

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування інституту, назва факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

бакалавр

(ступінь вищої освіти)

на тему ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ДЛЯ АВТОМАТИЗАЦІЇ СИСТЕМИ
СЛУЖБИ ДОСТАВКИ ВАНТАЖІВ
SOFTWARE FOR AUTOMATED CARGO DELIVERY SYSTEM

Виконав: студент(ка) 4 курсу, групи КНТ-119сп
Спеціальності 121 Інженерія програмного
забезпечення

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

Зінов'єв А.А.

(прізвище та ініціали)

Керівник

Зайко Т.А.

(прізвище та ініціали)

Рецензент

Коцур М.І.

(прізвище та ініціали)

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) _____
Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	Зайко Т.А., доцент		
Нормоконтролер	Липовець М.В., асистент		

7. Дата видачі завдання “04” квітня 2022 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Срок виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області.	1 тиждень	Розділ 1
3	Вибір мови програмування та інших технологій	2 тиждень	Розділ 2
	розробки		
4	Розробка архітектури програми.	2 тиждень	Розділ 3
5	Розробка програми.	3-4 тижні	Розділи 3,4
6	Тестування та експериментальне дослідження	5 тиждень	Розділ 4
	програмного забезпечення.		
7	Оформлення пояснювальної записки та документів	6 тиждень	Додатки
	до неї.		
8	Нормоконтроль та рецензування.	7 тиждень	
9	Захист роботи.	8 тиждень	

Студент(ка)

(підпис) Зінов'єв А.А.
(прізвище та ініціали)

Керівник проєкту (роботи)

(підпис) Зайко Т.А.
(прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: 73 с., 33 рис., 1 табл., 4 дод., 23 джерела.

АВТОМАТИЗОВАНА СИСТЕМА, ПОШТА, ВІДПРАВЛЕННЯ, ОПЕРАТОР, ВАНТАЖ, ВІДДІЛЕННЯ, C#, ACCESS, VISUAL STUDIO, КУР'ЄР.

Об'єкт дослідження – процес програмної реалізації автоматизованої системи служби доставки вантажів.

Предмет дослідження – методи реалізації автоматизованої системи служби доставки вантажів.

Метою роботи є розроблення автоматизованої системи служби доставки вантажів для зменшення ризику впливу людського фактору.

Матеріали, методи та технічні засоби: об'єктно-орієнтоване програмування, мова програмування C#, інтегроване середовище розробки Microsoft Visual Studio, система керування базою даних Microsoft Access.

Результати. Розроблено програмне забезпечення у вигляді автоматизованої системи для поштових компаній.

Висновки. Розроблено автоматизовану систему для служби доставки вантажів.

Галузь використання – оператори поштових компаній.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 73 pages, 33 figures, 1 tables, 4 appendixes, 23 sources.

AUTOMATED SYSTEM, MAIL, DEPARTURE, OPERATOR, CARGO, DEPARTMENT, C #, ACCESS, VISUAL STUDIO, COURIER.

The object of research is the process of software implementation of the automated system of cargo delivery service.

The subject of research - methods of implementing an automated system of cargo delivery service.

The aim of the work is to develop an automated cargo delivery system for fast operation of the operator and eliminate the risk of the human factor.

Materials, methods and technical means: object-oriented programming, C # programming language, Microsoft Visual Studio integrated development environment, Microsoft Access database management system.

Results. Software in the form of an automated system for postal companies has been developed.

Conclusions. An automated cargo delivery system has been developed.

Field of use - operators of postal companies.

ЗМІСТ

	С.
Перелік скорочень та умовних познач.....	8
Вступ.....	9
1 Аналіз предметної області.....	10
1.1 Поштова компанія	10
1.2 Огляд існуючих методів вирішення завдання	11
1.3 Огляд існуючих рішень	12
1.4 Постановка завдання	16
1.5 Висновки за розділом 1	18
2 Розробка архітектури програми	19
2.1 Вибір інструментарію розробки	19
2.2 Структурна схема розробки	20
2.3 Висновки за розділом 2	21
3 Основні рішення щодо реалізації компонентів системи	22
3.1 Проєктування моделей системи	22
3.2 Моделювання системи з точки зору прецедентів.....	24
3.3 Фізичне проєктування	25
3.4 Проєктування інтерфейсу	27
3.5 Висновки за розділом 3	28
4 Експлуатація, тестування та експериментальне дослідження програми	29
4.1 Опис застосунку	29
4.2 Умови виконання програми	29
4.3 Інструкція по експлуатації програми.....	29
4.4 Методика та результати тестування.....	31
Висновки	34
Перелік джерел посилання	35
Додаток А Технічне завдання	37
Додаток Б Опис програми	42
Додаток В Текст програми	46

Додаток Д Слайди презентації.....	67
-----------------------------------	----

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

DB	– Data Base;
DBMS	– Database Management System;
HDD	– Hard Disk Drive;
OS	– Operating System;
PC	– Personal Computer;
RAM	– Random Access Memory;
SSD	– Solid State Drive;
URL	– Uniform Resource Locator,
UML	– Unified Modeling Language;
VS	– Microsoft Visual Studio.

ВСТУП

У наші часи для прискорення наукового і технічного прогресу в усіх сферах життєдіяльності людини водночас з іншими заходами передбачається широке впровадження інформаційних технологій як фундаменту подальшого розвитку людства.

Керівництво будь-якого закладу сьогодні буде в захваті від зменшення впливу людського фактору на процес роботи а також збільшення прибутку. Це досягається за рахунок підвищення продуктивності праці. Саме тому спостерігається інтенсивне впровадження у виробництво досягнень науково-технічного прогресу, застосування новітніх революційних технологій. Впровадження автоматичної реєстрації відправлення є одним зі способів скорочення штату працівників та пришвидшенню їх роботи, корегування дій. А це означає не тільки зменшення кількості помилок а ще й менші затрати на виплату заробітної плати та ін.

Програмне забезпечення для автоматизації доставки вантажів може використовуватися у будь-якій сфері людської діяльності. У нашому випадку сферою використання застосунку є служба доставки або пошта.

Служба доставки вантажу – це можливість швидко переправляти різний вантаж по всій мережі відділень пошти, в будь-яку частину країни та світу.

Отже, актуальність даної дипломної роботи, яка полягає у створенні можливості автоматизації відправлення вантажу, складно переоцінити. Тепер кожен працівник поштових зв'язків може швидко створювати відправлення клієнта, відстежувати його в логістичній системі, та за потреби корегувати, сортувати, аналізувати дані. А для керівництва відділень впровадження цього програмного продукту означає зменшення впливу людського фактору на організацію процесу праці, а саме зменшення вірогідності допущення помилок при запису клієнтів та зменшення затрат на утримання персоналу, який відповідає за цей процес.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

Даний розділ роботи присвячено питанням системи служби доставки та робочого місця оператора. Розглянуто основні питання предметної області, досліджено існуючі аналогічні рішення та визначено функціональні вимоги, межі розробки та системи для подальшої реалізації проєкту.

1.1 Поштова компанія

Багато людей потребує перевезення власного майна, від техніки з магазину до переїзду на інше місце проживання. У кого є можливість самому поїхати та доставити потрібний вантаж, це може потребувати багато часу та коштів. В таких випадках користуються кур'єрськими службами так як це швидко, вигідно та надійно. Для передачі того самого відправлення можна використовувати різні кур'єрські служби, Укрпошту, міжміський транспорт, Нова пошта [1].

Але якщо вантаж занадто великий – його неможливо відправити Укрпоштою, в Justin або міжміським транспортом. На допомогу приходять вантажні відділення.

Поштовий відділок або відділ зв'язку – об'єкт поштового зв'язку, який здійснює приймання, сортування, відправлення, обробку та доставку пошти [2].

Є дуже багато перевізників вантажу. Найпопулярніші з них: Укрпошта, Нова пошта, Justin, Fedex, DHL Express та інші. В різних країнах для перевозки вантажу ви бачили схожі компанії.

Всі подібні сервіси працюють за одним принципом. Користувач просто віддає потрібний вантаж, заповнює куди воно прямує і через деякий час інший клієнт отримує своє відправлення [2].

Поштове відправлення – загальне визначення для відправлень, які доручаються спеціалізованій установі для пересилки та доставки адресатові [2].

Переваги поштових послуг. Кожна поштова компанія має свої особливості. Наприклад максимальний обсяг завантаження відправлення, якість упакування або швидкість перевезення, ціна за послуги. Розглянемо переваги кур'єрських компаній в загальному [1]:

- вони дозволяють без проблем пересилати великі вантажі;
- можна використовувати для економії коштів;
- всього за кілька днів можна отримати відправлення.

Недоліки кур'єрських компаній. Не дивлячись на значні плюси, недоліки у них теж є:

- велика кількість комісій. Саме так заробляють власники подібних закладів;
- обмеження на місця перевезення;
- обмеження на вид пакування;
- може трапитись помилка через яку відправлення загубиться;
- викрадення відправлення.

1.2 Огляд існуючих методів вирішення завдання

Керівництву поштових компаній необхідно впровадження програмного продукту для зменшення впливу людського фактору на організацію процесу праці, а саме зменшення вірогідності допущення помилок при створенні, модифікації та аналізуванні. Зменшення витрат на утримання персоналу, який відповідає за цей процес [1].

В даний час, існують приклади вирішення таких завдань. Але вони мають певні недоліки, серед яких складний до опанування інтерфейс. Система, що розробляється вирішує цю проблему, адже процес проходить у максимально зручний для користувача спосіб.

1.3 Огляд існуючих рішень

ПЗ робочого місця оператору розробляється з метою маніпулювання з різними типами відправлень, для спрощення та прискорення процесу роботи. Передбачається, що завдяки програмному продукту процес роботи з відправленнями можна буде значно полегшити та пришвидшити. Такий ресурс дає можливість для аналізу даних.

Для розробки конкурентоспроможного застосунку слід на початку проаналізувати існуючі аналогічні рішення та виявити їх переваги або недоліки.

Перед розглядом аналогів, слід зазначити, що переважна більшість існуючих рішень створені лише для внутрішнього використання, що ускладнює процес порівняння.

1.3.1 Сервіс «Нова пошта»

«Нова пошта» – поштово-вантажна компанія, яка забезпечує легку доставку кожному клієнту – до відділення, поштамату, за адресою.

Компанія надає логістичні та дистрибуційні послуги, доставляючи як дрібні посилки, так і великі вантажі. Нова пошта дозволяє тисячам підприємців створювати та розвивати бізнес не лише в Україні, а й за кордоном. Мережа компанії налічує понад 9300 відділень по всій Україні та понад 11000 поштоматів, а кількість відправлення за 2020 рік перевищила 300 млн посилок та вантажів.

Місія компанії – легкість доставки для життя та бізнесу. Для цього команда Нової пошти впроваджує нові продукти та сервіси, орієнтуючись на міжнародні стандарти та найкращий світовий досвід. Головну сторінку сервісу Нова пошта представлено на рис. 1.1 [3].

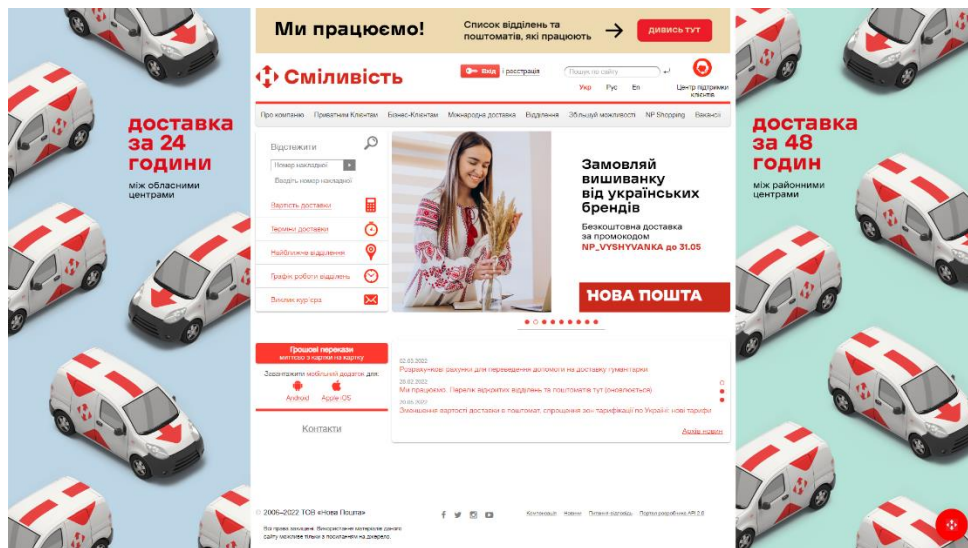


Рисунок 1.1 – Головна сторінка сервісу Нова пошта [3]

1.3.2 Компанія «Укрпошта»

Укрпошта — національний оператор поштового зв'язку, історія якого розпочалась у 1994 році. Сьогодні це сучасна компанія з новим брендом, новою командою, новими відділеннями, послугами й підходами до роботи.

Мережа Укрпошти найбільша в Україні — поштові відділення є у найвіддаленіших регіонах країни. 73 000 працівників щодня приймають, сортують, везуть, доставляють, організовують, аналізують та допомагають клієнтам. Головну сторінку сервісу Укрпошта представлено на рис. 1.2 [4].

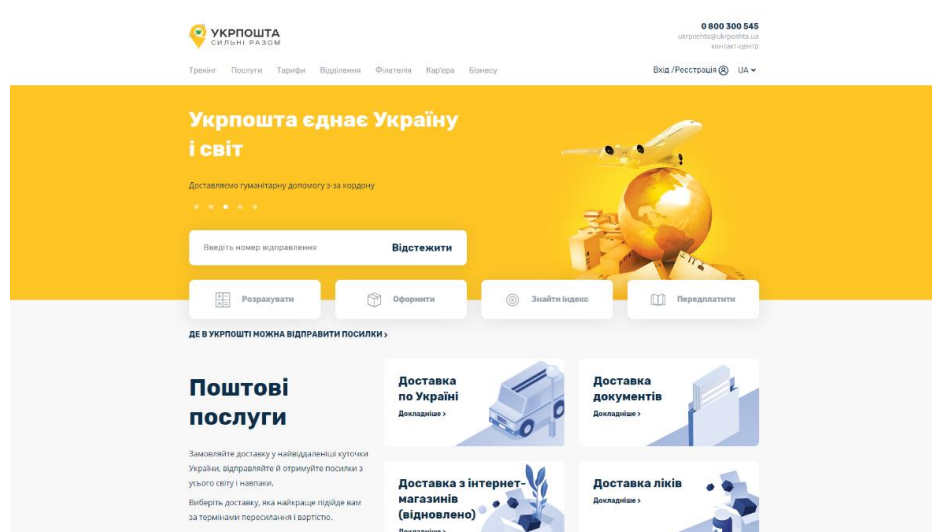


Рисунок 1.2 – Головна сторінка Укрпошти [4]

1.3.3 Компанія «FedEx»

FedEx Corporation – американська компанія, що надає поштові, кур'єрські та інші послуги логістики по всьому світу. Компанія була заснована у 1971 році під назвою Federal Express.

