парковки та перевірка наявності вільних місць.

Для початку роботи адміністратору парковки необхідно завантажити схему парковки. Діаграма цього бізнес-процесу представлена на рисунку 2.4.

Рисунок 2.4 – Діаграма бізнес-процесу «Завантаження схеми парковки»

Характеристика завантаження схеми парковки наведено у таблиці 2.4.

Таблиця 2.4 – Характеристика завантаження схеми парковки

Характеристика	Її значення
Ім’я бізнес-процесу	Завантаження схеми парковки
Основні учасники	Адміністратор
Вхідна подія	Відкриття діалогового вікна
Вхідні документи	XML-файл
Вихідна подія	Завантаження схеми у програму
Вихідні документи	Модель схеми парковки
Клієнт бізнес	Адміністратор

Після завантаження схеми парковки система візуалізує дану схему та підраховує кількість місць, в’їздів та виїздів на парковці.

Діаграму бізнес-процесу «Перегляд схеми парковки» зображено на рисунку 2.5.

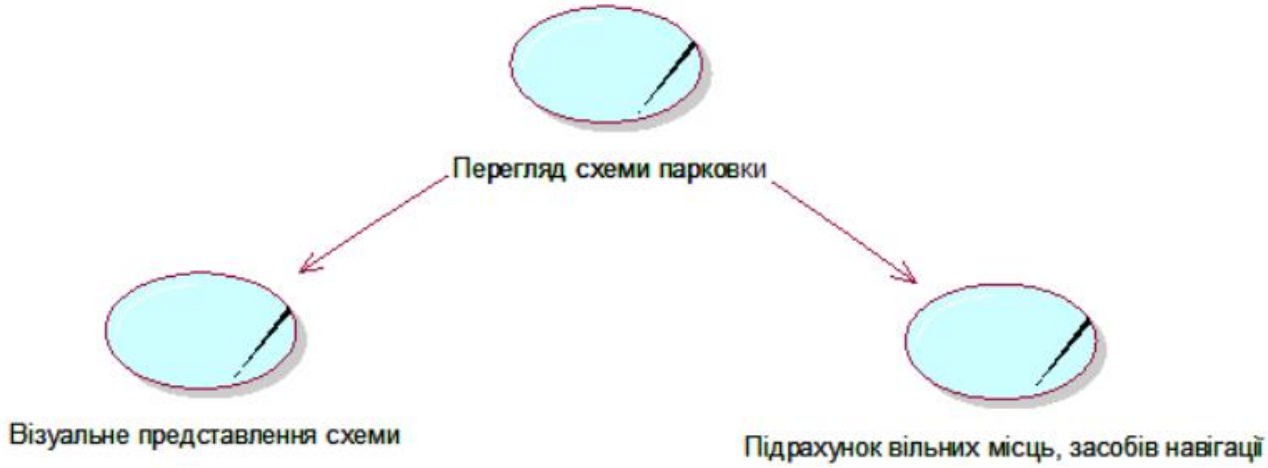

Рисунок 2.5 – Діаграма бізнес-процесу «Перегляд схеми парковки»

Характеристику бізнес-процесу «Перегляд схеми парковки» детально зображено у таблиці 2.5.

Таблиця 2.5 – Характеристика перегляду схеми парковки

Характеристики	ЇЇ значення
Ім’я бізнес-процесу	Перегляд схеми парковки
Основні учасники	Адміністратор
Вхідна подія	Візуалізація схеми
Вхідні документи	XML-файл
Вихідна подія	Підрахунок місць, в’їздів\виїздів, вказівних та додаткових знаків
Вихідні документи	Дані про парковку
Клієнт бізнес	Адміністратор

Основним бізнес-процесом є «Робота з БД». Діаграма цього бізнес-процесу представлена на рисунку 2.6.

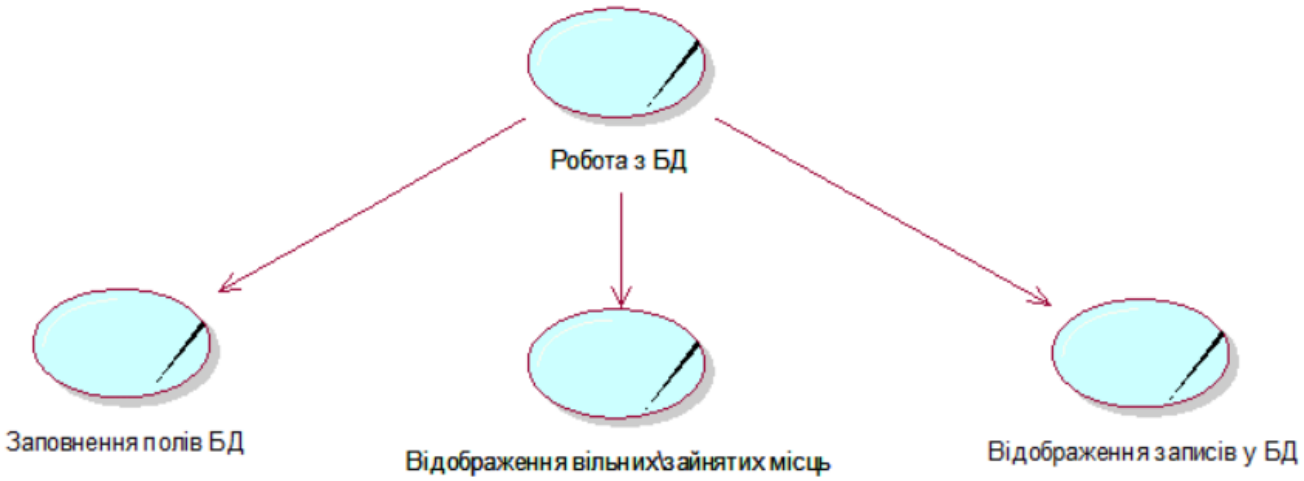

Рисунок 2.6 – Діаграма бізнес-процесу «Робота з БД»

Характеристику бізнес-процесу «Робота з БД» детально зображено у таблиці 2.6.

Таблиця 2.6 – Характеристика роботи з БД

Характеристика	Значення
Ім’я бізнес-процесу	Робота з БД
Основні учасники	Адміністратор
Вхідна подія	Представлення схеми та інформації про парковку
Вхідні документи	Дані про парковку
Вихідна подія	Формування бази клієнтів
Вихідні документи	Записи про користувачів парковки
Клієнт бізнес	Адміністратор

2.3 Специфікація вимог до розроблюваної системи

Опис (практично повний) поведінки розроблюваної системи називається специфікацією вимог до ПЗ. Також це опис різновидів використання ПЗ, тобто, аналіз того, як система реагує на зовнішні запити.[5]

Працюючи над описом специфікацій вимог до системи, потрібно згадати про функціональні та нефункціональні вимоги.

Функціональні вимоги описують функції, які повинна виконувати розроблювана система (що вона повинна робити) внутрішня робота системи та її поведінка: обчислення та зміна даних, їх обробка і інші специфічні функції системи. Ці функції витікаю з вимог користувача. Функціональні вимоги також розглядають як завдання для команди розробників.

Нефункціональні вимоги – це фіксовані умови, які, безпосередньо, не зв’язані з поведінкою системи, а слугують для опису зовнішнього середовища, у якому рішення повинно залишатись ефективним. Ці вимоги часто називають атрибутами – показниками якості, які мають бути властиві системі. Поділяються на категорії: вдосконалення (маштабування, відновлюваність і т.п.) властивостей систем та покращення (надійність, зручність у використанні і т.п.).

Отже, проаналізувавши доступні аналоги системи, зваживши усі їх переваги

і недоліки, можна, приступити до формування специфіки вимог до системи, а саме:

– створення словника вузьконаправлених термінів;
– опис варіантів використання;
– поставлення функціональних та нефункціональних вимог.

На початку роботи потрібно створити словник для пояснення термінів з досліджуваної предметної області і які будуть використовуватися в процесі проектування, реалізації та експлуатації програмного продукту.

Ці терміни описані у таблиці 2.7.

Таблиця 2.7 – Глосарій

Термін	Інтерпретація терміну
Паркінг, парковка, стоянка	Площа, місце, будівля призначені для зберігання транспортних засобів, у більшості випадків – автомобілів.
Генератор схем	Програмне забезпечення, яке призначене для формування карт парковки (розташування і кількість місць, знаків, локацій, в'їздів\виїздів) і на виході дає користувачу схему парковки зашифровану в XML-файл.
Адміністратор	Особа, яка здійснює, як створення схеми стоянки, так і роботу з клієнтами: завантаження схеми, додавання, редагування записів про користування паркінгом, моніторинг вільних місць.
XML-файл	Файл з розширеною мовою розмітки для опису документів.
Розроблювана система	Сукупність програм, призначених для адміністрування паркінгу.

Продовження табл. 2.7

Windows From	Інтерфейс програмування розроблюваної системи, який також відповідає за графічний інтерфейс користувача.
Desktop-система	Пакет програм, який буде експлуатуватися на персональних, настільних комп'ютерах.
Клієнти паркінгу	Особи, які володіють транспортним засобом (автомобіль) та виявили бажання скористатися стоянковим місцем.
Карта парковки	Інформація для клієнтів паркінгу про місце, час, вартість отримання послуг.
Вказівні та додаткові знаки	Символи, позначки, які вказують напрямок руху або позначення локації, відповідно.

Нехай визначено низку термінів для розроблюваної системи. Після цього необхідно описати взаємодію системи з зовнішніми запитами, тобто, сформувати діаграму варіантів використання. Оскільки, розроблювана система включає у себе

дві самостійних програми, то для parkingDesign (генератор схеми) діаграма буде виглядати так як показано на рисунку 2.7.

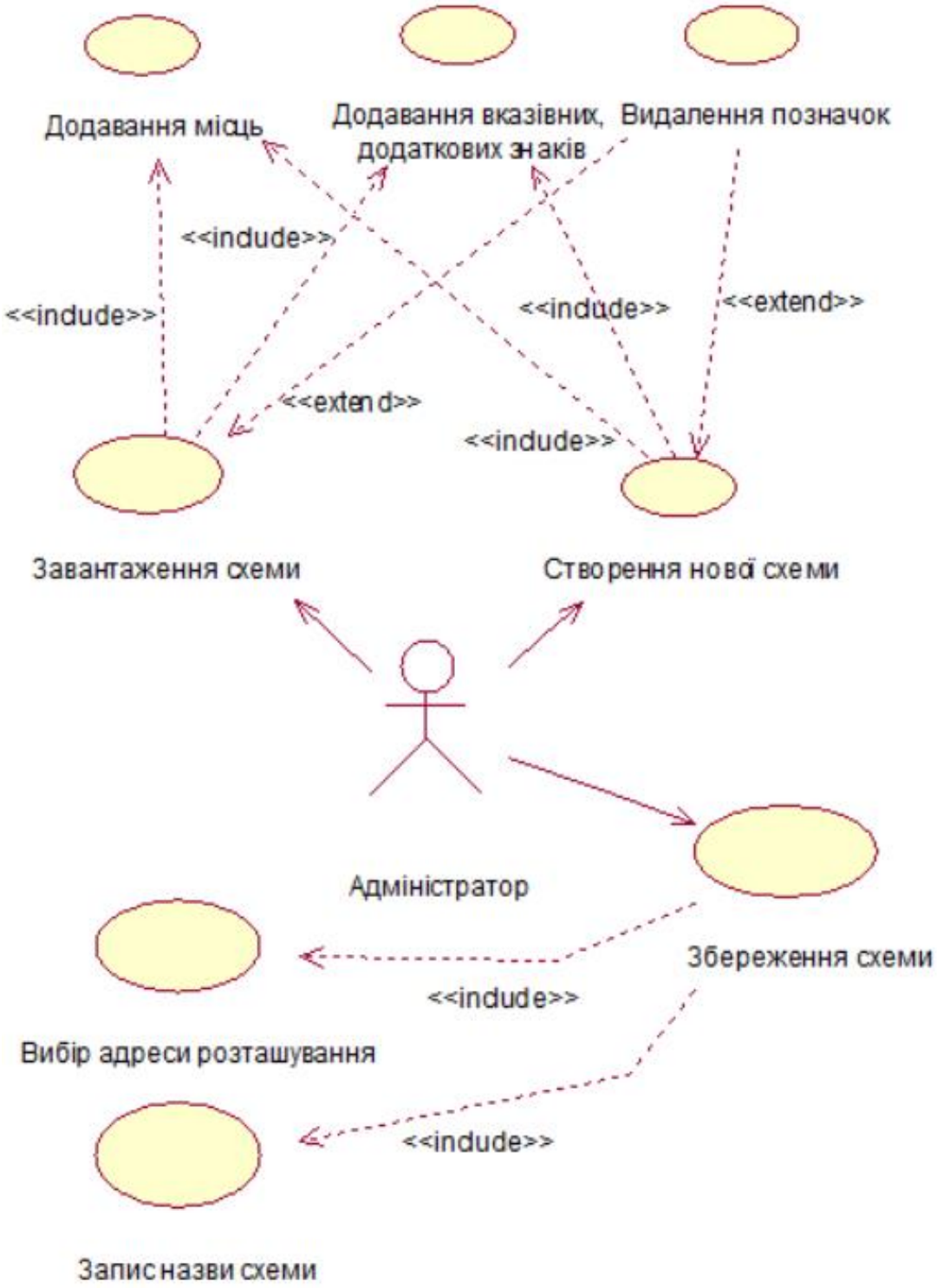

Рисунок 2.7 – Діаграма прецедентів для parkingDesign

Беручи до уваги те, що ці програми можуть використовуватися роздільно одна від одної, було прийнято рішення описати для кожної системи варіанти використання.

На рисунку 2.8 проілюстровано діаграму варіантів використання для програми parkingCRM (адміністрування паркінгу).

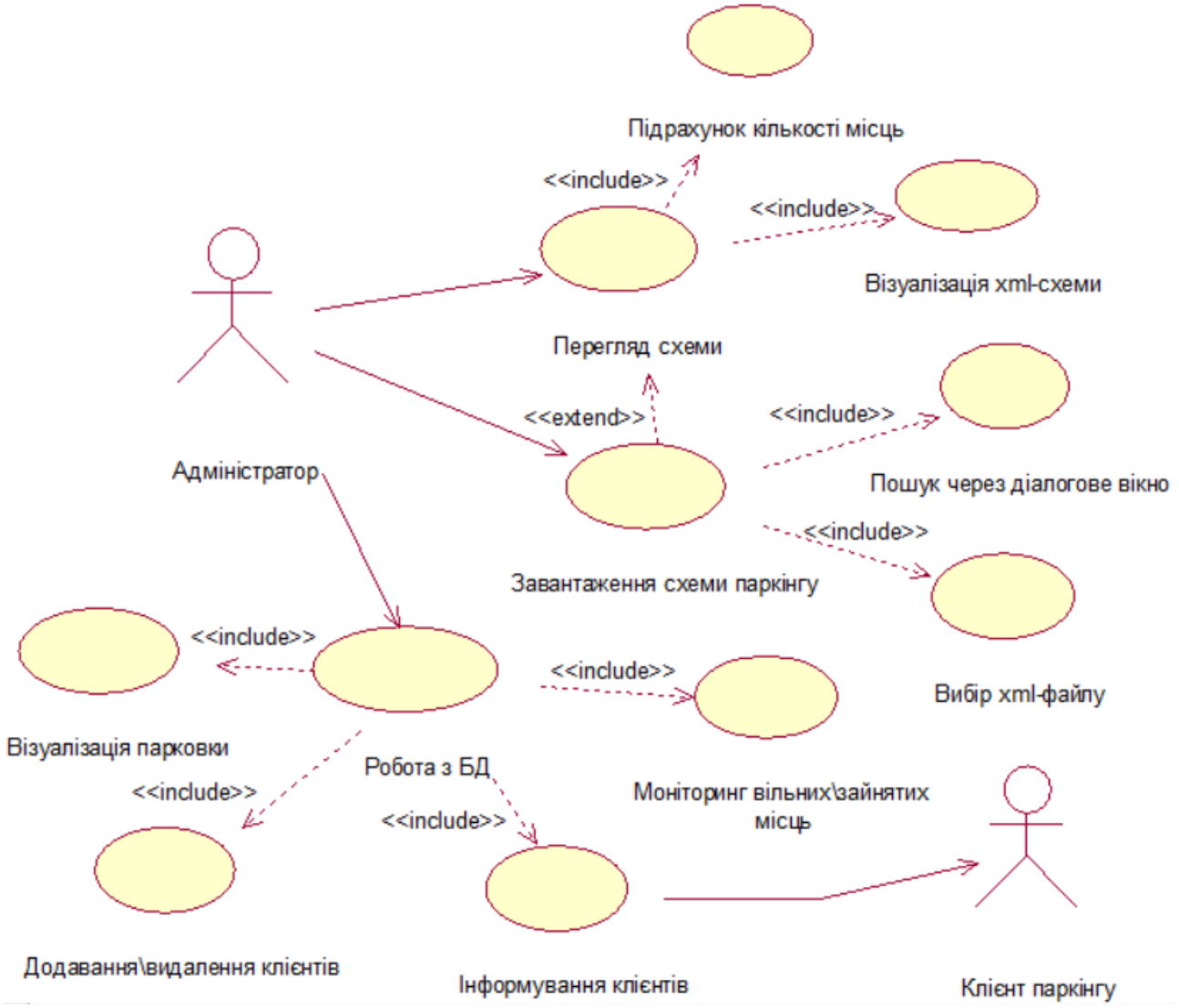

Рисунок 2.8 – Діаграма прецедентів для parkingCRM

Представивши діаграми варіантів використання для розроблюваної системи, слід більш детально охарактеризувати кожен процес.

Нижче буде подано такі варіанти використання: завантаження схеми, побудова нової схеми, перегляд схеми паркінгу, робота з базою даних.

Для побудова нової схеми потрібно перемістити місця для паркінгу, додаткові знаки на майбутню карту стоянки. Варіант використання побудова нової схеми можна побачити у таблиці 2.8.

