Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

Інститут електроенергетики
(інститут)
факультет інформаційних технологій
(факультет)
Кафедра інформаційних технологій та комп'ютерної інженерії
(повна назва)

ПОЯСНЮВАЛЬНА ЗАПИСКА
кваліфікаційної роботи ступеня ___магістра___
(бакалавра, спеціаліста, магістра)

Студента _____Удачиної Катерини Олександрівни_____
(ПІБ)
академічної групи _____126м-22з-1_____
(шифр)
спеціальності _____126 «Інформаційні системи та технології»_____
(код і назва спеціальності)
за освітньо-професійною програмою _____
_____«Інформаційні системи та технології»_____
(офіційна назва)
на тему _____Розробка інформаційної системи логістичного підприємства_____
_____з актуалізацією шляхів перевезення на базі GMap.NET_____
(назва за наказом ректора)

Керівники	**Прізвище, ініціали**	**Оцінка за шкалою**		**Підпис**
		рейтинговою	**інституційною**	
кваліфікаційної роботи	Олевський В.І.			
розділів				

Рецензент	Журба А.О.			

Нормоконтролер	Коротенко Г.М.			

Дніпро
2023

ЗАТВЕРДЖЕНО:

завідувач кафедри
інформаційних технологій та комп'ютерної інженерії
(повна назва)
_____ Гнатушенко В.В.
(підпис) (прізвище, ініціали)

«_____»_____20___ року

ЗАВДАННЯ
на кваліфікаційну роботу
ступеня _____магістр_____
(бакалавра, спеціаліста, магістра)

студенту _____Удачиній К.О._____ **академічної групи** ___126м-22з-1___
(прізвище та ініціали) (шифр)
спеціальності __126 «Інформаційні системи та технології»_____
за освітньою-професійною програмою _____
_____«Інформаційні системи та технології»_____
на тему _Розробка інформаційної системи логістичного підприємства _____
_____ з актуалізацією шляхів перевезення на базі GMap.NET._____
затверджену наказом ректора НТУ «Дніпровська політехніка» від 09.10.23_№ 1228-С

Розділ	Зміст	Термін виконання
Розділ 1	Аналіз стану області рішення задачі	09.10.2023 – 25.10.2023
Розділ 2	Проєктування інформаційної системи	26.10.2023 – 15.11.2023
Розділ 3	Програмна реалізація інформаційної системи	16.11.2023 – 08.12.2023

Завдання видано _____ _____
(підпис керівника) (прізвище, ініціали)
Дата видачі _____
Дата подання до екзаменаційної комісії _____
Прийнято до виконання _____ ___Удачина К.О.____
(підпис студента) (прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 70 стор., 32 рис., 2 додатки, 41 джерело.

Об'єкт дослідження: автоматизація процесів діяльності підприємств транспортної логістики.

Предмет дослідження: інформаційні системи і технології для побудови оптимальних шляхів перевезення товару.

Мета магістерської роботи: створення інформаційної системи на основі платформи .NET для формування оптимального маршруту постачання товарів та відображення його на карті за допомогою інструмента GMap в режимі онлайн.

У вступі наведено актуальність, подано стан проблеми та визначено задачі дослідження. У першому розділі представлено опис предметної області, досліджено існуючі підходи, інформаційні системи і технології для формування оптимального маршруту доставки товарів, виконано постановку завдання. У другому розділі описано методику для вирішення задачі, обґрунтовано вибір методу гілок та меж, наведено архітектуру, спроєктовано базу даних, розроблено інтерфейс користувача системи. У третьому розділі наведено елементи програмної реалізації системи, представлено результати тестування програмного модуля.

Наукова новизна отриманих результатів кваліфікаційної роботи полягає у застосуванні комбінаторних методів у поєднанні з використанням картографічної бібліотеки GMap.NET для та актуалізації шляхів перевезення.

Практична цінність результатів полягає в тому, що розроблена в роботі інформаційна система дозволяє визначати послідовність адрес доставки товарів, мінімізуючи при цьому витрати, та будувати оптимальний маршрут перевезення, обчислювати його довжину і час в режимі онлайн.

ОПТИМАЛЬНИЙ МАРШРУТ, МІНІМІЗАЦІЯ ВИТРАТ, МЕТОД ГІЛОК І МЕЖ, GMAP.NET, ФРЕЙМВОРК, ОБ’ЄКТНО-ОРІЄНТОВАНА ТЕХНОЛОГОГІЯ.

ABSTRACT

Explanatory note: 70 pages, 32 figures, 2 applications, 41 sources.

Object of research: activity processes of transport logistics enterprises.

Subject of research: information systems and technologies for building optimal ways of transporting goods.

Purpose of Master's thesis: development of the information system based on the .NET platform for forming the optimal route for the supply of goods and displaying it on map using the GMap tool online.

The introduction includes the relevance has given, the state of the problem has presented, the tasks of the research have defined. In the first chapter, a description of the subject area has presented, the existing approaches, information systems and technologies for the formation of the optimal route for the delivery of goods have investigated, the task of developing the system has performed. In the second chapter the methodology for solving the problem of forming the optimal route for the delivery of goods has describes, the choice of the method of branch and bound has justified, the architecture has given, the functional model has presented, the database has designed, and the user interface of the system has developed. In the third section, the elements of the software implementation of the system have given, and the results of testing the software module have presented.

The scientific novelty of the obtained results of the qualification work consists in the application of combinatorial methods to determine the sequence of goods delivery points in combination with the use of the GMap.NET cartographic library to perform geocoding of data and update transportation routes.

The practical value of the results is that the information system developed in the work allows you to determine the sequence of addresses for the delivery of goods, while minimizing costs, and to build an optimal transportation route, calculate its length and time online.

OPTIMUM ROUTE, COST MINIMIZATION, BRANCH AND BOUND, GMAP.NET, FRAMEWORK, OBJECT-ORIENTED TECHNOLOGY.

ЗМІСТ

ВСТУП

Логістичні процеси відіграють значну роль в різних сферах діяльності. Однією з актуальних задач виступає побудова оптимального шляху доставки товарів, оскільки помилкова маршрутизація призводить до додаткових витрат і впливає на прибуток компанії.

Для побудови оптимальних маршрутів адресної доставки товарів важливо мати точні дані про місцезнаходження клієнта та оптимально використовувати геокоординати клієнтів. І саме використання сучасних інформаційних технологій допоможе вирішити питання геокодування даних і побудови маршруту в режимі онлайн.

Питання автоматизації логістичних процесів досліджували багато дослідників: Марченко В. М., Шутюк В. В. [1] Біліченко В. В., Бурєнніков Ю. Ю. [2], Н. П. Резнік , С. В. Руденко , К. М. Пилипчук [3], К. А. Заріпова [4], О. А. Біловодська, Н. В. Гайдабрус [5] та інші. Враховуючи внесок вказаних вчених, слід зазначити, що виникає необхідність розгляду функціонування логістичної сфери відповідно до подій останніх років, адаптації підходів управління логістичними процесами під сучасні умови, які змінюються дуже швидко.

Зв’язок роботи з науковими програмами, планами, темами. Кваліфікаційна робота виконана згідно з планом наукових досліджень кафедри інформаційних технологій та комп’ютерної інженерії НТУ «ДП» як складова частина науково-дослідницької роботи на тему «Моделі й інформаційні технології обробки та аналізу даних в складних комп'ютерних системах і мережах» (державний реєстраційний номер 0121U114523).

Мета і завдання дослідження. Метою роботи виступає створення інформаційної системи на основі платформи .NET для формування оптимального маршруту постачання товарів та відображення його на карті за допомогою інструмента GMap.NET в режимі онлайн.

Для досягнення поставленої мети необхідно вирішити наступні завдання:

– дослідити існуючі інформаційні системи формування оптимального маршруту доставки товарів;

– описати технологію для вирішення задачі: виконати аналіз алгоритмів формування оптимального шляху перевезення, обґрунтувати вибір алгоритму;

– навести архітектуру інформаційної системи, представити функціональну модель, пояснити взамозв’язок компонентів;

– спроєктувати організацію даних інформаційної системи, навести опис структури бази даних;

– розробити інтерфейс користувача системи;

– виконати програмну реалізацію спроєктованої системи, об’єднати всі компоненти: підключити базу даних, налаштувати карту GMap;

– виконати тестування інформаційної системи.

Об’єктом даної роботи виступають автоматизація процесів діяльності підприємств транспортної логістики. Предметом роботи є інформаційні системи і технології для побудови оптимальних шляхів перевезення товару.

Методологічна основа дослідження: теорія графів, методи комбінаторної оптимізації – при розробці алгоритму формування оптимального маршруту доставки товарів, інформаційні технології: фреймворк .NET, бібліотека GMap.NET, СУБД MS SQL Server – при розробці інформаційної системи.

Наукова новизна отриманих результатів полягає у застосуванні комбінаторних методів для визначення послідовності пунктів доставки товарів у поєднанні з використанням картографічної бібліотеки GMap.NET для виконання геокодування даних та актуалізації шляхів перевезення.

Практичне значення отриманих результатів. Використання результатів дослідження дозволяє підвищити ефективність діяльності підприємства за рахунок мінімізації витрат на організацію поставки товарів.

Апробація результатів дослідження. Основні положення дослідження були оприлюднені на Всеукраїнській науково-практичній конференції здобувачів вищої освіти та молодих учених «Сучасні інформаційні технології: теорія, практика, перспективи» (м. Дніпро, 2023 р.).

Публікації. За результатами дослідження опубліковано 1 роботу [6].

РОЗДІЛ 1
1. АНАЛІЗ СТАНУ ОБЛАСТІ РІШЕННЯ ЗАДАЧІ

1.1. Опис предметної області

Логістичні процеси відіграють суттєву роль у економіці країни і взаємодіють з різними галузями діяльності. Логістика призначена для вирішення різноманітних завдань, таких як зберігання, транспортування, постачання, управління запасами, розподіл посередників, організація різновидів транспорту (автомобільний, залізничний, морський, повітряний) для перевезення вантажів [1, 6].

Термін "логістика" походить від досліджень різних вчених та спочатку був введений у військовій сфері для забезпечення армійських підрозділів. Найбільш поширене визначення логістики розглядає її як науку про управління всіма видами потоків у економічних системах [2, стор. 8].

Логістика включає різні підсистеми [7]:

– виробнича логістика;
– закупівельна логістика;
– логістика запасів;
– транспортна логістика;
– інформаційна логістика;
– митна логістика;
– складська логістика.

Важливою складовою логістики виступає транспортна логістика, яка виконує функцію управління матеріальними потоками під час їх переміщення від постачальника до споживача. Вартість товару, який доставляється, залежить від величини витрат, пов’язаних з його транспортуванням. На ці витрати впливає вид транспорту, маршрут руху, кількість учасників транспортного процесу та інші фактори. Відповідно до цього визначається попит на послуги логістичних провайдерів.

Основні етапи транспортної логістики включають [1, стор. 202]:

– організація маршруту для перевезення;

– розробка графіку переміщення;

– планування завантаженості транспортних засобів;

– вибір оптимального маршруту та методу перевезення;

– виконання операцій з навантаження та розвантаження;

– забезпечення зберігання вантажів на місцях навантаження та розвантаження;

– узгодження вартості транспортування;

– пошук і фрахтування транспортних засобів;

– подача транспортних засобів на місця навантаження / розвантаження;

– оформлення митних і супровідних документів;

– виконання операцій із завантаження/розвантаження, дроблення, консолідації та зберігання вантажів з метою оптимізації транспортного процесу;

– здійснення документального супроводу;

– оформлення дозволів на перевезення негабаритних і небезпечних вантажів;

– супровід вантажів;

– контроль над процесом перевезення.

Багато вчених присвятили свої роботи дослідженню питань логістичної сфери: Г. Лейбніц розглядав логістику як математичну логіку, А.-А. Жоміні можна вважати основоположником перших наукових праць з воєнної логістики, Є. Слуцький був першим економістом-математиком, хто використав термін «логістика» в Україні [1]. Серед сучасних дослідників системи управління ланцюгами постачань розглядали Резнік Н. П., Руденко С. В., Пилипчук К. М. [3], Заріпова К. А. [4] аналізувала функції логістичних інформаційних систем, Біловодська О. А. та Гайдабрус Н. В. [5] проводили дослідження у сфері логістичного сервісу як складової

інноваційної діяльності підприємства. Не зменшуючи значення внеску попередніх дослідників, виникає необхідність розгляду функціонування логістичної сфери відповідно до подій останніх років, адаптації підходів управління логістичними процесами під сучасні умови, які змінюються дуже швидко.

Останні роки в сфері логістики в Україні пройшли період інтенсивних змін. Спочатку, у зв'язку з пандемією, була необхідність перегляду ланцюгів постачання для зменшення відстані між країнами виробництва та реалізації товарів. На початку 2022 року виникли нові виклики, які вимагали невідкладних рішень [8]:

1. Відмова від накопичення товарів на складах: зменшення запасів на складах призвело до збільшення оборотності та зменшення заморожених коштів, що має важливе значення для країни. З ризиком атак на продукцію на складах, як у випадку пожежі на одному з найбільших логістичних центрів України, West Gate Logistic, площею 100 тис. кв. метрів;

2. Зміна умов зберігання: через бойові дії у Київській області, де розташована більшість складів, довелося транспортувати товари на захід України, де не вистачало необхідної площі для зберігання продукції;

3. Проблеми з закупівлею товарів: обмежена пропозиція товарів, відсутність постачальників, блокування портів та навантаження на залізничний транспорт;

4. Ускладнення логістичних операцій: обмеження часу через комендантську годину, затримки на блокпостах, перевірки, розробка додаткових маршрутів через проведення бойових дій;

5. Дефіцит водіїв та транспорту: зростання термінових поставок гуманітарної допомоги, зупинка діяльності міжнародних компаній в Україні (Raben Group, CMA CGM, MSC, HWL, Hapag-Lloyd і Maersk) [9].

Таким чином, враховуючи поточну ситуацією, оптимізація процесів транспортування є однією з актуальних задач транспортної логістики.

Для оптимізації процесу транспортування необхідно проаналізувати поставки товарів, визначити тарифи на транспортування та вказати обмеження при побудові плану, який може зменшити загальні витрати та покращити процес. Враховуючи, що транспортні витрати становлять значну частину витрат підприємства, то керівникам доцільно зосередитися на цьому питанні.

Звісно, в умовах сьогодення існує багато ризиків у сфері транспортної логістики, але разом з тим доцільно розглянути шляхи її покращення [10].

Використання автоматизованого програмного забезпечення (ПЗ).

Для великих організацій слід використовувати автоматизоване ПЗ для моніторингу процесів. Якщо виробники великогабаритних товарів мають широку базу клієнтів, то вони мають і значну кількість поставок, при ручній обробці яких існує високий ризик допущення помилок у процесі транспортування. А використання автоматизованої системи допомагає відстежувати певні помилки, якщо такі виникають.

Впровадження автоматизованого програмного забезпечення в роботу транспортних компаній передбачає виникнення певних питань, що стосуються технічного обслуговування або ремонту ще до того, як станеться поломка. За рахунок цього, у водія з’являється час для вирішення інших питань, поки його транспорт знаходиться на обслуговуванні.

Також сучасні програмні додатки мають вбудовані GPS-трекери, які виконуються функції відслідковування транспортних потоків, врахування погодних умов, цін на паливо, що призводить до побудову найкращого можливого маршруту. Побідні системи не лише можуть виявляти проблеми, а також запобігати настанню непередбачених випадків. Як наприклад, підбирачі і трекери на навантажувачах використовуються для безпечного транспортування та зберігання товарів, що мінімізує ймовірність виникнення збоїв.

Аналіз маршрутів, режимів та тарифів.

Перегляд усіх маршрутів розподілу, використовуваних видів транспорту та понесених витрат дозволяє краще керувати транспортом та оптимізувати його. Виконуючи аналіз всіх шляхів перевезення, можна згрупувати вантажі по географічним регіонах і виконувати їх поставку разом, зменшуючи при цьому транспортні витрати і час на перевезення.

Також дослідження різних видів транспорту та їх тарифів дозволяє визначити найбільш дороговартісні напрямки. У подальшому це допоможе визначити більш економічні способи перевезення товару. Диференціація способів доставки дозволить, наприклад, замість використання дорогих авіаперевезень і відправки частини товари морем досягти значної економії витрат на транспортування.

Використання прозорого процесу транспортування.

Важливе значення для клієнтів має можливість відстеження пересування товару, адже клієнт має бути впевнений, що про його замовлення не забули, що товар вже дійсно відправлений і якщо виникає певна затримка, то про це одразу повідомляють. Це підвищує довіру клієнта до транспортної компанії, а також дозволяє йому планувати свій графік з урахуванням витрат часу на отримання товару.

Прозорість також є важливою і для виконавця, оскільки при постійному моніторі знаходження товару можна оперативно відреагувати на його затримку і усунути причини її виникнення.

Захист даних.

Процес транспортування передбачає обробку даних клієнтів: прізвища, контактні дані, електронні адреси, фізичне місцерозташування, інформацію по товару. Звісно, цими даними обмінюються робітники транспортної компанії, а також інші залучені служби. Тому захист цих даних є одним з найголовніших питань, які впливають на відносини між клієнтом і виконавцем.

5. Свідомий вибір партнера доставки.

При виборі транспортного постачальника першим фактором є ціна, яка впливає на витрати при перевезенні товару. В свою чергу, ціна залежить від терміновості доставки, розміру і ваги товару.

Високі витрати на перевезення мають вплив на ефективність діяльності певного підприємства, оскільки зменшують його прибуток. Тому з економічної точки зору питання тарифів відіграє ключову роль, звісно, що при цьому треба враховувати і якість послуг. Тобто, знаходити ефективне співвідношення ціни і якості.

Слід звертати також увагу на те, що входить до переліку послуг. Важливими є функції обслуговування маршрутів та пунктів доставки товару, його відстеження, повідомлення про затримку за необхідності та підтвердження прибуття.

Вибір авторитетної компанії – крок до успіху, навіть за більші кошти. Для цього можна перевірити правдивість інформації, дослідивши цю компанію на ринку послуг, зібравши про неї певні відгуки. Якщо компанія є застрахованою, то це також має позитивне значення у випадку пошкодження або втрати товару.

Отже, до питання вибору транспортного партнеру треба підходити свідомо і виважено, оскільки це визначає безпечну та своєчасну доставку товарів. Найкращою стратегією буде вибір того виконавця, який відповідає бюджету та якості послуги, що надається

6. Транспорт у непіковий період.