Флот вантажних літаків FedEx є одним з найбільших у світі і станом на квітень 2009 року складається з 379 середньо- та далекомагістральних машин та 291 близькомагістрального. Головну сторінку сервісу «FedEx» представлено на рис. 1.3 [5].

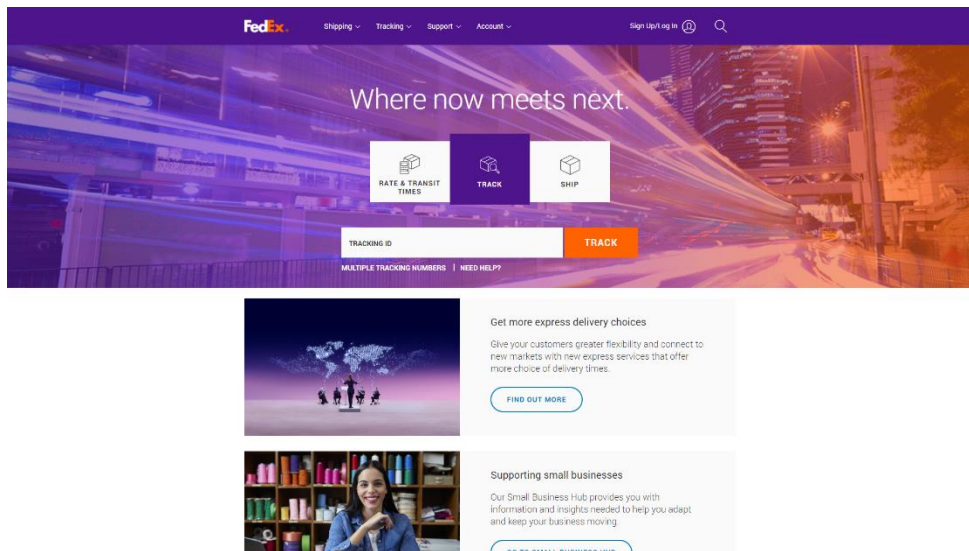


Рисунок 1.3 – Головна сторінка компанії «FedEx» [5]

1.3.4 Порівняння існуючих аналогів

Проведемо порівняння за певними критеріями, хід якого представимо у вигляді таблиці (табл. 1.1). Серед критеріїв виділимо наступні:

- ціна послуг – загальна оцінка ціни послуг, що надається клієнтам на вільній основі;
- швидкість перевезення – за який час відправлення потрапляє до свого отримувача;

- розподіл часу на утримання вантажу в сховищі – можливість довгочасного збереження відправлення для вільного витрачання особистого часу;
- обмеження за розміром відправлення – чи накладає сервіс обмеження на вантаж певного виду.

Сервіс – можливість та якість надання клієнтам різних інформаційних послуг, працівниками компанії.

З результатів порівняння можна зробити висновок, що в більшості випадків компанії достатньо обмежені за графіком роботи [6].

Таблиця 1.1 – Порівняння існуючих аналогів

Критерій порівняння	Нова пошта	Укрпошта	FedEx
Ціна послуг	Дорога, від розміри відправлення	Дуже дешево	Дорога, від відстані перевезення
Швидкість перевезення	Швидко	Повільно	Дуже швидко
розподіл часу на утримання вантажу в сховищі	Неділя	Місяць	День
обмеження за розміром відправлення	Є	Є	Немає
Сервіс	Низької якості	Середня якість	Якісний

Деякі з них не приймають важкий або великий вантаж, мають обмеження на розміри, кількість місць та форму, пакування вантажу. Так як це впливає на швидкість перевезення.

Саме тому, можна зробити висновок, що враховуючи велику кількість існуючих поштових компаній із різним функціоналом та можливостями, вони усі мають низку недоліків та незручностей.

1.4 Постановка завдання

Метою проєкту є реалізація автоматизованої системи для оператора вантажного відділення, що дозволить створювати, зберігати, відстежувати та аналізувати на сервері файли у приватній та загальнодоступній формі, ділитися ними з іншими працівниками компанії.

1.4.1 Межі проєкту

Автоматизована система — організаційно-технічна система, що складається із засобів автоматизації певного виду чи кількох видів діяльності людей та персоналу, що здійснює цю діяльність. Автоматизована система виробничого призначення здійснює збирання інформації з об'єкта керування, передає, перетворює й обробляє її, формує керуючі команди та виконує їх на керованому об'єкті, тобто ті функції, які піддаються автоматизації. Людина визначає цілі та критерії керування й коригує їх, коли змінюються умови, зокрема, виконує функції нагляду за роботою автоматизованих пристроїв, а в разі потреби змінює програму їхньої роботи і приймає загальні рішення щодо керування в змінених або складних ситуаціях [7].

Проєкт надаватиме користувачам доступ до створення та модифікування відправлення певного об'єму. Отже, першочергово слід реалізувати саме функцію створення та огляду відправлення, зміну даних вантажу.

За допомогою мови програмування C# буде створено функції реєстрації, редагування та огляду відправлень, а також додатковий функціонал (пакування, додаткові послуги і т. д.). Такі функції як оплата відправлення буде реалізована готовим кодом, що пропонується сайтом інтернет-гаманця (LiqPay) [8].

1.4.2 Функціональність системи

Система має підтримувати:

- створення відправлення;
- додавання пакування;
- додаткові послуги;
- модифікування;
- оплата за послуги;
- друк експрес накладної.

1.4.3 Вимоги до інтерфейсу

Інтерфейс системи повинен бути простим та інтуїтивно-зрозумілим для користувачів системи. Вся інформація системи повинна бути чітко структурована і розділена на розділи.

1.4.4 Вимоги до продуктивності

Система та всі її компоненти повинні працювати стабільно і з максимально можливою швидкістю, видавати правильні результати і не бути надто ресурсоємкою [9].

Реєстрація відправлення з занесенням його даних в базу даних зареєстрованих відправлень повинна займати не більше 5 хвилин. Система повинна підтримувати до 5 запитів на реєстрацію в секунду [10].

1.4.5 Вимоги до безпеки

Авторизація користувача обов'язкова так як це програма для внутрішнього використання, реєстрація лише адміністраторами компанії які за це відповідають, для збереження внутрішньої інформації компанії [11].

Незареєстровані користувачі не можуть мати доступ до роботи с програмою. Зареєстровані можуть створювати відправлення, переглядати та змінювати інформацію про них. Для безпечного зберігання особистих даних користувача, всі данні будуть збережені в зашифрованому вигляді [11].

1.4.6 Експлуатаційні вимоги

Для користування автоматизованою системою, користувач повинен мати девайс з доступом до мережі Інтернет, підключений принтер та бути зареєстрованим [10].

1.4.7 Не функціональні вимоги до продукту

Для роботи програми необхідно мати IBM-сумісний ПК з встановленою операційною системою Windows 7 та вище, з мікропроцесором Intel Core I3 2 GHz та вище, оперативна пам'ять 2Gb і вище, вільного простору на жорсткому диску 200 Мб, SVGA-монітор, клавіатура та маніпулятор типу “миша”. Для виводу на друк потрібен принтер.

1.5 Висновки за розділом 1

В першому розділі відбулося ознайомлення із проблемною людського фактору у роботі на поштові компанії. Більш детально розглянуто проблеми та перспективи існуючих кур'єрських служб. На основі виявлених недоліків сформовано функціональні вимоги до нового застосунку, що буде реалізовувати функції автоматизованої системи. Для подальшого обґрунтування проєктних питань та рішень було сформовано перелік функціональних та не функціональних вимог до застосунку.

2 РОЗРОБКА АРХІТЕКТУРИ ПРОГРАМИ

В даному розділі буде детально проаналізовано та обрано мову програмування для реалізації функціональної частини, програмні комплекси та засоби для оформлення ПЗ [9].

2.1 Вибір інструментарію розробки

2.1.1 Вибір мови програмування

Для реалізації проєкту обрано мову C# - об'єктно-орієнтована мова програмування прикладного рівня.

Вона розроблена в 1998-2001 роках групою інженерів компанії Microsoft під керівництвом Андерса Хейлсберг і Скотта Вільтаумота як мову розробки додатків для платформи Microsoft .NET Framework.

C # відноситься до сім'ї мов з C-подібним синтаксисом, з них його синтаксис найбільш близький до C ++ і Java [12].

Мова має статичну типізацію, підтримує поліморфізм, перевантаження операторів (в тому числі операторів явного і неявного приведення типу), делегати, атрибути, події, властивості, узагальнені типи і методи, ітератори, анонімні функції з підтримкою замикань, LINQ, виключення, коментарі у форматі XML [13].

2.1.2 Огляд особливостей обраного компілятора

Microsoft Visual Studio - лінійка продуктів компанії Microsoft, що включають інтегроване середовище розробки програмного забезпечення і ряд інших інструментальних засобів. Дані продукти дозволяють розробляти як консольні додатки, так і додатки з графічним інтерфейсом, в тому числі з підтримкою технології Windows Forms, а також web-сайти, web-додатки, web-служби як в рідному, так і в керованому кодах для всіх платформ,

підтримуваних Windows, Windows Mobile, Windows CE, .NET Framework, Xbox, Windows Phone .NET Compact Framework і Silverlight [14].

Visual Studio включає в себе редактор вихідного коду з підтримкою технології IntelliSense і можливістю найпростішого рефакторінга коду. Вбудований відладчик може працювати як відладчик рівня вихідного коду, так і відладчик машинного рівня. Решта вбудовуються інструменти включають в себе редактор форм для спрощення створення графічного інтерфейсу додатку, web-редактор, дизайнер класів і дизайнер схеми бази даних. Visual Studio дозволяє створювати і підключати сторонні додатки (плагіни) для розширення функціональності практично на кожному рівні, включаючи додавання підтримки систем контролю версій вихідного коду (як, наприклад, Subversion і Visual SourceSafe), додавання нових наборів інструментів (наприклад, для редагування і візуального проєктування коду на предметно-орієнтованих мовах програмування) або інструментів для інших аспектів процесу розробки програмного забезпечення (наприклад, клієнт Team Explorer для роботи з Team Foundation Server) [15].

2.2 Структурна схема розробки

Структурна схема — схема, яка визначає основні функціональні частини виробу, їх взаємозв'язки та призначення. Під функціональною частиною розуміють складову частину схеми: елемент, пристрій, функціональну групу, функціональну ланку. Структурна схема призначена для відображення загальної структури проєкту, тобто його основних блоків, вузлів, частин та головних зв'язків між ними. Із структурної схеми повинно бути зрозуміло, навіщо потрібний даний проєкт і як він працює в основних режимах роботи, як взаємодіють його частини [16].

На рисунку 2.1 наведено структурну схему ПЗ, що розробляється.

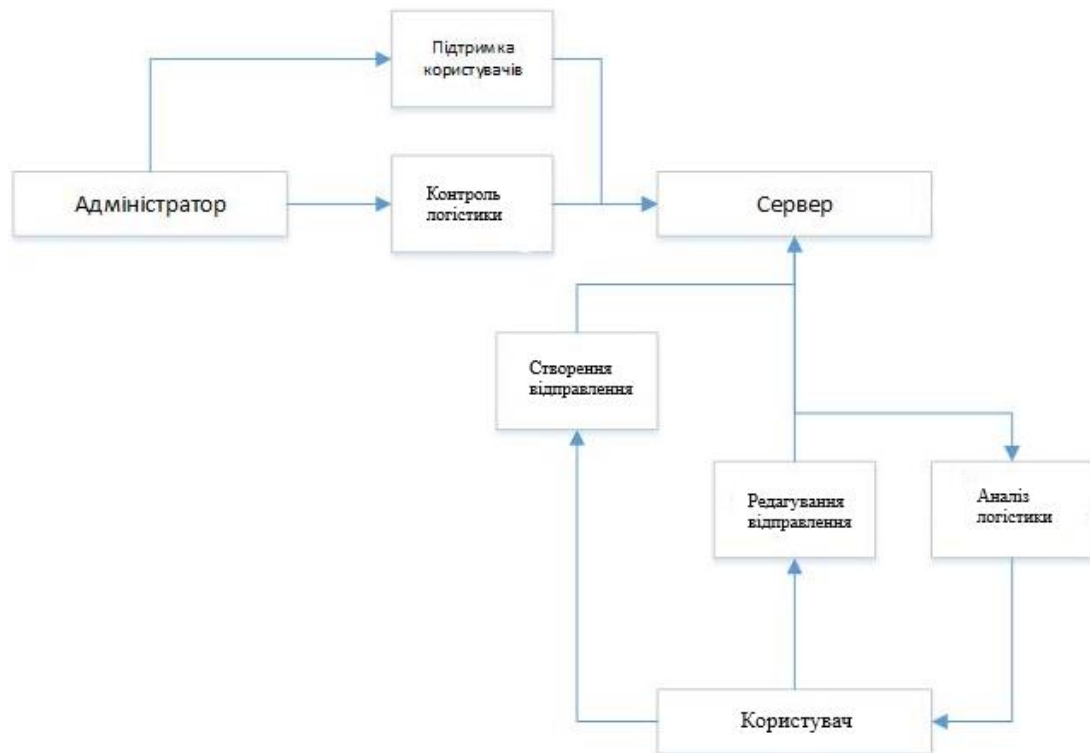


Рисунок 2.1 – Загальна схема системи за обраною темою

2.3 Висновки за розділом 2

В другому розділі було проведено детальне дослідження загальних технічних питань розробки. Для розробки автоматизованої системи було обрано C# через низку переваг. Для полегшення та пришвидшення розробки було обрано середовище розробки VS та засоби розробки, які вона надає.

