МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ХАРЧОВИХ ТЕХНОЛОГІЙ

Інститут (факультет) Автоматизації і комп'ютерних систем
Кафедра інформаційних систем

«До захисту в ЕК»		**«До захисту допущено»**	
Директор інституту (декан факультету)		Завідувач кафедри	
_____ (підпис)	Андрій Форсюк (ім'я та прізвище)	_____ (підпис)	Сергій Чумаченко (ім'я та прізвище)
«___» _____ 2022 р.		«___» _____ 2022 р.	

КВАЛІФІКАЦІЙНА РОБОТА
НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ БАКАЛАВРА

зі спеціальності 122 «Комп'ютерні науки»
(код та назва спеціальності)
освітньо-професійної програми_____
на тему: Створення моделі управлінням проектом розробки та впровадження інформаційної системи для логістичного відділу транспортної компанії

Виконав: здобувач 4 курсу, групи КН-4-3СК Корнієвський А.С.
(прізвище та ініціали)

Керівник Мазуренко Ольга Олександрівна _____
(прізвище та ініціали) (підпис)

Консультанти_____ _____
(ім'я та прізвище) (підпис)

_____ _____
(ім'я та прізвище) (підпис)

_____ _____
(ім'я та прізвище) (підпис)

Рецензент Ярослав Смітюх___ _____
(ім'я та прізвище) (підпис)

Я як здобувач(ка) Національного університету харчових технологій розумію і підтримую політику університету з академічної доброчесності. Я не надавав(-ла) і не одержував(-ла) недозволеної допомоги під час підготовки цієї роботи. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

Здобувач_____
(підпис)

Київ - 2022р.

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ХАРЧОВИХ ТЕХНОЛОГІЙ

Інститут (факультет) Автоматизації і комп'ютерних систем
Кафедра Інформаційних систем
Освітній ступінь бакалавр
Спеціальність 122 «Комп'ютерні науки»
(код і назва)
Освітньо-професійна програма «Комп'ютерні науки»
(назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри Чумаченко С.М.

"____" _____ 2022 року

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧА

Корнієвський Андрій Сергійович
(прізвище, ім'я, по батькові)

1. Тема роботи: Створення моделі управлінням проектом розробки та впровадження інформаційної системи для логістичного відділу транспортної компанії.
керівник роботи Мазуренко Ольга Олександрівна, к.т.н.,ст.викл,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом закладу вищої освіти від "31 " 03 2022 року №163 -кс
2. Строк подання здобувачем роботи 05 червня 2022 р
3. Вихідні дані до роботи *дані про транспортну компанію , документи та нормативи, що використовуються в процесі функціонування логістичного відділу компанії*
4. Зміст пояснювальної записки (перелік питань, які потрібно розробити)
Системний аналіз діяльності логістичного відділу, задачі автоматизації та їх вирішення.
5. Перелік графічного матеріалу
1. Організаційна структура логістичного відділу транспортної компанії
2. Контекстна діаграма функціональної моделі та діаграми декомпозиції
3.Логічний та фізичний рівень моделі даних
4. Схема бази даних у MS SQL Server
5.Скріншоти інтерфейсу системи

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Вступ			
I	Мазуренко Ольга Олександрівна, к.т.н., ст.викл.		
II	Мазуренко Ольга Олександрівна, к.т.н., ст.викл.		
Висновок			

7. Дата видачі завдання___00 квітня 0000 року_____

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів виконання кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Системний аналіз логістичного відділу транспортної компанії	**15.03.2022 – 26.03.2022**	Виконано
2	Розробка моделі даних та створення бази даних	**01.04.2022 – 18.04.2022**	Виконано
3	Визначення та реалізація функцій інформаційної системи	**19.04.2022 – 09.05.2022**	Виконано
4	Оформлення пояснювальної записки	**12.05.2022 – 30.05.2022**	Виконано
5	Розробка презентації	**30.05.2022 – 31.05.2022**	Виконано
6			Виконано
7			Виконано
8			Виконано
9			Виконано

Здобувач _____ __Корнієвський А.С.____
(підпис) (прізвище та ініціали)

Керівник роботи _____ ____Мазуренко О.О._____
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Головною метою даної бакалаврської роботи є створення моделі управлінням проектом розробки та впровадження інформаційної системи для логістичного відділу транспортної компанії.