Вибір днів доставки відіграє також значну роль, оскільки, наприклад, багато відправлень здійснюється наприкінці місяця або перед святами доступність водіїв є проблемною. Тому потрібно тримати зв'язок з постачальником і домовлятися про оптимальне співвідношення ціни та терміни доставки.

Отже, транспортна логістика дозволяє вирішувати багато важливих питань, одним з яких є побудова оптимального шляху перевезення товарів.

Оскільки сьогодні це питання є актуальним, у подальшому підрозділі будуть розглянуті технології, які дозволяють автоматизувати процес вирішення цієї задачі.

1.2. Огляд існуючих інформаційних систем формування оптимального маршруту доставки товарів

Сьогодні IT-ринок пропонує широкий вибір інформаційних систем для управління процесами перевезень, зокрема для формування оптимальних ланцюгів поставки.

ANT-Logistics (Мурашина логістика) – хмарний TMS-сервіс, призначений для планування маршрутів та контролю за транспортом [11].

ANT-Logistics виконує наступні функції:

– розраховує маршрути для 2500 адресних пунктів за 18 хвилин;

– враховує трафік доріг при визначенні маршрутів;

– на основі хмарного сервісу «Мобільна торгівля» організовує роботу та контроль торгових агентів;

– містить вебінтерфейс супервайзера для роботи з агентами;

– виконує багатофакторну оптимізацію при плануванні маршрутів: враховує понад 70 параметрів та понад 15 способів планування маршрутів;

– використовує різні моделі розрахунку відповідно до кожного способу: з оптимізацією за витратами та відстанню, без запізнень, без перевантаження машин, без переробки водія;

– можливість налаштування семи рівнів доступів користувачів до функціоналу;

– наочне представляє точки доставки на карті, відображає маршрут;

– порівнює плановий маршрут та фактичні дані GPS-трекерів, виводить кілометраж, інформацію про запізнення та відхилення від маршруту;

– виконує аналітику: надає готові звіти та друковані форми, формує власні звіти у вигляді таблиць або інтерактивних інформаційних панелей з їх подальшою печаткою;

– забезпечує доступ до маршруту в мобільних додатках Android та IOS, що дає водієві можливість завжди бачити маршрут, переходити в режим навігації, бачити повідомлення при зміні маршруту;

– контролює доставку: дозволяє спостерігати за водієм, самостійно налаштовувати певні правила, перевіряти порушення;

– надає можливість ставити і контролювати завдання по кожній точці доставки;

– забезпечує об'єднання API з системою обліку, що дозволяє завантажувати заявки на доставку та отримувати оптимальні маршрути, налаштовувати взаємодію з довідниками автомобілів, торгових точок та інших елементів;

– дозволяє надсилати SMS-повідомлення клієнтам.

За середніми показниками впровадження сервісу очікується зменшення пробігу транспортних засобів на 20%, скорочення транспортних витрат на 25%, збільшення кількості обслужених точок на 30% за той самий період часу, а скорочення часу, який логіст витрачає на планування маршрутів, - на 75%.

Інтерфейс додатку представлено на рисунку 1.1.

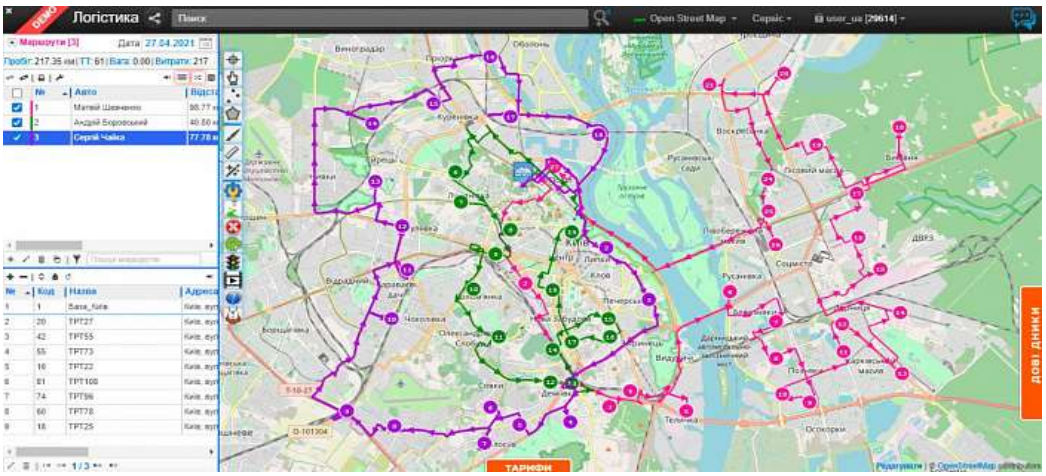

Рис. 1.1. Інтерфейс програми ANT-Logistics [11]

Компанія AVADA MEDIA виконує розробку програмних рішень для сфери транспорту та логістики [12]. Пропонує CRM-системи для логістичного бізнесу, які виконують оптимізацію та планування операційної діяльності, автоматизацію рутинних завдань, забезпечують надійне зберігання та швидкий доступ до клієнтських даних, ефективне планування стратегії розвитку компанії виходячи з реальних аналітичних даних, дозволяють вести маркетингову політику.

Також компанія розробляє інтелектуальні системи управління зерновими та контейнерними терміналами, використання яких убезпечує компанію від виникнення різних помилок у процесі прийому, відвантаження та транспортування зернових культур. Надає програмне забезпечення для складської логістики, що використовується для обліку, керування та розподілу товарних запасів; програмне забезпечення для морських та контейнерних перевезень для оптимізації документообігу з урахуванням митного законодавства різних країн, а також автоматизації розрахунку вартості транспортування та взаємодії з клієнтами; мобільні додатки для водіїв, які надають корисну інформацію щодо оптимізації маршрутів, документації та обслуговування автомобіля. Також компанія створює системи контролю та управління доступом транспорту (СКУД), що призначені для обмеження доступу автотранспорту на об’єкти підприємства з метою безпеки та контролю робочого часу робітників; автоматизовані системи вагогабаритного контролю, що здатні автоматично зчитувати потрібні дані про автомобілі з апаратних пристроїв та приймати рішення щодо авторизації проїзду транспорту; системи автоматизації комерційних доріг та переправ, які дозволяють дозволяє компаніям повністю автоматизувати процес оплати та пропуску транспортних засобів, а також вести точну статистику для аналізу бізнес-ефективності (рис. 1.2) [12].

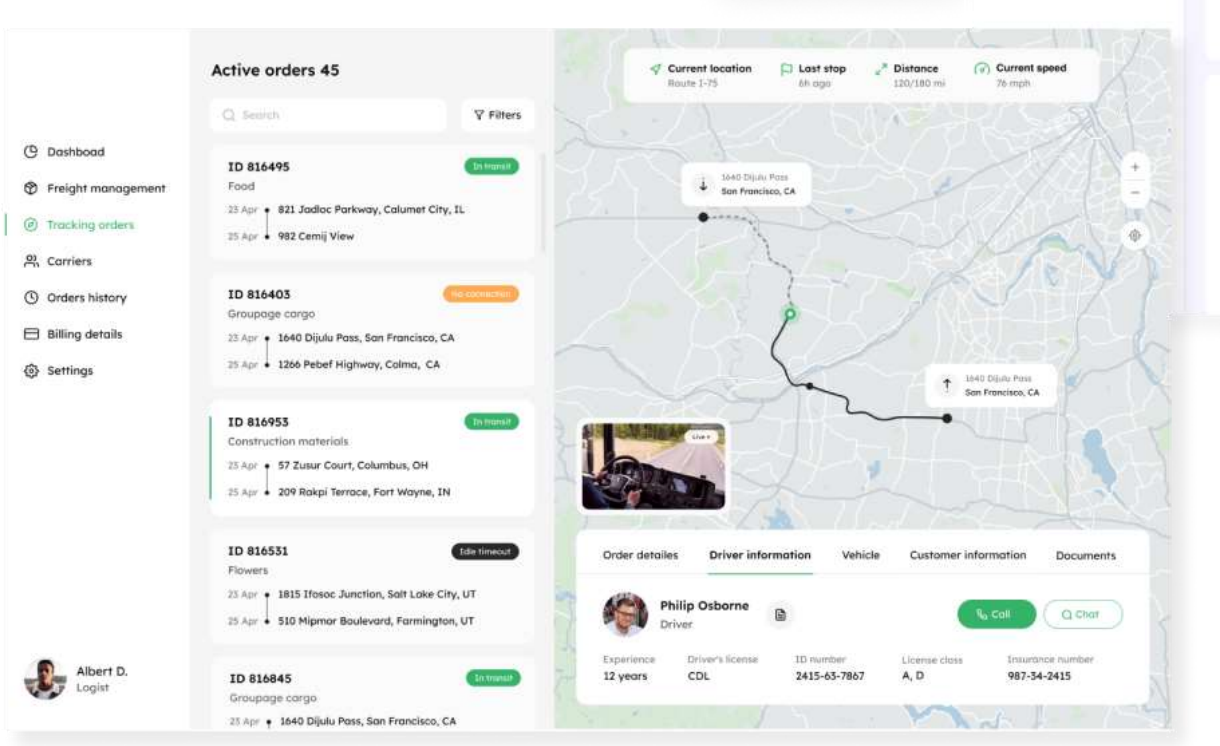

Рис. 1.2. Діалогове вікно системи автоматизації комерційних доріг та переправ [11]

TMS – базове логістичне програмне забезпечення для оптимізації маршруту, яке здатне відстежувати, регулювати та контролювати логістичні процеси; мінімізує вплив людських помилок і втоми, а також скорочує час, витрачений на планування та пошук причин помилок [13].

Зарубіжний ринок також пропонує програмне забезпечення для управління логістичними процесами [14].

Програма Upper призначена для ефективної оптимізації маршрутів транспортних засобів: дозволяє легко додавати різні зупинки, розставляти пріоритети для важливих поставок і призначати точні часові вікна для гарантованого своєчасного прибуття з метою покращення логістичних операцій; створює оптимізовані маршрути для кількох транспортних засобів одночасно, що значно економить час, а також зменшує витрати на паливо; автоматично сповіщає клієнтів про час прибуття, затримки, зміцнюючи тим самим довіру клієнтів [14].

Особливості системи Upper:

– функція імпорту декількох зупинок за допомогою файлів CSV або Excel;

– оптимізація маршруту для кількох транспортних засобів для економії часу та коштів;

– розрахунковий час прибуття (ETA) для простоти та зручності клієнтів;

– автоматичні сповіщення для кращого спілкування з клієнтами;

– зміна маршрутів між водіями для гнучкості;

– розширене планування маршруту для підвищення ефективності;

– відстеження водія в реальному часі для кращого керування автомобілями;

– функція підтвердження доставки за допомогою фотографій, приміток і електронних підписів;

– функція доставки до узбіччя для більш плавного висадження;

– розширені звіти та аналітика для оцінки ефективності;

– можливості інтеграції зі сторонніми платформами;

– спеціальний додаток для водіїв для ефективного керування маршрутами;

– контактна книга для ефективного управління адресами;

– виділена та легкодоступна технічна підтримка;

– пріоритезація термінових поставок для дотримання термінів;

– можливість призначення часу обслуговування на виділених зупинках.

Fishbowl – це програмне рішення ERP, яке допомагає організаціям працювати зі своїми запасами, виконувати замовлення та транспортувати процеси. Це додатково згладжує логістичну діяльність шляхом подальшого розвитку кваліфікації та зниження витрат. Однією з значних переваг Fishbowl є його інтеграція з такими системами, як WMS або TMS, для спрощення процесу ланцюжка поставок. Fishbowl – це корисний інструмент для великих

компаній, включаючи виробників, роздрібних і оптовиків, який допомагає їм масштабувати свою діяльність [14].

Rose Rocket – спеціалізоване програмне рішення, призначене для оптимізації роботи сучасних транспортних і логістичних компаній; виконує роль об'єднаного центру, де організації можуть наглядати та координувати весь свій робочий процес транспортування. Початкова ціна Rose Rocket становить 99,00 доларів США на місяць і може змінюватися залежно від потреби [14].

Oracle Netsuite – це одне з найпотужніших логістичних рішень і програмного забезпечення для управління запасами. Односистемний підхід Oracle спрямований на зменшення витрат на IT та інші пов'язані з ними витрати на обслуговування. Однак підписки на Oracle недешеві і в довгостроковій перспективі можуть коштувати дуже дорого.

Oracle NetSuite надає підприємствам єдину платформу для ефективного керування різними аспектами їх діяльності; дозволяє користувачам отримувати доступ до даних і програм з будь-якої точки, де є доступ до Інтернету, що особливо важливо для віддаленої роботи; є масштабованим і дає можливість встановлювати певні налаштування, що дозволяє компаніям адаптувати систему до своїх конкретних процесів і робочих процесів [14].

PTV Route Optimizer – програмне забезпечення для оптимізації маршрутів, розроблене для вирішення проблем доставки товарів. Підтримує виробників, роздрібних торговців, оптовиків і транспортних компаній у регіональному розповсюдженні продуктів і чудово справляється зі складнощами міської доставки [15].

Інструмент створює ефективні та надійні маршрути для автопарків, які зменшують пробіг і витрати, враховуючи важливі аспекти планування та обмеження. Це дозволяє компаніям оцифровувати процеси, централізовано зберігати знання та підвищувати прозорість і продуктивність доставки [15].

Компанія Vilmate пропонує рішення для планування маршрутів, яке складається з мобільного додатку та веб-порталу. Призначене для побудови

найбільш оптимального автомобільного маршруту між двома точками за різними критеріями. Ці критерії включають день тижня, час доби та годину, поточні дорожні роботи та перекриття, поточні або можливі затори тощо (рис. 1.3) [16].

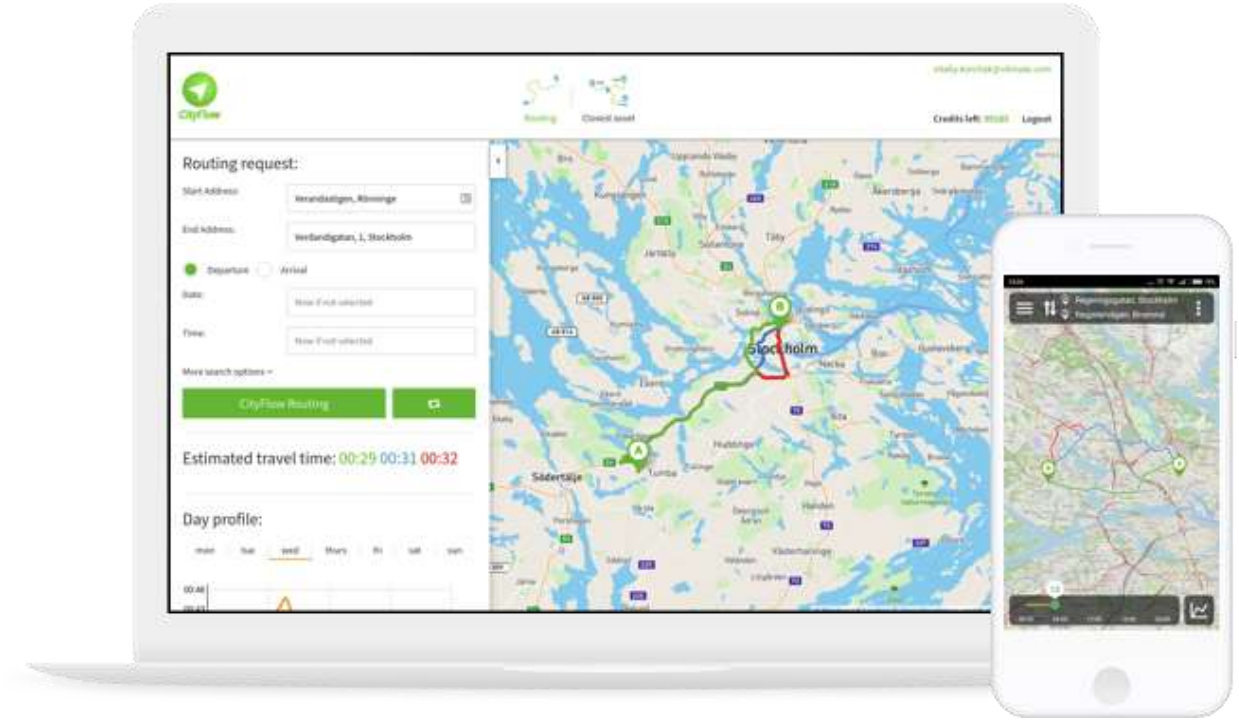

Рис. 1.3. Інтерфейс ПЗ для планування маршрутів компанії Vilmate [16]

Якщо звернутися до відомого сервісу Google Maps (рис. 1.4), то він є одним із найзручніших додатків для онлайн-навігації, який виконує пошук адреси, побудову маршрутів у урахуванням заторів і способу пересування. Але в разі вибору декількох адрес на карті треба передбачити пріоритетність, що не є дуже зручно, наприклад, при багатьох пунктах доставки товарів [17-18].