Таблиця 2.8 –Варіант використання «Побудова нової схеми»

Характеристика	Значення
Контекст використання	Побудова нової схеми
Дійові особи	Адміністратор
Передумова	Інстальована програма parkingDesign
Тригер	Відкрита програма parkingDesign
Сценарій	1. Відкриваємо програму parkingDesign. 2. Додаємо на макет місця для автомобілів. 3. Додаємо вказівні знаки. 4. Додаємо додаткові знаки. 5. Натискаємо кнопку «Зберегти». 6. Вказуємо шлях та назву схеми. 7. Дублюємо пункт 5.
Постумова	Створена нова схема

Також існує варіант завантаження уже раніше створеної схеми. Для його використання користувач повинен через діалогове вікно вказати xml-файл з схемою. У додатку Б подано XML-документація програми parkingDesign

Таблиця 2.9 – Варіант використання «Завантаження схеми»

Характеристика	Значення
Контекст використання	Завантаження схеми
Дійові особи	Адміністратор
Передумова	Створена раніше схема
Тригер	Відкрита програма parkingDesign
Сценарій	1. Відкриваємо програму parkingDesign. 2. Натискаємо кнопку «Завантажити». 3. У діалоговому вікні, що з’явилось, вибираємо xml-файл. 4. Натискаємо кнопку «Відкрити».
Постумова	Завантажена попередньо створена тема

На рисунку 2.9 показано один з кроків сценарію варіанту використання «Завантаження схеми».

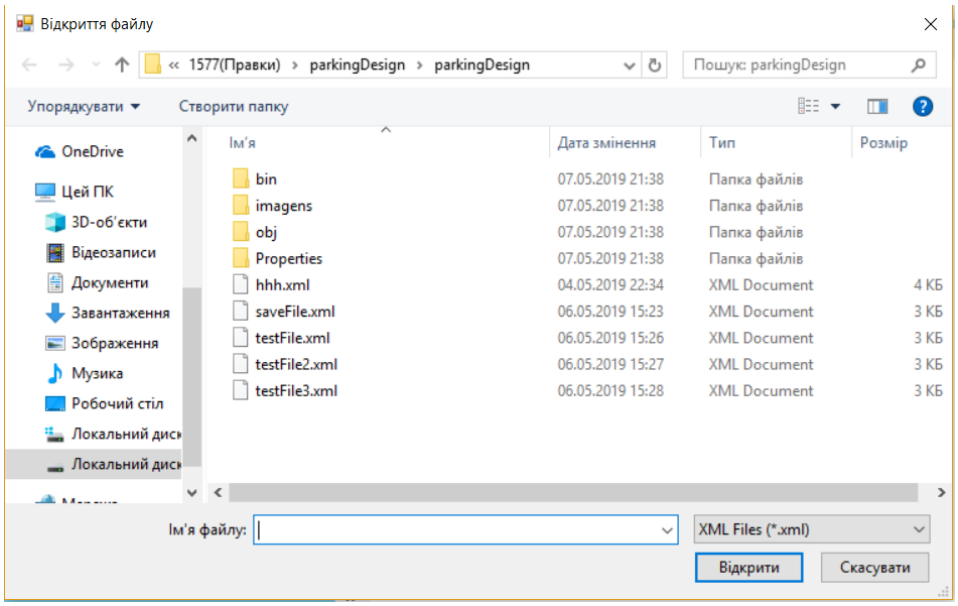

Рисунок 2.9 – Діалогове вікно для завантаження схеми паркінгу

Коли схема паркінгу сформована, можна, перейти, безпосередньо, до завантаження та перегляду схеми у системі для обліку місць на стоянці. Після того, як користувач завантажить xml-файл, система автоматично підрахує кількість місць, в'їздів та виїздів.

У лістингу 2.2 наведено XML-документація програми parkingDesign.

Варіант використання «Перегляд схеми паркінгу» описано у таблиці 2.10.

Таблиця 2.10 – Варіант використання «Перегляд схеми паркінгу»

Характеристика	Значення
Контекст використання	Перегляд схеми паркінгу
Дійові особи	Адміністратор
Передумова	Завантажена схема
Тригер	Відкрита програма parkingCRM
Сценарій	1. Відкриваємо програму parkingCRM. 2. Клікаєм по кнопці «Завантажити схему паркінгу». 3. У діалоговому вікні вибираємо потрібну схему. 4. Перевіряємо правильність вибору та інформацію про схему. 5. Натискаємо кнопку «Продовжити».
Постумова	Завантажена та перевірена схема.

Після виконаних вище кроків, адміністратор має можливість приступити до роботи з базою даних. Суть цієї роботи полягає у моніторингу вільних\зайнятих

місць та реєстрації нових клієнтів паркінгу. У таблиці 2.11 можна переглянути варіант використання «Робота з базою даних».

Таблиця 2.11 – Варіант використання «Робота з базою даних»

Характеристика	Значення
Контекст використання	Робота з базою даних
Дійові особи	Адміністратор
Передумова	Завантажена та перевірена схема
Тригер	Відкрита програма parkingDesign
Сценарій	1. Заповнюємо відповідні поля бази даних. 2. Перевіряємо наявність вільних місць. 3. Натискаємо кнопку «Додати запис»\«Редагуватизапис»\«Видалити запис». 4. У базі даних клікаєм по необхідному запису. 5. Натискаємо кнопку «Картка клієнта». 6. Клікаєм «Роздрукувати картку».
Постумова	База даних з записами про клієнтів паркінгу

Розкадровка варіанту використання «Робота з базою даних» показана на рисунку 2.10. На даному рисунку продемонстрований один з кроків інформування користувачів.

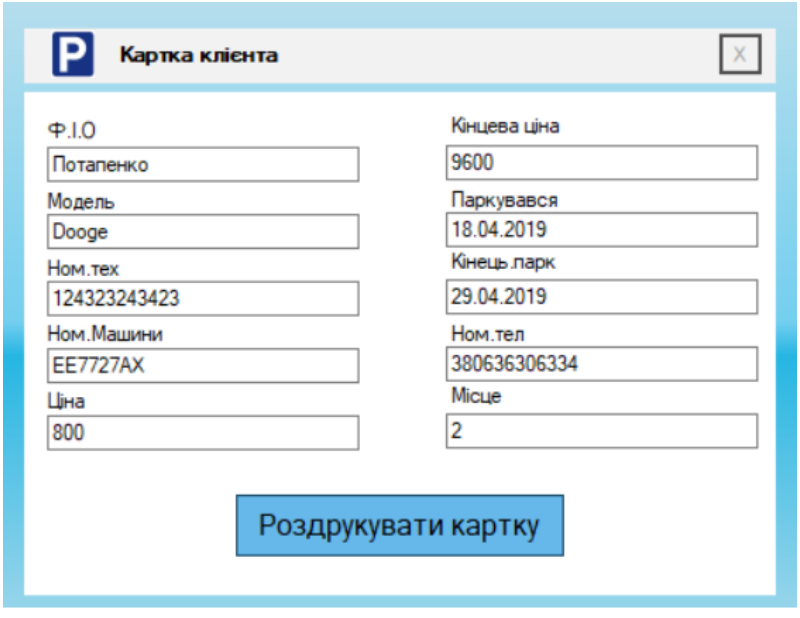

Рисунок 2.10 – Друк картки клієнта

Специфікація функціональних вимог систем представлена у таблиці 2.12. Атрибути характеризуються пріоритетом, складністю та контактом.

Таблиця 2.12 – Список функціональних вимог системи

Ідентифікатор вимоги	Назва вимоги	Атрибути вимог		
		Пріоритет	Складність	Контакт
1	Побудова нової схеми	Високий	Низька	Адміністратор
2	Завантаження схеми	Низький	Низька	Адміністратор
3	Перегляд схеми паркінгу	Високий	Низька	Адміністратор
4	Робота з базою даних	Високий	Середня	Адміністратор

Для оцінки якості роботи системи було описано нефункціональні вимоги, які відображені у таблиці 2.14.

Таблиця 2.14 – Специфікація нефункціональних вимог

№	Назва вимоги	Характеристики
1	Застосовність	
1.1	Відповідність іншим CRM-системам	Система містить зрозумілу та стандартизовану логіку
1.2	Час відгуку для типових завдань	Максимально наближений до аналогів
2	Надійність	
2.1	Середній час безвідмовної роботи	12 годин
2.2	Середній час відновлення бази даних	30 хвилин
3	Проектні обмеження	
3.1	Мова програмування	C#
3.2	Тип підключення бази даних	Локальний хостинг

2.3 Висновки до другого розділу

У другому розділі дипломної роботи сформульовано основні функціональні можливості і завдання сервісу, проаналізовано основні бізнес-процеси паркінгу автомобілів (зокрема старт роботи генератора схем парко-місць, створення парко-місць, кінець роботи генератора схем). Також розроблені вимоги до сервісу: функціональні (завдання для розробника) та нефункціональні (атрибути – показники якості).

3 ПРАКТИЧНЕ ДОСЛІДЖЕННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ СЕРВІСУ

3.1 Створення проекту у MS VisualStudio.

Проект – це основна робоча одиниця програміста. Він повинен самостійно вибрати тип проекту, а VisualStudio створює шаблон проекту відповідно до обраного типу.

У VisualStudio практично ніколи не створюється порожній проект, в який потім вписується код. Замість цього вказується в середовищі VisualStudio тип проекту, який потрібно створити, та IDE самостійно автоматично генерує файли і програмний код, що будуть основою для обраного типу проекту. Після цього користувач може працювати над додатком, вписуючи код в створений шаблоном проект [6].

Наприклад, якщо потрібно створити проект, заснований на Windows Forms, то середовище VisualStudio 2013 згенерує проект з порожньою формою. Якщо буде потрібно створити консольний додаток, то VisualStudio згенерує файл program.cs, в якому вже будуть основи простору імен, клас Program і точка входу в додаток статистичний метод Main ().

Основні способи для створення проектів у Visual C#:

– використання стандартних шаблонів проектів, наприклад «Шаблон консольного застосування» WCF (C#), для пришвидшення процесу побудови простих проектів;

– можливість використання майстра налаштування додатків для допомоги у побудові рішення. Стандартне рішення може складатися з безлічі проектів і написано на мові у складі VisualStudio;

– можливість побудови простого текстового файлу та подальшого його збереження з типом cs. Після створення проекту можливість керувати різними його аспектами.

Для створення нового проекту, потрібно вибрати File | New | Project в меню VisualStudio. При цьому відкриється діалогове вікно New Project з різноманітними видами проектів. Для розгляду роботи в інтегрованому середовищі розробками

створений простий консольний додаток з ім’ям ConsoleApp (рисунок 3.1).

Консольний додаток дозволить обмежитися невеликим обсягом коду і зосередитись на найбільш важливих питаннях вивчення роботи в середовищі розробки VisualStudio 2013 [7].

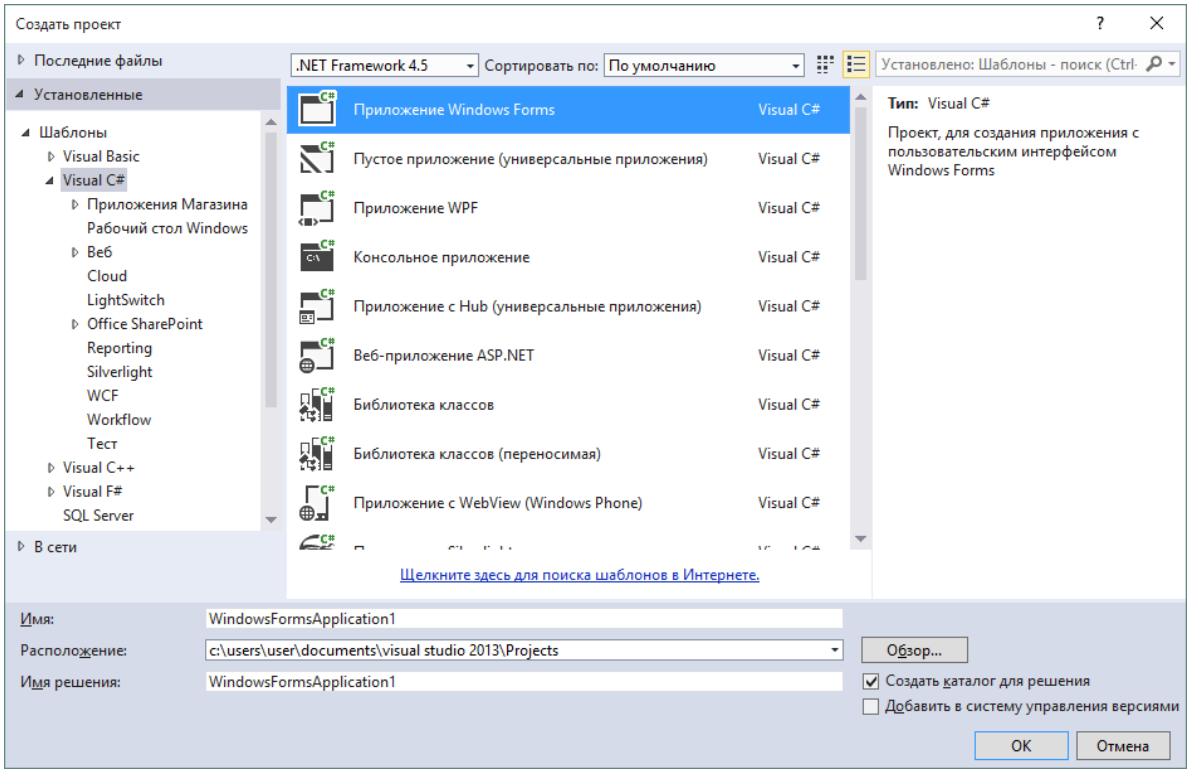

Рисунок 3.1 – Діалогове вікно New Project

На рисунку 3.2 показано вікно Solution Explorer, в якому відображається дерево файлів проекту, файл з кодом Program.cs і додатковий файл AssemblyInfo.cs в папці Properties.

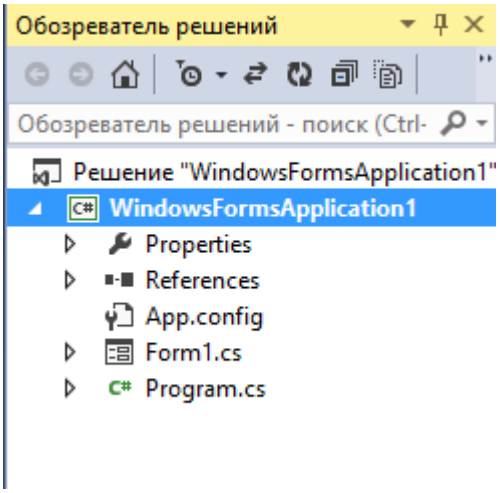

Рисунок 3.2 – Дерево проекту в вікні Solution Explorer

Існує кілька способів відкомпілювати і виконати програму з середовища VisualStudio.

Наприклад, можна натиснути комбінацію клавіш <Ctrl> + <Shift> + <B> або вибрати в меню пункт Build | BuildSolution.

В якості альтернатив можна клацнути по кнопці Build, розташованої на однойменній панелі інструментів.

Щоб виконати програму можна натиснути комбінацію клавіш <Ctrl> + <F5> або вибрати в меню пункт Debug | StartWithoutDebugging.

Програму можна виконати, не виділяючи компіляцію в окремий етап. Залежно від обраних параметрів середовище IDE збереже файл, відкомпілює і виконає його.

Конструктор класів підтримує класи C #і відображає власні класи так само, як фігури класів VisualBasic і Visual C ++, за винятком того, що класи можуть володіти відносинами множинного спадкоємства. Для відображення полів і методів у класі можна розгорнути фігуру класу або згорнути її для економії місця.

При перетягуванні в схему класів більше одного класу, що мають відношення спадкування, між цими класами з'являється з'єднання у вигляді стрілки. Стрілка вказує в напрямку базового класу.

Наприклад, якщо відобразити в схемі наступні класи, то вони будуть з'єднані стрілкою, що вказує від B до A:

```
class A {};
class B: A {};
```

Можливість перетягувань в схему класів тільки один клас B і переглянути його базові класи, клацнувши правою кнопкою миші фігуру класу B і потім вибравши команду Показати базові класи.

Конструктор класів підтримує наочне уявлення відносин множинного успадкування класів.

Множинне успадкування використовується, якщо у похідного класу є атрибути більш ніж одного базового класу.

Приклад множинного успадкування:

```
classBird {};
classSwimmer {};
classPenguin: publicBird, publicSwimmer {};
```

При перетягуванні в схему класів більше одного класу з’явиться стрілка, що з’єднує класи, які володіють відносинами множинного спадкоємства. Стрілка вказує в напрямку базових класів.

Реалізацію списку показано у лістингу 3.1.

Лістинг 3.1 – Реалізація списку

```
usingSystem;
usingSystem.Collections.Generic;
namespaceCollections
{
    classProgram
    {
        staticvoidMain(string[] args)
        {
            List<int>numbers = newList<int>() { 1, 2, 3, 45 };
            numbers.Add(6); // додавання елемента
            numbers.AddRange(newint[] { 7, 8, 9 });
            numbers.Insert(0, 666); число 666
            numbers.RemoveAt(1); //
            foreach(intiinnumbers)
            {   Console.WriteLine(i);      }
            List<Person>people = newList<Person>(3);
            people.Add(newPerson() { Name = "Петро"});
            people.Add(newPerson() { Name = "Андрій"});
            foreach(Person p inpeople)
 Console.WriteLine(p.Name);
            }
            Console.ReadLine();        }
    }
    classPerson
    {   publicstringName { get; set; }     }
}
```

Клас LinkedList<T> представляє двозв’язний список, у якому кожний елемент зберігає посилання одночасно на наступний і на попередній елемент.

Якщо в простому списку List<T> кожний елемент являє об’єкт типу T, то в LinkedList<T> кожен вузол представляє об’єкт класу LinkedListNode<T>.

Приклад двозв’язного списку показаний у лістингу 3.2.

Лістинг 3.2 – Двозв’язний список

```
usingSystem;
usingSystem.Collections.Generic;
namespaceCollections
{classProgram
    {  staticvoidMain(string[] args)
   {
LinkedList<int>numbers = newLinkedList<int>();
 numbers.AddLast(1); //
// так як в списку нема вузлів, то останній елемент стає першим
numbers.AddFirst(2);
//вставляєм вузол із значеням 2 в останнє місце
numbers.AddAfter(numbers.Last, 3); // вставляєм вузол із значеням 3
в останнє місце
//тепер у нас є список: 2, 1, 3
foreach(intiinnumbers)
{
Console.WriteLine(i);
}
LinkedList<Person>persons = newLinkedList<Person>();
// додаємо persona в список і отримуємо об'єкт
LinkedListNode<Person>, в якому зберігається імя Петро
     .....
```

Тут створюються і використовуються два списки: для чисел і для об'єктів класу Person. Далі буде продемонстровано роботу із класом Person.

Методи вставки (AddLast, AddFirst) при додаванні в список повертають посилання на доданий елемент LinkedListNode<T> (в нашому випадку LinkedListNode<Person>). Потім керуючи властивостями Previous і Next, ми можемо отримати посилання на попередній і наступний елементи списку.