Сама робота поділяється на два етапи, які в подальшому вилились в два розділи описаних в пояснювальній записці.

Перший етап - аналіз логістичного відділу транспортної компанії.

Другий етап - це створення програмного забезпечення та обрання технологій які були використані в створенні інформаційної системі.

Об'єктом дослідження в кваліфікаційній роботі є логістичний відділ транспортної компанії.

Бакалаврська робота містить 48 сторінок, 4 таблиць, 53 рисунків, 5 додатків і 11 літературних джерел.

КЛЮЧОВІ СЛОВА: ІНФОРМАЦІЙНА СИСТЕМА, ЛОГІСТИКА, ТРАНСПОРТНА КОМПАНІЯ, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, БАЗА ДАНИХ.

ANNOTATION

The main purpose of this bachelor's thesis is to create a project management model for the development and implementation of an information system for the logistics department of a transport company.

The work itself is divided into two stages, which later resulted in two sections described in the explanatory note.

The first stage is the analysis of the logistics department of the transport company.

The second stage is the creation of software and the choice of technologies that were used in the creation of the information system.

The object of research in the qualification work is the logistics department of the transport company.

The bachelor's thesis contains 48 pages, 4 tables, 53 figures, 5 appendices and 11 references.

KEY WORDS: INFORMATION SYSTEM, LOGISTICS, TRANSPORT COMPANY, SOFTWARE, DATABASE.

Зміст

ВСТУП

Логістично-транспортні компанії складають велику частку підприємств нашої держави, так їх послуги є необхідними для перевезень товарів в Україні та безпосередньо для експорту продукції в інші країни.

Діяльність транспортних компаній зазвичай пов’язана з обліково-бухгалтерською роботою.

Існує багато різновидів інформаційних систем для таких видів задач, однак вони мають в собі велику кількість зайвих для компанії функцій, і із-за чого мають велику вартість.

Розглянемо що саме мається на увазі під ІС, а це є сукупністю організаційних і технічних засобів для збереження та обробки даних з метою забезпечення потреб користувачів.

Метою цього проекту є отримання готової системи зі зручним інтерфейсом користувача

РОЗДІЛ 1.СИСТЕМНИЙ АНАЛІЗ ДІЯЛЬНОСТІ ТРАНСПОРТНОЇ КОМПАНІЇ

1.1 Загальна характеристика транспортної компанії

Транспортна галузь є однією з найважливіших для України, оскільки для повноцінної роботи будь якого виробничого підприємства неабияку роль відіграє логістичні послуги.

В рамках проходження практики було досліджено транспортну компанію ФОП “Мудрук”.

Транспортна компанія має в собі головну суть, а саме, надання широкого вибору вантажоперевезень та логістичних послуг юридичним та фізичним лицям.

В умовах постійної конкуренції серед транспортних компаній всі транспортні підприємства повинні мати постійний розвиток, тобто підприємства не повинні завжди нарощувати клієнтську базу та поповнювати транспортний парк новою технікою.

Транспортна компанія працює з клієнтами з всієї України, але в деяких випадках – і на території інших сусідніх держав.

В собі транспортна компанія налічує близько 20-30 працівників різних напрямків: бухгалтерів, логістів, водіїв, юристів тощо.

Транспортна компанія працює в своєму циклі, який складається із прийому вантажу, який транспортна компанія отримає від відправника, перевірку вантажу на вміст заборонених предметів та речовин, тобто сканування вантажу за допомогою рентгенівського апарату, потім їде формування документації на вантаж та його подальше транспортування. Після цього, якщо компанія отримає нове замовлення, цикл починається заново.

1.2 Організаційна структура транспортної компанії

1.2.1 Загальна схема організаційної структури транспортної компанії

Транспортна компанія ФОП “Мудрук” — не велике, порівняно з іншими транспортними компаніями-монополістами, але і в нього є достатня організаційна структура.