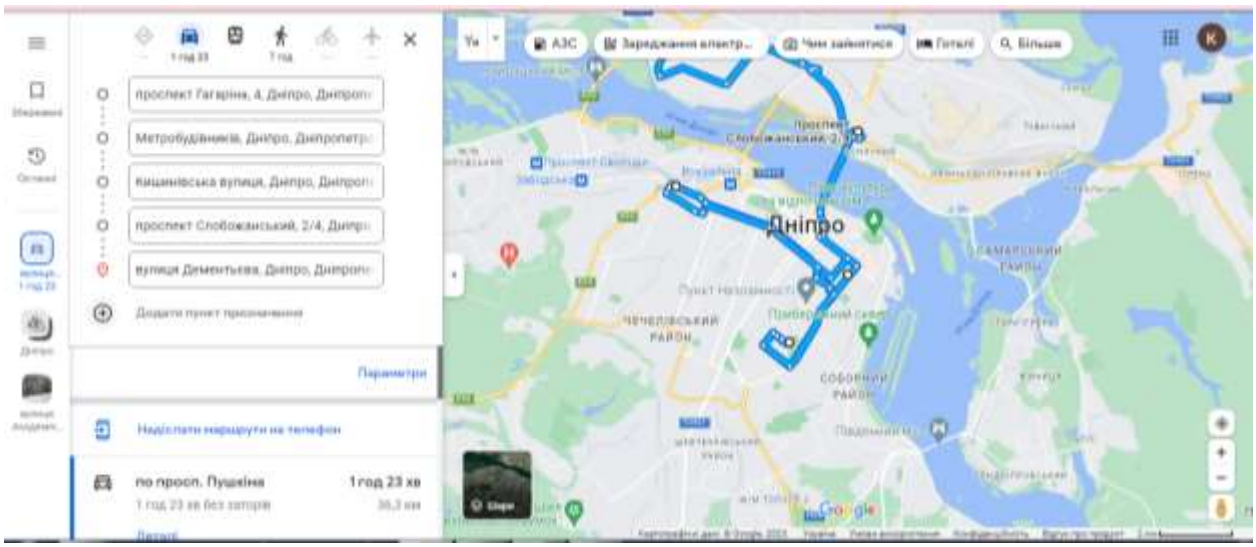

Рис. 1.4. Інтерфейс сервісу Google Maps

Розглянувши існуючі інформаційні системи, що вирішують завдання у сфері логістики, можна зробити висновок про відсутність програмного забезпечення у відкритому доступі, яке б вирішувало задачу формування оптимального маршруту. Хіба що пропонуються пробні версії. Переважно розробка вищевказаних програмних продуктів виконується під замовлення певного споживача, враховуючи особливості специфіки діяльності підприємства. Також у подальшому необхідний супровід і переналаштування параметрів під конкретні умови, що призводить до додаткових витрат. Або є безкоштовні інструменти, наприклад, Google Maps, який будує маршрут у тій послідовності, яку вказує користувач, що не обов’язково відповідає умові оптимальності. Тобто, водій самостійно виконує маршрутизацію своїх рейсів. А для цього потрібно використовувати додаткові сервіси для впорядкування пунктів доставки і ручного перенесення адрес на карту, що призводить до незручностей та вимагає додаткових витрат часу

Саме тому виникає потреба у створенні відкритої інформаційної системи для логістичного підприємства, яка б виконувала функцію впорядкування адрес доставки товарів побудови оптимального маршруту перевезення на онлайн-карті і збереження результатів до бази даних.

1.3. Постановка завдання на розробку систему

Останнім часом спостерігається збільшення звернень до компаній, що пропонують програмні рішення для управління процесами поставок товарів. Окрім завищених очікувань клієнтів, їх звички до стабільності, додалися непередбачувані збої, викликані спочатку всесвітньою пандемією, а потім збройними конфліктами. Ці фактори сприяли збільшенню попиту на інформаційні системи, що стосуються сфери транспортної логістики.

Пандемія COVID-19 відкрила нову еру видимості ланцюга поставок, позначену потрясіннями та збоями. У часи невизначеності вищий рівень безпеки став обов’язковим. Цифровізація ланцюгів постачання пережила значний стрибок, особливо з точки зору видимості. Можливість відстежувати місцезнаходження відправлень і оцінювати час їх прибуття стала критично важливою.

Причин інвестувати у програмне забезпечення для автоматизації транспортної логістики є декілька: економічна доцільність, відповідність вимогам бізнесу, актуальність технологій.

Логістика та автоматизація створили необхідність у переосмисленні, уточненні та покращенні внутрішніх процесів транспортних компаній. Зручне вебрішення для ефективної комунікації та обміну даними має вирішальне значення для компаній, які хочуть йти в ногу з часом, бути готовими до змін у ланцюжку поставок та моніторити процес транспортування.

Розробка інформаційної системи формування оптимального маршруту постачання товарів дозволить знизити витрати на перевезення товару, зменшити час доставки за рахунок оптимального шляху, забезпечити оперативний доступ до клієнтської бази.

Отже, метою даною роботи є створення інформаційної системи на основі платформи .NET для формування оптимального маршруту постачання

товарів та відображення його на карті за допомогою інструмента GMap.NET в режимі онлайн.

Для досягнення поставленої мети треба вирішити наступні завдання:

– описати технологію для вирішення задачі: виконати аналіз алгоритмів формування оптимального шляху перевезення, обгрунтувати вибір алгоритму, враховуючи критерії швидкодії та мінімального споживання пам’яті;

– представити функціональну модель проектованої системи, пояснити взамозв’язок компонентів;

– спроєктувати організацію даних інформаційної системи, навести опис структури бази даних;

– розробити інтерфейс користувача системи;

– виконати програмну реалізацію спроєктованої системи, об’єднати всі компоненти: підключити базу даних, налаштувати карту GMap;

– виконати тестування інформаційної системи.

Інформаційну систему пропонується розробляти мовою програмування C#, вибір якої обґрунтовується рядом переваг.

Мова C# є об’єктно-орієнтованою мовою програмування, а тому базується на чотирьох основних принципах [19]:

Абстракція. Моделювання необхідних атрибутів і взаємодій сутностей у вигляді класів для визначення абстрактного представлення системи.

Інкапсуляція. Приховування внутрішнього стану і функцій об'єкта і надання доступу тільки через відкритий набір функцій.

Наслідування. Можливість створення нових абстракцій на основі існуючих.

Поліморфізм. Можливість реалізації наслідуваних властивостей або методів, що відрізняються способами в рамках безлічі абстракцій.

Останніми роками мова C# входить в п’ятірку найпопулярніших і найперспективніших мов програмування.

Дана мова постійно розвивається та удосконалюється, з’являються нові оновлення.

C# підтримує велику кількість фреймворків для розробки додатків різної функціональності. Одним з популярних є .NET Framework – платформа, яка підтримує створення та виконання веб-служб і програм Windows [20]. Дана технологія створена. щоб:

– підтримувати узгоджене об'єктно-орієнтоване середовище програмування для локального збереження та виконання об'єктного коду, для локального виконання коду, що розподіляється в Інтернеті, або для виконання в дистанційному режимі;

– надавати середовище виконання коду, де зводиться до мінімуму ймовірність виникнення помилок у процесі роботи ПЗ та управління його версіями;

– виконання коду було безпечним, незалежно від того, хто виступає розробником: внутрішній робітник або стороння особа;

– підтримувати єдині принципи розробки для різного ПЗ;

– зв'язок відбувався на основі промислових стандартів, що забезпечує інтеграцію коду платформи .NET Framework з будь-яким іншим кодом.

Для відображення маршруту доставки товарів пропонується використовувати крос-платформну картографічну бібліотека GMap.NET, яка побудована на основі фреймворку .NET.

1.4. Аналіз підходів для вирішення поставленої задачі

Для побудови найкоротшого шляху використовують різні алгоритми на графах: [21-22]:

– пошук у ширину (Breadh-first search, BFS);

– пошук у глибину (Depth-first search, DFS);

– хвильовий алгоритм (ребрами одиничної довжини);

Алгоритм пошуку у ширину (BFS – Breadth First Search) використовує структуру даних черги для пошуку найкоротшого шляху в орієнтованому незваженому графі (рис. 1.5). При такому підході спочатку переглядаються всі вузли на одному рівні, перш ніж переходити до наступного рівня [23]. Обчислювальна складність алгоритму становить $O\ (n + m)$), де n – кількість вершин, m – кількість ребер.

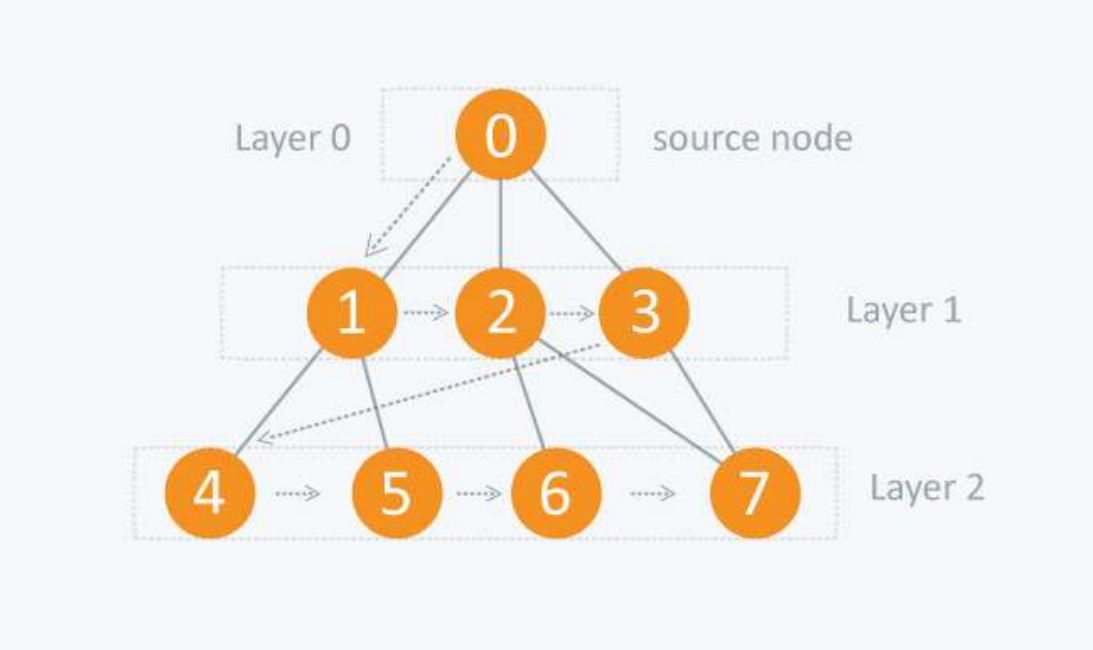

Рис. 1.5. Обхід графа в ширину [23]

Алгоритм пошуку у глибину (DFS Depth First Search) використовує структуру даних стека (рис. 1.6). При такому підході обхід починається з кореневого вузла і продовжується через вузли настільки далеко, наскільки це можливо, доки не досягається вузол без невідвіданих сусідніх вузлів. Обчислювальна складність алгоритму залежить від реалізації графа: через матрицю суміжності алгоритмічна складність $O\ (n^2)$ якщо граф має n вершин; у найгіршому випадку – $O\ (n + m)$, де m – кількість ребер [24].

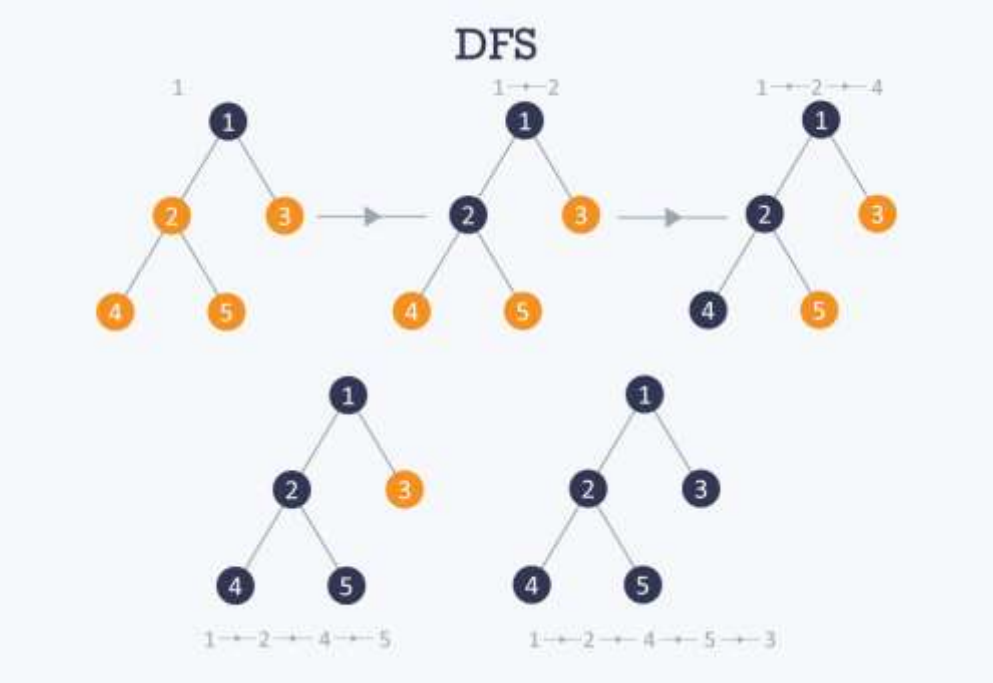

Рис. 1.6. Обхід графа в глибину [25]

Алгоритм Лі (хвильовий), розроблений у 1961 році, базується на алгоритмі пошуку в ширину, використовує хвильове поширення визначення найкоротшого шляху від початкової вершини до цільової вершини. Назва алгоритму дано не випадково, поведінка алгоритму відповідає поширенню хвилі, хвиля огинає перешкоди, поступово заповнюючи весь простір (рис. 1.7) [26]. Складність алгоритму становить *O (m*n).*

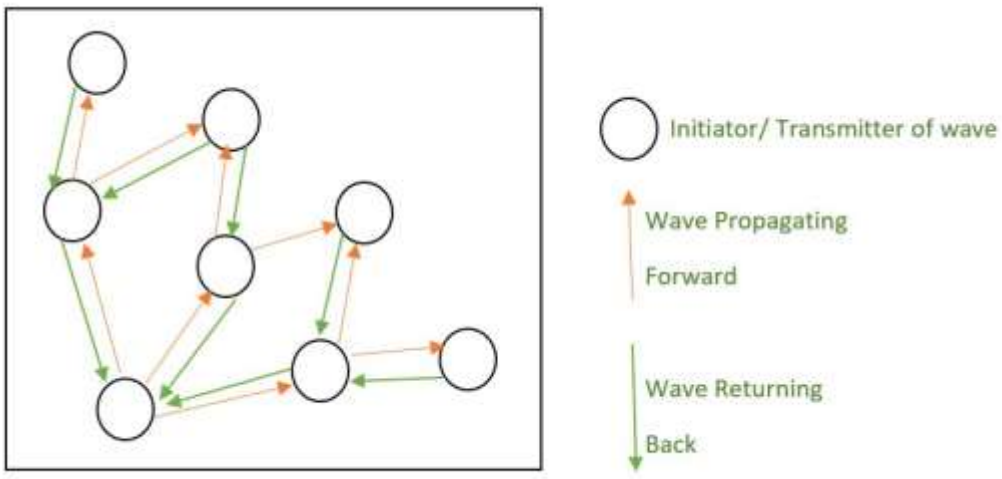

Рис. 1.7. Пошук шляху на основі хвильового алгоритму [26]

Розглянуті вище алгоритми працюють з класичними графами, ребра яких є рівноцінними, а довжина шляху дорівнює сумі довжини ребер, які він містить. Досить часто в задачі кожному ребру оцінюється певним параметром: довжиною ребра, його вартістю, часом проходження. Такий параметр прийнято називати вагою ребра, а граф, що містить зважені ребра, – зваженим [22].

Для пошуку шляху у зваженому графі використовують наступні алгоритми [22]:

– алгоритм Беллмана-Форда;
– алгоритм Дейкстри;
– алгоритм пошуку A*;
– алгоритм Флойда-Воршелла (зважений граф, негативні ребра).

Алгоритм Беллмана-Форда – це ітеративний алгоритм, який знаходить найкоротші шляхи від однієї вершини графа до решти. Запропонований незалежно американцями Річардом Белманом та Лестором Фордом.

Вважається одним з найпростіших з алгоритмічної точки зору, проте й одним з найповільніших. Його перевагою є те, що враховує від'ємні ребра. Але алгоритм використовує повний перебір всіх вершин графа, що можна вважати недоліком, який призводить до значних втрат часу і споживання великого обсягу пам'яті обчислювальної машини. Обчислювальна складність алгоритму дорівнює *O (m*n)* (рис. 1.8) [27].

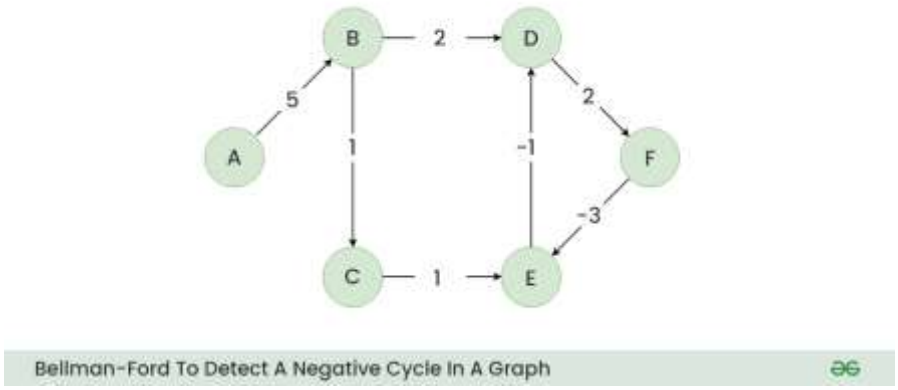

Рис. 1.8. Обхід графа на основі алгоритму Беллмана-Форда [27]

Алгоритм Дейкстри є відомим та поширеним алгоритмом, ої розроблений нідерландським вченим Едсгером Дейкстрой у 1959 році. Використовується для знаходження шляху у зваженому графі від однієї вершини до інших. Алгоритм працює виключно для графів із ребрами позитивної ваги. Обчислювальна складність алгоритму становить O (n^2) (рис. 1.9) [28].

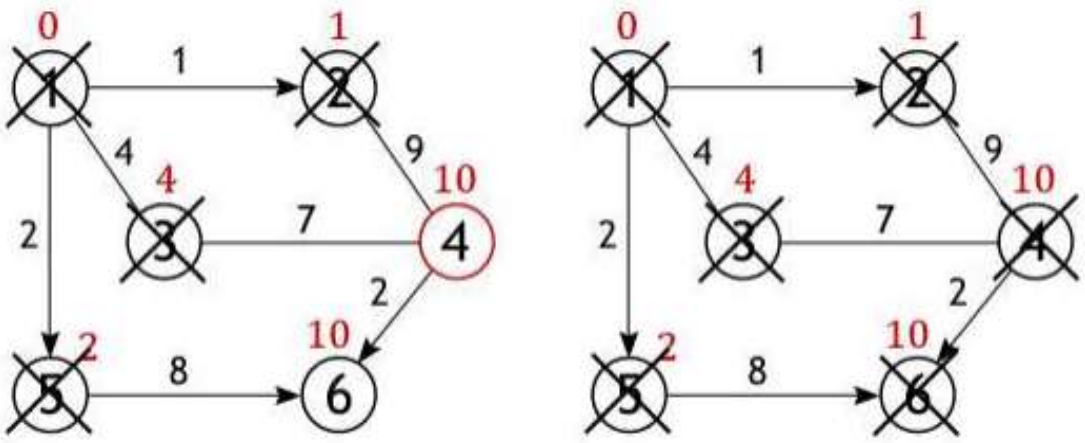

Рис. 1.9. Пошук найкоротшого шляху на основі алгоритму Дейкстри

Даний алгоритм вважається одним із простих. Вирішує задачі у графах з невеликою кількістю вершин і не працює з від'ємними вагами ребер. Може виникнути зациклювання, якщо граф має цикли [28].