3.2 Реалізація частини сервісу для створення схеми паркінгу

Перша частина системи відповідає за створення схеми паркінгу для подальшого її використання у основній частині додатку. На рисунку 3.3 зображено основне вікно програми.

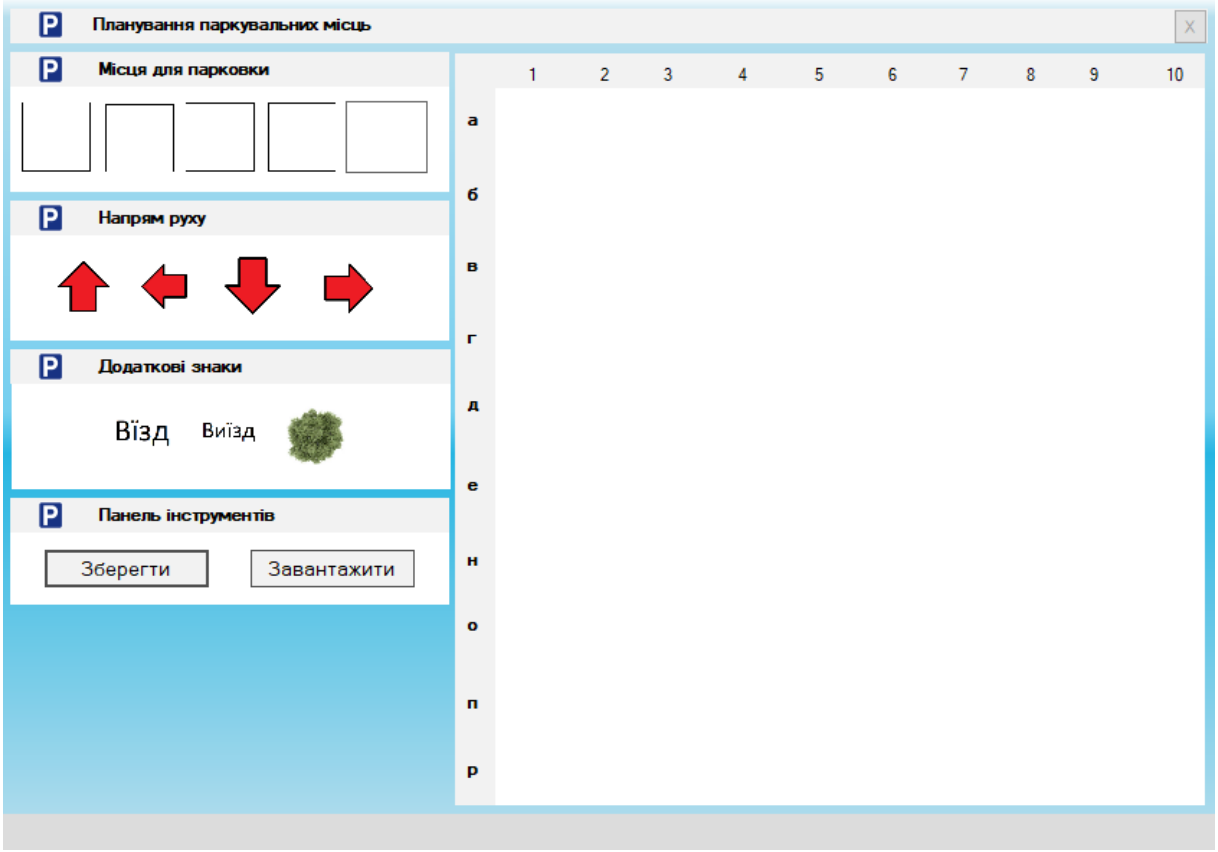

Рисунок 3.3 – Головне вікно додатку

Як видно з рисунку, головне вікно програми поділене на декілька секцій, кожна з яких відповідає за певне функціональну частину паркінгу.

У лістингу 3.3 подано метод завантаження схеми паркінгу (parkingCRM).

Лістинг 3.3 – метод завантаження схеми паркінгу

```
private void LoadXml_Clik(object sender, EvenArg e)
        {   OpenFilDialog ofd = new OpenFilDialog();
            ofd.Filter = "XML Files (*.xml)|*.xml";
            ofd.FilterIndex = 0;
            ofd.DefaultExt = "xml"; }
            { if (!String.Equals(Path.GetExtension(ofd.FileName),
                                 ".xml",
StringComparison.OrdinalIgnoreCase))
          MessageBoxButtons.OK,
              MessageBoxIcon.Error);        }
                else
                {    XmlDocument xmlDoc = new XmlDocument();
                    xmlDoc.Load(ofd.FileName);
                    locationSource = ofd.FileName;
                    int count = 0, countExit = 0, countEnter = 0;
                    for (int i = 0; i < 100; i++)
                    {
                        XmlNodeList listElement =
```

Продовження лістингу 3.3

```
xmlDoc.GetElementsByTagName("picturBox");
                        string textGetFromXML =
listElement[i].InnerText;
                        if (textGetFromXML == "bot1.PNG" ||
                            textGetFromXML == "left1.PNG" ||
                            textGetFromXML == "right1.PNG" ||
                            textGetFromXML == "top1.PNG")
                        {   count++;
                            LoadXml.Visible = false;
                            label7.Visible = false;
                            panel5.BackColor = Color.Red;
                            panel6.BackColor = Color.Red;
                            label9.Visible = true;
                            label10.Visible = true;
                            label11.Visible = true;
                            label12.Visible = true;
                            groupBox1.Visible = true;
                            Cont.Visible = true;
                        }
                        if (textGetFromXML == "toEnter.PNG")
                        {  countEnter++;       }
```

На рисунку 3.4 показано варіанти паркінгових місць.

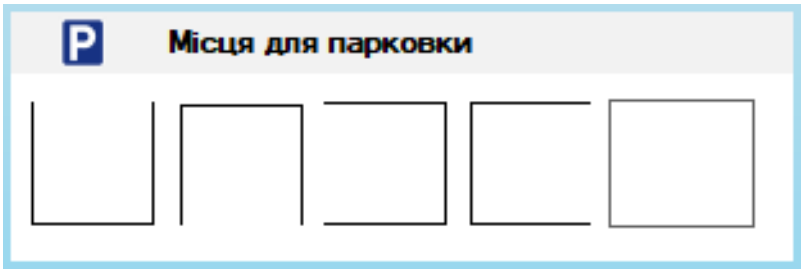

Рисунок 3.4 – Місця для паркінгу

Такий підхід дозволяє не прив’язувати місце до одного вигляду і однієї позиції, що дає можливість будувати схему з будь-якими варіантами розміщення паркінгових місць.

Для того, щоб рух по паркінгу не був хаотичний, було створено секцію, яка відповідає за розміщення напрямку руху по паркінговій зоні (рисунок 3.5).

Рисунок 3.5 – Напрям руху по паркінговій зоні

Кожний паркінг має свій в’їзд і виїзд, і інколи буває так, що їх є більше ніж одна пара. Для того, щоб була можливість створювати декілька в’їздів і виїздів, а також відображати стіни і глухі зони паркінгу, було створена секція, зображена на рисунку 3.6.

Рисунок 3.6 – Додаткові знаки розмітки схеми паркінгу

Для реалізації збереження даних про паркінг і завантаження вже створеної схеми, була створена секція форми, показана на рисунку 3.7.

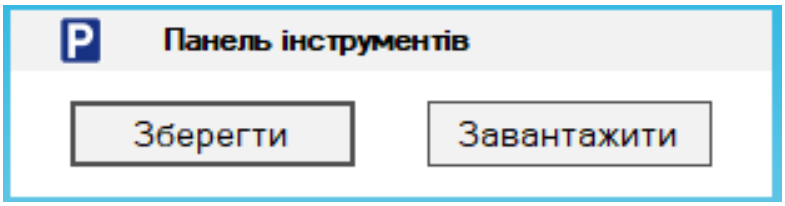

Рисунок 3.7 – Панель інструментів

Для того, щоб почати створювати схему паркінгу, необхідно вибрати потрібний елемент з вище описаних і перемістити його на робочу область, яка представлена у вигляді таблиці (рисунок 3.8).

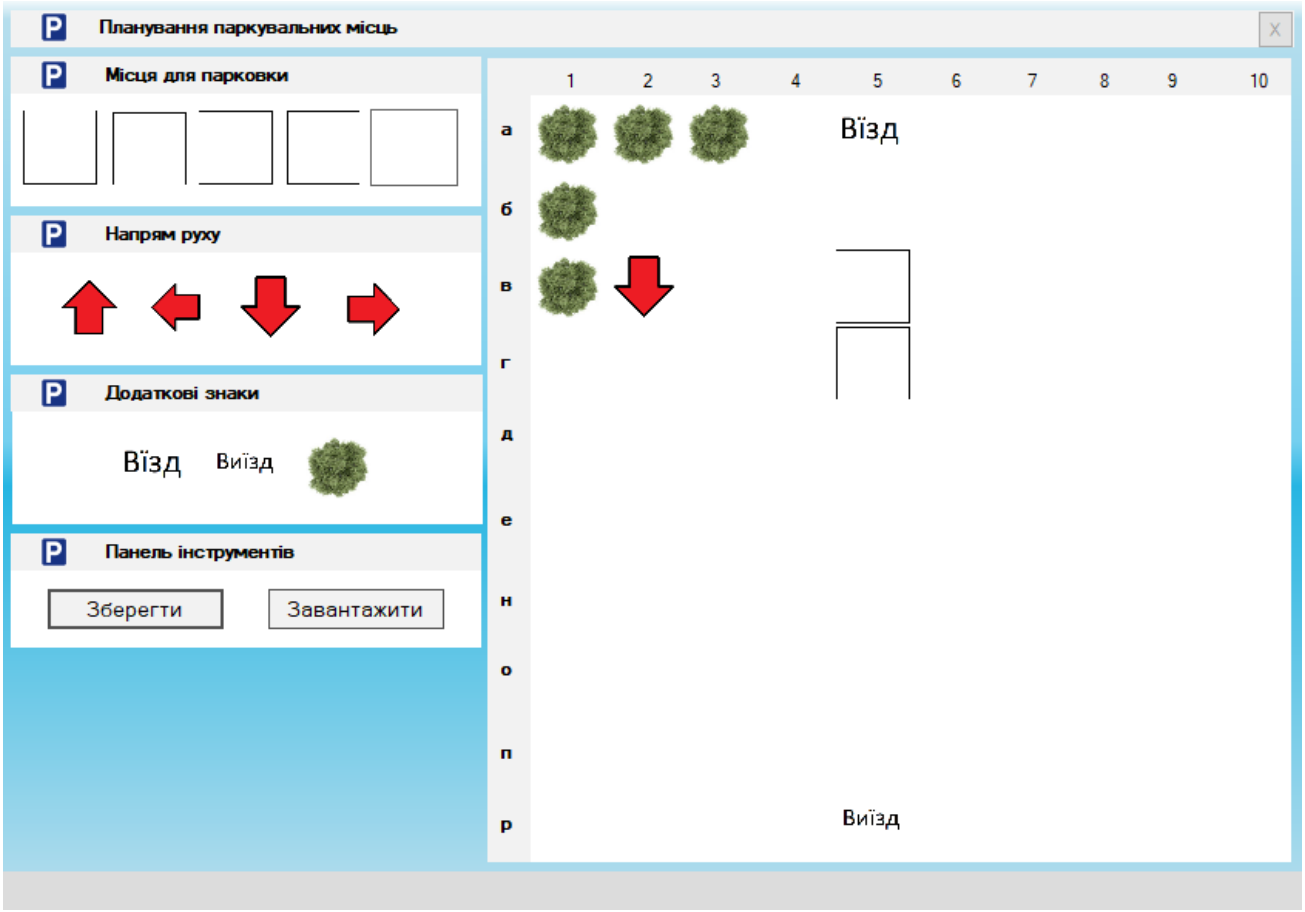

Рисунок 3.8– Розміщення елементів схеми паркінгу на робочу область

Робоча область програми реалізована у вигляді набору компонентів PictureBox, а розміщення елементів – за допомогою подій DragAndDrop.

У лістингу 3.4 подано методи для завантаження даних у головну форму.

Лістинг 3.4 – Методи для завантаження даних у головну форму

```
private void Cont_Click(object sender, EventArgs e)
        {   timer1.Start();
            label13.Visible = true;
            progressBar1.Visible = true;
            label9.Visible = false;
            label10.Visible = false;
            label11.Visible = false;
            label12.Visible = false;
            groupBox1.Visible = false;
            Cont.Visible = false;
            panel8.BackColor = Color.Red;
            panel7.BackColor = Color.Red;
            Zmina.Visible = false;        }
        float timerTick = 0;
        private void timer1_Tik(object sender, EvenArg e)
        {   timerTick += timer1.Interval / 20;
```

Продовження лістингу 3.4

```
if (timerTick <= 100)
    progressBar1.Value = Convert.ToInt32(timerTick);
if (progressBar1.Value == 100)
{   timer1.Stop();
    DBwork fs = new DBwork(locationSource);
    fs.Show();  this.Hide();
    ViewXml fo = new ViewXml();
    fo.Close();
    label9.Visible = false;
    label10.Visible = false;
    label11.Visible = false;
    label12.Visible = false;
    label13.Visible = false;
    groupBox1.Visible = false;
    Cont.Visible = false;
    progressBar1.Visible = false;
    LoadXml.Visible = false;
    Cont.Visible = false;
    panel7.Visible = false;
    panel8.Visible = false;
    panel4.Visible = false;
    panel6.Visible = false;
    panel5.Visible = false;
    panel7.Visible = false;
    label2.Visible = false;
    label3.Visible = false;
```

Переміщаючи елементи схеми паркінгу, можна створити повноцінну архітектуру, яка буде представлена у вигляді XML-файлу. На рисунку 3.9 показано готову схему паркінгу.

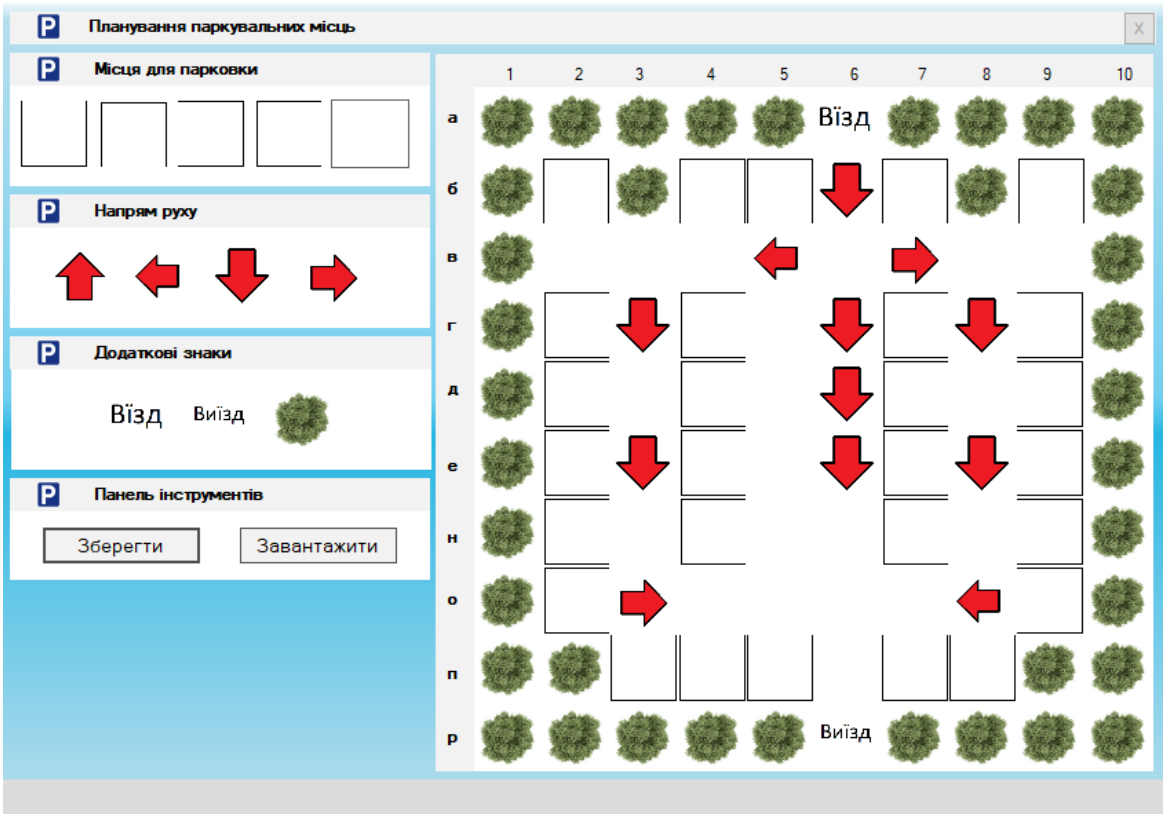

Рисунок 3.9 – Готова схема паркінгу

Код, який відповідає за збереження даних схеми, показаний у лістингу 3.5.

Лістинг 3.5 – Збереження схеми паркінгу

```
privatevoidsaveToXml_Click(object sender, EventArgs e)
        {
XmlDocumentdoc ╞ newXmlDocument();
doc.LoadXml("<item1></item1>");
for (int i = 0; i < 100; i++)
        {
XmlElement newElem1 = doc.CreateElement("pictureBox1");
                newElem1.InnerText
Convert.ToString(matrixEvent[i]);
doc.DocumentElement.AppendChild(newElem1);
         }
doc.PreserveWhitespace = true;
doc.Save("data1.xml");
        }
```

Алгоритм збереження полягає у створенні XML-документу, в який через цикл, записуються дані про компоненти PictureBox, в яких наявні елементи схеми паркінгу. Після чого, дані зберігається у файл data.xml, який знаходиться в кореневій папці додатку

3.3 Реалізація частини сервісу для ведення обліку паркінгових місць

Друга частина системи полягає у реалізації відображення даних про клієнтів паркінгу. Вона також поділяється на наступні: завантаження схеми, створеної у попередній частині, візуальне відображення результату, показ форми про клієнтів.

Запустивши додаток, на екрані буде показане її головне вікно (рисунок 3.10).