3 ОСНОВНІ РІШЕННЯ ЩОДО РЕАЛІЗАЦІЇ КОМПОНЕНТІВ СИСТЕМИ

3.1 Проєктування моделей системи

Unified Modeling Language (UML) — це мова моделювання загального призначення. Основна мета UML - визначити стандартний спосіб візуалізації того, як була розроблена система. Він дуже схожий на креслення, які використовуються в інших галузях техніки.

UML – це не мова програмування, це скоріше візуальна мова. Ми використовуємо діаграми UML, щоб зобразити поведінку та структуру системи. UML допомагає інженерам програмного забезпечення, бізнесменам і системним архітекторам моделювати, проєктувати та аналізувати. Група управління об'єктами прийняла уніфіковану мову моделювання як стандарт у 1997 році. Відтоді нею керує OMG. Міжнародна організація зі стандартизації опублікувала UML як затверджений стандарт у 2005 році. UML переглядався протягом багатьох років і періодично переглядається [17, 18].

Діаграма класів UML – це графічне позначення, яке використовується для побудови та візуалізації об'єктно-орієнтованих систем [17-19]. Діаграма класів на UML – це тип діаграми статичної структури, яка описує структуру системи, показуючи:

- класи;
- їх атрибути;
- операції (або методи);
- і відносини між об'єктами [17]-[19].

Діаграма класів автоматизованої системи, що розробляється зображена на рис. 3.1.

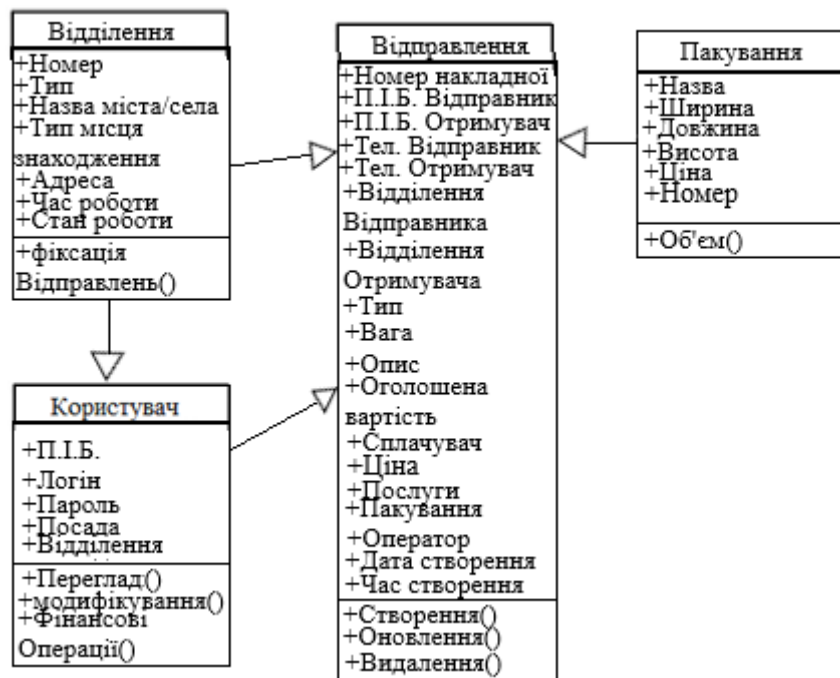


Рисунок 3.1 – Діаграма класів

Діаграми пакетів – це структурні діаграми, які використовуються для показу організації та розташування різних елементів моделі у вигляді пакетів [17]-[19]. Пакет – це групування пов’язаних елементів UML, таких як діаграми, документи, класи чи навіть інші пакети. Кожен елемент вкладений в пакунок, який зображений у вигляді папки файлів на діаграмі, а потім впорядкований ієрархічно в межах діаграми. Діаграми пакетів найчастіше використовуються для наочної організації багатошарової архітектури в будь-якому класифікаторі UML, наприклад, у програмній системі [17]-[19].

Діаграма пакетів для автоматизованої системи, що розробляється на рис. 3.2.

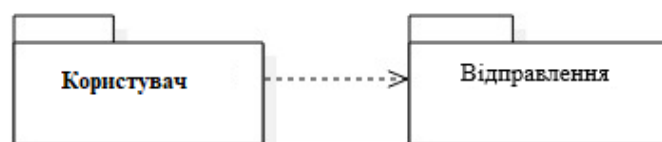


Рисунок 3.2 – Діаграма пакетів

Мета діаграми компонентів – показати взаємозв’язок між різними компонентами в системі. Для цілей UML 2.0 термін «компонент» відноситься

до модуля класів, які представляють незалежні системи або підсистеми з можливістю взаємодії з рештою системи [20]. Існує цілий підхід до розробки, який обертається навколо компонентів: розробка на основі компонентів. У цьому підході діаграми компонентів дозволяють планувальнику ідентифікувати різні компоненти, щоб вся система виконувала те, що вона повинна робити. Частіше, в підході до програмування, діаграма компонентів дозволяє старшому розробнику групувати класи разом на основі спільної мети, щоб розробник та інші могли дивитися на проєкт розробки програмного забезпечення на високому рівні [20].

Діаграми компонентів автоматизованої системи, що розробляється зображена на рис. 3.3.

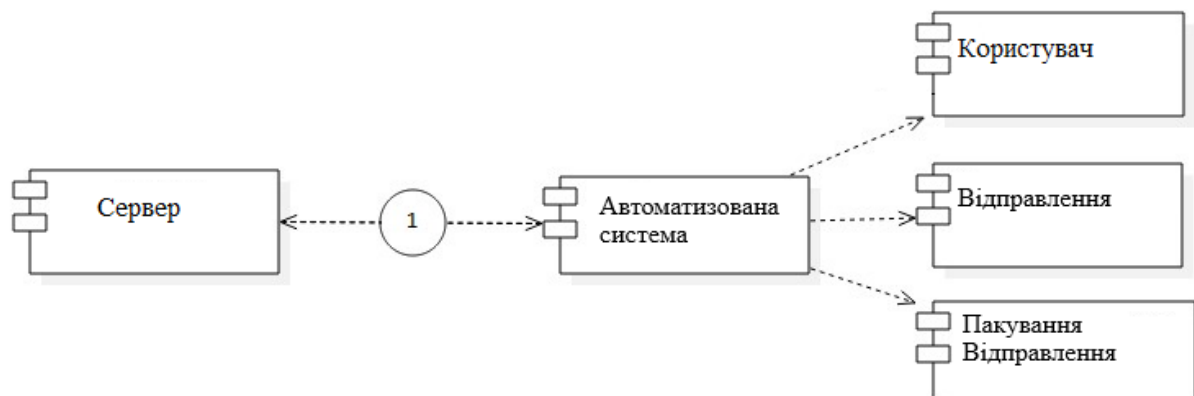


Рисунок 3.3 – Діаграма компонентів

3.2 Моделювання системи з точки зору прецедентів

Діаграма варіантів використання – це графічне зображення можливої взаємодії користувача з системою [21]. Діаграма варіантів використання показує різні варіанти використання та різні типи користувачів, яких має система, і часто супроводжується діаграмами інших типів. Варіанти використання представлені або колами, або еліпсами. Актори часто зображені у вигляді фігурок [21].

Прецедент: Оформлення відправлення клієнта.

Основна діюча особа: Робітник поштової компанії

Основний позитивний сценарій:

- користувач авторизується в програмі;
- заходить до вікна створення відправлення;
- вводить потрібну інформацію;
- натискає на кнопку створення відправлення.
- друк накладної про відправлення.

Альтернативний потік АП1:

- користувач невірно заповнює дані у формі;
- виводиться повідомлення про помилку.

Прецедент оформлення відправлення представлено на рис. 3.4.

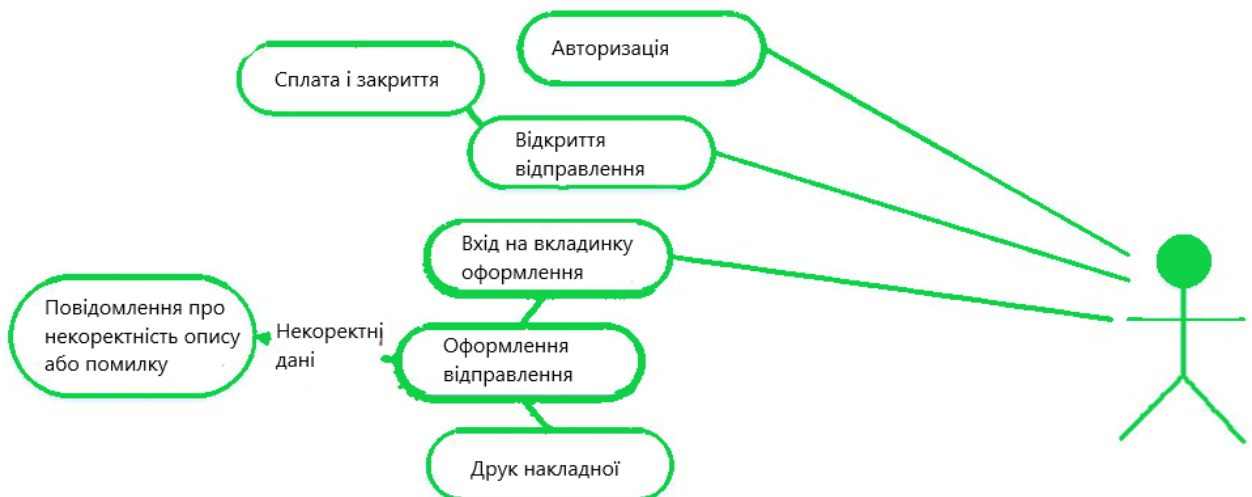


Рисунок 3.4 – Прецедент «Оформлення відправлення клієнта»

3.3 Фізичне проєктування

Діаграма активностей (видів діяльності) – один із доступних видів діаграм, що підтримуються Flexberry Designer [22]. Вона, як і діаграма станів, відбиває динамічні аспекти поведінки системи. По суті, ця діаграма є блок-схемою, яка наочно показує, як потік управління переходить від однієї діяльності до іншої [22].

Діаграма діяльності для автоматизованої системи, що розробляється зображено на рис. 3.5.

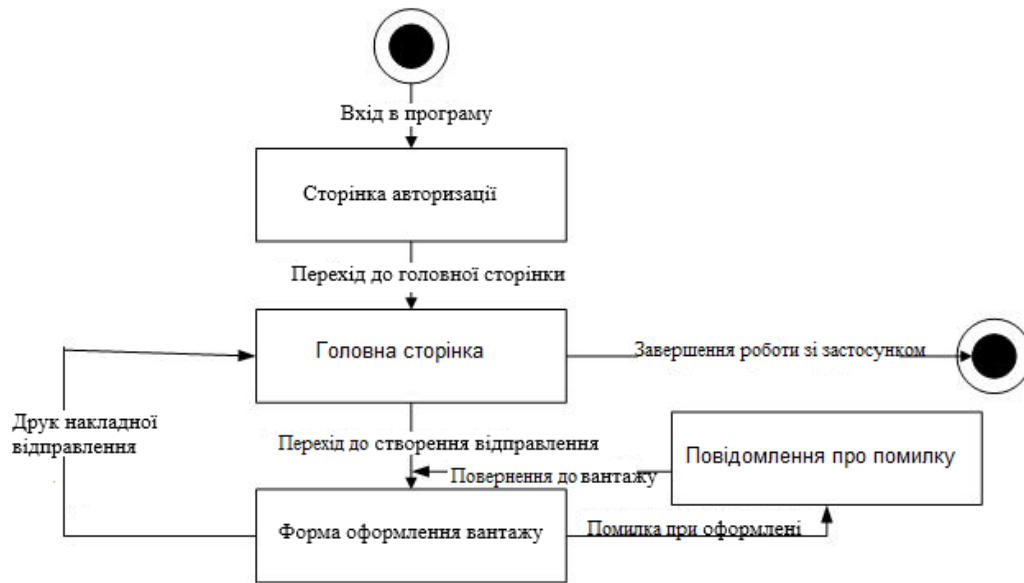


Рисунок 3.5 – Діаграма станів системи

Діаграми кооперацій використовуються, щоб показати, як об'єкти взаємодіють, щоб виконувати поведінку конкретного варіанту використання або частини варіанту використання [23]. Поряд із діаграмами послідовності, кооперація використовується дизайнерами для визначення й уточнення ролей об'єктів, які виконують певний потік подій у варіанті використання [23]. Вони є основним джерелом інформації, яка використовується для визначення обов'язків класу та інтерфейсів [23].

Діаграми кооперацій для автоматизованої системи, що розробляється зображені на рис. 3.6 та 3.7.



Рисунок 3.6 – Діаграма кооперації: Оформлення відправлення

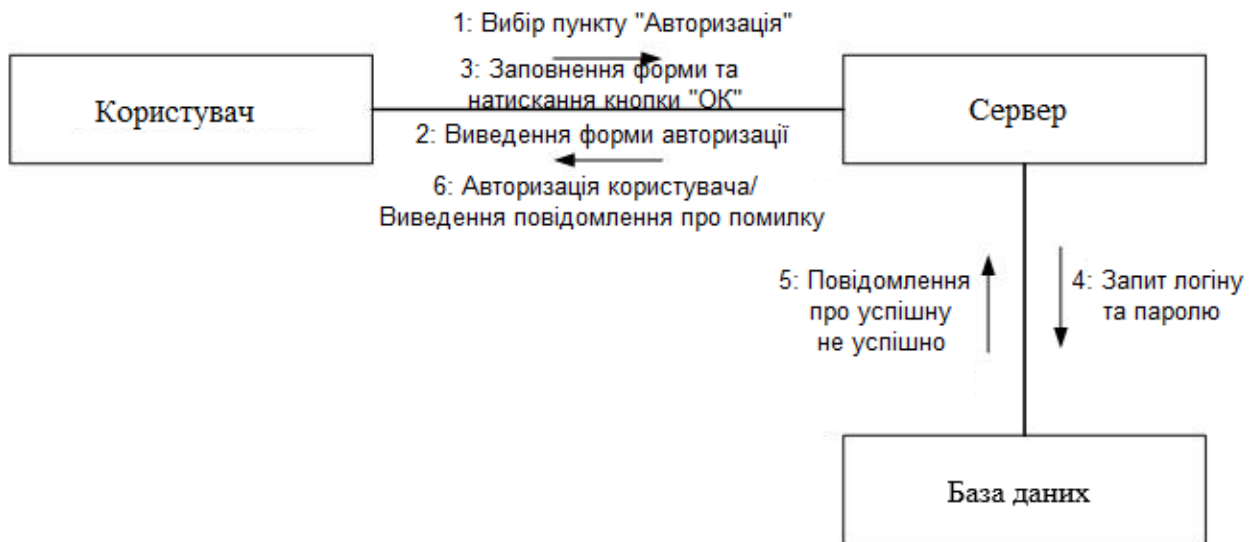


Рисунок 3.7 – Діаграма кооперації: Авторизація користувача

3.4 Проектування інтерфейсу

Програма повинна бути зручною в засвоєнні і використанні, а програмний інтерфейс повинен бути інтуїтивно зрозумілим.