Алгоритм пошуку A*. Був розроблений в 1968 Пітером Хартом, Нільсом Ніль-соном і Бертрамом Рафаелем. Цей алгоритм виступає удосконаленим алгоритмом Дейкстри, але є більш продуктивним за рахунок введення в роботу алгоритму евристичної функції (рис. 1.10) [29].

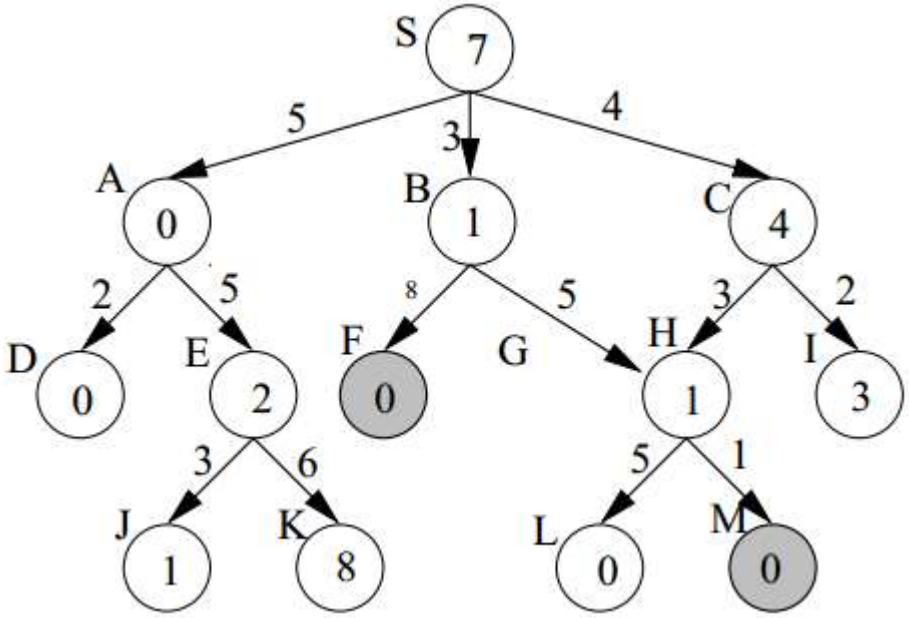

Рис. 1.10. Пошук найкоротшого шляху на основі алгоритму A* [29]

Алгоритм A* виконує пошук, під час якого визначає маршрут з найменшою вартістю від однієї вершини (початкової) до іншої (цільової, кінцевої), шукаючи перший найкращий збіг на графі. Порядок обходу вершин визначається за допомогою евристичної функції «відстань + вартість».

Даний алгоритм послідовно переглядає всі шляхи, що ведуть від початкової вершини до кінцевої, поки не знайде мінімальний. Його недоліком є те, що він здійснює пошук серед усіх вершин графа. У випадку, якщо граф матиме велику кількість вершин, цей алгоритм буде неефективним. Тимчасова обчислювальна складність даного алгоритму залежить від евристики. У гіршому випадку число вершин, що досліджуються алгоритмом, зростає експоненційно у порівнянні з довжиною оптимального шляху, але складність стає поліноміальною, коли простір пошуку є деревом. Але більшою проблемою, ніж часова складність, є споживання великого обсягу пам’яті.

Алгоритм Флойда-Воршелла, розроблений у 1962 році Робертом Флойдом та Стівеном Уоршеллом. Є динамічним алгоритмом для пошуку найкоротших відстаней між усіма вершинами виваженого орієнтованого

графа. Ваги ребер можуть бути негативними, обчислювана складність алгоритму становить $O(n^3)$. При застосуванні його до вирішення завдань на значно розгалуженому графі даний алгоритм матите потребу в додаткових ресурсах обчислювальної машини *(рис. 1.11) [22]*.

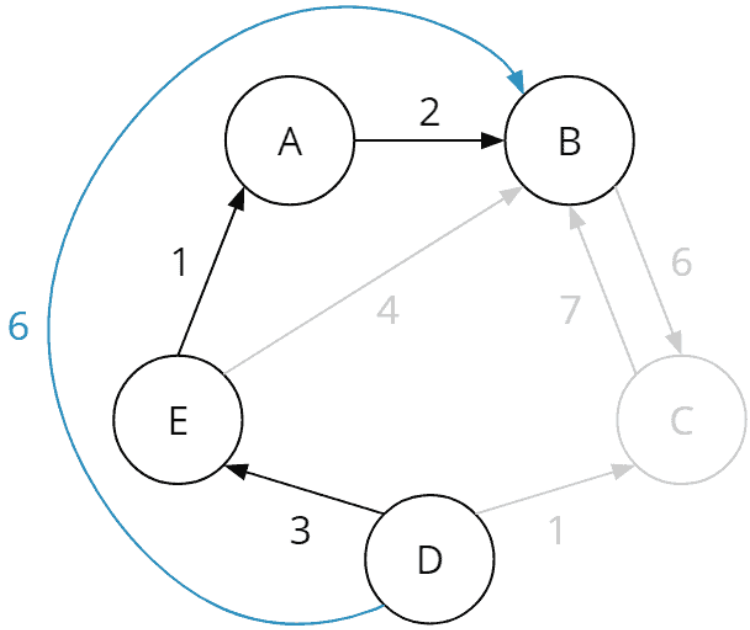

Рис. 1.11. Пошук найкоротшого шляху на основі алгоритму Флойда-Воршелла [22]

Для вирішення задачі формування оптимального шляху між містами можна побудувати мінімальне остовне дерево (minimum spanning tree) графа – це його ациклічний зв'язковий підграф, в який входять всі його вершини, що володіє мінімальною вагою ребер (рис. 1.12) [22].

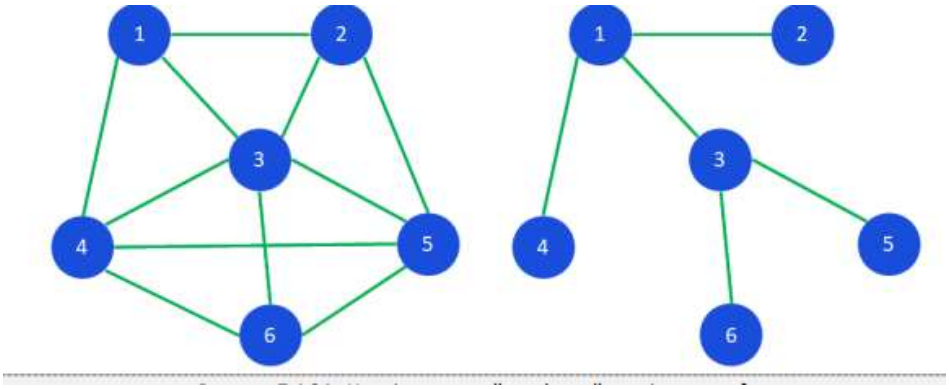

Рис. 1.12. Неорієнтований граф та його каркасне дерево [22]

Граф може містити кілька мінімальних остовних (каркасних) дерев. Мінімальною довжиною дерева зв’язного неорієнтованого графа виступає мінімально можлива вага – сума ваг його ребер, яку можна знайти на основі алгоритмів Прима, Крускала, Борувки.

Перераховані алгоритми дійсно дозволяють знайти найкоротший шлях від однієї вершини до іншої, але для побудови оптимального маршруту серед багатьох вершин слід знайти сумарний шлях, який був би мінімальним, тобто вирішити задачу комівояжера.

РОЗДІЛ 2
ПРОЄКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

2.1. Опис методики для вирішення задачі формування оптимального маршруту доставки товарів

Задача побудови оптимального маршруту перевезення полягає у визначенні найбільш вигідного маршруту об'їзду. Як міра оцінки маршруту може бути сумарний час у дорозі, сумарна вартість дороги або довжина маршруту. Така задача називається задачею комівояжера.

Задача комівояжера вирішує важливе завдання транспортної логістики – галузі, що займається плануванням транспортних перевезень. Комівояжеру, з метою продажу товарів має об'їхати *n* пунктів і зрештою повернутися назад, вихідного пункту. Прийнято вважати, що шлях має проходити через кожен пункт тільки єдиний раз. Це означає, що відповідь знаходиться серед гамільтонових циклів [30].

Задача комівояжера відноситься до задач комбінаторної оптимізації. Аналітична модель виглядає наступним чином [31]:

цільова функція (мінімізація витрат):

$$\sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij} x_{ij} \to \min, \tag{2.1}$$

де c_{ij} – матриця відстаней між усіма містами: $i, j = \overline{1, n}$;

x_{ij} – булеві змінні, 1 – переїзд з міста i до j;

обмеження на одноразовий виїзд з міста:

$$\sum_{j=1}^{n} x_{ij} = 1 \tag{2.2}$$

,

обмеження на одноразовий в’їзд до міста:

$$\sum_{i=1}^{n} x_{ij} = 1 \quad , \tag{2.3}$$

обмеження на наявність єдиного циклу в задачі:

$$u_i - u_j + nx_{ij} \le n-1; i \ne j \, , \tag{2.4}$$

де u – порядковий номер вершини.

До простих методів вирішення завдання комівояжера входять [30]:

– повний лексичний перебір;
– жадібні алгоритми;
– метод включення найближчого міста;
– метод найдешевшого включення;

На практиці застосовують різні модифікації ефективніших методів:

– метод гілок і меж;
– метод генетичних алгоритмів;
– алгоритм мурашиної колонії.

Завдання може бути вирішене грубим перебором усіх варіантів об'їзду та вибором оптимального. Але кількість можливих маршрутів дуже швидко зростає зі зростанням вершин. Для орієнтованого графа з n вершин складність алгоритму становитиме $n!$, для чого потрібно дуже багато часу при значній кількості вершин.

Жадібний алгоритм (greedy algorithm) – алгоритм дозволяє визначати мінімальну відстань методом вибору найкоротшого, ще не вибраного ребра, за умови, що воно не утворює цикл з вже вибраними ребрами. Цей алгоритм на кожній ітерації робить локально найкращий вибір, рухаючись таким чином до оптимального рішення. Недоліком жадібних алгоритмів є незнання

про те, що стоїть перед поточним жадібним станом. Жадібна стратегія може бути неправильною або призвести до неоптимального рішення.

Метод найближчого міста на кожному кроці алгоритму визначає допустимий маршрут по поточній підмножині пунктів обходу, що вже включені до маршруту, додаючи до нього новий пункт із ще не включених у маршрут, у якого буде найближчий сусід з пунктів, що вже належать до маршруту.

Метод найдешевшого включення полягає у тому, що на кожному кроці алгоритму визначається допустимий маршрут по поточній підмножині вершин обходу, які вже додані до маршруту, включаючи до нього нову вершину, додавання якої між деякими суміжними пунктами призводить до мінімального збільшення вартості (довжини) маршруту.

Генетичний алгоритм представляє собою евристичний метод для пошуку, оптимізації та моделювання, що використовує методи послідовного відбору, комбінування та варіації параметрів на основі біологічного принципу природного відбору. Цей метод належить до категорії еволюційних обчислень. Одна з особливостей генетичного алгоритму полягає в застосуванні оператора "схрещування", який включає операцію рекомбінації рішень-кандидатів, схожої на природну функцію схрещування в живій природі [32].

Мурашиний алгоритм – це алгоритм оптимізації, що залежить від поведінки мурах при пошуку шляху від однієї вершини до іншої. Оптимізація за алгоритмом мурашиної колонії була вперше запропонована Марко Доріго у 1992 році. Ця техніка оптимізації будується на стратегії, якої дотримуються мурахи, прокладаючи шлях у пошуках їжі. Крім того, даний алгоритм – це окремий випадок ройового інтелекту. Ройовий інтелект - це різновид штучного інтелекту, де для вирішення завдань застосовується децентралізована колективна поведінка Мурашиний алгоритм використовується для вирішення комбінаторних завдань, зокрема завдання комівояжера. В алгоритмі створюється безліч віртуальних мурах, які

послідовно приймають рішення про вибір наступного міста для відвідування на основі локальної інформації про відстані та феромони [33].

Метод гілок та меж (branch and bound) – загальний алгоритмічний метод оптимізації. Загальна ідея методу у тому, щоб розділити величезну кількість варіантів, що перебираються, на групи і отримати оцінки (нижню оцінку – в задачі мінімізації, верхню оцінку – в задачі максимізації) для цих груп, щоб мати можливість відкидати варіанти не по одному, а одразу цілими групами. Складність методу полягає в тому, щоб знайти такий поділ на групи (гілки) та такі оцінки (кордони), щоб процедура була ефективною.

За своєю суттю він нагадує метод перебору, в процесі реалізації якого задача розбивається на безліч підзадач, кожна з якої, у свою чергу, знову розбивається і т.д.

Даний метод пов'язують із деревом пошуку оптимального (*Topt*) рішення, яке будується у процесі обробки вихідних даних завдання. Рішення – це гілки, що з'єднують вузли дерева. Використання поняття меж та їх розрахунок стимулює чи гальмує зростання гілок у такому дереві. Важливу роль грає процедура розбиття на вузли області допустимих рішень (ОДР) вихідного завдання, тобто на менші непересічні підмножини та їх оцінювання.

Інша процедура, названа процедурою розгалуження, реалізує розбиття на безліч допустимих значень змінної x на підобласті менших розмірів. Ще один важливий елемент методу гілок та меж – процедура обчислення оцінок, яка полягає у пошуку значень меж цільової функції для вирішення задачі. Обчислення нижньої межі цільової функції є найважливішим, ключовим елементом запропонованої схеми.

Таким чином, метод гілок і меж базується на ідеї послідовного розбиття безлічі допустимих рішень на менші області (стратегія «розподіляй і володарюй») та оцінювання отриманих при розбитті частин. Кожен крок алгоритму розбиття супроводжується перевіркою умови, містить конкретна підмножина оптимальне рішення чи ні.

Отже, послідовність виконання дій методу гілок та меж наступна:

1. побудувати матрицю відстаней;

2. виконати редукцію рядків матриці: знайти мінімальне значення кожного рядка і відняти це значення від кожного елемента по рядках;

3. виконати редукцію колонок: знайти мінімальне значення в кожній колонці і відняти це значення від кожного елемента по колонках;

4. розрахувати довжину маршруту (суму мінімальних елементів по строках та по колонках) – поточну локальну нижню межу, яка обчислюється один раз;

5. оцінити кожну нульову клітинку редукційної матриці (обчислити суму мінімальних елементів відповідного рядка та колонки);

6. серед нульових клітинок знайти максимальну оцінку: оцінка на перетині певного рядка і стовпчика і означатиме певний шлях, наприклад, (5, 3) – маршрут з пункту 5 до пункту 3;

7. після цього розглянути дві гілки: з включенням знайденого шляху на попередньому етапі і з виключенням даного шляху;

8. обчислити суму мінімальних значень по рядках і стовпчиках, яку додати до довжини маршруту поточної локальної нижньої межі, порівняти довжини і далі рухатися за гілкою з мінімальним значенням.

На кожному кроці ітерації на основі порівняння нижніх меж підмножин розгалуження виконувати тільки з вершини, що має менше значення нижньої межі.

Значенням цільової функції буде мінімальна довжина всіх відстаней між вершинами графа.

Алгоритм методу гілок та меж представлено на рисунку 2.1.

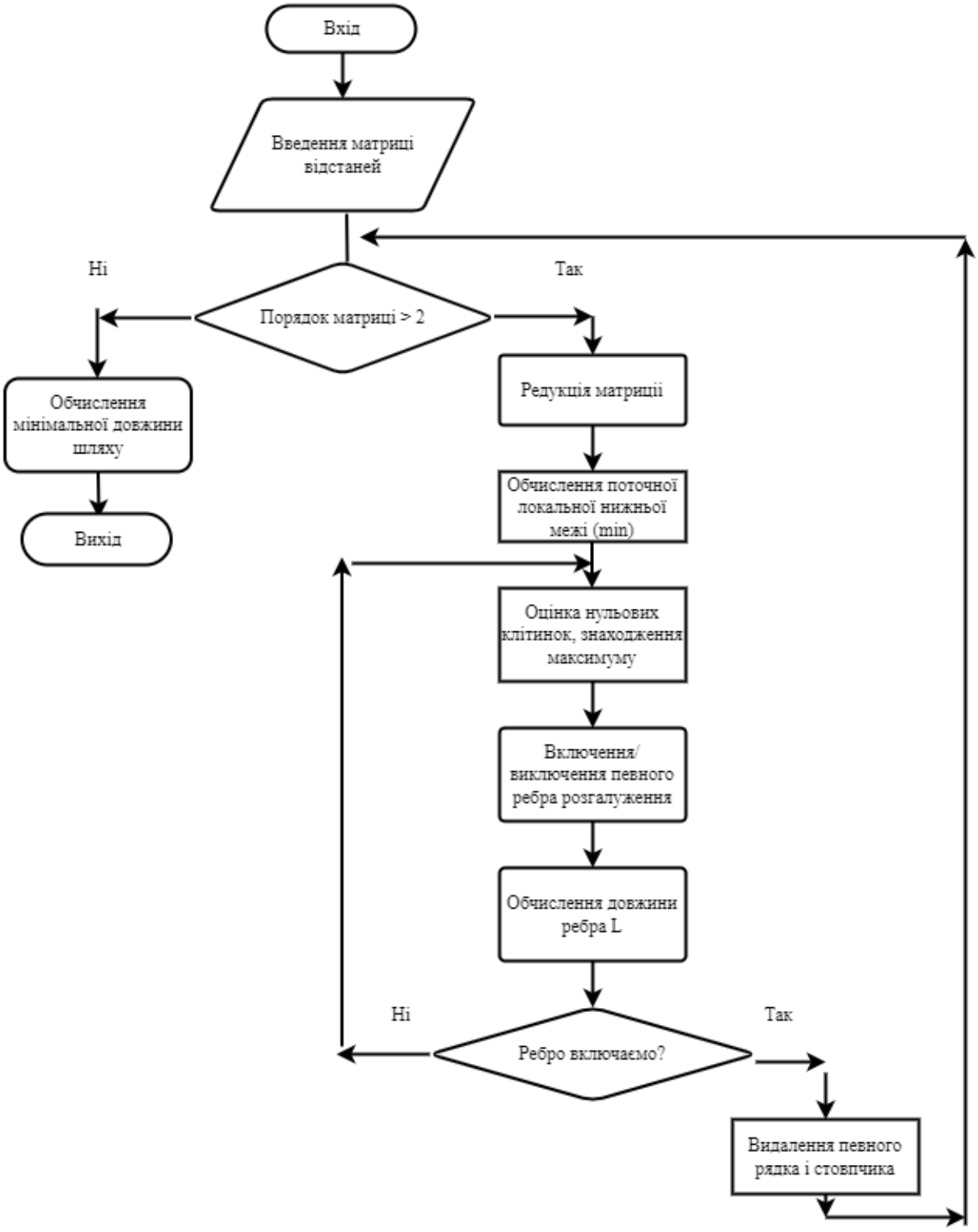

Рис. 2.1. Алгоритм методу гілок та меж *(побудовано на основі [34])*

Отже, серед методів, які вирішують саме задачу комівояжера, прості методи потребують багато часу та пам’яті при значній кількості вершин; генетичний алгоритм та алгоритм мурашиної колонії є евристичними

(наближеними методами), що не дозволяє отримати точного результату. Серед точних методів найбільш ефективним є метод гілок і меж, саме тому на основі нього і вирішуватиметься задача побудови оптимального маршруту доставки товарів.