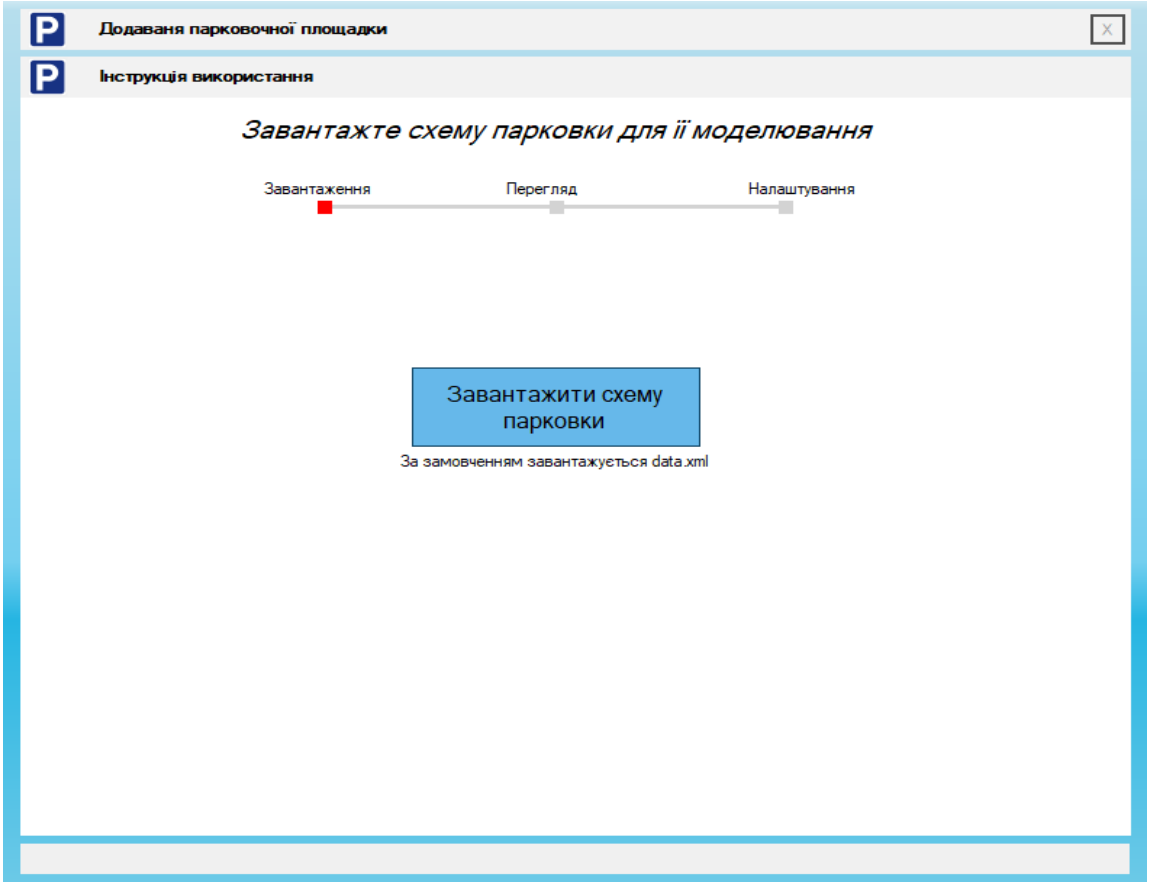

Рисунок 3.10 – Головне вікно програми

Як видно з рисунку, дана частина форми відповідає за завантаження створеного раніше XML-файлу, який містить в собі інформацію про паркінг. Для того, щоб завантажити файл, необхідно натиснути на кнопку «Завантажити схему парковки». Також, під кнопкою є підказка, яка каже, що за замовчуванням, буде завантажуватися файл, який знаходиться у кореневій папці програми для створення схем.

Програмний код, який відповідає за завантаження схеми, показаний у лістингу 3.8.

Лістинг 3.8 – Код завантаження схеми паркінгу

```
privatevoidbutton1_Click(object sender, EventArgs e)
        {
XmlDocumentxmlDoc = newXmlDocument();
xmlDoc.Load("data1.xml");
intcount = 0, countExit = 0, countEnter = 0;
for (int i = 0; i < 100; i++)
{
XmlNodeListlistElement = xmlDoc.GetElementsByTagName("pictureBox");
stringtextGetFromXML = listElement[i].InnerText;
if (textGetFromXML == "bot1.PNG" ||
textGetFromXML == "left1.PNG" ||
textGetFromXML == "right1.PNG" ||
textGetFromXML == "top1.PNG")
{
count++;
                    button1.Visible = false;
                    label7.Visible = false;
                    panel5.BackColor = Color.Red;
                    panel6.BackColor = Color.Red;
                    label9.Visible = true;
                    label10.Visible = true;
                    label11.Visible = true;
                    label12.Visible = true;
                    groupBox1.Visible = true;
                    button3.Visible = true;
                }
if (textGetFromXML == "toEnter.PNG")
{
countEnter++;
                }
if (textGetFromXML == "toExit.PNG")
                {
countExit++;
                }
            }
            label10.Text = "Кількість парковочних місць: " + count;
            label11.Text = "Кількість виїздів: " + 1;
                 label12.Text = "Кількість виїздів: " + 1;
```

Після натискання на кнопку завантаження схеми, буде показана частина форми, зображена на рисунку 3.11.

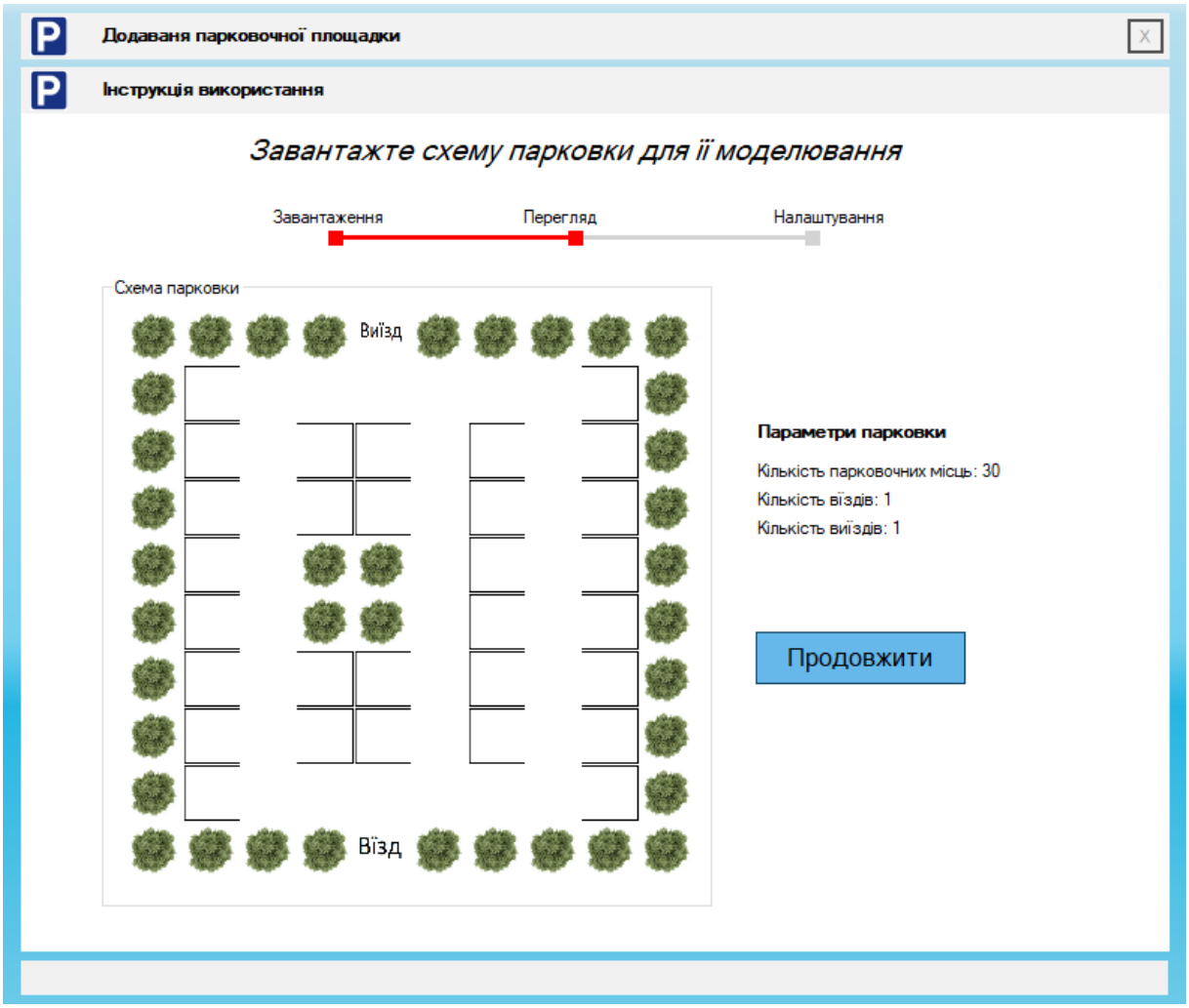

Рисунок 3.11 – Перегляд створеного паркінгу

Як видно з форми, в її лівій частині показано паркінг, який було створено в іншій, призначеній для цього, програмі. А в правій частині показано інформацію про неї, яка включає в себе кількість паркувальних місць, кількість в'їздів і виїздів.

Для того, щоб перейти до налаштування самого паркінгу, необхідно натиснути на кнопку «Продовжити», після чого система повідомить користувачу, що йде завантаження необхідних даних їх налаштування (рисунок 3.12).

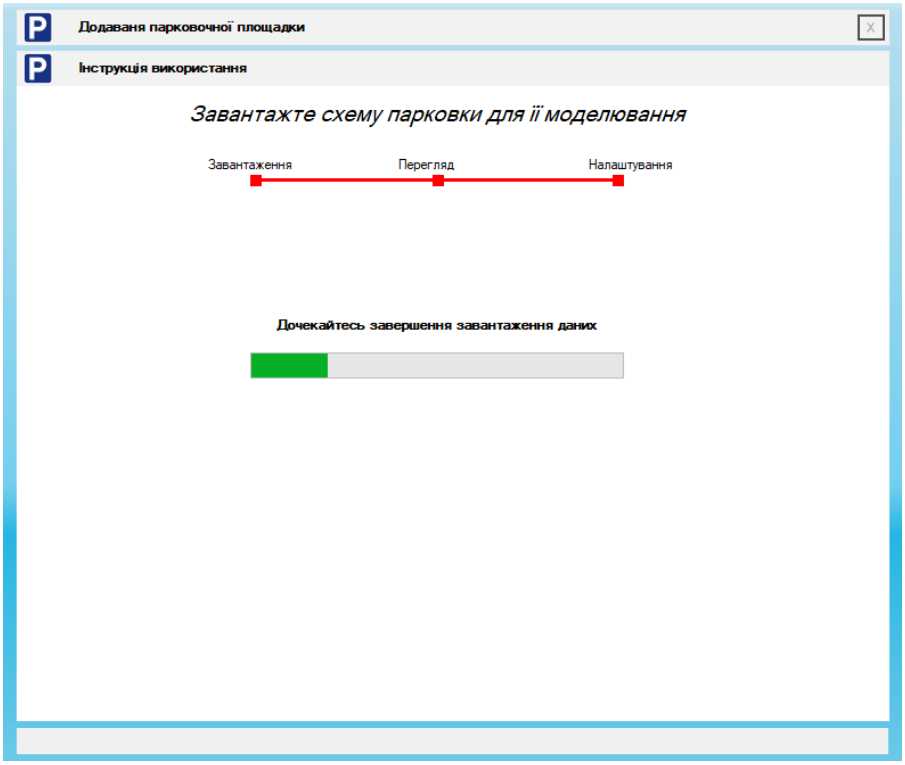

Рисунок 3.12 – Завантаження даних паркінгу

Код, який відповідає за завантаження усіх необхідних даних, показаний у лістингу 3.9.

Лістинг 3.9 – Код процесу завантаження даних

```
privatevoidbutton3_Click(object sender, EventArgs e)
        {
            timer1.Start();
            label13.Visible = true;
            progressBar1.Visible = true;
            label9.Visible = false;
            label10.Visible = false;
            label11.Visible = false;
            label12.Visible = false;
            groupBox1.Visible = false;
            button3.Visible = false;
            panel8.BackColor = Color.Red;
            panel7.BackColor = Color.Red;     }
floattimerTick = 0;
privatevoidtimer1_Tick(object sender, EventArgs e)
        {timerTick += timer1.Interval / 20;
if (timerTick<= 100)
            progressBar1.Value = Convert.ToInt32(timerTick);
if (progressBar1.Value == 100)
            {         timer1.Stop();
                Form2 fs = new Form2();
fs.Show();
this.Hide();
```

Продовження лістингу 3.9

```
                Form1 fo = new Form1();
fo.Close();
                label9.Visible = false;
                label10.Visible = false;
                label11.Visible = false;
                label12.Visible = false;
                label13.Visible = false;
                groupBox1.Visible = false;
                button3.Visible = false;
                progressBar1.Visible = false;
                button1.Visible = false;
                button3.Visible = false;
                panel7.Visible = false;
                panel8.Visible = false;
                panel4.Visible = false;
                panel6.Visible = false;
                panel5.Visible = false;
                panel7.Visible = false;
                label2.Visible = false;
                label3.Visible = false;
                label4.Visible = false;
                label6.Visible = false;   }}
                                                    }
```

Після завантаження усіх необхідних даних, буде показана форма, яка відповідає за ведення обліку паркінгових місць (рисунок 3.12).

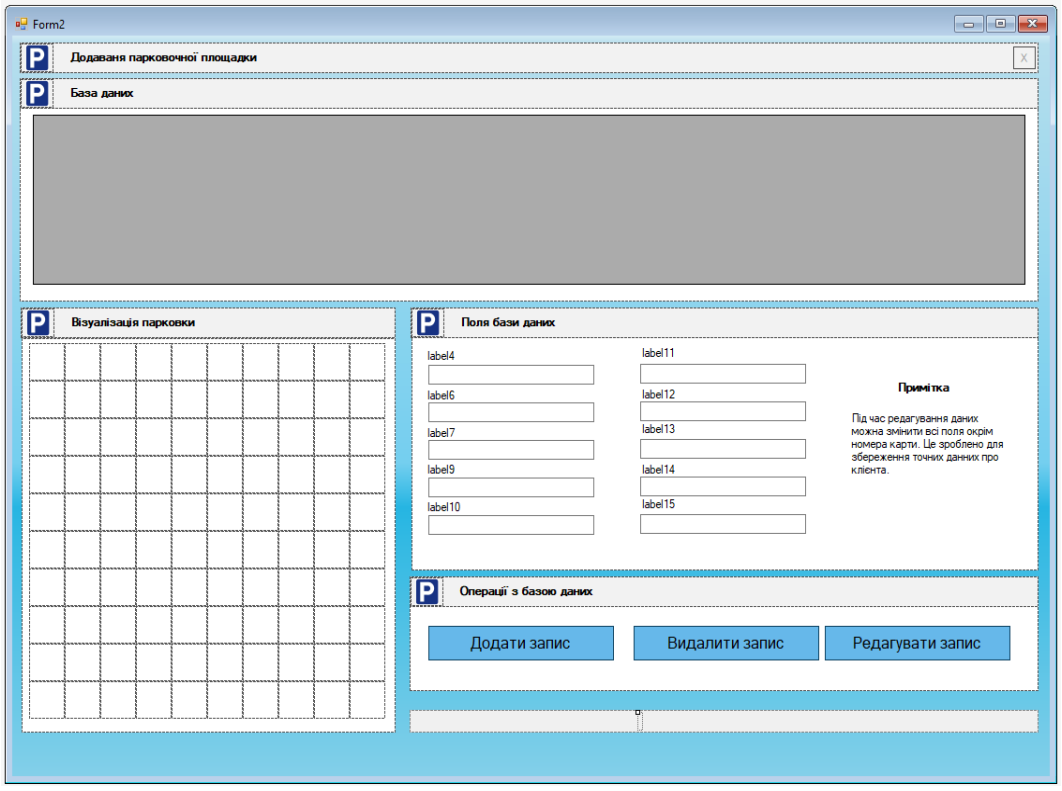

Рисунок 3.12 – Форма обліку паркінгових місць

3.4 Висновки до третього розділу

У третього розділі дипломної роботи наведено процес створення проекту у MS Visual Studio. Також показано програмна реалізація частин сервісу для створення схеми паркінгу та для ведення обліку паркінгових місць. Наведено окремі лістинги для різних прогрманих методів.

4 СПЕЦІАЛЬНА ЧАСТИНА

4.1 Мова програмування C#

C# живе за принципом «будь-яка сутність є об'єкт». Її зараховують до об'єктно-орієнтованої, а точніше об'єктної, мови програмування. «Мова заснована на строгій компонентній архітектурі і реалізує передові механізми забезпечення безпеки коду» – так прийнято характеризувати її. Прихильники C# називають її найбільш мультипарадигменною, універсальною, сучасною і зручною у використанні мовою програмування. Враховуючи той факт, що за нею стоїть платформа Microsoft .NET, число таких прихильників досить велике. [8]

Далекі предки C# з'явилися ще в 60-х роках. Все почалося з появи мови B, яка у 1969 році була створений колективом розробників з Технологічного інституту Масачусетсу (MIT). Головним автором B є Кен Томпсон. Тоді команда працювала над операційною системою UNIX. Мова PL/I, яка застосовувалася у той час для мейнфреймів виробництва компанії IBM, була досить громіздкою і менше підходила для поставленої задачі. Тому вчені вирішили створити нову мову, яка і отримав назву B. Вона є типовим представником ранніх імперативних мов програмування.

Після B, пішла C, яка була винайдена в 1972 році. Основою для нової мови служила саме B. Творцями C були Кен Томпсон і Денис Рітчі, які працювали в дослідницькій лабораторії компанії AT&T (AT&T Bell Telephone Laboratories). У 1971 році Рітчі почав створювати розширену версію B. Спочатку він назвав її NB (New B), але коли мова стала сильно відрізнятися від B, назву змінили на C. B розширилася за рахунок явного використання типів, структур та низки нових операцій.

У 1984 році Б'ярні Страуструп (Bell Labs) виступив з проектом мови C++. Спочатку вона отримала назву «Сі з класами» (C with classes). Назву «C++» придумав Рік Месчитти. "++" — це оператор інкремента в C, що як би натякає на те, що мова C++, щось більше, ніж просто C.

Microsoft вирішила відзначити Міленіум випуском нових програмних продуктів. До 2000 року компанія підготувала промислові версії нових

компонентних технологій і рішень в області обміну повідомленнями і даними, а також створення Internet-додатків (COM+, ASP+, ADO+, SOAP, Biztalk Framework). У підтримку цих нововведень Microsoft випустила інструментарій для розробки додатків – платформу .NET. Вона також об'єднувала під одним дахом кілька мов програмування, що було в новинку для того часу. Ще одним нововведенням платформи .NET була технологія активних серверних сторінок ASP.NET (Active Server Page). З її допомогою можна було відносно швидко розробити веб-додатків, взаємодіючих з БД.