Генеральний директор в транспортній компанії, частіше за все, є і власником або співвласником цього підприємства. Далі генеральний директор має підопічних по ієрархії: керівника фінансового відділу, Керівника логістичного відділу, керівника технічного відділу та керівника юридичного відділу.

Керівник фінансового відділу зазвичай потрібен для забезпечення складання різних фінансових документів, які, в свою чергу потрібні для функціонування компанії.

Керівник логістичного відділу відповідає за надання клієнтам логістичних послуг.

Керівник технічного відділу в свою чергу потрібен для контролю якості роботи працівників автопарку компанії.

Юридичний відділ потрібен для розв’язання спорів між компанією та клієнтами у разі виникнення суперечок щодо надання послуг компанії.

Загальний вигляд організаційної структури підприємства зображено на рис.1.1.

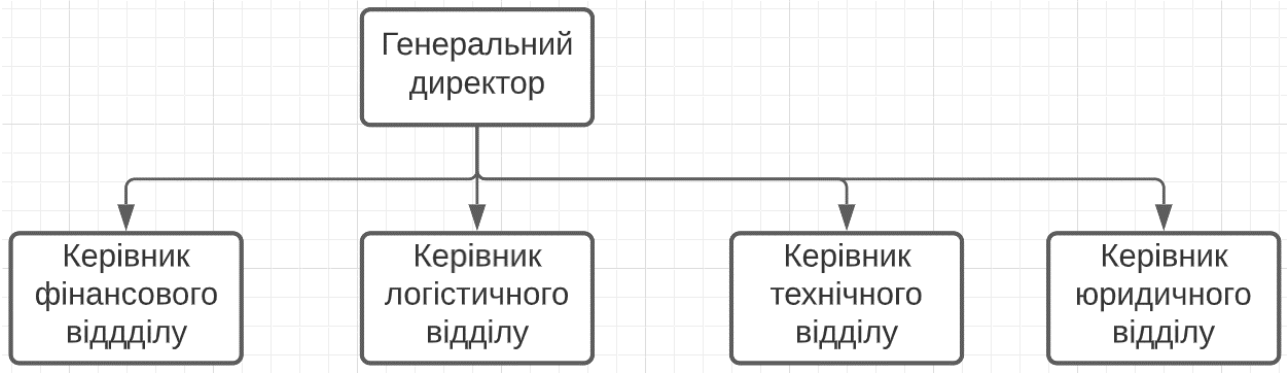


Рис.1.1. Організаційна структура підприємства

Кожний підрозділ компанії та посада має свою зону відповідальності, свої задачі та обов'язки. Перелік їх задач наведено у таблиці 1.1.

Таблиця 1.1. Обов'язки та задачі підрозділів на підприємстві

№	Підрозділ/Посада	Задачі
1	Генеральний директор	В більшості випадків на цій керуючій посаді находиться сам власник підприємства або співвласник, задає саме напрямок того як саме буде розвиватись підприємство та які послуги будуть надаватись клієнтам.
2	Керівник фінансового відділу	Відділ працівники якого потрібні для: Забезпечення оплати витрат для ефективного функціонування компанії, забезпечення складання рахунків по зарплаті, нарахувань і перерахувань податків і зборів до бюджетів різних рівнів, платежів у банківські установи, Контроль за своєчасним і правильним оформленням бухгалтерським документообігом, здійснення заходів щодо усунення втрат і непродуктивних витрат, Забезпечення ведення бухгалтерського обліку, складання фінансової звітності підприємства

Таблиця 1.1. Обов'язки та задачі підрозділів на підприємстві (Продовження)

3	Керівник логістичного відділу	Керує відділом працівники якого повинні бути атестовані особи, тобто яка пройшла відповідне навчання та перевірку знань з питань логістики, і відповідає за прокладання маршрутів доставки вантажів.
4	Керівник технічного відділу	Керує відділом в якому підлеглі повинні вміти утримувати автопарк компанії в належному стані, а саме: проводити заходи догляду та ремонту транспорту компанії.

5	Керівник юридичного відділу	Відділ працівників які юридично підковані. Вони допомагають функціонувати компанії на юридичному рівні.

1.2.2 Схема організаційної структури логістичного відділу транспортної компанії

Структурна схема структури логістичного відділу представлена на рис.1.2.