2.2. Архітектура інформаційної системи

Пропонується створити інформаційну систему, що базується на використанні інформаційних технологій, систем для зберігання даних та інструментарію для розробки програмного забезпечення.

Архітектура системи відображає наступні зв'язки компонентів (рис. 2.2):

– взаємодія з користувачем;
– збереження вхідних даних та результатів розрахунків у базі даних;
– виконання обчислень на основі представленого алгоритму;
– візуалізація результатів за допомогою онлайн-карти.

Рис. 2.2. Архітектура інформаційної системи формування оптимального маршруту постачання товарів

У блоці взаємодії користувача з системою виконуються операції перегляду, редагування, пошуку, видалення інформації, візуального представлення даних. Інтерфейс дозволяє користувачу взаємодіяти з системою під час введення інформації, під час отримання результатів обчислень або рекомендацій.

У блоці зберігання даних використовується база даних типу MS SQL Server, в якій міститься інформація про клієнтів, пункти доставки, виконання замовлень.

Блок обробки інформації забезпечує визначення координат адрес доставки, побудови оптимального шляху та відображення його на карті.

2.3. Функціональна модель проектованої системи

Запропонована інформаційна система працюватиме під управлінням операційних систем Windows. Модуль спілкування передбачає можливість вводу користувачем адреси поставки товарів. Програма обробляє дані, виконує обчислення, відображає результати та зберігає їх у базі даних. Функції системи наведені у таблиці 2.1.

Таблиця 2.1

Функції інформаційної системи

№з/п	Функції	Категорія
1.	Введених даних	Очевидна
2.	Корегування введених даних	Очевидна
3.	Видалення помилково введеної інформації	Очевидна
4.	Пошук інформації у базі даних	Прихована
5.	Визначення координат пунктів поставок	Прихована
6.	Побудова оптимального шляху перевезення	Прихована
7.	Відображення на карті маршруту	Очевидна
8.	Розрахунок відстані та часу доставки	Прихована
9.	Збереження даних у базі даних	Прихована

Проєктування інформаційної системи здійснюється на основі уніфікованої мови програмування UMl, яка використовується при об’єктно-орієнтованому підході.

Проєктовану систему можна представити у вигляді діаграми прецедентів (рис. 2.3), що складається з виконавців, прецедентів (варіантів використання), обмежених границею системи (прямокутником), та асоціацій між виконавцями та прецедентами.

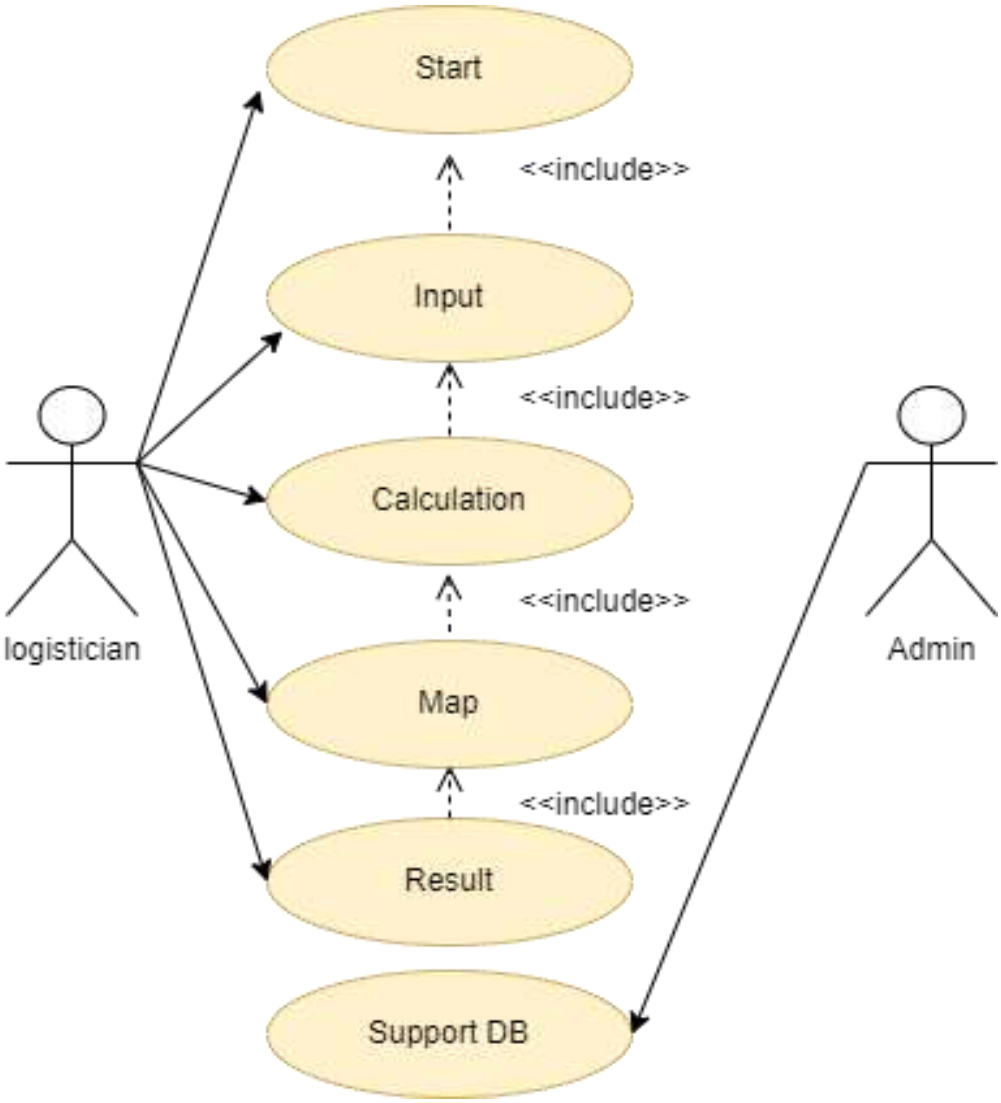

Рис. 2.3. Діаграма прецедентів інформаційної системи

Діаграма прецедентів відображає елементи моделі варіантів використання. Опис кожного реального варіанту використання містить:

сценарій варіанта використання, типовий перебіг подій сценарію, винятки сценарію варіанту використання.

Між акторами та прецедентами встановлено відношення комунікації – Unidirectional Association. Відношення між прецедентами – відношення включення (include), оскільки для побудови маршруту і відображення його на карті, спочатку треба виконати обчислення, а перед цим ввести дані і запустити систему.

Діаграма варіантів використання описує сценарії поведінки та прецеденти, а також виступає вихідним концептуальним відображенням системи під час її проектування та розробки,

Для візуалізації цілісної роботи системи побудовано діаграму діяльності (Activity Diagram), яка візуалізує процес використання та демонструє потік повідомлень від однієї дії до іншої. (dou.ua) Діаграма відображає динамічні аспекти поведінки системи.

Діаграми діяльності призначені для опису координації діяльності для надання послуги, що може виступати на різних рівнях абстракції. Зазвичай подія досягається декількома етапами, особливо якщо операція забезпечує досягнення різних спрямувань, які вимагають координації.

За цих допомогою цих діаграм: визначаються можливі способи використання за допомогою дослідження бізнес-процесів; передумови та післяумови для випадків використання; (dspace.nuft.edu.ua) моделюються робочі процеси між варіантами використання та всередині них; моделюються складні робочі процеси в операціях над об’єктами

На розробленій діаграмі показано комплекс дій, які виконуються користувачем та системою. Діаграма має послідовні та розгалужені потоки (рис. 2.4).

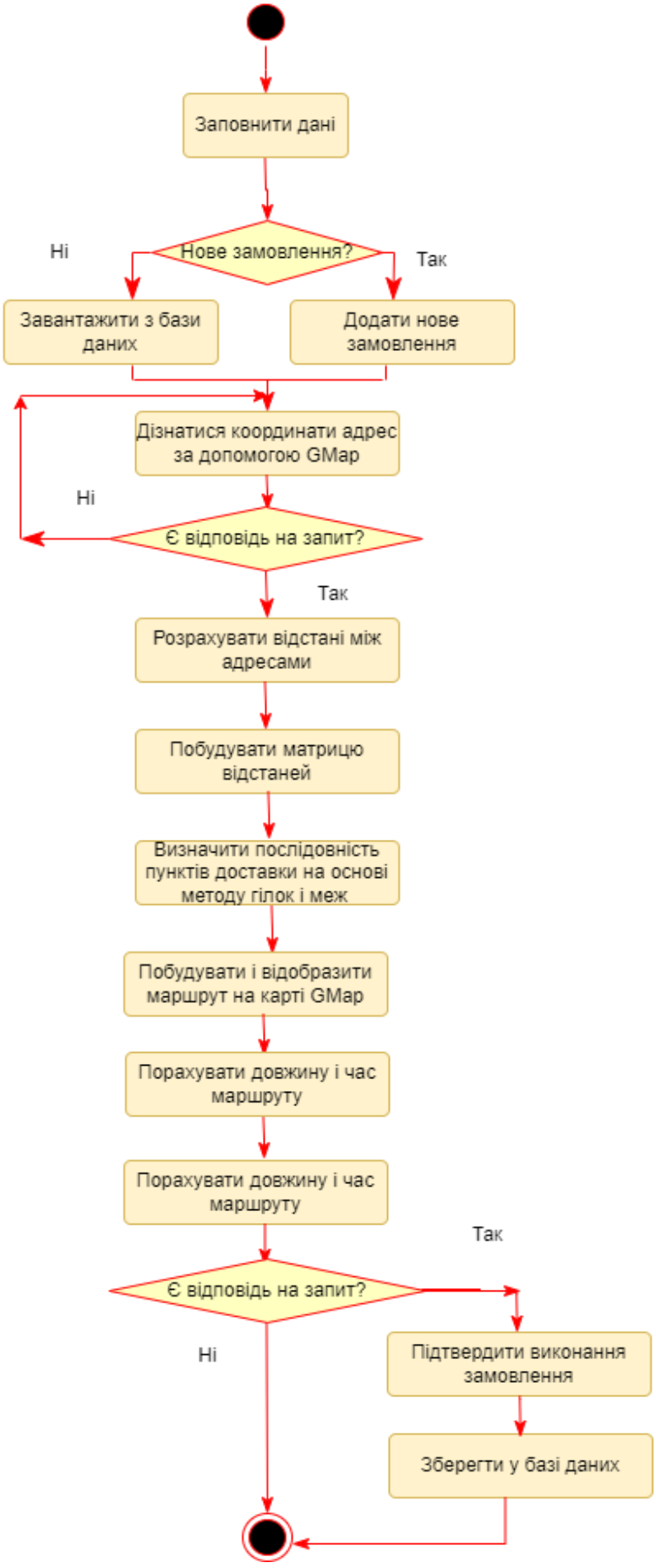

Рис. 2.4. Діаграма діяльності проєктованої інформаційної системи

Отже, в даному підрозділі визначено функції інформаційної системи, для візуалізації роботи системи побудовано діаграму варіантів використання та діаграму діяльності.

2.4. Проєктування бази даних інформаційної системи

Для проєктування бази даних (БД) використовувалася реляційна модель, яка визначає схему відносин між сутностями. Коректність структури бази даних забезпечується процедурою нормалізації.

Існують такі рівні нормалізації [33]:

- перша нормальна форма (1НФ);
- друга нормальна форма 2НФ;
- третя нормальна форма (3НФ);
- нормальна форма Бойса-Кодда (БКНФ);
- четверта нормальна форма (4НФ);
- п’ята нормальна форма (5НФ).

Але сьогодні жодна з реляційних систем управління базами даних не забезпечує належну підтримку усім п'яти нормальним формам. Це відбувається через жорсткі вимоги до продуктивності, оскільки при дотриманні всіх п’яти форм треба з’єднати велику кількість таблиць. Тому на практиці використовують лише перші три рівня нормалізації – 1НФ, 2НФ, ЗНФ [35].

Для виконання нормалізації даних визначено наступні сутності:

- замовники;
- робітники;
- транспорт;
- замовлення;
- виконані замовлення.

Для приведення даних до першої нормальної форми усунуто повторювані групи даних. Передбачено первинний ключ для забезпечення унікальності даних у кожному рядку.

Для приведення даних до другої нормальної форми передбачено залежність кожного атрибута сутності від усього ключа.

На етапі приведення даних до третьої нормальної форми перевірено, чи немає неключових полів (атрибутів), які залежать від інших неключових полів, і чи немає похідних полів. Такі поля відсутні, тому дана вимога виконана.

Отже, до кожної сутності наведено атрибути, встановлено ключові поля. При створенні таблиць вказано властивості кожного поля (рис. 2.5-2.9).

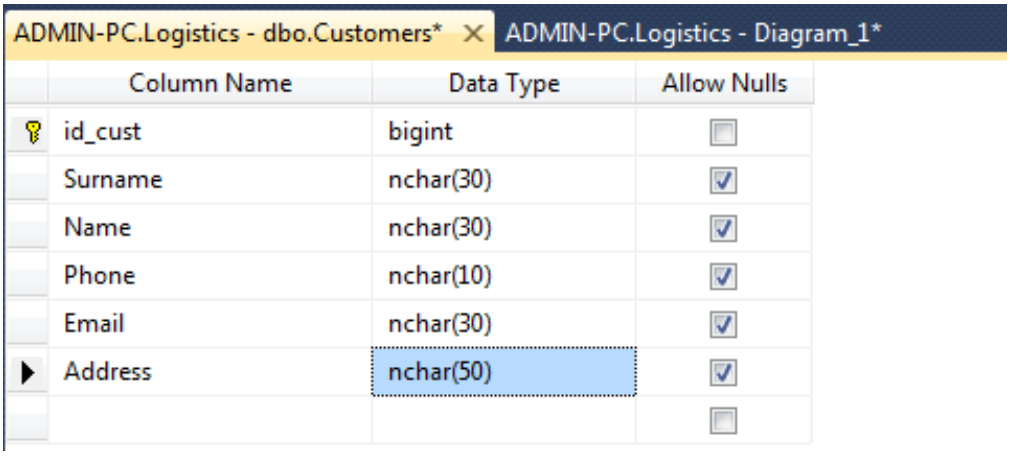

Рис. 2.5. Конструктор таблиці Customers

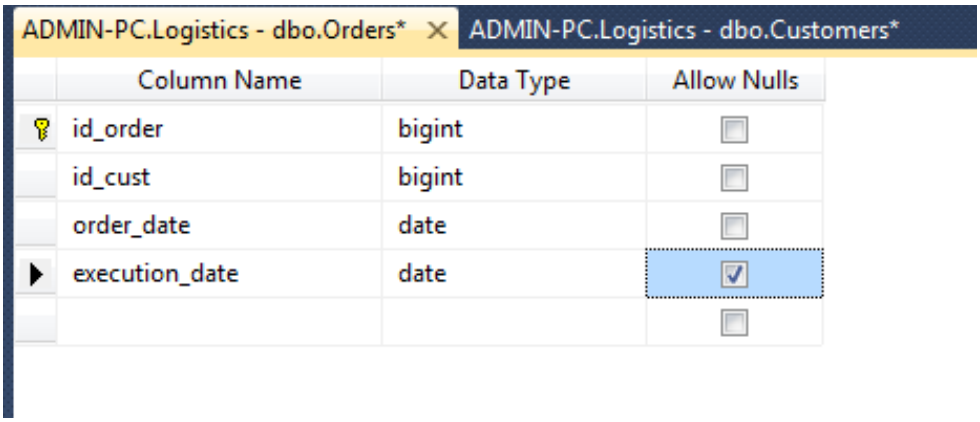

Рис. 2.6. Конструктор таблиці Orders

ADMIN-PC.Logistics - dbo.Workers* | ADMIN-PC.Logistics - dbo.Orders*

Column Name	Data Type	Allow Nulls
id_worker	bigint	☐
id_transport	bigint	☐
Surname	nchar(30)	☐
Name	nchar(30)	☐
phone	nchar(10)	☐
email	nchar(20)	☑
		☐

Рис. 2.7. Конструктор таблиці Workers

ADMIN-PC.Logistics - dbo.Transport* | ADMIN-PC.Logistics - dbo.Workers*

Column Name	Data Type	Allow Nulls
id_transport	bigint	☐
name	nchar(30)	☐
type	nchar(30)	☑
weight	nchar(30)	☑
		☐

Рис. 2.8. Конструктор таблиці Transport

ADMIN-PC.Logistic....Completed _orders* | ADMIN-PC.Logistic....Completed _orde

Column Name	Data Type	Allow Nulls
id_order	bigint	☐
id_worker	bigint	☐
fact_date	date	☐
len_route	float	☐
time	float	☐
		☐

Рис. 2.9. Конструктор таблиці Completed_orders

На рисунку 2.10 наведемо структуру нормалізованої бази даних із вказівкою первинних ключів і зв'язків між таблицями. Дана діаграма містить зв’язки «один-до-одного» та «один-до-багатьох».