Мова програмування C# була створена спеціально для ASP.NET. На C# повністю була написана і сама ASP.NET. Назва «Сі шарп» (від англ. sharp — дієз) несе «сакральний» сенс. Знак «#» (музичної нотації читається як «дієз») означає підвищення висоти звуку на півтону. З іншого боку, назва «C#» виходить шляхом наступної «еволюційної ланцюга»: C → C++ → C++++(C#), так як символ «#» можна скласти з 4-х знаків «+».

Внаслідок технічних обмежень на відображення (стандартні шрифти, браузери тощо) і того, що знак дієз ♯ не представлений на стандартній клавіатурі, знак # був обраний для представлення знака дієз при запису імені мови програмування. Ця угода відображена в Специфікації Мови C# ECMA-334. Назви мов програмування не прийнято переводити, тому мову слід називати англійською «Сі шарп».

Авторами цієї мови програмування стали Скотт Вілтамут і Андерс Хейлсберг — творець Турбо Паскаля і Дельфи, який перейшов в 1996 році в Microsoft.

C# підтримує всі три «стовпи» об'єктно-орієнтованого програмування: інкапсуляцію, успадкування і поліморфізм. Крім того, в ній була реалізована автоматична «збірка сміття», обробки винятків, динамічне зв'язування. C# створювалася як мова компонентного програмування, і в цьому одне з головних переваг мови, спрямоване на можливість повторного використання створених компонентів. З інших об'єктивних факторів відзначимо наступні: C# створювався паралельно з каркасом FrameworkNet і повною мірою враховує всі його можливості - як FCL, так й CLR; C# є повністю об'єктно-орієнтованою мовою, де навіть типи, вбудовані в мову, представлені класами; C# є потужною об'єктною

мовою з можливостями спадкування й універсалізації; C# є спадкоємцем мов C/C++, зберігаючи кращі риси цих популярних мов програмування. C# безпосередньо пов'язана із C, C++ і Java. Ці три мови - найпопулярніші і найулюбленіші мови програмування в світі. Більш того, майже всі професійні програмісти сьогодні знають C і C++, і більшість знає Java. Оскільки C# побудована на міцному, зрозумілому фундаменті, то перехід від цих "фундаментальних" мов до "надбудови" відбувається без особливих зусиль з боку програмістів. Оскільки Андерс Хейлсберг не збирався винаходити нову мову, він зосередився на введенні удосконалень.

Прамовою C# є мова C. Від C мова C# успадкувала синтаксис, багато ключових слів і оператори. Крім того, C# побудована на покращеній об'єктній моделі, визначеній в C++.

4.2 Розширена мова розмітки XML

XML— спеціальна мова, призначена для опису ієрархічних структур даних в World Wide Web. Розроблена робочою групою W3C в 1996 р.; безсумнівно, входить до найбільш перспективних технологій WWW, чим пояснюється інтерес, який приділяється ій і корпораціями-розробниками, і широкою публікою [9]. Це мова розмітки документів, призначена для зберігання структурованих даних, обміну інформацією між програмами, а також для створення на її основі спеціалізованих похідних мов. XML є набором символів або послідовностей, які вставляються в текст, таким чином, текстовий документ, розмічений з допомогою XML, містить не тільки сам текст, а й додаткову інформацію про його структуру.

Розмітка може бути розділена на стилістичну розмітку, структурну та семантичну: для опису структури документа використовується структурна розмітка, для опису логіки - семантична, а за зовнішній вигляд відповідає стилістична розмітка. XML-документ зазвичай складається з процесингових інструкцій, елементів, атрибутів, сутностей і коментарів [10].

Процес обробки XML-документа полягає в наступному. Його текст аналізується спеціальною програмою, яка називається XML-процесором. XML-процесор нічого не знає про семантику даних у документі, він тільки виробляє

синтаксичний розбір (parsing) тексту документа і перевіряє його правильність з точки зору правил XML. Якщо документ правильно оформлений (well-formed), то результати розбору тексту передаються XML-процесором прикладній програмі, яка виконує їх змістовну обробку, якщо ж документ оформлений невірно, тобто містить синтаксичні помилки, то XML-процесор повинен повідомити про них користувачеві . HTML не висловлює змісту документів.

Мова HTML була створена для опису структури документів (назва, заголовки, списки, абзаци і т. п.) і, в деякій мірі, правил їх відображення (напівжирний шрифт, курсивний шрифт і т. п.). Вона ні в якій мірі не призначена для опису сенсу написаних на ньому документів, а в багатьох випадках саме дані складають тіло документа, будь-то біржовому зведенні чи наукова публікація. Тому з’явилася необхідність в мові опису даних, причому даних, організованих в ієрархічні структури. HTML громіздкий і негнучкий. За останні роки HTML перетворився на накопичення тегів, які часто дублюють один одного і аж ніяк не вносять ясності в текст документа. Якщо додати сюди ще і нестандартні розширення HTML, якими грішать всі розробники браузерів, то створення малих складних HTML-документів стає серйозним завданням. З іншого боку, раз і назавжди зафіксований набір тегів часто виявляється недостатньо гнучким для вираження потрібного нам змісту.

Можна виділити наступні сильні сторони XML:

– XML є потужною метамовою, надає легко використовуваний механізм, з допомогою якого можуть бути розроблені інші мови розмітки для спеціалізованих потреб або бізнес-доменів (CML, VoxML, VML);

– XML є «легко мовою, яка легко читається». При розгляді XML документа можна легко зрозуміти структуру і дані, що містяться в цьому документі, і, завдяки цьому, можна більш легко керувати вмістом;

– поділ змісту і формату представлення. На відміну від HTML, XML чітко розмежовує міжсмисловий зміст документа і подання документа;

– загальний відкритий стандарт. Фактично XML є стандартом комп'ютерної індустрії, який знайшов широке визнання і застосування в різних галузях.

З усього вище перерахованого можна зробити висновок, що у XML є особливі переваги, які мають велике значення для інтеграції додатків

підприємства. Однак варто звернути увагу на обмеження XML при інтеграції додатків:

– обмеження семантичної розмітки. XML надає можливість створювати спеціалізовані теги, що описують конкретний об'єкт, однак семантична розмітка дозволяє виходити за межі визначених об'єктів - при інтеграції між системами повинні бути явним чином визначені значення тегів і їх наповнення;

– відсутнє перетворення даних об'єктів. XML породив безліч бізнес-специфічних форматів обміну, наприклад, такі як, OFX або RosettaNet. Однак при інтеграції різних додатків компанії XML є просто одним з багатьох існуючих форматів даних, тому необхідність перетворення даних як і раніше залишається основною проблемою.

3.3 СУБД MySQL

БД – це сукупність взаємопов’язаних (звичайно складноструктурованих) даних, яку можна спільно використовувати та керування якою здійснюється централізовано. Основні властивості БД:

– допущення для даних такої мінімальної надлишковості, яка сприяє їхньому оптимальному використанню в кількох програмних застосуваннях;

– незалежність даних від програм;

– наявність засобів для підтримки цілісності бази даних та захисту від неавторизованого доступу.

СУБД – це програмне забезпечення для ефективного, зручного і безпечного зберігання даних у БД, організації пошуку в ній та виведення даних на вимогу користувачів.

Найближчим попередником, що мав неабиякий вплив на появу й подальше формування баз даних, були файлові системи. Саме тому розглядати можливості, достоїнства й характерні особливості баз даних найлегше, порівнюючи їх з файловими системами.

Функціонування складної структури на зразок СУБД потребує наявності певних служб, відповідальних за підтримання експлуатаційних характеристик баз даних. Чи не найважливішою з них є служба адміністратора бази даних, функції

якої перелічено нижче:

– проектування й розробка описів даних на різних рівнях та їхнє узгодження;

– розробка структур зберігання і стратегій доступу до даних згідно з вимогами ефективного (чи економічного) зберігання, швидкої обробки та виведення даних за бажанням користувача;

– реструктуризація та реорганізація БД у випадку зміни вимог до характеристик зберігання й обробки даних;

– проектування та застосування механізмів захисту даних;

– реєстрація користувачів (імен, паролів), визначення їхніх прав доступу та повноважень;

– розробка й використання механізмів резервного копіювання даних та їхнє відновлення під час перебоїв;

– налаштування роботи БД (підвищення продуктивності механізмів обробки даних, підтримання планованої надлишковості, забезпечення ефективності їхнього зберігання).

В якості СУБД на даному етапі розвитку додатку була обрана MySQL. MySQL Server БД, реалізує підхід «клієнт-сервер»; багатопотоковість, підтримка декількох одночасних запитів; оптимізація зв’язків з приєднанням багатьох даних за один прохід; записи фіксованої і змінної довжини. MySQL – один з найпопулярніших в світі серверів БД з відкритим вихідним кодом. поєднує в собі швидкість, компактність, стабільність і портованість. MySQL – це популярна СКБД, дуже часто застосовується в поєднанні з PHP. [11]

БД являє собою структуровану сукупність даних. Ці дані можуть бути будь-якими – від простого списку майбутніх покупок до переліку експонатів картинної галереї або величезної кількості інформації в корпоративній мережі. Для запису, вибірки і обробки даних, що зберігаються комп’ютерній БД, необхідна СКБД, якою і є ПЗ MySQL. Оскільки комп’ютери чудово справляються з обробкою великих обсягів даних, управління БД відіграє центральну роль в обчисленнях. Реалізовано таке управління може бути по-різному – як у вигляді окремих утиліт, так і у вигляді коду, що входить до складу інших додатків. У реляційній БД дані зберігаються не всі разом, а в окремих таблицях, завдяки чому досягається виграш

в швидкості і гнучкості. Таблиці зв'язуються між собою за допомогою відношень, завдяки чому забезпечується можливість об'єднувати при виконанні запиту дані з декількох таблиць. SQL, як частина системи MySQL, може бути охарактеризована як мова структурованих запитів плюс найбільш поширений стандартний мови, який використовується для доступу до БД. MySQL – це ПЗ з відкритим кодом. Застосовувати його і модифікувати може будь-хто. Таке ПЗ можна одержувати через Internet використовувати безкоштовно. При цьому кожен користувач може вивчити вихідний код і змінити його відповідно до своїх потреб. Використання ПЗ MySQL регламентується ліцензією GPL (GNU General Public License), http://www.gnu.org/licenses/, в якій зазначено, що можна і чого не можна робити з цим ПЗ в різних ситуаціях.

MySQL є дуже швидким, надійним і легким у використанні ПЗ. MySQL володіє також рядом зручних можливостей, розроблених в тісному контакті з користувачами. Спочатку сервер MySQL розроблявся для управління великими БД метою забезпечити більш високу швидкість роботи в порівнянні з існуючими на той момент аналогами. І ось уже протягом кількох років даний сервер успішно використовується в умовах промислової експлуатації з високими вимогами. Незважаючи на те що MySQL постійно вдосконалюється, він уже сьогодні забезпечує широкий спектр корисних функцій. Завдяки своїй доступності, швидкості і безпеці MySQL дуже добре підходить для доступу до БД по Internet.

MySQL є системою клієнт-сервер, яка містить багато SQL-серверів, що забезпечує підтримку різних обчислювальних машин БД, а також кілька різних клієнтських програм і бібліотек, засоби адміністрування і широкий спектр програмних інтерфейсів (API). Доступна також велика кількість ПЗ для MySQL, в більшій частині – безкоштовного.

Сервер MySQL постійно працює на комп'ютері. Клієнтські програми (наприклад, скрипти PHP) посилають до сервера MySQL SQL-запити через механізм сокетів (тобто за допомогою мережевих засобів), сервер їх обробляє і запам'ятовує результат. Тобто скрипт (клієнт) вказує, яку інформацію він хоче отримати від сервера БД. Потім сервер БД посилає відповідь (результат) клієнту (скрипту).

Структура MySQL трирівнева: БД – таблиці – записи. БД таблиці MySQL

фізично представляються файлами з розширеннями frm, MYD, MYI. Логічно – таблиця являє собою сукупність записів. А записи – це сукупність полів різного типу. Ім’я БД MySQL унікальне в межах системи, а таблиці – в межах БД, поля – в межах таблиці. Один сервер

MySQL може підтримувати відразу кілька БД, доступ до яких може розмежовуватися логіном і паролем. Знаючи ці логін і пароль, можна працювати з конкретною БД. Наприклад, можна створити або видалити в ній таблицю, додати записи і т.д. Зазвичай ім’я-ідентифікатор і пароль призначаються хостинг-провайдерами, які і забезпечують підтримку MySQL для своїх користувачів.

3.4 Висновки до четвертого розділу

У четвертому розділі дипломної роботи наведено основні можливості програмного забезпечення, яке використовується для створення сервісу, зокрема мови програмування C#, мови розмітки документів XML та СКБД MySQL.

5 ОБҐРУНТУВАННЯ ЕКОНОМІЧНОЇ ЕФЕКТИВНОСТІ

Метою дипломної роботи є створення сервісу для адміністрування і обліку роботи автомобільної парковки.

Головною метою розділу є обґрунтування економічної ефективності впровадженої даної розробки.

Щоб виконати оцінку економічної ефективності необхідно розрахувати трудомісткість реалізації проекту, витрати на оплату праці найманим працівникам, витрати апаратного і програмного забезпечення, амортизаційні відрахування, витрати енергоресурсів та інші витрати які є основними пунктами виконання обчислень, а також показники економічної ефективності розробки проекту.

5.1 Розрахунок норм часу на виконання науково-дослідної роботи

Реалізація проекту створення сервісу, який призначений для адміністрування і обліку роботи автомобільної парковки складається з низки послідовних та взаємопов’язаних етапів.

Кожен із етапів реалізації проекту характеризується метою та змістом, оцінкою часу виконання, кількістю та спеціалізацією виконавців, а також приблизною оцінкою вартості.

Реалізація проекту створення сервісу складається із підготовчого етапу, етапу технічної пропозиції, створення технічного завдання, проектування системи, практичної реалізації, тестування, верифікації та заключного етапу.

Норми часу на виконання науково-дослідницької роботи розраховуватимуться на основі середнього часу виконання стадії в годинах, що наведені в таблиці 5.1 разом із інформацією про виконавців і сумарною кількості затраченого часу.

Таблиця 5.1 – Операції технологічного процесу та їх час виконання

№ п/п	Назва операції (стадії)	Виконавець	Середній час виконання операції, год.
1	Підготовча стадія	Проектний менеджер	10
		Інженер-програміст	
2	Технічна пропозиція	Проектний менеджер	10
		Інженер-програміст	
3	Створення технічного завдання	Проектний менеджер	20
		Інженер-програміст	
4	Проектування системи	Інженер-програміст	20
5	Практична реалізація	Інженер-програміст	135
6	Тестування системи	Тестувальник	20
7	Верифікація системи	Тестувальник	20
		Інженер-програміст	
		Проектний менеджер	
8	Створення документації	Інженер-програміст	20
9	Заключна стадія	Проектний менеджер	10
Разом			265

В підсумку на реалізацію проекту сервісу, який призначений для адміністрування і обліку роботи автомобільної парковки необхідно 265 людино-годин, залучення трьох спеціалістів та виконання дев’яти різноманітних стадій реалізації проекту.

5.2 Визначення витрат на оплату праці та відрахувань на соціальні заходи

Визначення витрат на оплату праці та відрахувань на соціальні заходи прямо залежить від кількості витраченого працівниками часу на роботу, ставки в годину чи місяць, кількість відрахувань на соціальні заходи встановлених в законному порядку на час розрахунку.

В результаті розрахунку потрібно визначити основну та додаткову заробітну плату, витрати на соціальні заходи та на основі цих даних визначити

сумарні витрати на оплату праці. Основна заробітна плата нараховується за виконану роботу за тарифними ставками, відрядними розцінками чи посадовими окладами. Додаткова заробітна плата – це складова заробітної плати працівників, до якої включають витрати на оплату праці, не пов'язані з виплатами за фактично відпрацьований час.

При розрахунку заробітної плати кількість робочих днів у місяці слід в середньому приймати – 24,5 дні/міс., або ж 196 год./міс. (тривалість робочого дня – 8 год.).

Наймані працівники для розробки сервісу, який призначений для адміністрування і обліку роботи автомобільної парковки працюють згідно контракту, який в якому вказано їхню погодинну ставку. Тобто розрахунок заробітної плати працівників відбуватиметься на базі тарифної ставки та кількості відпрацьованих годин.

У штаті найманих працівників для розробки сервісу залучено проектного менеджера, інженера-програміста і тестувальника.

Тарифні ставки учасників процесу розробки сервісу, який призначений для адміністрування і обліку роботи автомобільної парковки:

– Проектний менеджер – 150 грн./год.

– Інженер-програміст – 130 грн./год.

– Тестувальник – 100 грн./год.

Основна заробітна плата розраховується за формулою 5.1:

$$З_{осн.} = Т_с \cdot К_г, \qquad (5.1)$$

де $Т_с$ – тарифна ставка, грн.; $К_г$ – кількість відпрацьованих годин.

Оскільки всі види робіт в виконує три спеціаліста, то основна заробітна плата буде розраховуватись за даною формулою 5.1;

$$З_{осн.} = 150 \cdot 35 + 130 \cdot 200 + 100 \cdot 30 = 34250 \text{ грн.}$$

Додаткова заробітна плата становить 10–15 % від суми основної заробітної плати й визначається за формулою 5.2.

Коефіцієнт додаткових виплат працівникам становить 0,1.

$$Здод. = Зосн. \cdot Кдопл., \quad (5.2)$$

де $К_{допл}$ – коефіцієнт додаткових виплат працівникам

$$З_{дод} = 34250 \cdot 0,1 = 3425 \text{ грн.}$$

Звідси загальні витрати на оплату праці (фонд заробітної плати) визначаються за формулою 5.3:

$$Во.п. = Зосн. + Здод. \quad (5.3)$$

$$В_{о.п.} = 34250 + 3425 = 37675 \text{ грн.}$$

З цієї суми утримуються обов'язкові відрахування на заробітну плату:

– Єдиний соціальний внесок (ЄСВ), що становить 22%;

– Військовий збір (ВЗ), що становить 1,5%;

Сума відрахувань становить 23,5% від фонду оплати праці та визначається за формулою 5.4:

$$В_{с.з.} = Ф_{оп} \cdot 0,235 \quad (5.4)$$

де $Ф_{оп}$ – фонд оплати праці, грн.

$$В_{с.з.} = 37675 \cdot 0,235 = 8853,62$$

Усі витрати обчислюються детально наведені в таблиці 5.2 та обчислюються за формулою 5.5:

$$Взп = ФЗП + ФОП \quad (5.5)$$

$$В_{зп} = 34250 + 8853,62 = 43103,62 \text{ грн.}$$

Таблиця 5.2 – Розрахунки витрат на оплату праці

з/п	Категорія працівників	Основна заробітна плата, грн.			Додаткова заробітна плата, грн.	Нарахув. на ФОП, грн.	Всього витрати на плату праці, грн. (6=3+4+5)
		Тарифна ставка, грн.	Кількість відпрацьова них год.	Фактично нарах. з/пл., грн.			
А	Б	1	2	3	4	5	6
1.	Проектний менеджер	150	35	5250	525	-	-
2.	Інженер-програміст	130	200	26000	2600	-	-
3.	Тестувальник	100	30	3000	300	-	-
Разом		380	265	34250	3425	8853,62	43103,62

Опираючись на розрахунки витрат на оплату та таблицю результатів 5.2 видно, що всього витрати на плату праці становлять 43103,62 грн.