Користувач повинен довіряти програмі і не боятися її використовувати. Якщо користувач вводить неправильні дані, програма повинна повідомити його про це.

Спроектований інтерфейс зображений на рис. 3.8–3.11.

The screenshot shows a login interface with a light blue background. At the top, the text 'Ласкаво просимо' (Welcome) is displayed in a large, bold, black font. Below this, there are two input fields: 'Логін:' (Login) and 'Пароль:' (Password). Each label is followed by a white rectangular input box with a thin black border. At the bottom of the form, there are two buttons: 'Увійти' (Login) and 'Вийти' (Logout). Both buttons are light gray with black text and a thin black border.

Рисунок 3.8 – Сторінка авторизації

Рисунок 3.9 – Головна сторінка для авторизованого користувача

Рисунок 3.10 – Створення відправлення

Рисунок 3.11 – Вікно візиту.

3.5 Висновки за розділом 3

Було проведено ознайомлення з видами структурних діаграм та особливостями їх застосування, розроблено абстрактну модель системи мовою UML на основі структурних діаграм. Розроблено абстрактну модель системи за обраною темою на основі діаграму класів, пакетів та компонентів, на основі діаграм поведінки, прецедентів, діяльності та кооперацій.

4 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОГРАМИ

4.1 Опис застосунку

Автоматизована система служби доставки вантажів призначена для зручного, надійного та швидкого створення відправлень клієнтів компанії, логістичних і аналітичних операцій.

4.2 Умови виконання програми

Для роботи програми необхідно мати IBM-сумісний ПК з встановленою операційною системою Windows 7 та вище, з мікропроцесором Intel Core I3 2 GHz та вище, оперативна пам'ять 2Gb і вище, вільного простору на жорсткому диску 200 Мб, SVGA-монітор, клавіатура та маніпулятор типу “миша”. Для виводу на друк потрібен принтер.

4.3 Інструкція по експлуатації програми

4.3.1 Характеристики програми

Дане програмне забезпечення являється автоматизованою системою, має з'єднання з сервером, працює цілодобово, поки використовується оператором та знаходиться в працездатному стані.

Проект має наступну файлову структуру (рис 4.1).

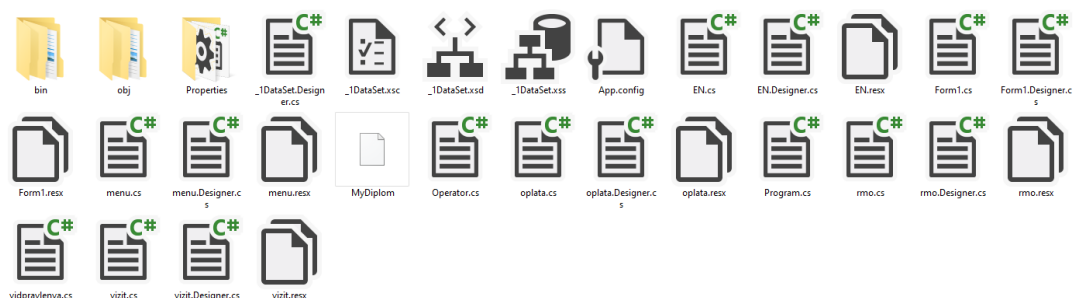


Рисунок 4.1 – Файлова структура проекту

У кореневому каталозі зберігаються файли конфігурацій та налаштувань

4.3.2 Звернення до програми

Для роботи з програмою слід натиснути на файл запуску програми. Для керування використовуються елементи інтерфейсу.

4.3.3 Вхідні й вихідні дані

Початковими даними програми є дані про користувача, що пройшов авторизацію, інформація про відправника, саме відправлення та отримувача. Вихідними даними є: виведена інформація про відправлення, та накладна з усією інформацією про відправлення, що користувач оформлював в автоматизованій системі.

4.3.4 Виконання програми

Для запуску застосунку потрібно виконати наступні дії:

- запустити автоматизовану систему.

Для створення відправлення:

- запустити автоматизовану систему;
- авторизуватись;
- обрати вкладку створення відправлення;
- заповнити потрібні дані про відправлення;
- натиснути на кнопку створення відправлення;
- надрукувати накладну відправлення.

Для видачі відправлення:

- після авторизації обрати вкладку видачі відправлення;
- вести номер відправлення та перевірити його;
- закрити відправлення та видати клієнту.

4.4 Методика та результати тестування

4.4.1 Аналіз реалізованих функціональних вимог

В ході реалізації програмного продукту, було реалізовано наступні функції, передбачені в попередніх планах розробки [17]:

- створення або видача відправлення клієнта;
- авторизація користувачів;
- оплата послуг;
- друк накладної;
- оформлення пакування.

4.4.2 Ергономічність автоматизованої системи

Існують чотири основні критерії оцінки ергономічності програмного продукту: ефективність, точність, остаточна пам'ять, емоціональна реакція [17].

Представлений програмний продукт у достатній мірі відповідає цим критеріям, оскільки являється простим і інтуїтивно зрозумілим.

Приклади основних сторінок застосунку продемонстровані на рис. 4.5 – 4.7.

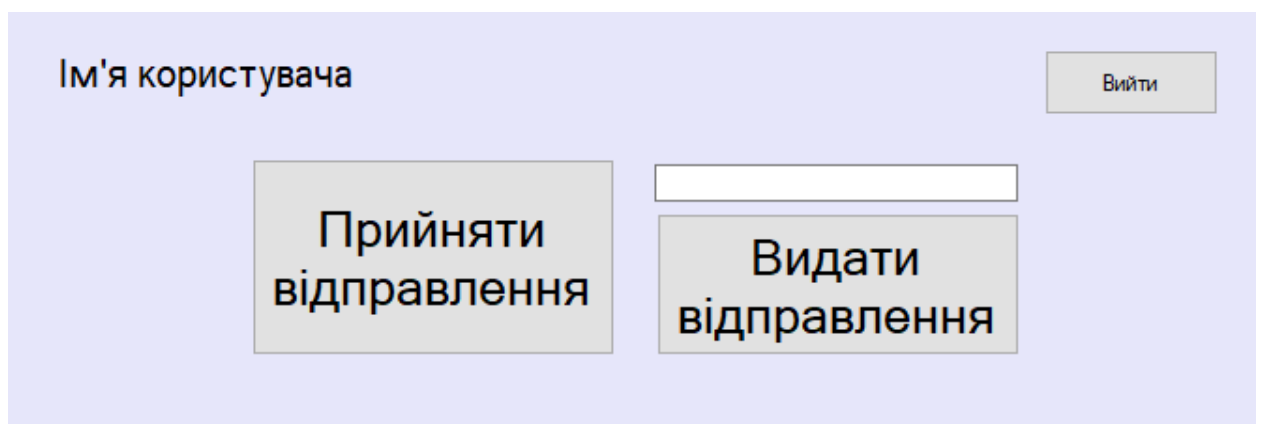


Рисунок 4.5 – Головна сторінка

Як видно зі знімку з екрану, робочу область поділено на дві частини: з ліва вкладка, що служить для оформлення вантажу клієнтів, та друга, де відображається відправлення на закриття [17].

Рисунок 4.6 – Вікно створення

Риунок 4.7 – Вікно авторизації

4.4.3 Тестування безпеки автоматизованої системи

Існують три основні вимоги до безпеки програмних продуктів [17]:

- конфіденційність - укриття певних ресурсів від деякої групи користувачів. Оскільки доступ до роботи мають тільки оператори, а доступ до усієї інформації має тільки адміністратор, додаток відповідає критерію;

- цілісність - дані відправлень можуть змінюватися тільки певною групою осіб (оператори) і зміна якихось важливих даних неможлива як для незареєстрованих так і для зареєстрованих користувачів;

- доступність - ресурси мають бути доступні авторизованому користувачу, внутрішньому об'єкту або пристрою.

Даний проєкт відповідає всім вищезазначеним критеріям безпеки.

4.4.4 Тестування стабільності

Даний програмний продукт повністю відповідає всім критеріям стабільності

- взаємодія між внутрішнім модулями;
- правильне відображення вкладок та їх елементів;
- деструктивне тестування.

ВИСНОВКИ

В ході виконання дипломної кваліфікаційної роботи бакалавра було виконано аналіз впливу людського фактору у роботі поштових компаній. Для цього було виконано наступні завдання:

- проаналізовано проблемну область створення та модифікування даних відправлення;
- було виконано огляд існуючих рішень завдання та аналогічного програмного забезпечення, проаналізовано його переваги та недоліки;
- спроектовано базу даних;
- розроблено власний програмний продукт.

Було розроблено максимально простий та зрозумілий інтерфейс, не перевантажений зайвими функціями.

В результаті виконання завдання було отримано систему, що відповідає всім вимогам визначеним на початку розробки та закріпленим в технічному завданні. Розроблена система є зручною для використання та виконує всі необхідні функції.

Розроблений програмний продукт повністю реалізує усі завдання дипломної кваліфікаційної роботи бакалавра.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Історія пошти України - Вікіпедія [Електрон. ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Історія_пошти_України.
2. Пошта – Вікіпедія [Електрон. ресурс]. – Режим доступу: <https://uk.wikipedia.org/wiki/Пошта>.
3. Сервіс Нова пошта [Електрон. ресурс]. – Режим доступу: <https://novaposhta.ua>.
4. Сервіс Укрпошта [Електрон. ресурс]. – Режим доступу: <https://www.ukrposhta.ua/ua>.
5. Сервіс FedEx [Електрон. ресурс]. – Режим доступу: <https://www.fedex.com/uk-ua/home.html>.
6. Валентина Дибська, Логістика складування для практиків / Альфа-Пресс. – 2005. – 208 с.
7. Власов К.П. Теорія автоматичного управління, 2013. – 544 с.
8. Джозеф Альбахари, С# 9.0.Довідник. Повний опис мови / Діалектика. – 2021. – 1056 с.
9. Роберт Мартін. Чистий код / Пітер. – 2019. - 464 с.
10. Стів Макконнелл. Досконалий код / Руська редакція. – 2019. – 895 с.
11. Безпечне програмування – Вікіпедія [Електрон. ресурс]. – Режим доступу : https://uk.wikipedia.org/wiki/Безпечне_програмування.
12. Ендрю Троелсен, Філіпп Джепікс. Мова програмування С# 7 та платформи .NET і .NET Core / Діалектика. – 2019. - 1328 с.
13. «Керівництво з мови С#» [Електрон. ресурс]. – Режим доступу: <https://docs.microsoft.com/ru-ru/dotnet/csharp/index>.
14. «Підручник з Microsoft Visual Studio» [Електронний ресурс]. – Режим доступу : <http://www.bhv.ru/books/book.php?id=188476>.

15. Kunal C. Mastering Visual Studio 2019: Become proficient in .NET Framework and .NET Core by using advanced coding techniques in Visual Studio, 2nd Edition. — Packt Publishing Ltd, 2019. — 374 с.
16. Структурна схема - Вікіпедія [Електрон. ресурс]. – Режим доступу: <https://uk.wikipedia.org/wiki/Блок-схема>.
17. Постіл С. Д. UML. Уніфікована мова моделювання інформаційних систем [Текст] : навч. посіб. / С. Д. Постіл ; Ун-т держ. фіск. служби України. - Ірпінь : Ун-т держ. фіск. служби України, 2019. - 321 с.
18. Гудзовата О. О. Використання уніфікованої мови візуального моделювання UML (Unified Modeling Language) як інструменту підтримки проектування інформаційних систем [Текст] / О. О. Гудзовата, В. І. Костирко, І. В. Артищук // Підприємництво і торгівля. - 2019. - Вип. 24. - С. 108-114.
19. Богдан І. Верифікація моделей об'єктно-орієнтованих програм: перевірка на несуперечливість та узгодженість [Текст] / І. Богдан // Техн. науки та технології. - 2017. - № 2. - С. 110-115.
20. Коваленко О. В. Імітаційна модель технології тестування безпеки на основі положень теорії масштабування / О. В. Коваленко // Безпека інформації. - 2018. - 24, № 2. - С. 110-117.
21. Гордон Сандер, Демон Шехтер, Логістика. Мистецтво управління ланцюжками поставок / Претекст. – 2010. – 140 с.
22. Віктор Трегуб, Автоматизація об'єктів періодичної дії / Ліра-К, 2017. - 136 с.
23. Діаграма станів (UML) - Вікіпедія [Електрон. ресурс]. – Режим доступу: [https://uk.wikipedia.org/wiki/Діаграма_станів_\(UML\)](https://uk.wikipedia.org/wiki/Діаграма_станів_(UML)).

ДОДАТОК А
Технічне завдання

Вступ

Програмне забезпечення, що розроблюється, має назву «автоматизована система служби доставки вантажів». У вік прискорення наукового і технічного прогресу в усіх сферах життєдіяльності людини передбачається широке впровадження інформаційних технологій як фундаменту подальшого розвитку людства.

Програмне забезпечення автоматизації системи служби доставки вантажів призначене для зменшення впливу людського фактору на процес роботи, прискорює алгоритм роботи.

A.1 Підстави для розробки

Розробка виконується на основі завдання на випускню дипломну роботу бакалавра, яке було визначене за темою «Програмне забезпечення для автоматизації системи служби доставки вантажів» і затверджене наказом № 102 від 27 квітня 2022 р. за Національним університетом «Запорізька політехніка».