Рис.1.2. Структура логістичного відділу

Керівник логістичного відділу, – працівник на посаді який має, беручи до уваги накази генерального директора , забезпечувати ефективне керування логістичним відділом.

Заступник керівника логістичного відділу – працівник, що має обов'язки так ж як і у керівника, однак виконує він їх тільки якщо керівника немає на робочому місті.

Оператор перевірки вантажу – працівник на посаді який повинний перевіряти вантаж на перелік заборонених речовин за допомогою пошукового апарату.

Оператор обробки замовлень – працівник який повинен обробляти нові замовлення від клієнтів, працювати з документацією, вести облік клієнтів.

Диспетчер – працівник який координує роботу відділу.

Вантажник – працівник, який має обов'язок завантажувати вантажі до транспорту компанії

Основною задачею відділу є доставка вантажу споживачеві в потрібне місце, вчасно, в потрібній кількості та потрібної якості.

Найважливіші функції цього відділу описані в таблиці 1.2

Таблиця 1.2 Задачі і функції відділу

№	Задачі	Опис функції
1	Скорочення витрат	Використання наявних коштів має бути ефективним. Наприклад, збірні перевезення, це вигідний варіант доставки.
2	Створення оптимальних маршрутів переміщення	Створення оптимального маршруту доставки. Щоб операція доставки займала якнайменш часу і зводилась до найкращого використання ресурсів.
3	Операції з обробки вантажу	Пакувально-маркувальні роботи. Навантажувально-розвантажувальні роботи. Облік і обробка замовлень.
4	Забезпечення перевірки багажу на вміст заборонених предметів та речовин	Проведення візуального огляду вантажа та більш детального огляду вантажа за допомогою пошукових апаратів або огляду з втручанням кінолога з собакою.
5	Забезпечення координації та взаємодії з замовником	Координація замовника та відповідь на можливі його запитання щодо стану його вантажу.

1.2.3 Взаємодія логістичного відділу з другими відділами

Логістичний відділ дуже тісно взаємодіє з усіма іншими відділами підприємства, що можна побачити у табл.1.3.

Таблиця 1.3. Взаємодія підрозділів

№	Підрозділ	Одержання	Надання
1	Технічний відділ	- актів про успішний огляд транспорту - відомостей для виписки квитанцій для повернення коштів за паливо	-Актів про порушення роботи транспорту -Актів про використане паливо під час транспортування
2	Фінансовий відділ	- Зарплат - Премій - Довідок щодо рівня з/п - Повернення коштів за паливо - Накладних	- Актів про порушення правил для подальшого введення штрафних санкцій проти персоналу - Відомостей про вагу та розміри вантажу

1.3 Діаграма діяльності роботи зерносховища

Під час дослідження функціонування логістичного відділу було проведено опитування працівників на рахунок всіх механізмів роботи відділу, тому приведена діаграма відповідає реальній діяльності логістичного відділу.

Діаграма діяльності має в собі декілька етапів.

Спочатку відбувається надходження вантажу від клієнта до компанії. Персонал проводить обробку замовлення і робить маркування вантажу на складі.

Далі вантаж проходить перевірку, і якщо перевірка була успішно пройдена, працівник формує логістично-транспортну накладну і паралельно до цього вантаж завантажується на транспорт компанії. (Див Рис 1.3.).