5.3 Розрахунок матеріальних витрат

Матеріальні витрати є невід'ємною частиною розробки сервісу, який призначений для адміністрування і обліку роботи автомобільної парковки та визначаються як добуток кількості витрачених матеріалів та їх ціни за формулою 5.6:

$$M_{ві} = q_i \cdot p_i , \quad (5.6)$$

де: q_i – кількість витраченого матеріалу і–го виду; p_i – ціна матеріалу і–го виду.

Звідси, загальні матеріальні витрати можна визначити за формулою 5.7:

$$З_{м.в.} = \sum M_{ві} . \quad (5.7)$$

Результати проведених розрахунків наведено у таблиці 5.3.

Таблиця 5.3 – Результати розрахунків матеріальних витрат.

№ п/п	Найменування матеріальних ресурсів	Од. виміру	Фактично витрачено матеріалів	Ціна одиниці, грн.	Загальна сума витрат, грн.
1	CD диски	шт.	2	7,45	14,90
2	Папір для друку	листів	500	0,15	75,00
3	Чорнила для принтера	шт.	1	80,00	80,00
Всього					169,90

Згідно проведених розрахунків, матеріальні витрати становлять 169,90 грн.

5.4 Розрахунок витрат на електроенергію

Однією із статей витрат є витрати на електроенергію під час проходження усіх етапів реалізації кінцевого продукту.

Затрати на електроенергію одиниці обладнання визначаються за формулою 5.8:

$$З_в = W \cdot T \cdot S, \qquad (5.8)$$

де W – необхідна потужність, кВт; T – кількість годин на реалізацію розробки; S – вартість кіловат-години електроенергії.

Вартість кіловат-години електроенергії слід приймати згідно існуючих на даний час тарифів. Отже, 1 кВт з ПДВ коштує 2,42 грн.

Потужність комп’ютерів для реалізації кінцевого продукту – 400 Вт, кількість годин роботи обладнання згідно таблиці 5.1 – 265 годин.

Визначимо витрати на електроенергію згідно формули 5.11:

$$З_в = 0{,}4 \cdot 265 \cdot 2{,}42 = 256{,}52 \text{ грн.}$$

Згідно формули затрати на електроенергію становлять 256,52 грн.

5.5 Розрахунок суми амортизаційних відрахувань

Для будь якої діяльності характерною є властивість зношування на зниження якості властивостей інструментарію та фондів за допомогою яких ведеться діяльність. Для вирішення проблеми із відновленням даних фондів використовується амортизація, що являє собою процес трансформації вартості основних фондів на вартість продукції, яка щойно була створена, задля повного відновлення основних фондів.

Для визначення амортизаційних відрахувань використовується формула 5.9:

$$A = \frac{Б_В \cdot Н_А}{100\%} \qquad (5.9)$$

де A – амортизаційні відрахування за звітний період, грн.; $Б_В$ – балансова вартість групи основних фондів на початок звітного періоду, грн.; $Н_А$ – норма амортизації.

Комп'ютери та оргтехніка належать до четвертої групи основних фондів. Для цієї групи річна норма амортизації дорівнює 60 % (квартальна – 15 %).

Річний робочий фонд становитиме 2352 годин, так як робочий день становить 8 годин, а кількість робочих днів в місяці становить 24,5 годин.

Для даної розробки засобом розробки є комп'ютер. Його сума становить 18500 грн. Отже, амортизаційні відрахування будуть рівні:

$$A = 18500 \cdot 5\% / 100\% = 925 \text{ грн.}$$

Згідно проведених обчислень амортизаційні відрахування становлять 925 грн.

5.6 Обчислення накладних витрат

Накладні витрати пов'язані з обслуговуванням виробництва, утриманням апарату управління спілкою та створення необхідних умов праці.

В залежності від організаційно-правової форми діяльності господарюючого суб'єкта, накладні витрати можуть становити 20–60 % від суми основної та додаткової заробітної плати працівників.

$$Н_в = В_{о.п.} \cdot 0{,}2…0{,}6\ ,\qquad (5.10)$$

де $Н_в$ – накладні витрати.

Отже, накладні витрати становлять згідно формули 5.10:

$$Н_в = 37675 \cdot 0{,}2 = 7535 \text{ грн.}$$

Накладні витрати згідно розрахунку формули, становить 7535 грн.

5.7 Складання кошторису витрат та визначення собівартості науково-дослідницької роботи

Результати проведених вище розрахунків наведено у таблиці 5.4.

Таблиця 5.4 – Кошторис витрат на НДР

Зміст витрат	Сума, грн.	В % до загальної суми
Витрати на оплату праці	43103,6	70,8
Відрахування на соціальні заходи	8853,6	14,6
Матеріальні витрати	169,9	0,3
Витрати на електроенергію	256,52	0,4
Амортизаційні відрахування	925	1,5
Накладні витрати	7535	12,4
Собівартість	60843,66	100

Собівартість ($С_в$) програмного продукту розрахуємо за формулою:

$$С_в = В_{о.п.} + В_{с.з.} + З_{м.в.} + З_в + А + Н_в\ .\qquad (5.11)$$

Отже, собівартість програмного продукту дорівнює:

Св = 43103,6+ 8853,6+ 169,90 + 256,52+ 925+ 7535= 60843,66 грн.

Загальний кошторис витрат та визначення собівартості науково-дослідницької роботи становить 60843,66 грн.

5.8 Розрахунок ціни програмного продукту

Ціну науково-дослідної роботи можна визначити можна визначити за формулою:

$$Ц = \frac{C_В \cdot (1 + P_{рен}) + К \cdot B_{н.і.}}{К} \cdot (1 + ПДВ), \qquad (5.12)$$

де $P_{рен.}$ – рівень рентабельності (30 %); $К$ – кількість замовлень, од. (встановлюється лише при розробці програмного продукту та мікропроцесорних систем); $B_{н.і.}$ – вартість носія інформації, грн. (встановлюється лише при розробці програмного продукту); $ПДВ$ – ставка податку на додану вартість (20%).

Оскільки розробка є прикладною, і використовуватиметься тільки для одного підприємства, то для розрахунку ціни не потрібно вказувати коефіцієнти К та $В_{н.і.}$, оскільки їх в даному випадку не потрібно.

Тоді, формула для обчислення ціни розробки буде мати вигляд:

$$Ц = C_В \cdot (1 + P_{рен}) \cdot (1 + ПДВ) \qquad (5.13)$$

Звідси ціна на роботу складе:

$$Ц = 60843,66 \cdot (1 + 0,3) \cdot (1 + 0,2) = 94916,11 \text{ грн.}$$

Загальний розрахунок ціни програмного продукту становить 94916,11 грн.

5.9 Визначення економічної ефективності і терміну окупності капітальних вкладень

Ефективність виробництва – це узагальнене і повне відображення кінцевих результатів використання робочої сили, засобів та предметів праці на підприємстві за певний проміжок часу.

Економічна ефективність (E_p) полягає у відношенні результату виробництва до затрачених ресурсів:

$$E_p = \frac{П}{C_B}, \tag{5.14}$$

де $П$ – прибуток; C_B – собівартість.

Плановий прибуток ($П_{пл}$) знаходимо за формулою:

$$П_{пл} = Ц - C_в \, . \tag{5.15}$$

Розраховуємо плановий прибуток:

$$П_{пл} = 94916{,}11 - 60843{,}66 = 34072{,}45 \text{ грн.}$$

Отже, формула для визначення економічної ефективності набуде вигляду:

$$E_p = \frac{П}{C_B}. \tag{5.16}$$

Тоді,

$$E_p = 34072{,}45 \,/\, 60843{,}66 = 0{,}56.$$

Поряд із економічною ефективністю розраховують термін окупності капітальних вкладень (T_p):

$$T_p = \frac{1}{E_p} \, , \tag{5.17}$$

Термін окупності дорівнює:

$$T_p = 1 / 0{,}5 = 1{,}78 \text{ р.}$$

Згідно формул плановий прибуток від розробки становить 34072,45 грн., економічна ефективність дорівнює 0,56, а термін окупності становить 1,78 роки що вважається доцільним та економічно вигідним.

5.10 Висновки до п'ятого розділу

В організаційно-економічній частині дипломної роботи освітнього рівня «магістр» було розраховано основні техніко-економічні показники створення сервісу, який призначений для адміністрування і обліку роботи автомобільної парковки (див. таблиця 5.5).

Орієнтоване значення економічної ефективності становить 0,56, що є достатньо високим значенням.

Період окупності повинен варіюватися від 1 до 3 років, тоді розвиток вважається доцільним та економічно вигідним. Термін окупності даної роботи становить 1,78 років.

Таблиця 5.5 – Техніко-економічні показники науково-дослідної роботи

№ п/п	Показник	Значення
1	Собівартість, грн.	60843,66
2	Плановий прибуток, грн.	34072,45
3	Ціна, грн.	94916,11
4	Економічна ефективність	0,56
5	Термін окупності, рік	1,78

На основі проведених обрахунків можна зробити висновок, що створення сервісу, який призначений для адміністрування і обліку роботи автомобільної парковки є доцільним у зв'язку з невеликим терміном окупності та великим обсягом планового прибутку.

6 ЕКОЛОГІЯ

6.1 Статистична оцінка техногенних впливів

Така оцінка містить наступні складові: комплексна оцінка техногенного впливу на біосферу; комплексна оцінка території; оцінка впливу на людину. Розглянемо ці складові частини окремо [20].

Комплексна оцінка техногенного впливу на біосферу. Процес незворотного перетворення людиною частин біосфери на техногенні об'єкти і території дістав назву техногенезу, а частина біосфери, штучно перетворена в результаті життєдіяльності людини і заповнена її продуктами, називається технічною оболонкою біосфери (техносферою). Основні завдання цієї оцінки: вивчення техногенних чинників забруднення довкілля; класифікація джерел забруднень; визначення джерел походження забруднень; всі проблемами людства, які породжують забруднення біосфери. Техногенні чинники забруднення довкілля об'єднують у такі групи: атмосферні - хімічне, фізичне, механічне і теплове забруднення; водні - океани і моря, забруднення поверхневих і підземних вод; грунтові - хімічне, ерозійне забруднення, ущільнення, засолення, заболочення тощо; геологічні - негативні екзогенні процеси - зсуви, підтоплення, обвали, абразії берегів тощо; біотичні - деградації екосистем, збіднення біорізноманіття, мутації, зникнення лісів і пасовищ, біогенна акумуляція шкідливих речовин тощо; комплексні - порушення природної структури ландшафтів, поява пустель, деградація земель.

Забруднення класифікують за галузевим принципом: промислові - хімічна промисловість, металургійна, видобувна тощо; транспортні - автотранспорт, авіаційний, морський тощо; енергетичні - теплові і атомні електростанції; сільськогосподарські - засоби захисту рослин, мінеральні та органічні добрива тощо; пов'язані з військовою діяльністю. Вплив техносфери на стан атмосфери: найбільший вплив на стан атмосфери чинять теплоенергетика, металургійна промисловість, підприємства хімічної та будівельної індустрії, автотранспорт, що викидають у повітря пил, важкі метали, вуглеводні, оксиди карбону, бенз(а)пірен та інші речовини; найбільший вплив на хімічний склад атмосферного повітря

чинить спалювання кам'яного вугілля; найпотужнішим негативним техногенним чинником є енергетика - підприємства чорної металургії утворюються пил та оксид сірки, хімічна і нафтохімічна промисловість продукують майже у два рази менше викидів при значно більшій різноманітності забруднюючих речовин; крім газоподібних речовин у повітря потрапляють рідкі і тверді частинки у вигляді аерозолів; серед усіх видів транспорту автомобільний посідає перше місце за кількістю і різноманітністю забруднюючих речовин, а також за кількістю незворотних змін ландшафтів та інших негативних впливів на довкілля. У містах з розвинутою промисловістю внесок автотранспорту в забруднення довкілля досягає 80% усіх забруднень. Проблеми, пов'язані з гідросферою, зумовлені нестачею прісної води для потреб людства, її забрудненням, порушенням природних кругообігів. Найбільшими забрудниками водних ресурсів є промисловість, комунальне і сільське господарства країни, яки в структурі забруднення водних ресурсів України складають стосовно 60, 20 і 17%.

Комплексна оцінка території. Суть такої оцінки полягає в дослідженні просторової структури історично складених природно-територіальних комплексів та проведенні на цій основі розділу території країни на природні зони (області), округи та райони. Основне завдання комплексної оцінки території в конкретних регіонах: виявлення комплексу несприятливих факторів, що складають необхідний вихідний матеріал для прогнозування можливих негативних наслідків господарської діяльності; визначення характеру і масштабів наслідків; виявлення причини на основі встановлення причинно-наслідкових зв'язків; розробці заходів, спрямованих на ліквідацію, попередженні і компенсації цих наслідків. Основною метою комплексної оцінки території є встановлення суспільної значимості наслідків за існуючих масштабів господарського впливу на рівновагу екосистем.

Процедура комплексної оцінки території грунтується на вивченні механізму взаємодії в системі «населення - господарство - природні системи». Ця оцінка спрямована на вивчення: спричиненого діяльністю людини впливу на природні екосистеми регіону; змін у природних екосистемах під впливом цієї діяльності; наслідків впливу змінених природних систем на суспільство і економіку в цілому.

Існують різні підходи до екологічної оцінки територій: економічна оцінка; соціальна оцінка; природно-ресурсна оцінка. Для отримання екологічних оцінок

застосовуються різні моделі: блокові; матричні; картографічні; статистичні. Для деяких видів залежностей відомі моделі, що адекватно їх описують: закон логарифмічно нормального розподілу, закон дифузії та ін. Більшість зв'язків потребує виявлення виду залежностей і значень коефіцієнтів, що часто мають регіональний характер.

Джерелами впливу на природні екосистеми є господарство та населення, які можуть аналізуватися на трьох територіальних рівнях: мікрорівень - окремі виробничі та сільськогосподарські підприємства або їх підрозділи, окремі об'єкти міського господарства або функціональні зони населених пунктів; мезорівень - промислові пункти, центри або вузли, великі сільськогосподарські підприємства, міські та сільські поселення; макрорівень - промислові та сільськогосподарські райони, агломерації, територіально-виробничі комплекси.

Залежно від показника виміру для екологічної оцінки території застосовується ряд кількісних і якісних методів оцінки: якісна бальна оцінка; кількісна оцінка на основі натуральних (абсолютних і відносних) показників; кількісна оцінка на основі вартісних показників.

Оцінка впливу на людину. Види негативного впливу на організм людини умовно можна об'єднати у дві групи: процеси прямого впливу і процеси непрямого впливу. Процеси прямого впливу обумовлені безпосереднім контактом людини з техногенними об'єктами (механізмами, машинами) або робочими агентами цих об'єктів (високою температурою, токсичними речовинами, електричним струмом, електромагнітними полями чи іншими формами енергетичного впливу, активними біологічними організмами, ін.), що можуть завдавати шкоди здоров'ю людини або навіть призводити до її загибелі. Процеси непрямого впливу на організм людини пов'язані з погіршенням умов життя і діяльності людини (склад повітря, температура, вологість, ін.), які зумовлюють процеси метаболізму в організмі людини. Погіршення якості їжі і питної води є однією з найбільш небезпечних форм непрямого впливу.