A.2 Призначення розробки

Автоматизовану систему можна віднести до категорії «Програмне забезпечення». Він створений для прискорення та контролю роботи оператора поштової компанії з метою швидкої роботи з відправленнями.

A.3 Основні вимоги до програми, що розробляється

A.3.1 Вимоги до функціональних характеристик

Система має підтримувати:

- створення відправлення;

- додавання пакування;
- додаткові послуги;
- модифікування;
- оплата за послуги;
- друк експрес накладної.

A.3.2 Вимоги до надійності

Система повинна запускатися з сервера, також повинна бути забезпечена збережуваність інформації при збоях та її захист від несанкціонованого доступу, а так само на всіх етапах збору, обробки та зберігання за рахунок створення страхових копій бази даних та застосування системи паролів при організації доступу до інформації [17].

Для захищення особових даних кожен користувач мусить мати свій логін та пароль.

A.3.3 Умови експлуатації

Для користування автоматизованою системою, користувач повинен мати девайс з доступом до мережі Інтернет, підключений принтер та бути зареєстрованим [20].

A.3.4 Вимоги до складу та параметрів технічних засобів

Для роботи програми необхідно мати IBM-сумісний ПК з встановленою операційною системою Windows 7 та вище, з мікропроцесором Intel Core I3 2 GHz та вище, оперативна пам'ять 2Gb і вище, вільного простору на жорсткому диску 200 Мб, SVGA-монітор, клавіатура та маніпулятор типу “миша”. Для виводу на друк потрібен принтер.

A.3.5 Вимоги до маркування та упакування

Перша версія матиме маркування 1.0, наступні версії матимуть маркування 1.1, 2.0 і т.д.

A.3.6 Вимоги до транспортування та збереження

Допускається транспортування програмного виробу в транспортній тарі усіма видами транспорту. При транспортуванні та зберіганні програмного виробу має бути передбачений захист від попадання пилу і атмосферних опадів.

A.3.7 Вимоги до програмної документації

До складу програмної документації повинні входити:

- документ опису вимог;
- технічне завдання;
- опис програми.

A.4 Стадії та етапи розробки

Розробка всього проєкту складається з 4х етапів:

- аналіз даних. Він включає в себе два під етапи: збір та аналіз даних і складання технічного завдання;
- підготовка до розробки. Він включає в себе такі підетапи: розробка архітектури проєкту та підбирання готових компонентів-модулів для системи;
- розробка програмного продукту;
- фінальне тестування системи та здача замовнику готового додатку з усіма необхідними документами.

A.5 Порядок контролю та приймання системи

Для контролю та приймання системи використовується широко розповсюджена методологія керування проєктами SCRUM. Тобто весь проєкт розбивається на декілька модулів, при завершенні розробки певного модуля проєкт представляє собою завершену повноцінно працюючу систему.

Коли програмне забезпечення повністю розроблене та протестоване, воно передається замовнику на флеш-носії, двома архівами з розширенням zip, які містять у собі: перший – файли готового проєкту та експортовану БД, другий – всю необхідну документацію.

ДОДАТОК Б
Опис програми

Б.1 Загальні відомості

Програмне забезпечення, що розроблюється, має назву «автоматизована система служби доставки вантажів». У вік прискорення наукового і технічного прогресу в усіх сферах життєдіяльності людини передбачається широке впровадження інформаційних технологій як фундаменту подальшого розвитку людства.

Програмне забезпечення автоматизації системи служби доставки вантажів призначене для зменшення впливу людського фактору на процес роботи, прискорює алгоритм роботи.

Б.2 Функціональне призначення

Система має підтримувати:

- створення відправлення;
- додавання пакування;
- додаткові послуги;
- модифікування;
- оплата за послуги;
- друк експрес накладної.

Б.3 Опис логічної структури

На рисунку Б.1 наведено структурну схему ПЗ, що розробляється.

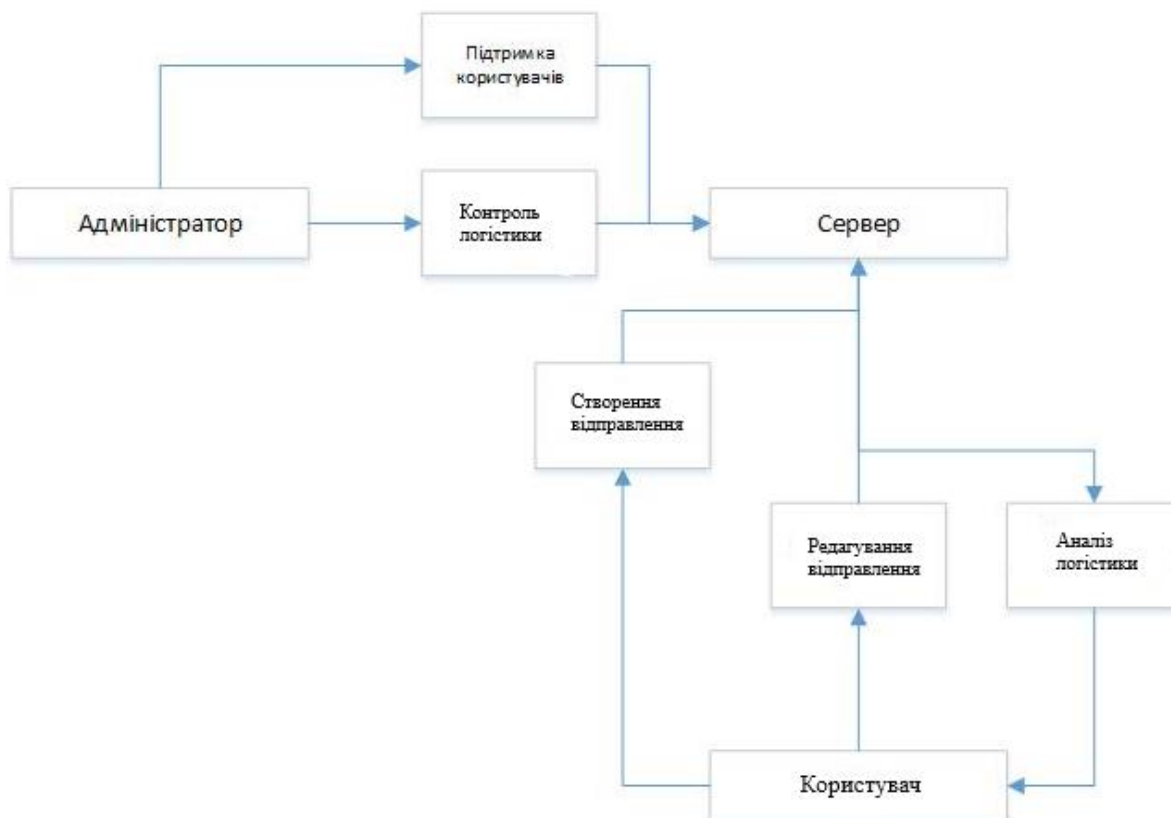


Рисунок Б.1 – Загальна схема системи за обраною темою

Microsoft Visual Studio - лінійка продуктів компанії Microsoft, що включають інтегроване середовище розробки програмного забезпечення і ряд інших інструментальних засобів. Дані продукти дозволяють розробляти як консольні додатки, так і додатки з графічним інтерфейсом, в тому числі з підтримкою технології Windows Forms, а також web-сайти, web-додатки, web-служби як в рідному, так і в керованому кодах для всіх платформ, підтримуваних Windows, Windows Mobile, Windows CE, .NET Framework, Xbox, Windows Phone .NET Compact Framework і Silverlight [14].

Б.4 Використані технічні засоби

Для роботи програми необхідно мати IBM-сумісний ПК з встановленою операційною системою Windows 7 та вище, з мікропроцесором Intel Core I3 2 GHz та вище, оперативна пам'ять 2Gb і вище, вільного

простору на жорсткому диску 200 Мб, SVGA-монітор, клавіатура та маніпулятор типу “миша”. Для виводу на друк потрібен принтер.

Б.5 Виклик та завантаження

Для роботи з програмою слід натиснути на файл запуску програми. Для керування використовуються елементи інтерфейсу.

Б.6 Вхідні та вихідні дані

Початковими даними програми є дані про користувача, що пройшов авторизацію, інформація про відправника, саме відправлення та отримувача. Вихідними даними є: виведена інформація про відправлення, та накладна з усією інформацією про відправлення, що користувач оформлював в автоматизованій системі.

ДОДАТОК В
Текст програми

В.1 Клас авторизації

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.OleDb;

namespace MyDiplom
{
    public partial class Form1 : Form
    {
        public static string ConnectionString1 = @"Provider=Microsoft.Jet.OLEDB.4.0; " +
            "Data Source= 1.mdb";
        public static string ConnectionString2 = @"Provider=Microsoft.ACE.OLEDB.12.0; " +
            "Data Source= 1.mdb";

        private OleDbConnection connection;
        public static string OperName;
        public static int OperVidd;
        public static int OperKod;
        public static string OperGorod;
        private volatile string LOGIN;
        private volatile string PASS;
        public Form1()
        {
            InitializeComponent();
            try
            {
                connection = new OleDbConnection();
                connection.ConnectionString = ConnectionString1;
                connection.Open();
            }
            catch (InvalidOperationException)
            {
                connection = new OleDbConnection();
                connection.ConnectionString = ConnectionString2;
                connection.Open();}}
        private void button1_Click(object sender, EventArgs e)
        {
            Operator oper = new Operator();
            oper.kod_user = 0;
            oper.FullName = "Nn";
            oper.Login = "Nn";
            oper.Pass = "Nn";
            oper.Position = "Nn";
            oper.User_vid = 0;
            oper.User_phone = "Nn";
            LOGIN = textBox1.Text;
            PASS = textBox2.Text;
            string sql = "SELECT User_vid.user_login, User_vid.user_pass,
            User_vid.user_fullname, User_vid.user_position, viddilenya.number_vid, User_vid.Kod_user,
            viddilenya.name_city FROM User_vid, viddilenya WHERE User_vid.user_vid =
            viddilenya.Kod_vid";
            OleDbCommand myCommand = new OleDbCommand(sql, connection);
            OleDbDataReader reader = myCommand.ExecuteReader();
            while (reader.Read()){

```

```

        if (LOGIN == reader[0].ToString() && PASS == reader[1].ToString()){
            oper.Login = reader[0].ToString();
            oper.Pass = reader[1].ToString();
            oper.FullName = reader[2].ToString();
            oper.Position = reader[3].ToString();
            oper.User_vid = Int32.Parse(reader[4].ToString());
            oper.kod_user = Int32.Parse(reader[5].ToString());
            OperName = oper.FullName;
            OperVidd = oper.User_vid;
            OperKod = oper.kod_user;
            OperGorod = reader[6].ToString();
            menu C = new menu();
            C.label1.Text = OperName;
            C.Show();}}
    reader.Close();
    textBox1.Text = "";
    textBox2.Text = "";
}
private void button2_Click(object sender, EventArgs e)
{
    Application.Exit();}}}

```

В.2 Клас головного меню

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.OleDb;

namespace MyDiplom
{
    public partial class menu : Form
    {
        public static string ConnectionString1 = @"Provider=Microsoft.Jet.OLEDB.4.0; " +
            "Data Source= 1.mdb";
        public static string ConnectionString2 = @"Provider=Microsoft.ACE.OLEDB.12.0; " +
            "Data Source= 1.mdb";
        private OleDbConnection connection;
        public menu()
        {
            InitializeComponent();
            try
            {
                connection = new OleDbConnection();
                connection.ConnectionString = ConnectionString1;
                connection.Open();
            }
            catch (InvalidOperationException)
            {
                connection = new OleDbConnection();
                connection.ConnectionString = ConnectionString2;
                connection.Open();
            }
        }

        private void button1_Click(object sender, EventArgs e)

```

```

{
    rmo A = new rmo();
    A.Show();
}

private void button2_Click(object sender, EventArgs e)
{
    Vidpravlenya viz = new Vidpravlenya();
    viz.VidpravKod = 0;
    viz.NameOperator = 0;
    viz.PhoneVidpravnika = "";
    viz.PhoneOtrimuvacha = "";
    viz.NameVidpravnika = "";
    viz.NameOtrimuvacha = "";
    viz.ViddilenyaVidpravnika = 0;
    viz.ViddilenyaOtrimuvacha = 0;
    viz.TypeVidpravlenya = false;
    viz.VesVidpravlenya = 0;
    viz.ObjemVidpravlenya = 0;
    viz.OpisVidpravlenya = "";
    viz.OgoloshVartist = 0;
    viz.Upakovka = 0;
    viz.DopUslugi = 0;
    viz.DateStvorenya = "";
    viz.TimeStvorenya = "";
    viz.DatePriezda = "";
    viz.GdeOtbito = "";
    viz.CityVidpravnika = "";
    viz.CityOtrimuvacha = "";
    viz.KtoPlatit = false;
    viz.Oplacheno = false; //dodav v vizit
    viz.VidpravlenyaPrice = 0;
    viz.ENalojka = false; // last
    viz.PriceNalojka = 0; // last
    viz.Vidp_zakrit = false;
    viz.TypeVidpravlenya = false;
    string sql = "SELECT * FROM vidprav WHERE vidprav.kod_vidp = " +
Int32.Parse(textBox1.Text);
    OleDbCommand myCommand = new OleDbCommand(sql, connection);
    OleDbDataReader reader2 = myCommand.ExecuteReader();
    while (reader2.Read())
    {
        viz.VidpravKod = Int32.Parse(reader2[0].ToString());
        viz.NameOperator = Int32.Parse(reader2[1].ToString());
        viz.PhoneVidpravnika = reader2[2].ToString();
        viz.PhoneOtrimuvacha = reader2[3].ToString();
        viz.NameVidpravnika = reader2[4].ToString();
        viz.NameOtrimuvacha = reader2[5].ToString();
        viz.ViddilenyaVidpravnika = Int32.Parse(reader2[6].ToString());
        viz.ViddilenyaOtrimuvacha = Int32.Parse(reader2[7].ToString());
        viz.TypeVidpravlenya = bool.Parse(reader2[8].ToString());
        viz.VesVidpravlenya = Int32.Parse(reader2[9].ToString());
        viz.OpisVidpravlenya = reader2[10].ToString();
        viz.OgoloshVartist = Int32.Parse(reader2[11].ToString());
        viz.KtoPlatit = bool.Parse(reader2[12].ToString());
        viz.VidpravlenyaPrice = Int32.Parse(reader2[13].ToString());
        viz.Oplacheno = bool.Parse(reader2[14].ToString());
        viz.DopUslugi = Int32.Parse(reader2[15].ToString());
        viz.Upakovka = Int32.Parse(reader2[16].ToString());
        viz.DateStvorenya = reader2[17].ToString();
        viz.TimeStvorenya = reader2[18].ToString();
        viz.DatePriezda = reader2[19].ToString();
        viz.GdeOtbito = reader2[20].ToString();
        viz.CityVidpravnika = reader2[21].ToString();
        viz.CityOtrimuvacha = reader2[22].ToString();
    }
}