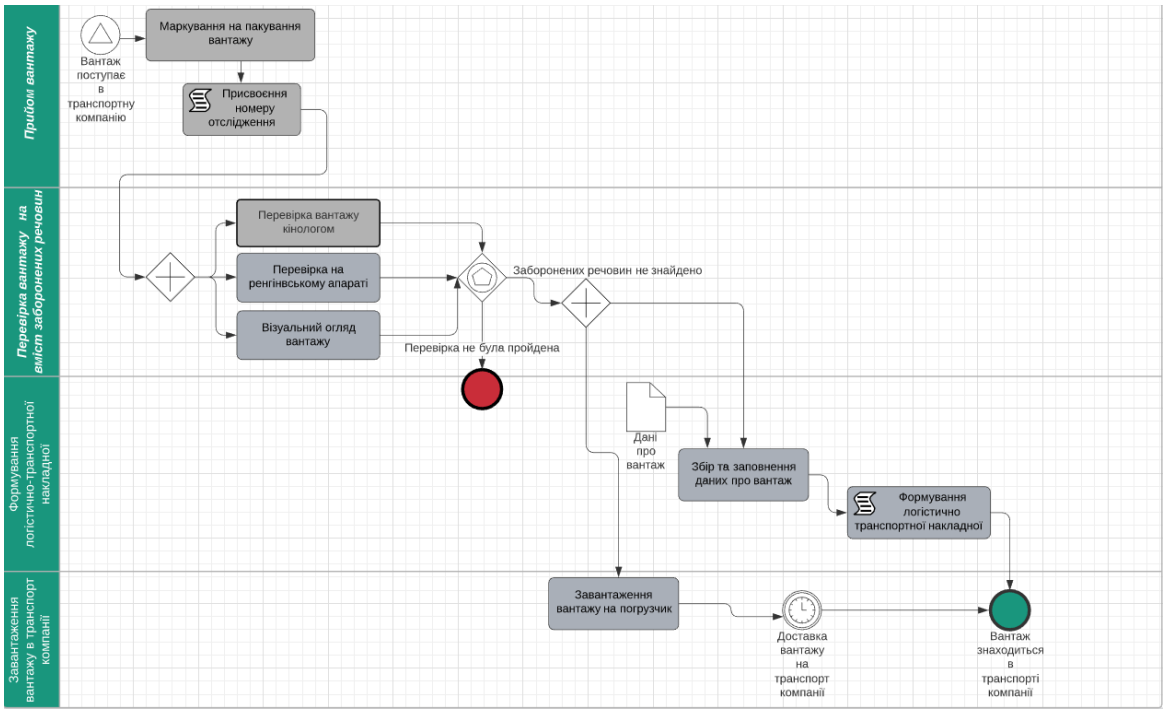


Рис.1.3 Функціональна схема роботи зерносховища

1.4 Стан автоматизації відділу

Важливу роль у діяльності логістичного відділу відіграє персонал який здійснює облік даних які необхідні для роботи компанії, та формування звітності та інших документів.

Програмне забезпечення, яким користуються на підприємстві, називається – «Документація 1.0».

«Документація 1.0» це вітчизняне програмне забезпечення, яке потрібно для автоматизації роботи підприємств. Також, за допомогою «Документація1.0» є можливість легкого адміністрування та автоматизації бізнес-процесів.

В логістичному відділу використовується декілька рішень для автоматизації, а саме автоматизації перевірки вантажу, автоматизації формування звітності.

Також, дане рішення забезпечує автоматизацію процесів роботи зі звітною документацією встановленого зразка.

1.5 Розроблення функціональної моделі та аналіз бізнес-процесів

1.5.1 Функціональна модель діяльності логістичного відділу

На базі того що задача є створення інформаційної системи, тобто, потрібно створити або доповнити ІС для логістичного відділу. Для цього потрібно мати інструмент або технологію. А для цієї задачі було обрано саме CASE.

CASE (Computer-Aided Software / System Engineering) – це технології та інструментарій CASE-засобів, що дозволяє максимально систематизувати і автоматизувати всі етапи розробки програмного забезпечення.

Розглянемо який склад має функціональна модель. Переглянути функціональну модель можливо в додатку А.

Дані, документи та матеріальні об’єкти, які потрібні для того щоб описати бізнес-процес у логістичному відділі транспортної компанії:

- Вантаж;
- Супровідна документація;

Всі дії в самому відділу, по моделі, керуються такими документами:

- Організаційні документи
- Штатно-посадові інструкції

В результаті відділ надає такі документи та товари:

- Вантаж що пройшов перевірку;
- Вантаж що не пройшов перевірку;
- Сформовані документи;

В моделі фігурують такі механізми, які забезпечують роботу відділу:

- Персональний комп’ютер
- Персонал відділу

Діаграма має в собі ще моделі, декомпозиція А-0, а саме А0 має в собі функції прийому вантажу (А1), перевірки вантажу на вміст заборонених речовин

(А2), формування документів (А3), завантаження вантажу в транспорт компанії (А4).