6.2 Методологічні основи обробки екологічної інформації на базі комп’ютерних технологій

Це комплексне питання містить наступні складові [20]:

— зведення і первинне оброблення статистичних даних (статистичне зведення — це первинне наукове оброблення даних спостереження для характеристики суцільного явища узагальнюючими показниками. Етапи: статистичне групування; підсумовування даних; табличне і графічне оформлення одержаних даних. За допомогою статистичного зведення розв'язують такі завдання: групування даних, розроблення системи показників для характеристики груп і всієї статистичної сукупності, обчислення групових і загальних показників, зведення результатів обчислення у статистичних таблицях. У результаті обробки та систематизації статистичних матеріалів отримуємо ряди цифрових показників, які характеризують окремі сторони явищ, що вивчаються, в просторі або зміну цих явищ у часі. Тому побудова статистичних рядів є основою будь-якого первинного оброблення статистичної інформації);

– статистична оцінка екологічного стану НПС і закономірностей його розподілу (властивістю статистичної сукупності є коливання, мінливість значень будь-якої ознаки, тобто варіація. Вона зумовлена дією безлічі взаємопов’язаних причин, серед яких є основні і другорядні. Основні причини формують центр розподілу, другорядні - варіацію ознак, сукупна їх дія - форму розподілу. Аналіз варіаційного ряду розподілу полягає у виявленні закономірностей зміни частот залежно від зміни кількісної ознаки, яка покладена в основу групування. При аналізі варіаційних рядів найуживанішими є такі групи показників: центра розподілу, розміру варіації, форми розподілу);

– статистичне групування в екології (мета - поділ сукупностей на однорідні типові групи за існуючими для них кількісними ознаками з метою всебічної характеристики їхнього стану, розвитку і взаємодії. Метод статистичних групувань робить статистику могутнім знаряддям соціального пізнання і використовується для вирішення трьох взаємопов'язаних завдань: виділення різних соціально-економічних типів явищ (процесів) та всебічна їх характеристика; дослідження структури масової сукупності; вивчення взаємодії між окремими ознаками сукупності. Суть цього методу полягає у тому, що складне масове явище розглядається не як єдине нероздільне ціле, а в ньому виділяються окремі групи одиниць із статистичними показниками, які дають

кількісну характеристику якісно своєрідній частині одиниць усієї сукупності);

– дисперсійний аналіз в екології (для кількісної оцінки взаємозв'язків і їхньої суттєвості при незначній кількості спостережень застосовується дисперсійний аналіз. Головне призначення дисперсійного аналізу — статистично виявити вплив різних факторів на мінливість ознаки, що вивчається. В результаті дисперсійного аналізу одержуються дані, що характеризують загальне розсіювання, або дисперсію ознаки, обумовлену дією всіх факторів; часткову або факторну дисперсію, викликану впливом організованих і врахованих дослідником факторів; та залишкову дисперсію, пов’язану з невідомими експериментатору, випадковими, неорганізованими факторами. за його допомогою розв'язуються такі завдання: кількісне вимірювання сили впливу факторних ознак та їх сполучень на результативну; визначення вірогідності впливу та його довірчих меж; аналіз окремих середніх та статистична оцінка їх різниці);

– кореляційний аналіз зв’язків в екології (поглибленням дослідження й кількісною оцінкою характеру та механізму взаємодії факторних і результативних ознак є метод аналізу регресії та кореляції, тобто кореляційний аналіз. При кореляційній залежності будь-якому значенню однієї змінної величини може відповідати декілька чи навіть безліч різноманітних, тобто варіюючих значень іншої змінної величини. Розрахунки на основі кореляційних моделей підвищують ступінь точності аналізу, часто виявляють недоліки попереднього аналізу. Перевага цього методу полягає також і в тому, що він дає можливість розв'язувати задачі, які не можна вирішити за допомогою інших методів економічного аналізу. У дослідженнях важливо вивчати не стільки міру кореляції, скільки форму її й характер зміни однієї ознаки залежно від зміни іншої. Ці задачі розв'язуються методами регресійного аналізу. Використання методу кореляції і регресії дозволяє вирішити такі основні завдання: встановити характер і тісноту зв'язку між досліджуваними явищами; визначити і кількісно виміряти ступінь впливу окремих факторів і їх комплексу на рівень досліджуваного явища; на підставі фактичних даних моделі залежності екологічних показників від різних факторів розраховувати кількісні зміни аналізованого явища при прогнозуванні показників і давати об'єктивну оцінку діяльності підприємств);

– статистичний аналіз тенденцій і закономірностей динаміки в екології

(екологічні процеси - явище не статичне, а динамічне, оскільки упродовж певного часу - місяць за місяцем, рік за роком змінюється стан забруднень природних сфер, рівень викидів забруднюючих речовин в навколишнє середовище, об'єм промислових і побутових відходів на звалищах тощо. Дослідження процесів зміни і розвитку явищ у часі відбувається на основі побудови і аналізу рядів динаміки. Для динамічного ряду характерні перелік хронологічних дат або інтервалів часу і конкретні значення відповідних статистичних показників, які називають рівнями ряду. Тому кожен ряд динаміки має елементи двох типів - рівні (цифри, з яких складається ряд) і періоди (дати, яким відповідають рівні ряду);

– індексний метод в екології (статистична практика при вивченні екологічних явищ широко використовує індекси. Знання методології побудови індексів значно розширює аналітичні можливості дослідника, збагачує результативну інформацію досліджень. За допомогою індексів можна характеризувати зміну в часі і просторі найрізноманітніших показників: обсяги викидів в атмосферу, скидів шкідливих речовин у водне середовище, інтенсивність забруднень і т. д. За допомогою індексного методу вирішуються такі завдання: характеризують загальну зміну складного економічного явища чи окремих його елементів (складових); виділяють вплив одного з факторів через елімінування впливу інших; відокремлюють впливу зміни структури явища на зміну індексованої величини).

6.3 Висновки до розділу

В цьому розділі описано статистичну оцінку техногенних впливів та методологічні основи обробки екологічної інформації на базі комп'ютерних технологій.

7 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

7.1 Відповідальність за порушення законодавства про охорону праці

Закон України «Про охорону праці» передбачає, що роботодавець зобов’язаний створити на робочому місці умови праці та забезпечити додержання вимог законодавства щодо прав працівників у галузі охорони праці. З цією метою роботодавець забезпечує функціонування системи управління охороною праці та несе безпосередню відповідальність за порушення вимог з охорони праці на підприємстві. Якщо роботодавцем не забезпечено належне функціонування системи управління охороною праці на підприємстві, до нього можуть застосовуватися штрафні санкції. [15]

Згідно із Законом України «Про охорону праці» за порушення законодавчих та інших нормативних актів про охорону праці (розділ VIII) встановлені різні види відповідальності: дисциплінарна, адміністративна, матеріальна, кримінальна. Передбачена відповідальність як підприємств, так і самих працівників.

Дисциплінарна відповідальність накладається у вигляді догани, звільнення з роботи Дисциплінарне стягнення не може бути накладене пізніше шести місяців з дня вчинення проступку. Одним з конкретних порушень законодавства про охорону праці, за яке роботодавець або уповноважений ним орган має право притягнути працівника до дисциплінарної відповідальності, є ухилення останнього від проходження обов'язкового медичного огляду У цьому випадку роботодавець або уповноважений ним орган зобов'язаний також відсторонити працівника від роботи без збереження заробітної плати.

До адміністративних порушень можна віднести протиправні дії чи бездіяльність, спрямовані на створення перешкод для діяльності посадових осіб органів державного нагляду і представників професійних спілок. Адміністративна відповідальність регулюється Кодексом про адміністративні правопорушення і реалізується у вигляді накладання штрафів на працівників і, зокрема, службових осіб підприємств, установ, організацій, а також громадян - роботодавців чи уповноважених ними осіб. Максимальний розмір штрафу не може перевищувати

п’яти відсотків середньомісячного фонду заробітної плати за попередній рік юридичної чи фізичної особи, яка відповідно до законодавства використовує найману працю. Притягнення до відповідальності посадових осіб і працівників за порушення законів та інших нормативно-правових актів з охорони праці здійснюється відповідно до Кодексу України про адміністративні правопорушення.

Підставою для матеріальної відповідальності на працівника є наявність прямої дійсної шкоди, вина працівника (умисел або необережність), протиправні дії (бездіяльності) працівника, а також наявність причинного зв'язку між виною, протиправними діями працівника та завданою шкодою Існують різні види матеріальної відповідальності залежно від того, чи є в діях працівника ознаки кримінального злочину . На працівника може бути накладено повну матеріальну відповідальність або обмежену відповідальність в межах середнього місячного заробітку Працівник звільняється як від кримінальної, так і матеріальної відповідальності, якщо ним заподіяно шкоду в стані крайньої необхідності або ж в стані необхідної оборони Матеріальною відповідальністю також передбачено відшкодування збитків, заподіяних підприємствами працівникам (або членам їх сімей), які постраждали від нещасного випадку або профзахворювання

Кримінальна відповідальність за порушення правил охорони праці (недотримання загальнодержавних, галузевих та локальних правил, інструкцій та інших підзаконних актів настає за порушення вимог законодавства та інших нормативних актів про охорону праці, якщо це порушення створило небезпеку для життя або здоров'я громадян. Порушення спеціальних правил, що забезпечують безпеку робіт, становлять окремі склади злочину і для кожного з них передбачено відповідальність в Кримінальному кодексі України

7.2 Заходи покращення умов праці в галузі IT

На перший погляд, робота за комп'ютером здається безпечною, але саме легковажність до неї може призвести до певних проблем у здоров'ї людини. Професія програміста та інших фахівців IT-технологій пов'язана з колосальним розумовим напруженням. Розробники – настільки захоплені люди, що навіть

відволікаючись від роботи над проектом, продовжують думати про роботу. Нерідко відпочинком вони вважають паралельну заміну основної діяльності, наприклад, читання профільної літератури, верстку сайтів, вивчення нових мов програмування. Однак мозок не може до безкінечності приймати виключно корисну інформацію, яку розробник прагне направляти в русло особистісного та професійного зростання.

Поява та впровадження нових інформаційно-комунікаційних технологій зумовлює необхідність подальшого вдосконалення охорони праці фахівців ІТ-індустрії. З метою належного правового забезпечення необхідно розширити ти доповнити перелік основних професій комп'ютерної галузі у національному класифікаторі ДК-003-2010 [16], а також підготувати відповідний випуск у кваліфікаційному довіднику посад фахівців ІТ-індустрії, що сприятиме вирішенню питань їх соціального захисту, пенсійного забезпечення, атестації робочих місць основних професій за умовами праці на предмет подальших певних видів пільг та компенсацій за важкі шкідливі і небезпечні умови праці. Важливим напрямом стосовно визначення професійної придатності фахівців з інформаційних технологій є проведення психофізіологічної експертизи відповідно до 5 статті Закону України «Про охорону праці».

ІТ-фахівці, як і будь-які інші працівники, повинні проходити навчання і перевірку знань з охорони праці або в навчальному центрі, або в самій організації. Якщо в ній є комісія з перевірки знань з охорони праці, атестованих в спеціалізованому навчальному центрі. Навчання охорони праці в організації проводять по самостійно розробленими програмами. Їх складають, спираючись на типові програми, а також з огляду на особливості галузі, в якій працює організація.

Робота з комп'ютерами нового покоління характеризується певним психофізіологічними перенавантаженнями, втомою зорового аналізатора, гіпокінезією, відсутність диференційованих норм праці при роботі з новою комп'ютерною технікою в залежності від віку, статі, категорії зорової роботи, режимів праці і відпочинку (протягом робочого дня, тижня, щорічного режиму відпусток). Все це потребує розробки нових нормативно-правових актів з регламентації праці та відпочинку фахівців ІТ-індустрії і стандартів підприємств,

центрів комп'ютерної техніки, центрів інформаційних технологій, сучасних комп'ютерних класів. Особлива роль з точки зору збереження та відновлення здоров'я працюючих в комп'ютерні галузі належить попереднім та періодичним наглядам з подальшої психофізіологічної експертизи і встановленням професійної придатності при роботі з комп'ютерами нового покоління, який супроводжується виникненням певних факторів професійного ризику електротравматизму при їх ремонті та обслуговуванні. В цьому зв'язку необхідне запровадження експертизи на предмет безпечної експлуатації ПЕОМ, тобто офіційне підтвердження фактичних параметрів електробезпеки, їх відповідності вимогам нормативної документації фахівців, які проводять таку експертизу повинні пройти навчання і перевірку знань відповідно до вимог ДНАОП 0.00-8.20-99 [17]. За результатами експертизи повинні прийматися рішення про відповідність ПЕОМ нормам безпеки, терміни чергової експертизи, оформлюються протоколи вимірювань і випробувань, проведені у разі потреби розрахунки та експертний висновок. Для підвищення розумової працездатності то зорової роботи повинна здійснюватися ергономічна оптимізація в рамках системи «оператор-термінал», яка сприятиме результативній фізичній та інтелектуальний працездатності і відновленню психосоматичного здоров'я фахівців ІТ-індустрії. Заслуговує на увагу зарубіжний досвід створення у приміщеннях та в зоні їх розміщення на територіях підприємств спеціальних візуальних комфортних умов та забезпечення вимог виробничої естетики, дотримання норм рівнів виробничого шуму та акустичної тиші за межами офісу. Також дуже важливим є використання в офісних приміщеннях та кабінетах психофізіологічного розвантаження функціональної музики, яка сприяє попередженню перевтоми і підтриманню необхідного рівня розумової працездатності фахівців комп'ютерної галузі. В цьому напрямі заслуговує на увагу створення при великих центрах інформаційних технологій кімнат (кабінетів) психофізіологічного розвантаження працівників галузі (на 5 місць).

Зарубіжний досвід охорони праці при використанні новітніх інформаційних технологій та сучасного комп'ютерного обладнання передбачає з метою попередження наслідків монотонної праці, підвищення рівня рухової активності і покращення розумової працездатності фахівців ІТ-індустрії під час технологічних

перерв участь у спеціальних облаштованих приміщеннях необхідним спортивним інвентарем та різними тренажерами відповідних фізичних вправ, індивідуальних тренінгових завдань відповідно до віку, статі та категорії зорової роботи. Такий підхід дозволяє зняти надлишкове психофізіологічне перевантаження, підвищити працездатність центральної нервової системи, попередити перевтому зорового аналізатора. Показана ефективність проведення різноманітних за своєю спрямованістю вправ робітників цієї галузі (приблизно на 5-30%). [18]

Всі наведені заходи щодо вдосконалення охорони праці фахівців IT-індустрії повинні контролюватися службою охорони праці та комісією з охорони праці підприємства. Особливе значення у соціальному захисті цієї категорії працівників належить прийняття комплексного договору, який може забезпечити фахівців додатковими пільгами та компенсаціями.

Таким чином, використання новітніх інформаційно-комунікаційних технологій вимагає від фахівців IT-індустрії додержання певних правил та вимог з точки зору безпеки праці, її нормування з урахуванням віку працюючих та загального інформаційного навантаження, розробки та впровадження індивідуальних, щотижневих та щорічних режимів праці та відпочинку, які сприятимуть профілактиці перевтомлення і підвищенню розумової працездатності працюючих. Особливу роль в цьому напрямі повинні відігравати ергономічні заходи стосовно створення робочих місць, оптимізації взаємодії людини в рамках системи «оператор-термінал». Всі ці вимоги повинні бути втілені у відповідних нормативно-правових актах (стандартах підприємств), що регламентують різноманітні питання охорони та психології безпеки праці фахівців IT-індустрії.

7.3 Оцінка дії електромагнітного імпульсу (EMI) на елементи сервісу для адміністрування і обліку роботи автомобільної парковки

У воєнний час при застосуванні ядерної зброї проти України на електронно-обчислювальне обладнання в першу чергу буде впливати EMI ядерного вибуху у вигляді короткого імпульсу, який вражає головним чином електричну та електронну апаратуру [19].

ЕМІ виникають в основному в результаті взаємодії гамма-випромінювання з атомами навколишнього середовища. На утворення ЕМІ йде невелика кількість ядерної енергії, але він здатен викликати високі імпульси струмів та напруг в кабелях повітряних і підземних ліній зв'язку, сигналізації, управління, електропередачі, в антенах радіостанцій. Вплив ЕМІ може привести до згортання чутливих електронних та електричних елементів, зв'язаних з великими антенами чи відкритими дротами, а також до порушень в обчислювальних пристроях. Вплив ЕМІ необхідно враховувати для всіх електричних та електронних систем. Для найбільш важливих приладів треба використовувати засоби захисту і підвищувати їх стійкість до ЕМІ.

Особливістю ЕМІ, як вражаючого фактору є його здатність розповсюджуватись на десятки і сотні кілометрів в оточуючому середовищі. Тому ЕМІ може вплинути своєю дією на об'єкти, там де вибухова хвиля, світлове випромінювання, проникаюча радіація втрачають своє значення, як вражаючі фактори. При наземних та низьких повітряних вибухах в лініях зв'язку та електрозабезпечення виникають напруги, які можуть викликати пробій ізоляції провідників та кабелів відносно землі, пробій ізоляції елементів приладів підключених до повітряних і підземних ліній. Степінь враження залежить від наведеного імпульсу напруги чи струму і також електричної міцності обладнання.

Найбільш піддані впливу ЕМІ системи зв'язку, сигналізації, управління. Використані в цих системах кабелі та апаратура мають обмежену електричну міцність не більше 10кВ імпульсної напруги, тоді як наведені імпульси напруги від ЕМІ можуть перевищувати ці значення. Найбільш піддана впливу ЕМІ апаратура виконана на напівпровідниках та інтегральних схемах, працюючих на малих струмах і напругах, і значить відчутних до впливу зовнішніх електричних і магнітних кіл, в тому числі і елементи програмного засобу для управління процесом міграції віртуальних машин в обчислювальній хмарі. ЕМІ пробиває ізоляцію, спалює елементи електричних схем радіоапаратури, викликає коротке замикання в радіопристроях, іонізацію діелектриків, змінює або повністю стирає магнітний запис. Встановлено, що при дії ЕМІ на апаратуру найбільша напруга наводиться на вході. В транзисторах відбувається така залежність: чим більший коефіцієнт підсилення транзистора, тим менша його електрична міцність.

ЕМІ пошкоджує також резистори, викликає іскріння в їх міжконтактних з'єднаннях і деяких областях провідної поверхні. Найбільшу небезпеку ЕМІ представляє для апаратури, яка встановлена в особливо міцних спорудах, які витримують великі тиски ударної хвилі. В цих спорудах апаратура не виходить з ладу від механічних пошкоджень, але ЕМІ може вивести з ладу всю незахищену апаратуру системи зв'язку, сигналізації і керування. Найбільших значень досягають напруги, які наводяться між кабелем і землею. Напруженість електромагнітного поля всередині споруди в деяких випадках недостатня для того, щоб вивести з ладу апаратуру, але такі поля в змозі викликати короткочасний збій роботи радіотехнічних пристроїв.

Розглянемо можливі шляхи рішення задачі захисту від ЕМІ сервісу для адміністрування і обліку роботи автомобільної пар. Ідеальним захистом від ЕМІ виявилося б повне укриття приміщення, в якому розміщена радіоелектронна апаратура, металевим екраном.

Водночас зрозуміло, що практично забезпечити такий захист у ряді випадків неможливо, тому що для роботи апаратури часто потрібно забезпечити її електричний зв'язок із зовнішніми пристроями. Тому використовуються менш надійні засоби захисту, такі, як струмопровідні сітки, або плівкові покриття для вікон, щільникові металеві конструкції для повітрезабірників і вентиляційних отворів і контактні пружинні прокладки, розміщувані по периметру дверей і люків.

Більш складною технічною проблемою рахується захист від проникнення ЕМІ в апаратуру через різноманітні кабельні входи. Радикальним рішенням даної проблеми міг би стати перехід від електричних мереж зв'язку до практично не схильних до впливу ЕМІ волоконно-оптичних. Проте заміна напівпровідникових приладів у всьому спектрі виконуваних ними функцій електронно-оптичними пристроями можлива тільки у віддаленому майбутньому. Тому в. даний час в якості засобів захисту кабельних входів найбільші широко використовуються фільтри, у тому числі волоконні, а також іскрові розрядники, металлоокисні варистори і високошвидкісні зенеровські діоди.

Всі ці засоби мають як переваги, так і недоліки. Так, ємнісно-індуктивні фільтри достатньо ефективні для захисту від ЕМІ малої інтенсивності, волоконні фільтри захищають у відносно вузькому діапазоні надвисоких

частот. Іскрові розрядники мають значну інерційність й в основному придатні для захисту від перевантажень, що виникають під впливом напруг і струмів, що наводяться в обшивці літака, кожусі апаратури й оплетенні кабеля.

Металоокисні варистори є напівпровідниковими приладами, що різко підвищують свою провідність при високій напрузі. Проте, при застосуванні цих приладів у якості засобів захисту від ЕМІ варто враховувати їх недостатньо високу швидкодію і погіршення характеристик при кількаразовому впливі навантажень. Ці недоліки відсутні у високошвидкісних зенеровських діодах, дія яких заснована на різкій лавиноподібній зміні опору від високого значення практично до нуля, при перевищенні прикладеної до них напруги граничного розміру. Крім того на відміну від варисторів характеристики зенеровських діодів після багатократних впливів високих напруг і переключень режимів не погіршуються.