```

```

        viz.ObjemVidpravlenya = Int32.Parse(reader2[23].ToString());
        viz.ENalojka = bool.Parse(reader2[24].ToString());
        viz.PriceNalojka = Int32.Parse(reader2[25].ToString());
        viz.Vidp_zakrit = bool.Parse(reader2[26].ToString());
    }
    reader2.Close();
    string namePakuv = "";
    string sql2 = "SELECT pakuv_name FROM pakuv WHERE pakuv.kod_pakuv = " +
viz.Upakovka;
    OleDbCommand myCommand2 = new OleDbCommand(sql2, connection);
    OleDbDataReader reader3 = myCommand2.ExecuteReader();
    while (reader3.Read())
    {
        namePakuv = reader3[0].ToString();
    }
    reader3.Close();
    string namePoslug = "";
    string sql3 = "SELECT poslug_name FROM poslugi WHERE poslugi.Код = " +
viz.DopUslugi;
    OleDbCommand myCommand3 = new OleDbCommand(sql3, connection);
    OleDbDataReader reader4 = myCommand3.ExecuteReader();
    while (reader4.Read())
    {
        namePoslug = reader4[0].ToString();
    }
    reader4.Close();
    vizit Bb = new vizit();
    Bb.label2.Text = viz.VidpravKod.ToString();
    Bb.label3.Text = viz.DateStvorenya;
    Bb.label4.Text = viz.TimeStvorenya;
    Bb.textBox1.Text = viz.PhoneVidpravnika;
    Bb.textBox2.Text = viz.NameVidpravnika;
    Bb.textBox3.Text = viz.OpisVidpravlenya;
    Bb.textBox4.Text = viz.OgoloshVartist.ToString();
    Bb.textBox5.Text = viz.VesVidpravlenya.ToString();
    Bb.textBox13.Text = namePakuv;
    Bb.label23.Text = viz.ObjemVidpravlenya.ToString();
    Bb.textBox9.Text = viz.PhoneOtrimuvacha;
    Bb.textBox10.Text = viz.NameOtrimuvacha;
    Bb.textBox11.Text = viz.CityOtrimuvacha;
    Bb.textBox12.Text = viz.ViddiljenjaOtrimuvacha.ToString();
    Bb.textBox14.Text = namePoslug;
    if (viz.KtoPlatit == true)
    {
        Bb.radioButton1.Checked = true;
    }
    else
    {
        Bb.radioButton2.Checked = true;
    }
    Bb.label18.Text = "Вартість послуг: " + viz.VidpravlenyaPrice.ToString() + "
€";
    Bb.Show();
}
private void button3_Click(object sender, EventArgs e)
{
    this.Close();
}
private void menu_Load(object sender, EventArgs e)
{
}
}
}

```

В.3 Клас створення відправлення

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.OleDb;
using System.Runtime.InteropServices;

namespace MyDiplom
{
    public partial class rmo : Form
    {
        public static string ConnectionString1 =
@"Provider=Microsoft.Jet.OLEDB.4.0; " +
        "Data Source= 1.mdb";
        public static string ConnectionString2 =
@"Provider=Microsoft.ACE.OLEDB.12.0; " +
        "Data Source= 1.mdb";

        private OleDbConnection connection;
        public static int NAMEOPERATOR;
        public static string PHONEVIDPRAVNIKA;
        public static string PHONEOTRIMUVACHA;
        public static string NAMEVIDPRAVNIKA;
        public static string NAMEOTRIMUVACHA;
        public static int VIDDILENYAVIDPRAVNIKA;
        public static int VIDDILENYAOTRIMUVACHA;
        public static bool TYPEVIDPRAVLENYA;
        public static int VESVIDPRAVLENYA;
        public static int OBJEMVIDPRAVLENYA;
        public static string OPISVIDPRAVLENYA;
        public static int OGOLOSHVARTIST;
        public static int VIDPRAVLENYAPRICE = 0;
        public static int BasePrice = 60;
        public static int KGPrice = 0;
        public static int DOPUSLUGI;
        public static int UPAKOVKA;
        public static string DATESTVORENYA;
    }
}

```

```

public static string TIMESTVORENYA;
public static string DATEPRIEZDA;
public static string GDEOTBITO;
public static string CITYVIDPRAVNIKA;
public static string CITYOTRIMUVACHA;
public static bool ENALOJKA;
public static int PRICENALOJKA;
public static bool VIDP_ZAKRIT;
public static int poslugG;
public static int upakG;
public static int Xo;
public static int Yo;
public static int Zo;
public static bool OPLACHENO = false;
public static int TypeCena = 0;
public static bool KTOPLATIT;
public static int OgoloshkaPrice;
public static int NumberEN;
public static string AllTime;
public rmo()
{
    InitializeComponent();
    try
    {
        connection = new OleDbConnection();
        connection.ConnectionString = ConnectionString1;
        connection.Open();
    }
    catch (InvalidOperationException)
    {
        connection = new OleDbConnection();
        connection.ConnectionString = ConnectionString2;
        connection.Open();
    }
}
private void button4_Click(object sender, EventArgs e)
{
    this.Close();
}
private void rmo_Load(object sender, EventArgs e)
{
    this.poslugiTableAdapter.Fill(this._1DataSet.poslugi);
    this.pakuvTableAdapter.Fill(this._1DataSet.pakuv);
}

```

```

DateTime date23 = new DateTime();
date23 = DateTime.Now;
AllTime = date23.ToLongDateString().ToString();
label13.Text = date23.ToShortDateString().ToString();
label14.Text = date23.ToShortTimeString().ToString();
DATESTVORENYA = date23.ToShortDateString().ToString();
TIMESTVORENYA = date23.ToShortTimeString().ToString();
DATEPRIEZDA = date23.AddHours(16).ToString();
}
private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
{
    upakG = 0;
    switch (comboBox1.SelectedIndex)
    {
        case 0: upakG = 0; break;
        case 1: upakG = 100; break;
        case 2: upakG = 80; break;
        case 3: upakG = 60; break;
        case 4: upakG = 150; break;
        case 5: upakG = 200; break;
        case 6: upakG = 350; break;
        case 7: upakG = 25; break;
        case 8: upakG = 70; break;
        case 9: upakG = 130; break;
        case 10: upakG = 25; break;
        case 11: upakG = 8; break;
        case 12: upakG = 10; break;
        case 13: upakG = 10; break;
        case 14: upakG = 15; break;
        case 15: upakG = 30; break;
        case 17: upakG = 40; break;
        case 18: upakG = 40; break;
        case 19: upakG = 50; break;
        case 20: upakG = 50; break;
        case 21: upakG = 18; break;
        case 22: upakG = 25; break;
        case 23: upakG = 5; break;
        case 24: upakG = 200; break;
        case 25: upakG = 25; break;
        case 26: upakG = 8; break;
        case 27: upakG = 10; break;
        case 28: upakG = 10; break;
        case 29: upakG = 15; break;
    }
}

```

```

        case 30: upakG = 30; break;
        case 31: upakG = 40; break;
        case 32: upakG = 40; break;
        case 33: upakG = 50; break;
        case 34: upakG = 50; break;
        case 35: upakG = 18; break;
        case 36: upakG = 25; break;
        case 37: upakG = 5; break;
        case 38: upakG = 200; break;

    }
}

private void comboBox2_SelectedIndexChanged(object sender, EventArgs e)
{
    poslugG = 0;
    switch (comboBox2.SelectedIndex)
    {
        case 0: poslugG = 0; break;
        case 1: poslugG = 100; break;
        case 2: poslugG = 60; break;
        case 3: poslugG = 120; break;
        case 4: poslugG = 30; break;
        case 5: poslugG = 30; break;
    }
}

private void button3_Click(object sender, EventArgs e)
{
    Vidpravlenya vdpr = new Vidpravlenya();
    vdpr.NameOperator = Form1.OperKod;
    vdpr.PhoneVidpravnika = textBox1.Text;
    vdpr.PhoneOtrimuvacha = textBox9.Text;
    vdpr.NameVidpravnika = textBox2.Text;
    vdpr.NameOtrimuvacha = textBox10.Text;
    vdpr.ViddilenyaVidpravnika = Form1.OperVidd;
    vdpr.ViddilenyaOtrimuvacha = Int32.Parse(textBox12.Text);
    vdpr.TypeVidpravlenya = TYPEVIDPRAVLENYA;
    vdpr.VesVidpravlenya = VESVIDPRAVLENYA;
    vdpr.ObjemVidpravlenya = OBJEMVIDPRAVLENYA;
    vdpr.OpisVidpravlenya = textBox3.Text;
    vdpr.OgoloshVartist = Int32.Parse(textBox4.Text);
    vdpr.Upakovka = 1+comboBox1.SelectedIndex;
    vdpr.DopUslugi = 1+comboBox2.SelectedIndex;
    vdpr.DateStvorenya = DATESTVORENYA;
}

```

```

vdpr.TimeStvorenya = TIMESTVORENYA;
vdpr.DatePriezda = DATEPRIEZDA;
vdpr.GdeOtbito = Form1.OperGorod;
vdpr.CityVidpravnika = Form1.OperGorod;
vdpr.CityOtrimuvacha = textBox11.Text;
vdpr.KtoPlatit = KTOPLATIT;
vdpr.Oplacheno = OPLACHENO;
vdpr.VidpravlenyaPrice = VIDPRAVLENYAPRICE;
vdpr.ENalojka = false;
vdpr.PriceNalojka = 0;
vdpr.Vidp_zakrit = false;
string sql = "INSERT INTO [vidprav] ( [vidp_name_operator]
,[vidp_phone_vidprav], [vidp_phone_otrim], [vidp_name_vidprav], [vidp_name_otrim],
[vidp_vidd_vidprav], [vidp_vidd_otrim], [vidp_type], [vidp-ves], [vidp_opis],
[vidp_ocenka], [vidp_splachue], [vidp_price], [vidp_oplacheno], [vidp_dop_poslug],
[vidp_pakuv], [vidp_date_stvorenya], [vidp_time_stvorenya], [vidp_date_priezd],
[vidp_otbito], [vidp_city_vidprav], [vidp_city_otrim], [vidp_massa], [vidp_nalojka],
[vidp_price_nalojka], [vidp_zakrito] ) VALUES ( " + vdpr.NameOperator + " , '" +
vdpr.PhoneVidpravnika + "' , '" + vdpr.PhoneOtrimuvacha + "' , '" + vdpr.NameVidpravnika
+ "' , '" + vdpr.NameOtrimuvacha + "' , " + vdpr.ViddilenyaVidpravnika + " , " +
vdpr.ViddilenyaOtrimuvacha + " , " + vdpr.TypeVidpravlenya + " , " + vdpr.VesVidpravlenya
+ " , '" + vdpr.OpisVidpravlenya + "' , " + vdpr.OgoloshVartist + " , " + vdpr.KtoPlatit
+ " , " + vdpr.VidpravlenyaPrice + " , " + vdpr.Oplacheno + " , " + vdpr.DopUslugi + " ,
" + vdpr.Upakovka + " , '" + vdpr.DateStvorenya + "' , '" + vdpr.TimeStvorenya + "' , '"
+ vdpr.DatePriezda + "' , '" + vdpr.GdeOtbito + "' , '" + vdpr.CityVidpravnika + "' , '"
+ vdpr.CityOtrimuvacha + "' , " + vdpr.ObjemVidpravlenya + " , " + vdpr.ENalojka + " , "
+ vdpr.PriceNalojka + " , " + vdpr.Vidp_zakrit + " )";

OleDbCommand myCommand = new OleDbCommand(sql, connection);
myCommand.ExecuteNonQuery();

string sql2 = "SELECT vidprav.kod_vidp FROM vidprav WHERE
vidprav.vidp_name_vidprav = '" + vdpr.NameVidpravnika + "' AND vidprav.vidp_name_otrim =
'" + vdpr.NameOtrimuvacha + "' AND vidprav.vidp_date_stvorenya = '" + vdpr.DateStvorenya +
"' AND vidprav.vidp_time_stvorenya = '" + vdpr.TimeStvorenya + "'";

OleDbCommand myCommand2 = new OleDbCommand(sql2, connection);
OleDbDataReader reader2 = myCommand2.ExecuteReader();
while (reader2.Read())
{
    NumberEN = Int32.Parse(reader2[0].ToString());
}
reader2.Close();

DialogResult dialogResult = MessageBox.Show("Данні введено вірно?" +
"\n\n" , " Підтвердження", MessageBoxButtons.OKCancel);
if (dialogResult == DialogResult.OK)

```

```

{
    EN Eee = new EN();
    Eee.label11.Text = Eee.label11.Text+" "+ NumberEN.ToString();
    Eee.label12.Text = Eee.label12.Text + " " + vdpr.DateStvorenya;
    Eee.label13.Text = Eee.label13.Text + " " + vdpr.DatePriezda;
    Eee.label15.Text = vdpr.NameVidpravnika;
    Eee.label16.Text = vdpr.CityVidpravnika;
    Eee.label18.Text = vdpr.PhoneVidpravnika;
    Eee.label10.Text = vdpr.NameOtrimuvacha;
    Eee.label11.Text = vdpr.CityOtrimuvacha;
    Eee.label13.Text = vdpr.PhoneOtrimuvacha;
    if (vdpr.TypeVidpravlenya == true)
    {
        Eee.label14.Text = Eee.label14.Text+" ВАНТАЖ";
    }
    else
    {
        Eee.label14.Text = Eee.label14.Text + " ПОСИЛКА";
    }
    Eee.label15.Text      =      Eee.label15.Text      +      "      "+
vdpr.VesVidpravlenya.ToString()+" ОБ'ЄМ: "+ vdpr.ObjemVidpravlenya.ToString();
    Eee.label17.Text      =      Eee.label17.Text      +      "      "      +
vdpr.OpisVidpravlenya;
    Eee.label18.Text      =      Eee.label18.Text      +      "      "      +
vdpr.OgoloshVartist.ToString();
    if (vdpr.KtoPlatit == true)
    {
        Eee.label19.Text = Eee.label19.Text + " ОТРИМУВАЧ";
    }
    else
    {
        Eee.label19.Text = Eee.label19.Text + " ВІДПРАВНИК";
    }
    Eee.label20.Text      =      Eee.label20.Text      +      "      "      +
vdpr.VidpravlenyaPrice.ToString() + " €";
    Eee.label22.Text = Form1.OperName;
    Eee.label23.Text = AllTime;
    Eee.Show();
}
else if (dialogResult == DialogResult.Cancel)
{
}
}

```