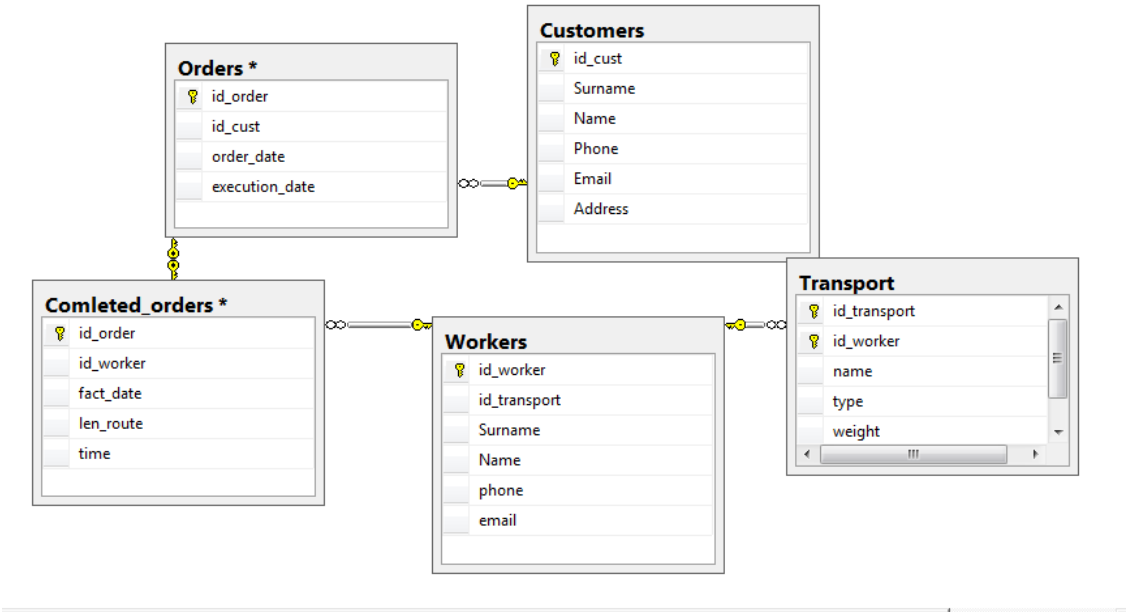

Рис. 2.10. Схема даних

Отже, при проєктування структури бази даних дотримано процедуру нормалізації, яка знижує ймовірність дублювання даних допомагає виявити та усунити функціональні залежності.

2.5. Розробка інтерфейсу користувача системи

Інтерфейс користувача розроблено з використанням WS Forms – платформи, призначеної для розробки графічного інтерфейсу користувача, яка є частиною Microsoft.NET Framework. WS Forms забезпечує доступ до елементів інтерфейсу Microsoft Windows за допомогою створення обгортки для Win32 API в керованому коді.

Інтерфейс користувача повинен відповідати ряду вимог для забезпечення зручності, ефективності та задоволення потреб користувачів. До основних вимог включаються:

Зрозумілість: Інтерфейс повинен бути легким для розуміння користувачем. Інформація та дії повинні бути представлені зрозуміло та логічно.

Доступність: Інтерфейс повинен бути доступним для різних категорій користувачів, включаючи тих, хто має обмеження чи використовує допоміжні технології.

Легкість навігації: Користувач повинен легко розуміти, як переходити між різними частинами інтерфейсу та виконувати необхідні завдання.

Консистентність: Елементи інтерфейсу повинні мати однаковий вигляд і поведінку у всій системі, для того щоб користувачі могли передбачати результати своїх дій.

Відповідність завданням: Інтерфейс має відповідати потребам та завданням користувачів, дозволяючи їм ефективно виконувати необхідні завдання.

Забезпечення повноцінного функціоналу: Інтерфейс повинен надавати доступ до усіх необхідних функцій системи, не обмежуючи користувачів у їхніх можливостях.

Ефективність та швидкість: Інтерфейс має бути ефективним та реагувати на дії користувача швидко, забезпечуючи позитивний досвід взаємодії.

Гнучкість та налаштування: Користувач повинен мати можливість налаштовувати інтерфейс з урахуванням своїх потреб та вподобань.

Візуальна привабливість: Інтерфейс повинен бути естетично приємним, щоб стимулювати користувачів взаємодіяти з системою.

Безпека: Забезпечення безпеки даних та інформації користувачів через відповідні заходи та контроль доступу.

Загальна мета – створити інтерфейс, який дозволяє користувачам ефективно та комфортно взаємодіяти з системою, враховуючи їхні потреби та очікування.

Отже, робота системи починається з відкриття діалогового вікна, де у користувача є можливість знайти в базі даних замовлення на певну дату, використовуючи календар, або додати до таблиці нове замовлення (рис. 2.11-2.12).

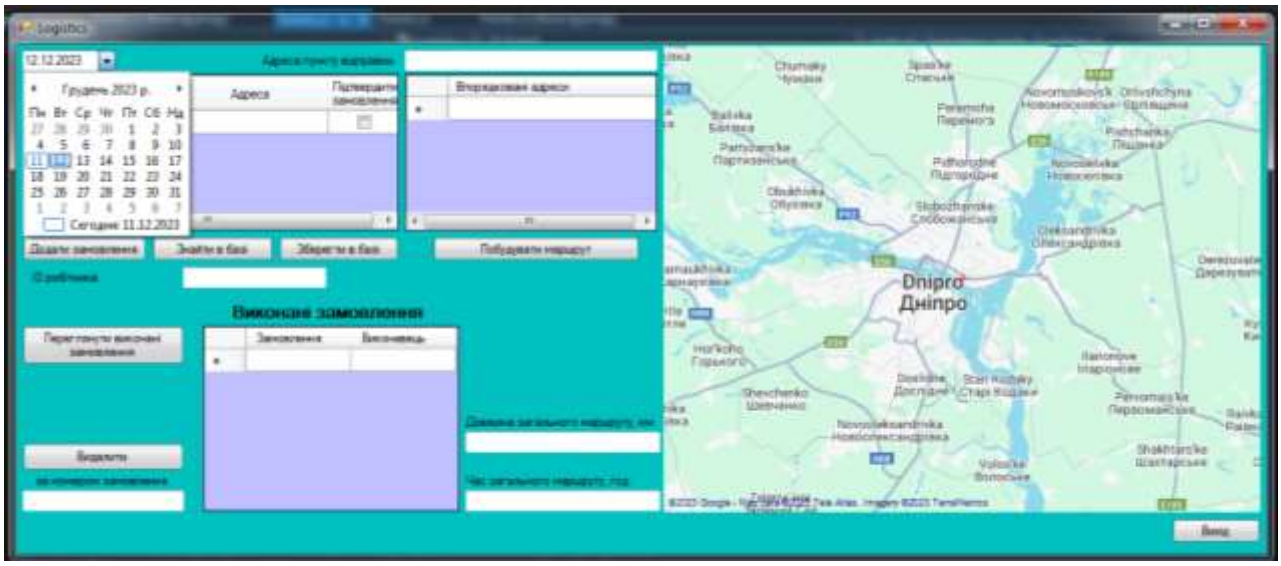

Рис. 2.11. Діалогове вікно IC Logistics

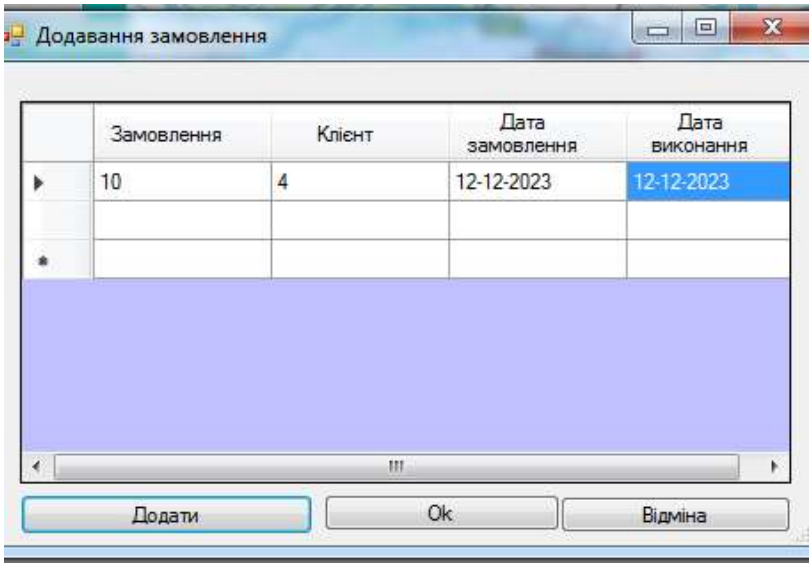

Рис. 2.12. Діалогове вікно додавання нового замовлення

Після натиснення кнопки «Знайти в базі» виконується запит, який повертає інформацію про замовлення, клієнта і адресу. Система виводить дані в таблицю (рис. 2.13).

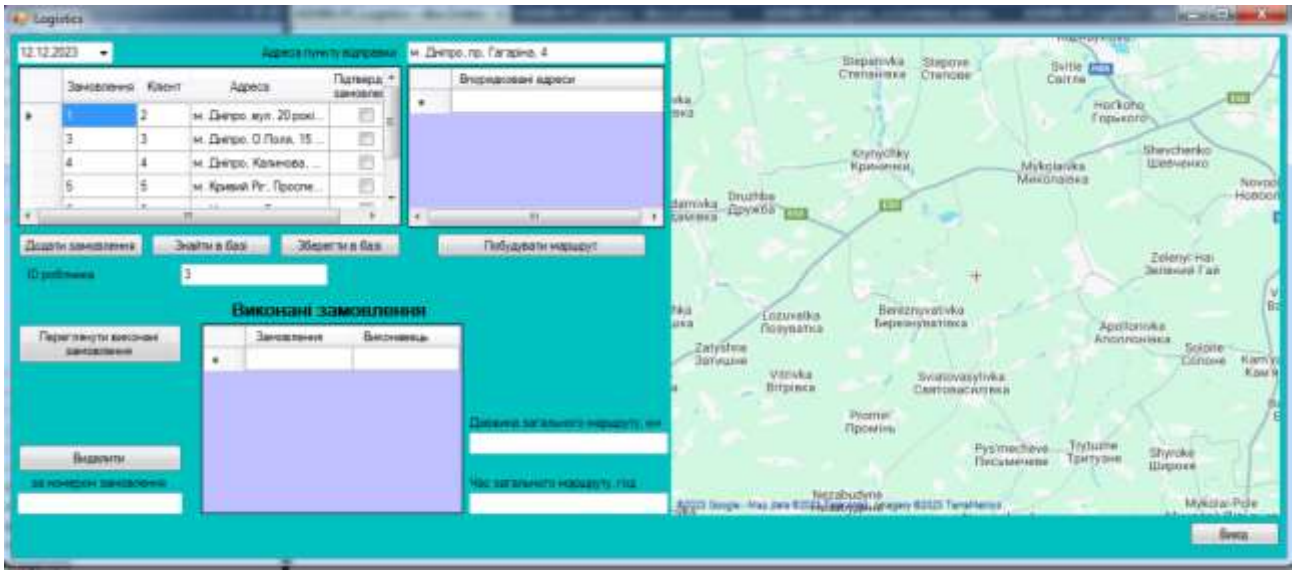

Рис. 2.13. Інтерфейс ІС (запит по замовленнях)

Після чого користувач натискає кнопку «Побудувати маршрут»: виконується обробник подій, на основі методу гілок і меж виводиться послідовність пунктів доставки товару. Система також будує маршрут, розраховує загальну довжину і час маршруту (рис. 2.14).

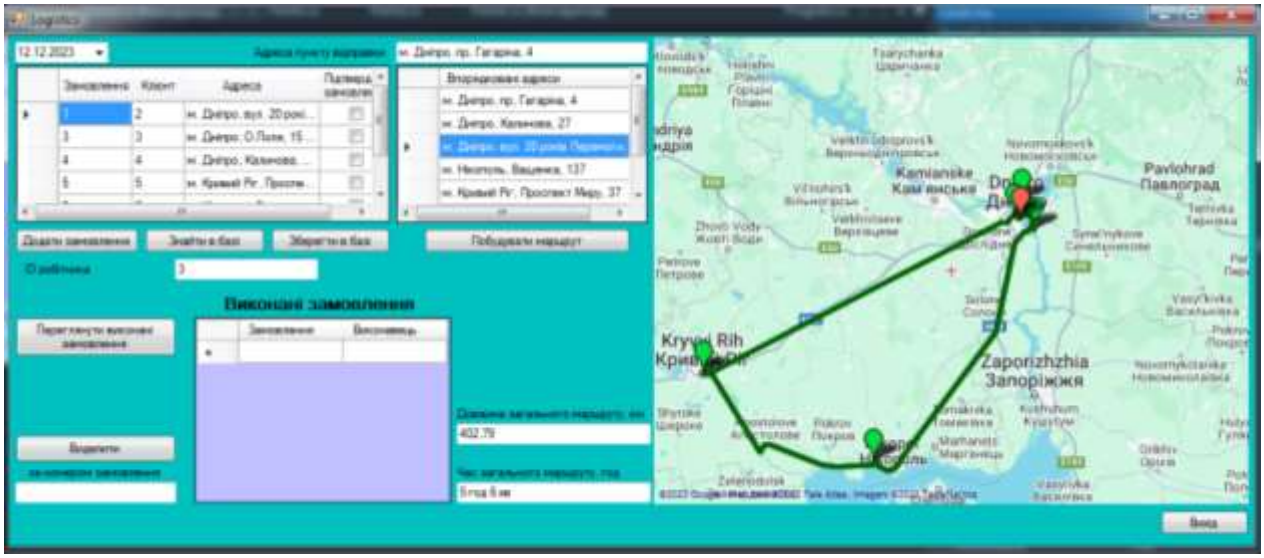

Рисунок 2.14. Інтерфейс ІС (Побудова маршруту на карті GMap)

Якщо доставка товару здійснено, то користувач натискає перемикач, чим підтверджує виконання замовлення, і зберігає інформацію в базі даних. Для перегляду виконаних замовлень слід натиснути на кнопку «Переглянути виконані замовлення» – в таблиці відобразиться інформація окремо по кожному замовленню і виконавцю (рис. 2.15). Також є можливість за необхідності видалити замовлення з бази даних.

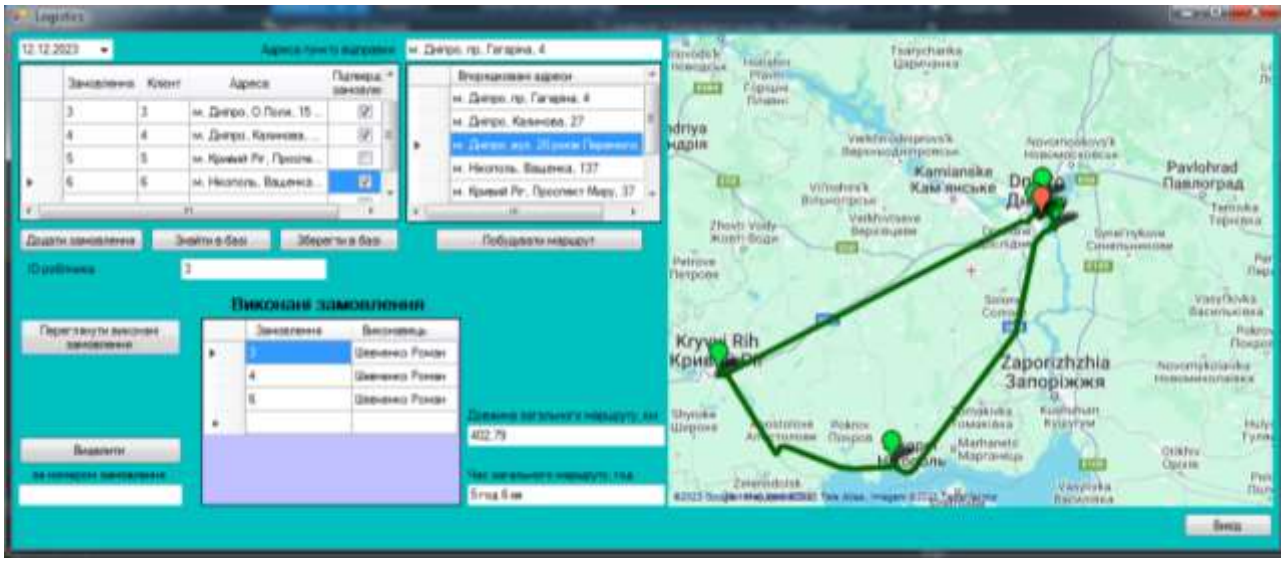

Рисунок 2.15. Інтерфейс ІС (Виконані замовлення)

Отже, розроблений інтерфейс інформаційної системи є доступним та зрозумілим для користувача.

РОЗДІЛ 3
ПРОГРАМНА РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

3.1. Елементи програмної реалізації проєктованої системи

Інформаційна система розроблена мовою програмування C# в середовищі MS Visual Studio 2019 з використанням програмної технології .NET Framework.

Платформа .NET Framework складається із загальномовного середовища виконання (CLR – Common Language Runtime) і бібліотеки класів .NET Framework [36]. CLR керує кодом під час виконання програми та виконує функції керування пам’яттю, потоками та видаленої взаємодії. При цьому накладаються обмеження типізації та інші види перевірки точності коду, що підтримують безпеку та надійність. Фактично головна задача середовища – це управління керованим та некерованим кодом. Підтримується бібліотека класів, що виступає інтегрованою об’єктно-орієнтованою колекцією повторно використовуваних типів, які використовуються для розробки звичайних додатків, додатків із графічним інтерфейсом (GUI) і завантажуючими додатками, що використовують останні технологічні можливості ASP.NET, наприклад веб-форми і веб-служби XML.

Framework. NET може включати некеровані компоненти, які завантажують середовище CLR, до власних процесів, і запускати виконання керованого коду, що дозволяє використовувати засоби як керованого, так і некерованого виконання. Платформа .NET Framework не тільки надає кілька базових засобів виконання, але й підтримує розробку базових засобів виконання незалежними виробниками [37].

Для створення графічного інтерфейсу використовувався фреймворк Windows Forms, який забезпечує взаємодію з користувачем через обробники подій.