В результаті виконаної роботи було створено функціональну модель яка відображає як саме влаштовано логістичний відділ транспортної компанії.

1.5.2 Виявлені проблеми

Основна проблема відділу заключається в тому що більша частина роботи займає формування саме документації та звітів по роботі компанії, бо вона виконується вручну в паперовому вигляді, а це займає доволі багато часу роботи працівників, та також існує імовірність втрати паперових записів, що в подальшому приведе до проблем. Також, за довгий час роботи фірми накопичується доволі багато записів, які, в свою чергу, подалі буде ускладнювати навігацію та пошук потрібної інформації.

Загалом були виявлені такі недоліки:

- Відсутність формування рахунку-фактури
- Відсутність експорту даних в формі файлу Excel
- Відсутність пошуку даних
- Введення обліку клієнтів в паперовому вигляді.

1.5.3 Задачі автоматизації

Головною ціллю системи є підвищення швидкості роботи персоналу компанії за рахунок автоматизованого введення, редагування, пошуку, вибірки даних, формування документів тощо.

Працівник зерносховища повинен виконувати великий обсяг роботи, від приймання зерна, до його догляду. І важливою частиною для підвищення результативності та ефективності діяльності працівників зерносховища, та всього зерносховища, відіграє покращення процесу внесення даних в інформаційну базу.

Користувачами даної підсистеми є тільки працівники. Гостей для цього виду підсистеми не повинні мати доступу.

Обов’язками користувача є заповнення необхідними даними відповідних таблиці, для подальшого використання їх у вигляді звіту.

Для модулів інформаційної системи були поставлені такі задачі:

- Експорт даних інформаційної системи в вигляді таблиць Excel;
- Можливість пошуку потрібних даних за будь-яким критерієм;
- Можливість додавання, редагування та видалення даних.
- Авторизація працівника в інформаційну систему;
- Забезпечити автоматичного обрахунку статистики роботи відділу.

1.6 Огляд існуючих рішень для розв’язання виявлених проблем

Для пошуку вирішення проблеми, потрібно розглянути існуючі на ринку інформаційні системи і обґрунтувати доречність створення нового програмного продукту відносно його вартості та витратам людино-годин на розробку.

Таблиця 1.3. Порівняльна таблиця існуючих систем

Назва системи	1С:Підприємство	Парус	Ананас
Доступні мови	Російська, англійська	Українська, Російська, Англійська та інші.	Російська
Операційна система	Microsoft Windows, Linux, Android, Mac OS та IOS	Microsoft Windows, Android/IOS/Mac OS	Microsoft Windows
Вартість	Від 25.000 грн	295грн за місяць.	Безкоштовно
Переваги	"Оперативний облік" призначена для обліку	Сервіс BookKeeper SaaS являє собою	Ананас – перша вільна облікова

	наявності та руху матеріальних та грошових коштів. Вона може використовуватись як автономно, так і спільно з іншими компонентами "1С:Підприємства". "1С:Торгівля та склад" призначена для обліку будь-яких видів торгових операцій. Завдяки гнучкості та настроюваності, система здатна виконувати всі функції обліку - від ведення довідників та введення первинних документів до отримання різних відомостей та аналітичних звітів. [8]	рішення автоматизації обліку на сучасній програмній платформі A2v10. Платформа A2v10 призначена для створення майже будь-яких бізнес-застосунків (ERP, CRM, BPM тощо) як з web-інтерфейсом, так і в десктопних варіантах. Платформа побудована на технологічному стеку .NET корпорації Microsoft і може бути використана для розробки як on-premise рішень, так і хмарних сервісів. [9]	платформа для користувачів Linux та Windows. Написаний на C++ (інтерфейс Qt), і працює з MySQL, PostgreSQL та SQLite. Код програми вільний, тому програмою можна користуватися як у домашніх, так і в комерційних цілях будь-якої кількості комп'ютерів.[10]

Отже розглянувши ті системи які вже існують на ринку, було прийнято рішення створити свій власний програмний продукт тому, що 1С є забороненою на території України, а інші інформаційні системи не підходять, завдяки відсутності функції автоматизованого формування документації.