Найбільш раціональним підходом до проектування засобів захисту від ЕМІ кабельних входів є створення таких роз'ємів у конструкції яких передбачені спеціальні заходи, що забезпечують формування елементів фільтрів і установку вмонтованих зенеровських діодів. Подібне рішення сприяє одержанню дуже малих значень ємності й індуктивності, що необхідно для забезпечення захисту від імпульсів, що мають незначну тривалість і, отже, потужну високочастотну складову. Використання роз'ємів подібної конструкції дозволить вирішити проблему обмеження малогабаритних характеристик пристрою захисту.

Складність рішення задачі захисту від ЕМІ і висока вартість розроблених для цих цілей засобів і методів змушують піти по шляху їхнього вибіркового застосування в особливо важливих системах зброї і військової техніки. Такий же шлях обраний і для захисту систем, що мають велику протяжність, керування і зв'язку. Проте основним методом рішення даної проблеми спеціалісти вважають створення так званих розподілених мереж зв'язку.

7.4 Забезпечення безпеки життєдіяльності користувачів ПЕОМ в умовах НС

Забезпечення безпеки (захист) населення у НС є основна задача ЦЗ населення. Основні способи захисту населення у НС відповідно до розділу IV Кодексу цивільного захисту України від 02.10.12 no 5403-vi:

– укриття в захисних спорудах;

– евакуація з небезпечних районів;

– застосування засобів індивідуального захисту.

При цьому захист населення може бути ускладнений такими факторами: ступінь підготовки населення до дій в НС; постійне спостереження за станом навколишнього середовища; своєчасне повідомлення населення про загрозу НС; захист систем водопостачання, продуктів, сировини від небезпечного забруднення; розробка та своєчасне введення в дію режимів захисту людей; організацією планування та проведення профілактичних, санітарно-гігієнічних, протипожежних заходів, обеззараження техніки, території, споруд та ін.

Роль того чи іншого заходу та його питома вага в загальному комплексі заходів захисту залежать від умов в яких він застосовується. Найбільший ефект може бути досягнутий лише при комплексному застосуванні всіх заходів. В Україні створена і функціонує Єдина державна система цивільного захисту населення від небезпечних наслідків, аварій та катастроф техногенного, екологічного, природного та воєнного характеру. Забезпечення безпечної життєдіяльності у НС базується на комплексі організаційних, інженерно-технічних заходів і засобів, спрямованих на збереження життя і здоров'я людини у всіх сферах її діяльності. Для цього необхідно: [18]

– прогнозувати та оцінити можливі наслідки;

– заздалегідь спланувати заходи із запобігання та зменшення вірогідності виникнення НС і скорочення масштабів прояву результатів НС;

– організація робіт в умовах НС та ліквідація її наслідків.

Необхідно заздалегідь планувати роботи необхідні для запобігання або зменшення можливості їх виникнення та скорочення масштабів наслідків для забезпечення стійкої роботи об'єктів народного господарства в умовах НС. Для здійснення цих заходів важливим є набуття населенням умінь, навичок поводження в умовах НС, що надалі сприятиме зменшенню негативних

результатів, ліквідації наслідків надзвичайної ситуацій. Отже, всі можливі дії у разі виникнення НС повинні бути заздалегідь чітко сплановані.

Кінцевий результат планування дій є документ - план, який повинен містити такі елементи: конкретні показники видів робіт, заходів, які треба провести в умовах НС та терміни виконання цих робіт. Важливим є перелік ресурсів, необхідних для виконання плану та зазначення конкретних обов'язків осіб, відповідальних за виконання кожного пункту плану; способи контролю за ходом його виконання. Текстова частина плану може складатися з двох розділів. В першому наведені висновки з оцінки обстановки, яка може скластися в результаті НС. Другий розділ висвітлює заходи щодо забезпечення безпеки населення при загрозі виникнення НС. В ньому треба вказати послідовність дій: порядок оповіщення; організація розвідки і спостереження; підготовка сил і засобів для проведення рятувальних та інших невідкладних робіт; заходи щодо попередження і пом'якшення наслідків НС; прискорене проведення робіт, необхідних для захисту людей і матеріальних цінностей; хід забезпечення медичного, дозиметричного і хімічного контролю; порядок проведення заходів щодо безаварійного припинення виробництва; організація захисту людей і видача населенню ЗІЗ та проведення евакуаційних заходів; керівництво управлінням, порядком і черговістю ведення рятувальних та інших невідкладних робіт у реальних умовах НС; надання повідомлень у вищі органи ЦЗ, в комісію з надзвичайних ситуацій.

Важливим є прогноз та оцінка можливих наслідків НС, які виникають в ході її розвитку і характеру її прояву. Для цього застосовують методи орієнтовного виявлення та оцінки обстановки, яка виникає в результаті стихійних лих, аварій і катастроф, воєнних конфліктів. Складність полягає в тому, що оцінка стану території, характеру і масштабу НС в умовах неповної і ненадійної інформації, дає можливість орієнтовно визначити характер і обсяг необхідних робіт з ліквідації її наслідків. На основі цього складають довгостроковий прогноз.

Прогнозування обстановки, пов'язаної з виникненням НС, здійснюють і математичними методами із застосуванням комп’ютерів. Вихідними даними для прогнозування обстановки є місця потенційно небезпечних об'єктів - запаси речовин, джерела енергії, чисельність і щільність населення [19].

Крім цього, з точки зору забезпечення безпеки життєдіяльності робітників та службовців, а також населення, що мешкає поблизу об'єкта, важливе місце займають заходи з недопущення виникнення вторинних вражаючих чинників - пожеж, вибухів, які можуть виникати як під впливом внутрішніх, так і зовнішніх причин.

7.5 Висновки до сьомого розділу

В цьому розділі розглянуто важливі питання охорони праці та безпеки в надзвичайних ситуаціях, зокрема відповідальність за порушення законодавства про охорону праці та заходи покращення умов праці в галузі ІТ. Також проведено оцінку дії ЕМІ на елементи сервісу для адміністрування і обліку роботи автомобільної парковки та описано процес забезпечення безпеки життєдіяльності користувачів ПЕОМ в умовах НС.

ВИСНОВКИ

У результаті виконання дипломної роботи було проаналізовано три технічні аналоги, які, за своїм функціоналом, схожі з розроблюваним сервісом. Створивши порівняльну таблицю цих аналогів, було виділено переваги, які необхідно вдосконалити та недоліки, яких варто уникнути. Спираючись на вище досліджені аналоги, було створено словник вузьконаправлених термінів, описано варіанти використання сервісу, поставлено список функціональних та нефункціональних вимог.

Для опису послідовності дій користувача, у програмній системі, було змодельовано бізнес-процеси, які мають існувати у сервісі.

Зважаючи на структуру досліджених аналогів, було прийнято рішення, спроектувати сервіс з архітектурою «Database-centric architecture».

Відштовхуючись від призначення сервісу, у процесі роботи отримані основні результати:

- спроектовано концепції вирішення проблеми адміністрування і обліку роботи автомобільної парковки;
- побудовано діаграми станів та класів;
- розроблено алгоритми роботи програм;
- описано класи та основні методи, які реалізують логіку роботи сервісу.

Сервіс було реалізовано у середовищі розробки Microsoft VisualStudio 2013 на мові програмування C#. База даних створена за допомогою MySQLCommandLine та наповнена інструментом для візуального проектування БД – MySQLWorkbench. Також було залучено мову розширеної розмітки XML.

Практичним результатом дипломної роботи є створення генератора схем паркувальної розмітки, що, у результаті своєї роботи, дає XML – документ з схемою парковки; реалізовано програму для обліку і візуалізації парковки. Також розроблено функціонал для створення бази клієнтів, де буде можливість отримання карти парковки, наявність вільних місць.

Подальшим розвитком проекту може бути реалізація мобільного додатку для інформування та моніторингу стоянки, клієнтами.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Автостоянки і гаражі для легкових автомобілів. ДБН В.2.3-15:2007: ДБН В.2.3-15:2007 – [Чинний від 2007-08-01] – К.: Мінбуду України, 2007. – 36 с. – (Державні будівельні норми України).
2. Галкина Н.Г. Исследование городских парковок. – Харьков: Вестник ХНАДУ, вып. 50, 2010, – с. 84-87.
3. Завальний О. В. Особливі характеристики місць для паркування індивідуального автотранспорту / О. В. Завальний, Н. В. Аношенко // Містобудування та територіальне планування. - 2010. - Вип. 36. - С. 166-174.
4. Надозірний В.Б. Програмний засіб для адміністрування і обліку роботи автомобільної парковки / В. Надозірний – Матеріали VII науково-технічній конференції «Інформаційні моделі, системи та технології» – Тернопіль, ТНТУ, 11-12 грудня 2019 р.– с. 73.
5. Анісімов А.В. Інформаційні системи та бази даних: Навчальний посібник для студентів факультету комп'ютерних наук та кібернетики. / Анісімов А.В., Кулябко П.П. − Київ : Київський університет, 2017. – 110 с.
6. Голощапов А. Л. Microsoft VisualStudio 2013. – БХВ-Петербург, Москва, 2011. – 544 с. – ISBN: 978-5-9554-0617-5.
7. Робота з Microsoft VisualStudio 2015. [Електронний ресурс] – Режим доступу: https://www.visualstudio.com/vs-2015-product-editions (дата звернення: 24.10.2019)
8. Шилдт Г. C#. Учебный курс / Пер. с англ. –СПб.: Питер; К.: BHV, 2002. – 512 с. – ISBN: 966-338-454-9.
9. World Wide Web Consortium. XML Technology. [Електронний ресурс] – Режим доступу: http://www.w3.org/standards/xml/ (дата звернення: 25.10.2019)
10. Одиночкина С. В. Основы технологий XML. – СПб: НИУ ИТМО, 2013. - 56 с.
11. СУБД SQL-Server: основні особливості та її застосування [Електронний ресурс] – Режим доступу: http://ukrefs.com.ua/171131-SUBD-SQL-Server-osnovnye-osobennosti-i-ee-primenenie.html (дата звернення: 27.10.2019).
12. Основи мови sql [Електронний ресурс] – Режим доступу: http://www.studfiles.ru/preview/5210288/page:2/ (дата звернення 12.11.2019).

13. Джигирей В.С. Екологія та охорона навколишнього природного середовища. Навчальний посібник. – К.: Знання, 2006. – 219 с.
14. Яким Р.С. Безпека життєдіяльності людини: Навч. посібник. - Львів: Видавництво "Бескид Біт", 2005. - 304 с.
15. Закон України «Про охорону праці». [Електронний ресурс] – Режим доступу: https://zakon.rada.gov.ua/laws/show/2694-12 - (дата звертання 26.11.2019)
16. Класифікатор професій ДК 003:2010/ [Електронний ресурс] – Режим доступу: https://zakon.rada.gov.ua/rada/show/va327609-10 - (дата звертання 26.11.2019)
17. ДНАОП 0.00-8.20-99. Порядок проведення експертизи електроустановок споживачів/ [Електронний ресурс] – Режим доступу: https://dnaop.com/html/43255/doc-%D0%94%D0%9D%D0%90%D0%9E%D0%9F_0.00-8.20-99 - (дата звертання 26.11.2019)
18. Зеркалов Д.В. Безпека життєдіяльності та основи охорони праці. Навчальний посібник. К.: «Основа». 2016. 267 с.
19. Сакевич В.Ф., Поліщук О.В. Цивільна оборона. Теоретичні основи. Навчальний посібник. — Вінниця : ВНТУ, — 2009. — 136 с.
20. Тарасова В.В. Екологічна статистика. – К.: «Центр учбової літератури», 2008. - 391с.

ДОДАТКИ

Додаток Б

Клас ViewXml.cs програми parkingCRM (фрагмент)

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.IO;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Xml;
using MySql.Data.MySqlClient;
namespace parkingCRM
{
    public partial class ViewXml : Form
    {
        bool isMove;
        int xMove;
        int yMove;

        public ViewXml()
        {
            InitializeComponent();
        }



        private void ViewXml_Load(object sender, EventArgs e)
        {
            this.FormBorderStyle =
System.Windows.Forms.FormBorderStyle.None;
            label9.Visible = false;
            label10.Visible = false;
            label11.Visible = false;
            label12.Visible = false;
            label13.Visible = false;
            groupBox1.Visible = false;
            Cont.Visible = false;
            progressBar1.Visible = false;
            Cont.Visible = false;
            Zmina.Visible = false;
        }

        private void Ext_Click(object sender, EventArgs e)
        {
            Close();
        }
        string locationSource = "";

        private void LoadXml_Click(object sender, EventArgs e)
        {
            OpenFileDialog ofd = new OpenFileDialog();
            ofd.Filter = "XML Files (*.xml)|*.xml";
            ofd.FilterIndex = 0;
            ofd.DefaultExt = "xml";
            if (ofd.ShowDialog() == DialogResult.OK)
```

```
{
                if (!String.Equals(Path.GetExtension(ofd.FileName),
                                   ".xml",
                                   StringComparison.OrdinalIgnoreCase))
                {
                    MessageBox.Show("The type of the selected file is not
supported by this application. You must select an XML file.",
                                    "Invalid File Type",
                                    MessageBoxButtons.OK,
                                    MessageBoxIcon.Error);
                }
                else
                {
                    XmlDocument xmlDoc = new XmlDocument();
                    xmlDoc.Load(ofd.FileName);
                    locationSource = ofd.FileName;
                    int count = 0, countExit = 0, countEnter = 0;
                    for (int i = 0; i < 100; i++)
                    {
                        XmlNodeList listElement =
xmlDoc.GetElementsByTagName("pictureBox");
                        string textGetFromXML = listElement[i].InnerText;
                        if (textGetFromXML == "bot.PNG" ||
                            textGetFromXML == "left.PNG" ||
                            textGetFromXML == "right.PNG" ||
                            textGetFromXML == "top.PNG")
                        {
                            count++;
                            LoadXml.Visible = false;
                            label7.Visible = false;
                            panel5.BackColor = Color.Red;
                            panel6.BackColor = Color.Red;
                            label9.Visible = true;
                            label10.Visible = true;
                            label11.Visible = true;
                            label12.Visible = true;
                            groupBox1.Visible = true;
                            Cont.Visible = true;
                        }
                        if (textGetFromXML == "toEnter.PNG")
                        {
                            countEnter++;
                        }
                        if (textGetFromXML == "toExit.PNG")
                        {
                            countExit++;
                        }
                    }
                    Zmina.Visible = true;
                    label10.Text = "Кількість парковочних місць: " + count;
                    label11.Text = "Кількість вїздів: " + 1;
                    label12.Text = "Кількість виїздів: " + 1;
                    XmlNodeList picture1 =
xmlDoc.GetElementsByTagName("pictureBox");
pictureBox1.ImageLocation = @"imagens\" + picture1[0].InnerText;
                    XmlNodeList picture2 =
xmlDoc.GetElementsByTagName("pictureBox");
pictureBox2.ImageLocation = @"imagens\" + picture2[1].InnerText;
```



```
  XmlNodeList picture3 = xmlDoc.GetElementsByTagName("pictureBox");
pictureBox3.ImageLocation = @"imagens\" + picture3[2].InnerText;
```

```
                    XmlNodeList picture4 =
xmlDoc.GetElementsByTagName("pictureBox");
pictureBox4.ImageLocation = @"imagens\" + picture4[3].InnerText;
                    XmlNodeList picture5 =
xmlDoc.GetElementsByTagName("pictureBox");
pictureBox5.ImageLocation = @"imagens\" + picture5[4].InnerText;
                    XmlNodeList picture6 =
xmlDoc.GetElementsByTagName("pictureBox");
pictureBox6.ImageLocation = @"imagens\" + picture6[5].InnerText;
                    XmlNodeList picture7 =
xmlDoc.GetElementsByTagName("pictureBox");
pictureBox7.ImageLocation = @"imagens\" + picture7[6].InnerText;
                    XmlNodeList picture8 =
xmlDoc.GetElementsByTagName("pictureBox");
pictureBox8.ImageLocation = @"imagens\" + picture8[7].InnerText;
                    XmlNodeList picture9 =
xmlDoc.GetElementsByTagName("pictureBox");
pictureBox9.ImageLocation = @"imagens\" + picture9[8].InnerText;
                    XmlNodeList picture10 =
xmlDoc.GetElementsByTagName("pictureBox");
pictureBox10.ImageLocation = @"imagens\" + picture10[9].InnerText;
                    XmlNodeList picture11 =
xmlDoc.GetElementsByTagName("pictureBox");
  pictureBox11.ImageLocation = @"imagens\" + picture11[10].InnerText;
                    XmlNodeList picture12 =
xmlDoc.GetElementsByTagName("pictureBox");
pictureBox12.ImageLocation = @"imagens\" + picture12[11].InnerText;
                    XmlNodeList picture13 =
xmlDoc.GetElementsByTagName("pictureBox");
pictureBox13.ImageLocation = @"imagens\" + picture13[12].InnerText;
                    XmlNodeList picture14 =
xmlDoc.GetElementsByTagName("pictureBox");
pictureBox14.ImageLocation = @"imagens\" + picture14[13].InnerText;
                    XmlNodeList picture15 =
xmlDoc.GetElementsByTagName("pictureBox");
pictureBox15.ImageLocation = @"imagens\" + picture15[14].InnerText;
                    XmlNodeList picture16 =
xmlDoc.GetElementsByTagName("pictureBox");
pictureBox16.ImageLocation = @"imagens\" + picture16[15].InnerText;
                    XmlNodeList picture17 =
xmlDoc.GetElementsByTagName("pictureBox");
pictureBox17.ImageLocation = @"imagens\" + picture17[16].InnerText;
                    XmlNodeList picture18 =
xmlDoc.GetElementsByTagName("pictureBox");
pictureBox18.ImageLocation = @"imagens\" + picture18[17].InnerText;
                    XmlNodeList picture19 =
xmlDoc.GetElementsByTagName("pictureBox");
pictureBox19.ImageLocation = @"imagens\" + picture19[18].InnerText;
                    XmlNodeList picture20 =
xmlDoc.GetElementsByTagName("pictureBox");
pictureBox20.ImageLocation = @"imagens\" + picture20[19].InnerText;
                    XmlNodeList picture21 =
xmlDoc.GetElementsByTagName("pictureBox");
pictureBox21.ImageLocation = @"imagens\" + picture21[20].InnerText;
.....................................
```