```

private void textBox6_TextChanged(object sender, EventArgs e)
{
    Xo = Int32.Parse(textBox6.Text);
    Yo = Int32.Parse(textBox7.Text);
    Zo = Int32.Parse(textBox8.Text);
    OBJEMVIDPRAVENYA = (Xo * Yo * Zo) / 4000;
    label23.Text = OBJEMVIDPRAVENYA.ToString();
}
private void textBox7_TextChanged(object sender, EventArgs e)
{
    Xo = Int32.Parse(textBox6.Text);
    Yo = Int32.Parse(textBox7.Text);
    Zo = Int32.Parse(textBox8.Text);
    OBJEMVIDPRAVENYA = (Xo * Yo * Zo) / 4000;
    label23.Text = OBJEMVIDPRAVENYA.ToString();
}
private void textBox8_TextChanged(object sender, EventArgs e)
{
    Xo = Int32.Parse(textBox6.Text);
    Yo = Int32.Parse(textBox7.Text);
    Zo = Int32.Parse(textBox8.Text);
    OBJEMVIDPRAVENYA = (Xo * Yo * Zo) / 4000;
    label23.Text = OBJEMVIDPRAVENYA.ToString();
}
private void button1_Click(object sender, EventArgs e)
{
    if (radioButton2.Checked)
    {
        OPLACHENO = true;
        label18.Text = "Вартість послуг: 0 €";
        VIDPRAVENYAPRICE = 0;
    }
    oplata opp = new oplata();
    opp.Show();
}

private void radioButton1_CheckedChanged(object sender, EventArgs e)
{
    if (radioButton1.Checked)
    {
        KTOPLATIT = true;
        VESVIDPRAVENYA = Int32.Parse(textBox5.Text);
        if (OBJEMVIDPRAVENYA > VESVIDPRAVENYA)

```

```

        {
            KGPrice = OBJEMVIDPRAVLHENYA;
        }
        else
        {
            KGPrice = VESVIDPRAVLHENYA;
        }
        if ((VESVIDPRAVLHENYA > 30) || (OBJEMVIDPRAVLHENYA > 30))
        {
            TYPEVIDPRAVLHENYA = true;
            TypeCena = 200;
        }
        VIDPRAVLHENYAPRICE = BasePrice+ OgoIoshkaPrice+ KGPrice + TypeCena
+ poslugG + upakG;

        label18.Text = "Вартість послуг: " + VIDPRAVLHENYAPRICE.ToString()
+ " €";
    }
}

private void radioButton2_CheckedChanged(object sender, EventArgs e)
{
    if (radioButton2.Checked)
    {
        KTOPLATIT = false;
        VESVIDPRAVLHENYA = Int32.Parse(textBox5.Text);
        if (OBJEMVIDPRAVLHENYA > VESVIDPRAVLHENYA)
        {
            KGPrice = OBJEMVIDPRAVLHENYA;
        }
        else
        {
            KGPrice = VESVIDPRAVLHENYA;
        }
        if ((VESVIDPRAVLHENYA > 30) || (OBJEMVIDPRAVLHENYA > 30))
        {
            TYPEVIDPRAVLHENYA = true;
            TypeCena = 200;
        }
        VIDPRAVLHENYAPRICE = BasePrice+ OgoIoshkaPrice+ KGPrice + TypeCena
+ poslugG + upakG;

        label18.Text = "Вартість послуг: " + VIDPRAVLHENYAPRICE.ToString()
+ " €";
    }
}

```

```

        private void textBox4_TextChanged(object sender, EventArgs e)
        {
            OGOLOSHVARTIST = Int32.Parse(textBox4.Text);
            OgoloshkaPrice = OGOLOSHVARTIST/ 100;
        }
    }
}

```

В.4 Клас перегляду відправлення

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.OleDb;

namespace MyDiplom
{
    public partial class vizit : Form
    {
        public static string ConnectionString1 = @"Provider=Microsoft.Jet.OLEDB.4.0; " +
            "Data Source= 1.mdb";
        public static string ConnectionString2 = @"Provider=Microsoft.ACE.OLEDB.12.0; " +
            "Data Source= 1.mdb";

        private OleDbConnection connection;
        public static bool OPLACHENO;
        public static int VIDPRAVLENYAPRICE;
        public static bool VIDP_ZAKRIT;
        public static int VIDP_EN;
        public vizit()
        {
            InitializeComponent();
            try
            {
                connection = new OleDbConnection();
                connection.ConnectionString = ConnectionString1;
                connection.Open();
            }
            catch (InvalidOperationException)
            {
                connection = new OleDbConnection();
                connection.ConnectionString = ConnectionString2;
                connection.Open();
            }
        }

        private void button1_Click(object sender, EventArgs e)
        {
            this.Close();
        }
        private void vizit_Load(object sender, EventArgs e)
        {
            this.poslugiTableAdapter.Fill(this._1DataSet.poslugi);
        }
    }
}

```

```

        this.pakuvTableAdapter.Fill(this._1DataSet.pakuv);
    }
    private void button4_Click(object sender, EventArgs e)
    {
        this.Close();
    }
    private void button3_Click(object sender, EventArgs e)
    {
        if (OPLACHENO = true)
        {
            VIDP_ZAKRIT = true;
            VIDP_EN = Int32.Parse(label2.Text);
            string sql = "UPDATE vidprav SET vidp_price = "+0+" , vidp_oplacheno =
"+true+" , vidp_zakrito = "+true+" WHERE kod_vidp = "+ VIDP_EN;
            OleDbCommand myCommand = new OleDbCommand(sql, connection);
            myCommand.ExecuteNonQuery();
            DialogResult dialogResult = MessageBox.Show(" Відправлення закрито" +
"\n\n", " Підтвердження");
        }
    }
    private void button1_Click_1(object sender, EventArgs e)
    {
        OPLACHENO = true;
        label18.Text = "Вартість послуг: 0 €";
        VIDPRAVLENYAPRICE = 0;
        oplata opp = new oplata();
        opp.Show();
    }
}
}

```

В.5 Клас експрес накладної

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace MyDiplom
{
    public partial class EN : Form
    {
        public EN()
        {
            InitializeComponent();
        }

        [System.Runtime.InteropServices.DllImport("gdi32.dll")]
        public static extern long BitBlt(IntPtr hdcDest, int nXDest, int nYDest,
int nWidth, int nHeight, IntPtr hdcSrc, int nXSrc, int nYSrc, int dwRop);
        private Bitmap memoryImage;
        private void CaptureScreen()
        {
            Graphics mygraphics = this.CreateGraphics();
            Size s = this.Size;
            memoryImage = new Bitmap(s.Width, s.Height, mygraphics);
        }
    }
}

```

```

        Graphics memoryGraphics = Graphics.FromImage(memoryImage);
        IntPtr dc1 = mygraphics.GetHdc();
        IntPtr dc2 = memoryGraphics.GetHdc();
        BitBlt(dc2, 0, 0, this.ClientRectangle.Width,
            this.ClientRectangle.Height, dc1, 0, 0, 13369376);
        mygraphics.ReleaseHdc(dc1);
        memoryGraphics.ReleaseHdc(dc2);
    }
    private void label2_Click(object sender, EventArgs e)
    {
    }
    private void button1_Click(object sender, EventArgs e)
    {
        button1.Visible = false;
        CaptureScreen();
        printDocument1.Print();
    }
    private void printDocument1_PrintPage(object sender,
        System.Drawing.Printing.PrintPageEventArgs e)
    {
        e.Graphics.DrawImage(memoryImage, 0, 0);
    }

    private void button2_Click(object sender, EventArgs e)
    {
        this.Close();
    }
}
}

```

В.6 Клас оплати

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace MyDiplom
{
    public partial class oplata : Form
    {
        public oplata()
        {
            InitializeComponent();
        }
        private void button1_Click(object sender, EventArgs e)
        {
            this.Close();
        }
        private void oplata_Load(object sender, EventArgs e)
        {
            webBrowser1.Navigate("https://i.pining.com/564x/03/b3/c5/03b3c53c7e44aa048e56218e6e9ca7f6.jpg");
        }
    }
}

```

```
}
```

В.7 Клас оператора

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace MyDiplom
{
    public class Operator
    {
        private int Kod_user;
        private string fullName;
        private string login;
        private string pass;
        private string position;
        private int user_vid;
        private string user_phone;
        public int kod_user
        {
            get => Kod_user;
            set => Kod_user = value;
        }
        public string FullName
        {
            get => fullName;
            set => fullName = value;
        }
        public string Login
        {
            get => login;
            set => login = value;
        }
        public string Pass
        {
            get => pass;
            set => pass = value;
        }
        public string Position
        {
            get => position;
            set => position = value;
        }
        public int User_vid
        {
            get => user_vid;
            set => user_vid = value;
        }
        public string User_phone
        {
            get => user_phone;
            set => user_phone = value;
        }
    }
}
```

B.8 Клас Vidpravleniya

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace MyDiplom
{
    public class Vidpravleniya
    {
        private int vidpravKod;
        private int nameOperator;
        private string phoneVidpravnika;
        private string phoneOtrimuvacha;
        private string nameVidpravnika;
        private string nameOtrimuvacha;
        private int viddilenyaVidpravnika;
        private int viddilenyaOtrimuvacha;
        private bool typeVidpravleniya;
        private int vesVidpravleniya;
        private int objemVidpravleniya;
        private string opisVidpravleniya;
        private int ogoloshVartist;
        private bool ktoPlatit;
        private bool oplacheno;
        private int vidpravleniyaPrice;
        private int dopUslugi;
        private int upakovka;
        private string dateStvorenya;
        private string timeStvorenya;
        private string datePriezda;
        private string gdeOtbito;
        private string cityVidpravnika;
        private string cityOtrimuvacha;
        private bool eNalojka;
        private int priceNalojka;
        private bool vidp_zakrit;
        public int VidpravKod

[System.Runtime.InteropServices.DllImport("gdi32.dll")]
        public static extern long BitBlt(IntPtr hdcDest, int nXDest, int nYDest,
int nWidth, int nHeight, IntPtr hdcSrc, int nXSrc, int nYSrc, int dwRop);
        private Bitmap memoryImage;
        private void CaptureScreen()
        {
            Graphics mygraphics = this.CreateGraphics();
            Size s = this.Size;
            memoryImage = new Bitmap(s.Width, s.Height, mygraphics);
            Graphics memoryGraphics = Graphics.FromImage(memoryImage);
            IntPtr dc1 = mygraphics.GetHdc();
            IntPtr dc2 = memoryGraphics.GetHdc();
            BitBlt(dc2, 0, 0, this.ClientRectangle.Width,
                this.ClientRectangle.Height, dc1, 0, 0, 13369376);
            mygraphics.ReleaseHdc(dc1);
            memoryGraphics.ReleaseHdc(dc2);
        }

        {
            get => vidpravKod;
            set => vidpravKod = value;
        }
    }
}

```

```

public int NameOperator
{
    get => nameOperator;
    set => nameOperator = value;
}

public string PhoneVidpravnika
{
    get => phoneVidpravnika;
    set => phoneVidpravnika = value;
}
public string PhoneOtrimuvacha
{
    get => phoneOtrimuvacha;
    set => phoneOtrimuvacha = value;
}
public string NameVidpravnika
{
    get => nameVidpravnika;
    set => nameVidpravnika = value;
}
public string NameOtrimuvacha
{
    get => nameOtrimuvacha;
    set => nameOtrimuvacha = value;
}
public int ViddilenyaVidpravnika
{
    get => viddilenyaVidpravnika;
    set => viddilenyaVidpravnika = value;
}
public int ViddilenyaOtrimuvacha
{
    get => viddilenyaOtrimuvacha;
    set => viddilenyaOtrimuvacha = value;
}
public bool TypeVidpravlenya
{
    get => typeVidpravlenya;
    set => typeVidpravlenya = value;
}
public int VesVidpravlenya
{
    get => vesVidpravlenya;
    set => vesVidpravlenya = value;
}
public int ObjemVidpravlenya
{
    get => objemVidpravlenya;
    set => objemVidpravlenya = value;
}
public string OpisVidpravlenya
{
    get => opisVidpravlenya;
    set => opisVidpravlenya = value;
}
public int OgoleshVartist
{
    get => ogoleshVartist;
    set => ogoleshVartist = value;
}
public bool KtoPlatit
{
    get => ktoPlatit;
    set => ktoPlatit = value;
}

```

```

}
public bool Oplacheno
{
    get => oplacheno;
    set => oplacheno = value;
}

public int VidpravlenyaPrice
{
    get => vidpravlenyaPrice;
    set => vidpravlenyaPrice = value;
}

public int DopUslugi
{
    get => dopUslugi;
    set => dopUslugi = value;
}

public int Upakovka
{
    get => upakovka;
    set => upakovka = value;
}

public string DateStvorenya
{
    get => dateStvorenya;
    set => dateStvorenya = value;
}

public string TimeStvorenya
{
    get => timeStvorenya;
    set => timeStvorenya = value;
}

public string DatePriezda
{
    get => datePriezda;
    set => datePriezda = value;
}

public string GdeOtbito
{
    get => gdeOtbito;
    set => gdeOtbito = value;
}

public string CityVidpravnika
{
    get => cityVidpravnika;
    set => cityVidpravnika = value;
}

public string CityOtrimuvacha
{
    get => cityOtrimuvacha;
    set => cityOtrimuvacha = value;
}

public bool ENalojka
{
    get => eNalojka;