Базу даних розроблено у середовищі SQL Server Management Studio. Система управління базами даних MS SQL Server має ряд переваг [38]:

– інтеграція з операційною системою Windows;
– висока продуктивність, відмовостійкість;
– підтримка багатокористувацького середовища;
– розширені функції резервування даних;
– робота з віддаленим підключенням.

Для роботи з картами використано бібліотеку GMap.NET, яка встановлюється за допомогою вільної системи керування пакетами NuGet [39].

Бібліотека GMap.NET забезпечує виконання ряду функцій:

– підтримка різних типів карт;
– побудова маршрутів та полігонів;
– накладання маркерів;
– маршрутизація та геокодування.

Нижче наведено перелік необхідних компонентів, які були підключені для розробки інформаційної системи (рис. 3.1).

```
1   using System;
2   using System.Collections.Generic;
3   using System.ComponentModel;
4   using System.Data;
5   using System.Drawing;
6   using System.Linq;
7   using System.Text;
8   using System.Threading.Tasks;
9   using System.Windows.Forms;
10  using GMap.NET.MapProviders;
11  using GMap.NET.WindowsForms.Markers;
12  using System.Xml;
13  using System.Net;
14  using System.IO;
15  using System.Data.SqlClient;
16
```

Рис. 3.1. Компоненти для розробки ІС

Для підключення до бази даних використано класи бібліотеки System.Data.SqlClient. Програмний код з'єднання з базою даних і пошуку замовлень по даті наведено на рисунку 3.2.

```
private void button1_Click(object sender, EventArgs e)
{
    string connectString = @"Data Source=ADMIN-PC;Initial Catalog=Logistics;" + "Integrated Security=true;";
    SqlConnection myConnection = new SqlConnection(connectString); // встановлюємо зв'язок з БД
    myConnection.Open();
    SqlCommand command = new SqlCommand("select id_order, Customers.id_cust, execution_date, Address " +
    "FROM Orders, Customers where Orders.id_cust = Customers.id_cust and execution_date = @execution_date",
    myConnection);
    command.Parameters.Add("@execution_date", SqlDbType.Date).Value = dateTimePicker1.Value; // пошук замовлень
    SqlDataReader reader = command.ExecuteReader();
    List<string[]> data = new List<string[]>();
    while (reader.Read())
    {
        data.Add(new string[3]);

        data[data.Count - 1][0] = reader[0].ToString();
        data[data.Count - 1][1] = reader[1].ToString();
        data[data.Count - 1][2] = reader[3].ToString();
    }
    reader.Close();
    myConnection.Close();
    dataGridView1.Rows.Clear();
    foreach (string[] s in data)
      dataGridView1.Rows.Add(s);
}
```

Рис. 3.2. Фрагмент програмного коду підключення до бази даних

Також створено інші запити мовою SQL.

Додавання нового замовлення:

string query = "INSERT INTO [Orders](id_order, id_cust, order_date, execution_date) VALUES('" + dataGridView1.Rows[0].Cells[0].Value + "', '" + dataGridView1.Rows[0].Cells[1].Value + "','" + dataGridView1.Rows[0].Cells[2].Value + "', '" + dataGridView1.Rows[0].Cells[3].Value +"')";

Збереження даних по виконаних замовленнях:

query = "INSERT INTO[Comleted_orders](id_order, id_worker, fact_date, len_route, time) VALUES('" + id + "', '" + Convert.ToInt32(textBox1.Text) + "', '" + date + "', '" + _len[i] + "', '" + _time[i] + "')";

Пошук виконаних замовлень:

SqlCommand command = new SqlCommand("select id_order, Surname, Name, fact_date, len_route, time FROM Comleted_orders, Workers where Workers.id_worker = Comleted_orders.id_worker", myConnection);

Видалення вибраних замовлень:

SqlCommand command = new SqlCommand("delete FROM Comleted_orders where id_order = " + textBox5.Text, myConnection);

Для завантаження і відкриття карти GMap встановлено наступні налаштування:

gMapControl1.MapProvider = GMapProviders.GoogleMap;

```
gMapControl1.Bearing = 0;
gMapControl1.CanDragMap = true;
gMapControl1.DragButton = MouseButtons.Left;
gMapControl1.GrayScaleMode = true;
gMapControl1.MarkersEnabled = true;
gMapControl1.MaxZoom = 18;
gMapControl1.MinZoom = 2;
gMapControl1.MouseWheelZoomType = GMap.NET.MouseWheelZoomType.MousePositionAndCenter;
gMapControl1.NegativeMode = false;
gMapControl1.PolygonsEnabled = true;
gMapControl1.RoutesEnabled = true;
gMapControl1.ShowTileGridLines = false;
gMapControl1.Zoom = 10;
gMapControl1.MapProvider = GMap.NET.MapProviders.GoogleMapProvider.Instance;
GMap.NET.GMaps.Instance.Mode = GMap.NET.AccessMode.ServerOnly;
```

Для взаємодії з сервісом Google Maps було створено API ключ, за допомогою якого можна виконати запит і отримати координати адрес для побудови маршруту.

Нижче наведено фрагмент програмного коду для виконання інтернет-запиту:

```
url = new string[count, count];
HttpWebRequest[,] request = new HttpWebRequest[count, count];
WebResponse[,] response = new WebResponse[count, count];
Stream[,] dataStream = new Stream[count, count];
StreamReader[,] sreader = new StreamReader[count, count];
string[,] responsereader = new string[count, count];
string[,] s = new string[count, count];
string[] addr = new string[count + 1];
 addr[0] = textBox4.Text;
 if (textBox4.Text != "")
 { addr[count] = textBox4.Text;
 for (int _i = 1; _i < count; _i++) {
addr[_i] = dataGridView1.Rows[_i - 1].Cells[2].Value.ToString(); // масив адрес
 }
 for (int _i = 1; _i < count + 1; _i++)
 { for (int j = 0; j < count; j++) {
 url[_i - 1, j] = string.Format(zapros, Uri.EscapeDataString(addr[_i - 1]), Uri.EscapeDataString(addr[j]),
Uri.EscapeDataString("driving"));
 // Виконуємо запити
 request[_i - 1, j] = (HttpWebRequest)WebRequest.Create(url[_i - 1, j]);
 // отримуємо відповіді
 response[_i - 1, j] = request[_i - 1, j].GetResponse();
 // читання даних з інтернет-ресурсу
 dataStream[_i - 1, j] = response[_i - 1, j].GetResponseStream();
```

```
//читаємо
sreader[_i - 1, j] = new StreamReader(dataStream[_i - 1, j]);
// читає поток від поточного положення і до кінця
responsereader[_i - 1, j] = sreader[_i - 1, j].ReadToEnd();
response[_i - 1, j].Close(); // закриваємо поток
} }
```

Далі виконано парсинг даних:

```
// таблиця відстаней, часу, довжини шляху між адресами
double[,] table = new double[count + 1, 7];
XmlDocument[,] xmldoc = new XmlDocument[count, count];
XmlNodeList[,] nodes = new XmlNodeList[count, count];
string[,] d = new string[count, count]; // матриця відстаней
for (int index= 0; index< count; index++)
{
for (int j = 0; j < count; j++)
{
xmldoc[index, j] = new XmlDocument();
xmldoc[index, j].LoadXml(responsereader[index, j]);
 if (xmldoc[index, j].GetElementsByTagName("status")[0].ChildNodes[0].InnerText == "OK")
{
nodes[index, j] = xmldoc[index, j].SelectNodes("//leg//step");
// матриця відстаней
d[index, j] = xmldoc[index, j].GetElementsByTagName("distance")[nodes[index, j].Count].ChildNodes[1].InnerText;
// відстані між точками для матриці гілок і меж
// додавання рядка в таблицю
if (j == index+ 1)
{
table[index, 0] = index;
table[index, 1] = XmlConvert.ToDouble(xmldoc[index, j].GetElementsByTagName("start_location")[nodes[index,
j].Count].ChildNodes[0].InnerText);
table[index, 2] = XmlConvert.ToDouble(xmldoc[index, j].GetElementsByTagName("start_location")[nodes[index,
j].Count].ChildNodes[1].InnerText);
table[index, 3] = XmlConvert.ToDouble(xmldoc[index, j].GetElementsByTagName("end_location")[nodes[index,
j].Count].ChildNodes[0].InnerText);
table[index, 4] = XmlConvert.ToDouble(xmldoc[index, j].GetElementsByTagName("end_location")[nodes[index,
j].Count].ChildNodes[1].InnerText); } } } }
```

Адреси впорядковано відповідно до результатів, отриманих за методом гілок і меж:

```
// перестановка адрес в xml
XmlDocument[] xmldoc_new = new XmlDocument[count];
XmlNodeList[] nodes_new = new XmlNodeList[count];
 for (int _i = 0; _i < count; _i++)
{ xmldoc_new[_i] = xmldoc[K[_i] - 1, K[_i + 1] - 1];
nodes_new[_i] = nodes[K[_i] - 1, K[_i + 1] - 1];  }
```

Окремо створено таблицю для збереження координат початкових і кінцевих точок, відстаней і часу окремих відрізків:

```
DataTable dtRouter = new DataTable();
dtRouter.Columns.Add("№ з/п");
dtRouter.Columns.Add("Початкова точка X (latitude)");
dtRouter.Columns.Add("Початковаточка У (longitude)");
dtRouter.Columns.Add("Кінцева точка X (latitude)");
dtRouter.Columns.Add("Кінцева точка У (longitude)");
dtRouter.Columns.Add("Відстань");
dtRouter.Columns.Add("Час шляху");
 // Координати початкової точки шляху
double[] latStart = new double[count];
double[] lngStart = new double[count];
 // Координати кінцевої точки шляху
double[] latEnd = new double[count];
double[] lngEnd = new double[count];
```

Для створення слоїв, маркерів і маршрутів використано наступні класи:

```
// Слої для маркерів і маршрутів
 GMap.NET.WindowsForms.GMapOverlay[]overlaymarshrut = new Map.NET.WindowsForms.GMapOverlay[count];
 // Маркери и їх підписи
 GMarkerGoogle[] markerG = new GMarkerGoogle[count];
 GMarkerGoogle[] markerR = new GMarkerGoogle[count];
 // Створюємо список контрольних точок для побудови маршруту
 List<GMap.NET.PointLatLng>[] list = new List<GMap.NET.PointLatLng>[count];
 GMap.NET.WindowsForms.GMapRoute[] r = new GMap.NET.WindowsForms.GMapRoute[count];
```

На основі зчитаних координат адрес побудовано і відображено маршрут на карті:

```
// Слої для маркерів и маршрутів
 overlaymarshrut[i] = new GMap.NET.WindowsForms.GMapOverlay("overlaymarshrut");
 // Маркери и їх підписи
 markerG[i] = new GMarkerGoogle(new GMap.NET.PointLatLng(latStart[i], lngStart[i]),
GMarkerGoogleType.green);
 markerR[i] = new GMarkerGoogle(new GMap.NET.PointLatLng(latEnd[i], lngEnd[i]), GMarkerGoogleType.red);
 overlaymarshrut[i].Markers.Add(markerG[i]);
 overlaymarshrut[i].Markers.Add(markerR[i]);
 markerG[i].ToolTipText = i + 1 + ". " + addr[K[i] - 1]; //Текст всплывающих подсказок при наведении
 // Створюємо список контрольних точок для побудови маршруту
 list[i] = new List<GMap.NET.PointLatLng>();
 // Зчитуємо з таблиці координати контрольних точок маршрутів і зберігаємо в список координат
 for (int z = 0; z < dtRouter.Rows.Count; z++)
 {
 double dbStartLat = double.Parse(dtRouter.Rows[z].ItemArray[1].ToString(),
System.Globalization.CultureInfo.InvariantCulture);
```

```
double dbStartLng = double.Parse(dtRouter.Rows[z].ItemArray[2].ToString(),
System.Globalization.CultureInfo.InvariantCulture);
list[i].Add(new GMap.NET.PointLatLng(dbStartLat, dbStartLng));
double dbEndLat = double.Parse(dtRouter.Rows[z].ItemArray[3].ToString(),
System.Globalization.CultureInfo.InvariantCulture);
double dbEndLng = double.Parse(dtRouter.Rows[z].ItemArray[4].ToString(),
System.Globalization.CultureInfo.InvariantCulture);
list[i].Add(new GMap.NET.PointLatLng(dbEndLat, dbEndLng)); }
// Очищуємо всі маршрути
overlaymarshrut[i].Routes.Clear();
// Створємо маршрут на основі списку контрольних точек
r[i] = new GMap.NET.WindowsForms.GMapRoute(list[i], "Route");
// Указуєм, що даний маршрут має відображатися
r[i].IsVisible = true;
// Встановлюємо колір
r[i].Stroke.Color = Color.DarkGreen;
// Додаємо маршрут
overlaymarshrut[i].Routes.Add(r[i]);
// Додаємо до компоненту, список маркерів і маршрутів
gMapControl1.Overlays.Add(overlaymarshrut[i]);
textBox2.Text = len.ToString();
textBox3.Text = Math.Round((time / 60)).ToString() + " год " + Math.Round((time % 60)).ToString() + " хв";}
```

Отже, у даному підрозділі описано вибрані технології програмування і наведено фрагменти програмного коду розробленої інформаційної системи.

3.2. Тестування інформаційної системи

Будь-яка інформаційна система перед її використанням має бути протестована.

Мета тестування полягає в тому, щоб переконатися, що додаток не містить помилок і працює відповідно до заданих функціональних вимог. Набори тестів (ключі) розробляються на етапі тестування і повинні бути перевірені. Вони також забезпечують виправлення зареєстрованих дефектів розробниками та повторне тестування виправлених дефектів тестувальниками. Ручне тестування перевіряє якість системи і надає клієнту продукт без помилок. Один із головних принципів тестування говорить про те, що 100% автоматизація неможлива. Саме тому ручне тестування є обов’язковим [40].

Для зменшення помилок при роботі з системою у програмному коді використовувалися оператори обробки виключень throw, try-catch, try-finallyи try-catch-finally.

Також для цього призначені вбудовані класи виключень, наприклад, ArgumentOutOfRangeExceptionInvalidOperationException. .NET також надає допоміжні методи для створення виключень у певних умовах: ArgumentNullException.ThrowIfNull, ArgumentException.ThrowIfNullOrEmpty. Або визначаються власні класи виключень, похідні від System.Exception [40].

При виникненні виключення середовище CLR шукає блок catch , який може обробляти це виключення. Якщо поточний метод catch не містить такого блоку, середовище CLR переглядає метод, який відкриває поточний метод в стеці викликів. Якщо блок catch не знайдено, середовище CLR завершує виконання потоку [40].

Оператор try можна використовувати в будь-якій із наступних форм: try-catch – для обробки виключень, які можуть виникнути під час виконання коду всередині блоку try; try-finally виконується при виході елемента try з блоку управління; try-catch -finally – у вигляді поєднання двох попередніх форм [41].

Якщо у створеній інформаційній системі користувач спробує побудувати маршрут, не вказавши при цьому адресу відправлення або вказавши дані невірного формату, то система виведе попередження (рис. 3.3).

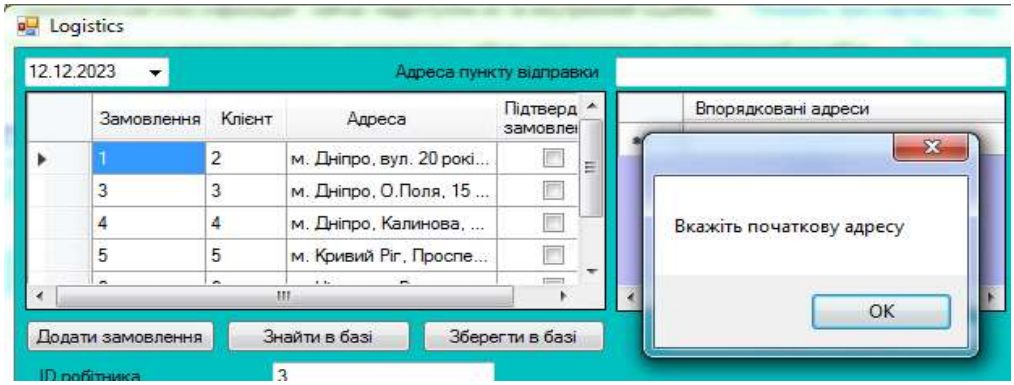

Рис. 3.3. Результат попередження про незаповнені дані або некоректний формат даних

Якщо немає доступу до інтернет-даних, то отримуємо наступне повідомлення (рис. 3.3).

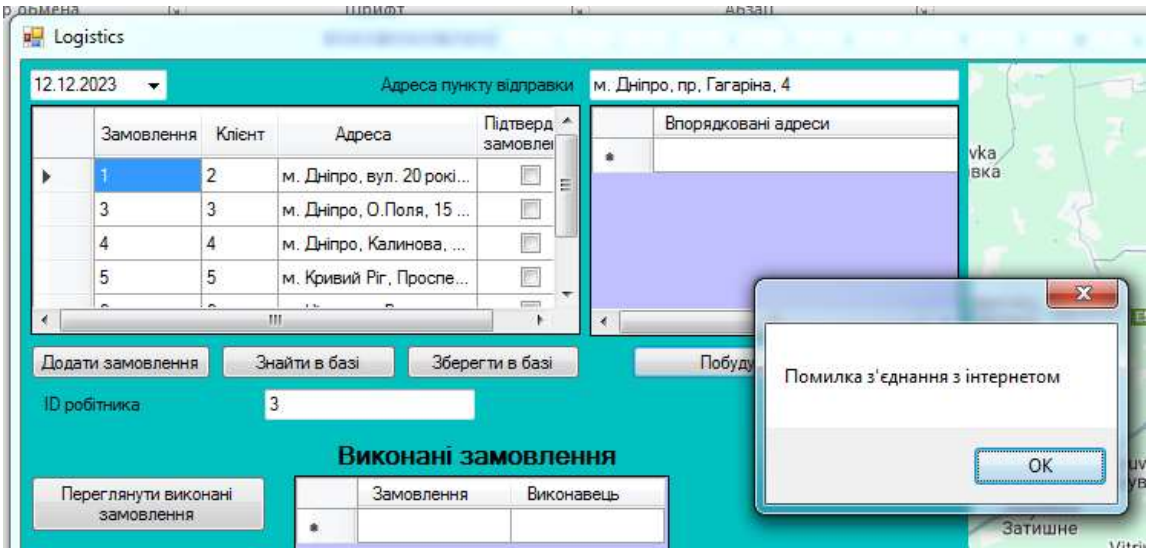

Рис.3.3. Результат попередження про відсутність доступу до інтернет-даних

У випадку збереження незаповнених даних у базу даних система відображає наступне повідомлення (рис. 3.4).