```

```
        set => eNalojka = value;
    }
    public int PriceNalojka
    {
        get => priceNalojka;
        set => priceNalojka = value;
    }

    public bool Vidp_zakrit
    {
        get => vidp_zakrit;
        set => vidp_zakrit = value;
    }
}
}
```

ДОДАТОК Д
Слайди презентації

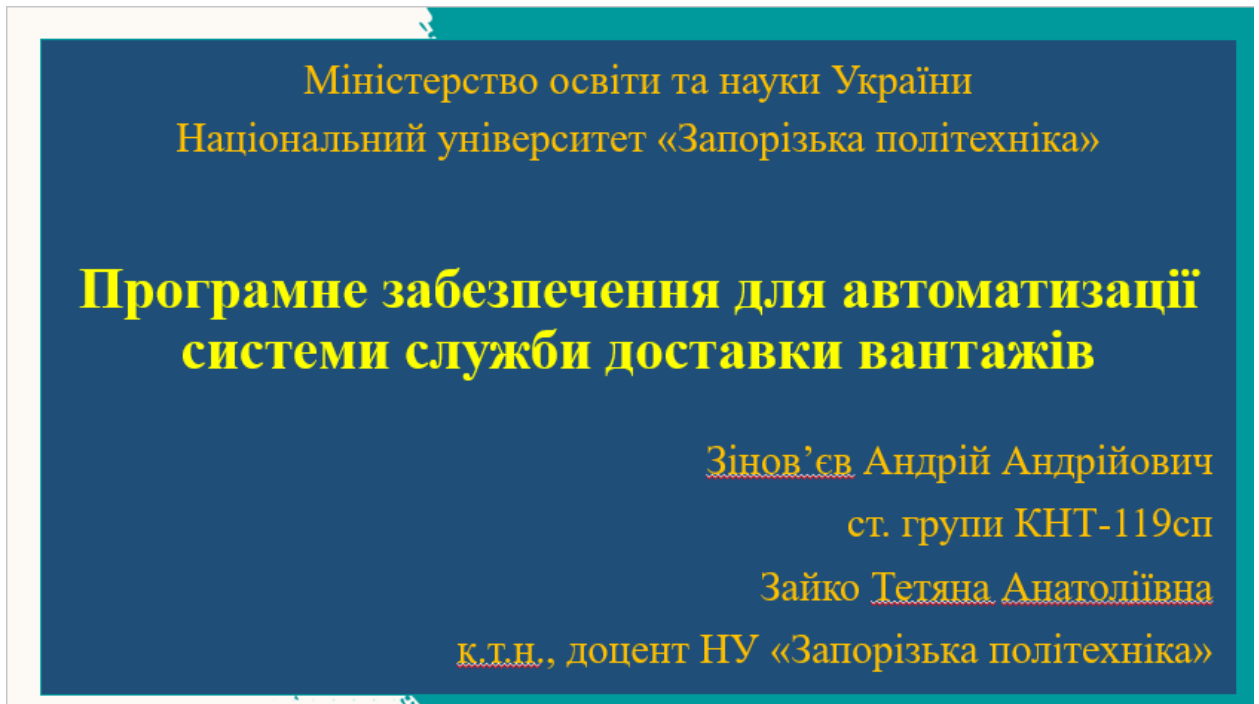


Рисунок Д.1 – Слайд 1

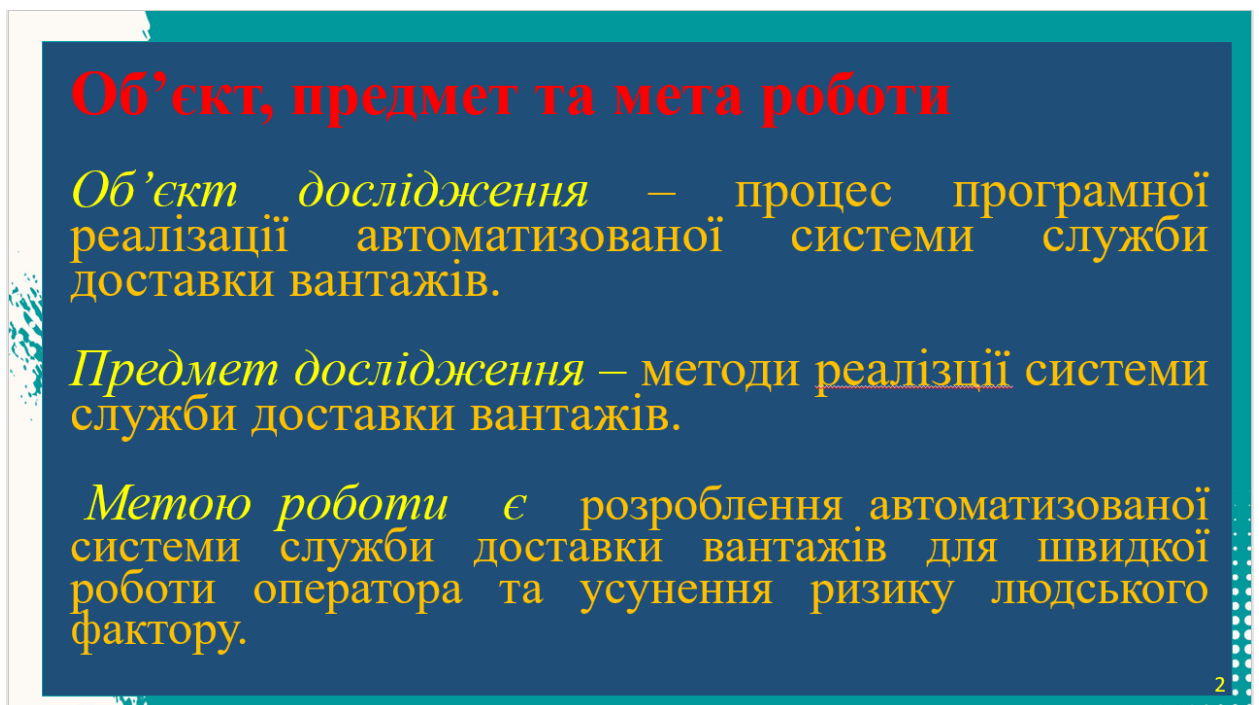


Рисунок Д.2 – Слайд 2

Постановка завдання

Метою проекту є реалізація автоматизованої системи для оператора вантажного відділення, що дозволить створювати, зберігати, відстежувати та аналізувати на сервері файли у приватній та загальнодоступній формі, ділитися інформацією з іншими працівниками компанії.

3

Рисунок Д.3 – Слайд 3

Порівняння існуючих аналогів

Критерій порівняння	Нова пошта	Укрпошта	FedEx
Ціна послуг	Дорога, від розміри відправлення	Дуже дешево	Дорога, від відстані перевезення
Швидкість перевезення	Швидко	Повільно	Дуже швидко
розподіл часу на утримання вантажу в сховищі	Неділя	Місяць	День
обмеження за розміром відправлення	Є	Є	Немає
Сервіс	Низької якості	Середня якість	Якісний

4

Рисунок Д.4 – Слайд 4

Порівняння мов програмування

Критерій	C#	Java	Python
Швидкість роботи та продуктивність	++	+	±
Складність синтаксису	+	+	++
Підтримка	++	+	++
Програмні бібліотеки	++	±	±

5

Рисунок Д.5 – Слайд 5

Порівняння існуючих середовищ розробки

	Visual Studio	IntelliJ IDEA	VS Code
Необхідність платної підписки	-	+	-
Інтелектуальне доповнення	+	+	+
Кросплатформовість	+	±	+
Зручний рефакторинг коду	++	+	+
Інтегрований Git-інструментарій	++	+	+
Можливість встановлення сторонніх розширень	++	-	+

6

Рисунок Д.6 – Слайд 6

Загальна схема системи за обраною темою

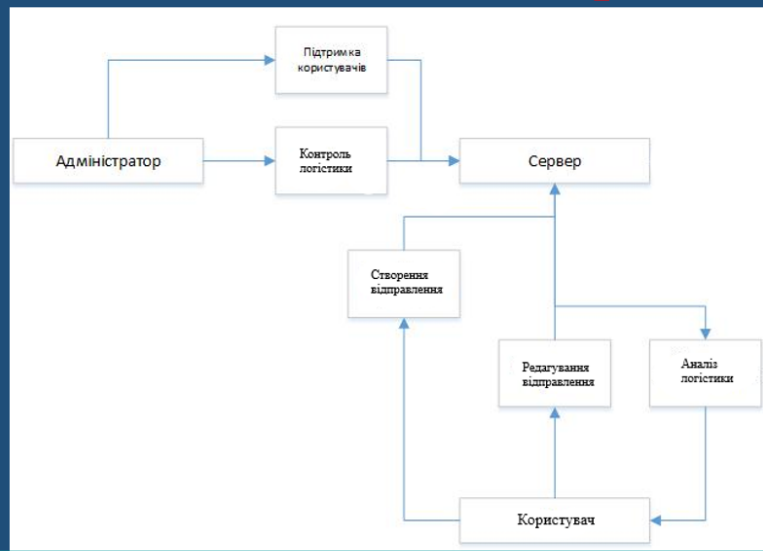


Рисунок Д.7 – Слайд 7

Проектування моделей системи

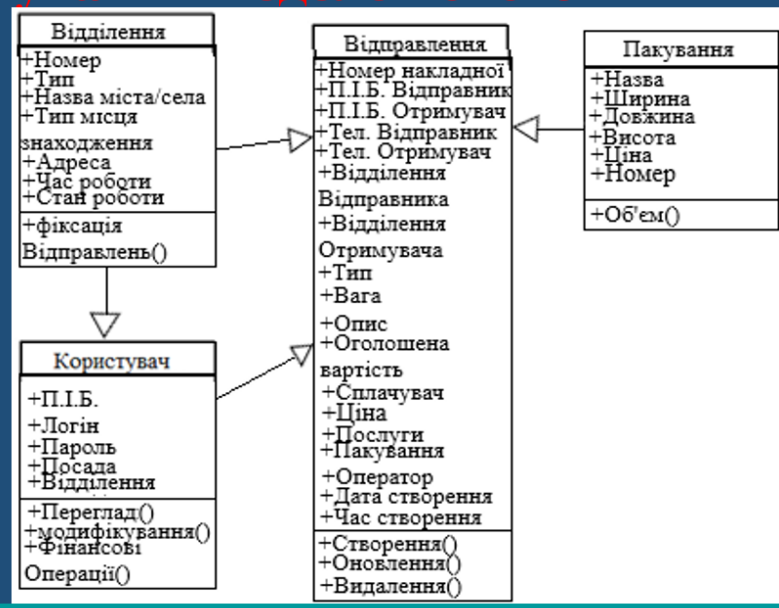


Рисунок Д.8 – Слайд 8



Рисунок Д.9 – Слайд 9

Тестування

Ласкаво просимо

Логін:

Пароль:

візит № 00000000 14.06.2022 12:53

Відправник номер телефону 0660459933 Прізвище Ім'я Побатькові ів'єв Андрій Андрійович	Відправлення Опис відправлення Оголошена вартість 1000 Пакунання: коробка 30 кг Ширина 40 Довжина 33 Висота 32 об'єм: 10	Автозапчастини Вага 23 Отримувач Телефон 0502829123 Прізвище Ім'я Побатькові Куц Сергій Русланович Місто Київ Відділення 23	Оплата Додаткові послуги: доставка в руки Платник ☒ Отримувач ☐ Відправник Вартість послуг: 243 ₴ Сплатити Створити
--	--	---	---

ЕКСПРЕС НАКЛАДНА № 66
ДАТА ОФОРМЛЕННЯ: 14.06.2022
ЧАС ДОСТАВКИ: 15.06.2022 5:03:45
ВІДПРАВНИК
Зіновійєв Андрій Андрійович
Запоріжжя
0660459933
ОДЕРЖУВАЧ
Куц Сергій Русланович
Київ
0502829123
ТИП ВІДПРАВЛЕННЯ: ПОСИЛКА
ВАГА: 23 ОБ'ЄМ: 14
ОПИС: Автозапчастини
ОГОЛОШЕНА ВАРТІСТЬ: 1000
ДОСТАВКУ СПЛАЧУЄ: ОТРИМУВАЧ
ЦІНА: 243 ₴
ПРЕДСТАНИК КОМПАНІЇ:
Зіно'єв Андрій Андрійович
14 июня 2022 г.
0-900-500-610
З умовами Публічного договору про надання послуг з організації перевезення відправлень, ознайомлений

10

Рисунок Д.10 – Слайд 10

Тестування

Зіно'єв Андрій Андрійович Вийти

**Прийняти
відправлення**

**Видати
відправлення**

Amount: 0.00 UAH

Description payment:

Pay with Privat24

24 Pay

G Pay **** 7142

Card Another method

Card number: **** * 0000 0000

Expiration: M M / Y Y CV2: *** ?

By pressing the button «Pay» you accept Terms and conditions

Pay

візит № 66 14.06.2022 13:03 Вийти

Відправник номер телефону: 0660459933 Прізвище Ім'я Побатькові: Зіно'єв Андрій Андрій	Відправлення Опис відправлення: Автозапчастини Оголошена вартість: 1000 Вага: 23 Пакування: коробка 30 кг Ширина: 0 Довжина: 0 Висота: 0 об'єм: 14	Отримувач Телефон: 0502829123 Прізвище Ім'я Побатькові: Куц Сергій Русланович Місто: Київ Відділення: 23	Оплата Додаткові послуги: доставка в руки Платник: ☒ Отримувач ☐ Відправник Вартість послуг: 243 ₪
---	---	--	---

Сплатити
Закрити

11

Рисунок Д.11 – Слайд 11

Висновки

Під час роботи було виконано наступні завдання:

- проаналізовано проблемну область створення та модифікування даних відправлення;
- був виконаний огляд існуючих рішень завдання та аналогічного програмного забезпечення, проаналізовано його переваги та недоліки;
- спроектовано базу даних;
- розроблено власний програмний продукт;
- Проведено тестування.

В результаті виконання завдання було отримано систему, що відповідає всім вимогам визначеним на початку розробки та закріпленим в технічному завданні. Розроблена система є зручною для використання та виконує всі необхідні функції. Розроблений програмний продукт повністю реалізує усі завдання дипломної кваліфікаційної роботи бакалавра.

12

Рисунок Д.12 – Слайд 12