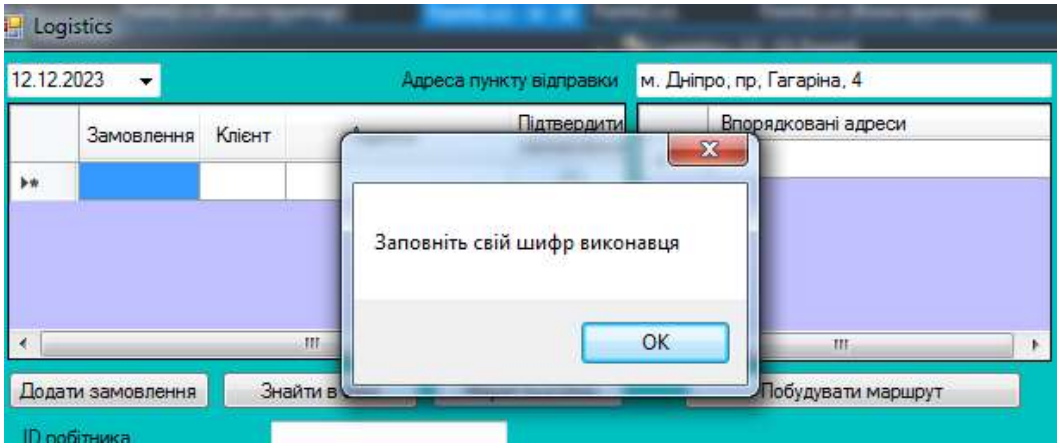

Рис. 3.4. Результат попередження про незаповнені дані або некоректний формат даних при збереженні до бази даних

При спробі видалити дані з бази даних, не вказавши номер замовлення, на екрані відкриється діалогове вікно (рис. 3.5).

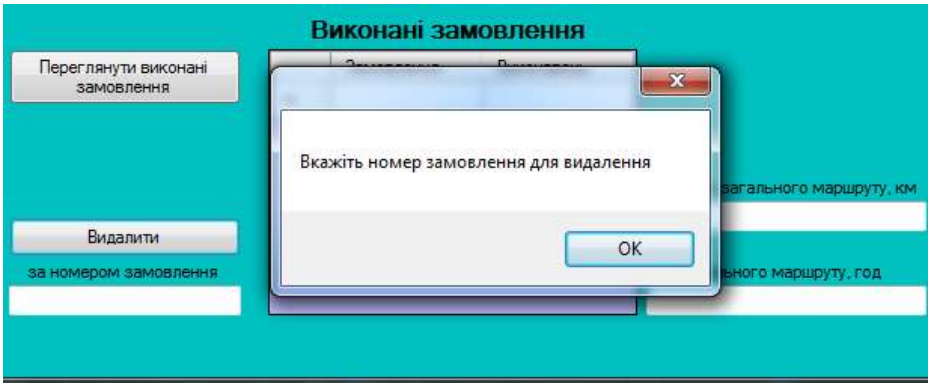

Рис. 3.5. Результат попередження про незаповнені дані або некоректний формат даних при видаленні з бази даних

Отже, у даному підрозділі наведено результати тестування розробленої системи.

ВИСНОВОК

Під час роботи над кваліфікаційною роботою магістра відповідно до поставленої мети було створено інформаційну системи на основі платформи .NET для формування оптимального маршруту постачання товарів та відображення його на карті за допомогою інструмента GMap.NET в режимі онлайн.

Для досягнення поставленої мети було досліджено предметну область, розглянуто існуючі інформаційні системи формування оптимального маршруту доставки товарів, проаналізовано підходи для вирішення поставленої задачі.

Описано методику для вирішення задачі формування оптимального маршруту доставки товарів, представлено алгоритм методу гілок та меж, вибір якого обумовлюється точністю та низькими вимогами до обсягів необхідної пам'яті.

Наведено архітектуру інформаційної системи, представлено функціональну модель, спроєктовано базу даних інформаційної системи, створено інтерфейс системи, який є простим та доступним для користувача.

Інформаційну систему розроблено мовою програмування C# у середовищі програмування MS Visual Studio 2019 з використанням Microsoft .NET Framework 4.8. Для геокодування даних та побудови маршруту в режимі онлайн підключено бібліотеку GMap.NET. Базу даних розроблено в середовищі MS SQL Server.

Практичне значення результатів кваліфікаційної роботи полягає у розробці додатку, який дозволяє визначати послідовність пунктів доставки товарів, відображати маршрут перевезення на онлайн-карті і взаємодіяти з клієнтською базою даних.

Отже, завдання на кваліфікаційну роботу магістра виконано, мети досягнуто.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Марченко В. М., Шутюк В. В. Логістика: Підручник. К.: Видавничий дім «Артек», 2018. 312 с.

2. Біліченко В. В., Бурєнніков Ю. Ю. Основи логістики : навчальний посібник. Вінниця : ВНТУ, 2017. 129 с.

3. Резнік Н. П., Руденко С. В., Пилипчук К. М. Основні характеристики поняття логістики і системи управління ланцюгами постачань. *Innovation and Sustainability*. 2022. (3). С. 95-102.

4. Заріпова К. А. Визначення, види та функції логістичних інформаційних систем. *Сучасні підходи до управління підприємством*. URL : http://conf.management.fmm.kpi.ua/proc/article/view/129610 (дата звернення: 22.10.2023)

5. Гайдабрус Н. В., Біловодська О. А. Аналіз сервісу як складової логістичного забезпечення інноваційної діяльності підприємства. *Проблеми науки*. 2013. №2 (146). С. 37-44.

6. Удачина К.О., Садиков Б. С. Особливості розвитку процесів логістичних підприємств України в сучасних умовах. Сучасні інформаційні технології: теорія, практика, перспективи : збірник тез доповідей Всеукраїнської науково-практичної конференції здобувачів вищої освіти і молодих учених, Дніпро, 4-5 грудня 2023 р. Дніпро : УДУНТ, 2023. С. 55-56.

7. Що таке логістика : вебсайт. URL : http://bigenergy.com.ua/fnansi/bznes-dlya-pdpri/1122-logistika--shho-ce-take-i-dlya-chogo-vona-potribna.html (дата звернення: 20.10.2023).

8. Логістика під час війни: як українському бізнесу організувати транспортні потоки : вебсайт. URL: http://surl.li/ddzjw (дата звернення: 22.10.2023).

9. Рік війни: як конфлікт в Україні вплинув на транспортний сектор Європи? : вебсайт. URL: https://trans.info/ua/rik-viyny-yak-konflikt-v-ukrayini-vplynuv-na-transportnyi-sektor-yevropy-328130 (дата звернення: 27.10.2023).

10. 6 Practices to Optimize Your Goods Transportation Process! : вебсайт. URL : https://supplychaingamechanger.com/6-practices-to-optimize-your-goods-transportation-process/ (дата звернення: 27.10.2023).

11. ANT Logistics : вебсайт : URL : https://www.softkey.ua/ua/catalog/cloud/ant-logistics/ (дата звернення: 27.10.2023).

12. Програмні рішення для сфери транспорту та логістики : вебсайт. URL : https://avada-media.ua/ua/services/programmnyye-resheniya-dlya-sfery-transporta-i-logistiki/ AVADA MEDIA (дата звернення: 27.10.2023).

13. Influence of technology on route management systems : вебсайт . URL : https://yalantis.com/blog/add-route-planning-to-an-app/ https://yalantis.com/blog/add-route-planning-to-an-app/ (дата звернення: 27.10.2023).

14. Top 10 Logistics Management Software for 2023 : вебсайт . URL : https://www.upperinc.com/blog/logistics-management-software/ (дата звернення: 27.10.2023).

15. PTV Route Optimiser 2023 : вебсайт . URL : https://www.ptvlogistics.com/en/products/ptv-route-optimiser (дата звернення: 30.10.2023).

16. CityFlow Commute 2023 : вебсайт . URL : https://vilmate.com/project/cityflow-commute/ (дата звернення: 30.10.2023).

17. Як прокласти маршрут у Google Картах : вебсайт . URL : https://stylus.ua/uk/articles/1296.html (дата звернення: 30.10.2023).

18. Як додавати проміжні точки на маршруті в Google Maps : вебсайт . URL : https://techtoday.in.ua/tips/yak-dodavaty-promizhni-tochky-na-marshruti-u-google-maps-160295.html (дата звернення: 30.10.2023).

19. Object-Oriented programming (C#) : вебсайт . URL : https://learn.microsoft.com/en-us/dotnet/csharp/fundamentals/tutorials/oop (дата звернення: 21.11.2023).

20. Overview of .NET Framework : вебсайт . URL : https://learn.microsoft.com/en-us/dotnet/framework/get-started/overview (дата звернення: 24.11.2023)

21. Difference between BFS and DFS : вебсайт . URL : https://www.geeksforgeeks.org/difference-between-bfs-and-dfs/ (дата звернення: 03.11.2023).

22. Креневич А.П. Алгоритми і структури даних: Підручник. К.: ВПЦ "Київський Університет", 2021. 200 с. URL : https://www.mechmat.univ.kiev.ua/wp-content/uploads/2021/09/pidruchnyk-alhorytmy-i-struktury-danykh.pdf (дата звернення: 03.11.2023).

23. 21 Пошук в ширину : вебсайт . URL : https://algoua.com/algorithms/graphs/bfs (дата звернення: 03.11.2023).

24. Depth First Search : вебсайт . URL : https://www.hackerearth.com/practice/algorithms/graphs/depth-first-search/tutorial/ (дата звернення: 03.11.2023).

25. Пошук в глибину : вебсайт . URL : https://algoua.com/algorithms/graphs/dfs/ (дата звернення: 03.11.2023).

26. Wave and Traversal Algorithm in Distributed System : вебсайт . URL : https://www.geeksforgeeks.org/wave-and-traversal-algorithm-in-distributed-system/ (дата звернення: 03.11.2023).

27. Bellman–Ford Algorithm : вебсайт . URL : https://www.geeksforgeeks.org/bellman-ford-algorithm-dp-23/ (дата звернення: 03.11.2023).

28. Алгоритм Дейкстри : вебсайт . URL : https://ua5.org/algorithm/1970-algorytm-dejkstry.html (дата звернення: 05.11.2023).

29. A* Algorithm Search : вебсайт . URL : http://surl.li/mwaem (дата звернення: 05.11.2023).

30. Задача комівояжера : вебсайт . URL : http://surl.li/mwcfh (дата звернення: 05.11.2023).

31. Задача комівояжера. Математична постановка задачі : вебсайт . URL : https://www.mathros.net.ua/zadacha-komivojazhera-matematychna-postanovka-zadachi.html (дата звернення: 05.11.2023).

32. Господінов А. М., Смирнов С. А. ГЕНЕТИЧНИЙ АЛГОРИТМ ДЛЯ РОЗВ'ЯЗАННЯ ЗАДАЧІ КОМІВОЯЖЕРА. *Математичні методи комп'ютерного моделювання та кібернетичної безпеки:* Всеукраїнська науково-практична конференція студентів, аспірантів та молодих вчених. URL : https://ela.kpi.ua/bitstream/123456789/25202/1/S.19-21.pdf (дата звернення: 11.11.2023).

33. Зіньков Р. В., Марчук Г. В. Принцип дії мурашиного алгоритму при вирішенні задачі комівояжера. Математичне моделювання та розробка програмного забезпечення. URL : https://conf.ztu.edu.ua/wp-content/uploads/2019/12/17-1.pdf (дата звернення: 11.11.2023).

34. Метод гілок і меж : вебсайт . URL : http://ebib.pp.ua/metod-gilok-i-mej-ekonomiko-matematichni-metodi-i-modeli-v-komertsiyniy-diyalnosti.html (дата звернення: 15.11.2023).

35. Нормалізація відношень при проектуванні БД : вебсайт . URL : https://rdb.dp.ua/uk/chapter_03 (дата звернення: 21.11.2023).

36. Object-Oriented programming (C#) : вебсайт . URL : https://learn.microsoft.com/en-us/dotnet/csharp/fundamentals/tutorials/oop (дата звернення: 21.11.2023).

37. Overview of .NET Framework : вебсайт . URL : https://learn.microsoft.com/en-us/dotnet/framework/get-started/overview (дата звернення: 24.11.2023).

38. SQL Server 2019 – провідні можливості для продуктивності й безпеки : вебсайт . URL : https://www.microsoft.com/uk-ua/sql-server/sql-server-2019 (дата звернення: 25.11.2023).

39. Мапа GMap.NET у застосунках C# : вебсайт . URL : http://surl.li/oepgx (дата звернення: 12.12.2023).

40. База знань : вебсайт . URL : https://qalight.ua/baza-znaniy/ruchne-ta-avtomatizovane-testuvannya/ (дата звернення: 08.12.2023).

41. Exception-handling statements : вебсайт . URL : https://learn.microsoft.com/en-us/dotnet/csharp/language-reference/statements/exception-handling-statements (дата звернення: 08.12.2023).

ДОДАТОК А

В І Д Г У К

на кваліфікаційну роботу магістра на тему:

«Розробка інформаційної системи логістичного підприємства з актуалізацією шляхів перевезення на базі GMap.NET»

студента групи 126м-22з-1 Удачиної Катерини Олександрівни

Метою даної кваліфікаційної роботи є створення інформаційної системи на основі платформи .NET для формування оптимального маршруту постачання товарів та відображення його на карті за допомогою інструмента GMap в режимі онлайн.

Питання побудови оптимального маршруту доставки товарів набуло особливої актуальності з початком періоду карантину та воєнних дій на території України. Сьогодні ринок інформаційних систем пропонує різноманітне програмне забезпечення для побудови оптимальних маршрутів перевезень. Але недолік подібних систем полягає у тому, що водії самі визначають порядок об'їзду клієнтів, тобто самостійно виконують маршрутизацію своїх рейсів. Звісно, що такий маршрут не завжди є оптимальним або потрібно використовувати додаткові сервіси для впорядкування пунктів доставки і ручного перенесення адрес на карту, а це призводить до незручностей та вимагає додаткових витрат часу.

Тому виникає потреба створити інформаційну систему для логістичного підприємства, яка б виконувала функцію впорядкування адрес доставки, побудови оптимального шляху перевезення товарів на онлайн-карті і збереження результатів до бази даних.

Тема кваліфікаційної роботи відповідає напрямку діяльності фахівця галузі знань 12 «Інформаційні технології» спеціальності 126 «Інформаційні системи та технології» – створення, дослідження та практичне застосування прикладних інформаційних систем та технологій.

Оригінальність даної роботи полягає у комплексному застосуванні комбінаторних методів та сучасних інформаційних технологій, зокрема крос-платформної картографічної бібліотеки GMap.NET, для створення інформаційної системи логістичного підприємства з актуалізацією шляхів перевезення.

Практичне значення результатів кваліфікаційної роботи полягає у розробці додатку, який дозволяє визначати координати пунктів доставки товарів на основі геокодування даних, будувати маршрут перевезення з урахуванням мінімізації витрат на онлайн-карті і взаємодіяти з базою даних.

Ступінь самостійності виконання кваліфікаційної роботи повною мірою відповідає другому освітньо-кваліфікаційному рівню вищої освіти, тобто ступеню магістра.

В роботі досить повно виконано огляд сучасного стану ринку інформаційних технологій логістичної сфери та проаналізовано методи та моделі для побудови оптимального маршруту доставки товарів.

Деякі дискусійні положення та несуттєві недоліки пов’язані з наступними моментами:

– робота інформаційної системи залежить від швидкості інтернету;

– для виконання функції геокодування даних потрібно мати ідентифікатор API;

– робота дещо перевантажена детальним описом алгоритмів, що використовуються для побудови оптимальних шляхів.

Незважаючи на вищевказані зауваження, кваліфікаційна робота в цілому заслуговує оцінки

«_____», а її виконавець, студент _____, присвоєння йому кваліфікації магістр за спеціальністю 126 «Інформаційні системи та технології».

Керівник кваліфікаційної роботи,
професор кафедри ІТКІ, д-р техн. наук **Віктор ОЛЕВСЬКИЙ**

ДОДАТОК Б

Рецензія

на кваліфікаційну роботу магістра на тему:
«Розробка інформаційної системи логістичного підприємства з актуалізацією шляхів перевезення на базі GMap.NET»
студента групи 126м-22з-1 Удачиної Катерини Олександрівни

В період пандемії зріс попит на послуги логістичних підприємств, а з початком війни загострилася необхідність в оперативній побудові маршрутів перевезень з урахуванням реального стану доріг. Розглянувши існуючі інформаційні системи, що вирішують завдання у сфері логістики, можна зробити висновок про те, що переважно програмного забезпечення немає у відкритому доступі, хіба що пропонуються пробні версії. Або створюються програмні продукти під замовлення певного споживача, враховуючи особливості специфіки діяльності підприємства, що потребує подальшого супроводу, налаштування параметрів під конкретні умови, що призводить до додаткових витрат.

Саме тому створення відкритої інформаційної системи для логістичного підприємства, яка б виконувала функцію побудови оптимального шляху перевезень товарів, відображення маршруту на онлайн-карті і збереження результатів до бази даних, є актуальною задачею, яка потребує вирішення.

До переваг даної роботи слід віднести використання точного методу для визначення послідовності адрес пунктів доставки товарів, а також використання бібліотеки GMap.NET, яка дозволяє виконувати геокодування даних, маршрутизацію, будувати полігони та накладати маркери.

Серед недоліків даної роботи слід виділити наступні:

– відсутність модуля авторизації та реєстрації користувача в системі;

– необхідність стабільного доступу до інтернету для роботи інформаційної системи.

Однак вказані зауваження не мають суттєвого впливу на загальну позитивну оцінку роботи.

На підставі вищевикладеного можна зробити висновок, що представлені матеріали цілком відповідають вимогам, що пред'являються до кваліфікаційних робіт другого рівня вищої освіти, тобто ступеня магістра.

З огляду на весь спектр створених компонентів, що забезпечують формування даної кваліфікаційної роботи, в цілому вона заслуговує на оцінку

«_____», а її виконавець, студентка Удачина К.О., присвоєння їй кваліфікації магістр за спеціальністю 126 «Інформаційні системи та технології».

Рецензент, доцент кафедри інформаційних
систем і технологій УДУНТ, к.т.н. **Анна ЖУРБА**