1.7 Обґрунтування доцільності проектування й розроблення інформаційної підтримки діяльності логістичного відділу транспортної компанії

Завдяки аналізу програмного продукту, що вже є у компанії, було вирішено, що задля зменшення виплати за щомісячні підписки на програмний продукт, та відповідно ключів для робочих місць, слід створити власний продукт, який буде мати в собі тільки потрібні функції для роботи підприємства

, а крім того, підприємство матиме змогу удосконалювати свій програмний продукт у майбутньому .

Дане рішення в короткострокових термінах не є привабливою так як розробка програмного забезпечення та подальше впровадження є довгою роботою. Однак в довгострокових термінах розробка унікального ПЗ для компанії є привабливою, бо в майбутньому не потрібно буде щомісячно платити підписки на інші програмні продукти.

Завдяки цьому можна зробити висновок що програмне забезпечення транспортної компанії в майбутньому зможе конкурувати з іншими подібними продуктами.

Отож, розроблення ІС для логістичного відділу транспортної компанії є доцільним вирішенням виявлених проблем.

РОЗДІЛ 2. ОПИС КОМПЛЕКСУ ЗАДАЧ АВТОМАТИЗАЦІЇ

2.1. Інформаційне забезпечення системи

Було використано програмний засіб CA ERwin Data Modeler для того щоб правильно вистроїти структуру БД, а також для подальшої генерації БД в MS SQL server.

Першим кроком буде створення логічної моделі яка буде відображати те як саме дані виглядають в реальному світі.

Подальшим кроком є створення на базі логічної моделі, фізичну, а саме ту модель, яка буде генеруватись як база даних в MS SQL Server.

Модель фізичного типу є універсальною та побудована на основі моделі процесів (див. Додаток Б).

Кожна сутність має свій набір атрибутів, а також є унікально ідентифікованою завдяки первинним ключам (PK). Окрім первинних ключів, сутності мають також альтернативні ключі (АК), які також є унікальними, та інверсні входи (IE). [2]

Фізична модель таблиці та її колонки мають велику кількість налаштувань, а саме валідація для даних, що будуть вводиться в атрибути. Властивості атрибутів мають можливість також настроїти який тип даних буде вводиться тобто числові, текстові та інші типи даних.

Фізичну модель можливо переглянути в Додатку В.

Структура фізичної моделі даних та наявні правила валідації описані в таблиці 2.1.

Таблиця 2.1. Структура фізичної моделі та правила валідації

№	Таблиці	Колонки та їх типи даних	Правила валідації
1	Акт прийняття вантажу	○ ІД_Акт_прийняття_вантажу(РК): integer ○ Дата: datetime ○ Код персоналу(FK): integer ○ ІД_Відправника(FK) : integer ○ Код вантажу(FK): integer	
2	Вантаж	○ Код вантажу (РК): integer ○ Вага_відправлення: integer ○ Тип_вантажу: text ○ Опис_вантажу: text ○ Код_накладної(FK): integer	
3	Відправник	○ ІД_Відправника (РК): integer ○ Супровідна_документація: text ○ Електронна_пошта: varchar(75) ○ Піб_відпраника : text	
4	Логістично - транспортна накладна	○ Код_накладної (РК): integer ○ Код_персоналу(FK): integer ○ Код_транспорту(FK): integer	
5	Персонал	○ Код_персоналу (РК): integer ○ ПІБ: text ○ Посада: text ○ Телефон: varchar(20)	Телефон: тільки цифри
6	Рахунок-фактура	○ Код_рахунку_фактури (РК): integer ○ Ціна доставки: integer ○ Код_персоналу(FK): integer ○ ІД_Відправника(FK) : integer ○ Дата: datetime ○ Код вантажу(FK): integer ○ ІД_Акт_прийняття_вантажу(FK): integer	Дата: автоматичне виставлення сьогоднішньої дати
7	Транспорт	○ Код_транспорту (РК): integer ○ Модель: text ○ Рік_випуску: integer ○ Вантажопідйомність: integer	Вантажопідйомність: не менше 0.

На основні фізичній моделі було згенеровано відповідну базу даних в середовищі СУБД MS SQL Server 2008. Схема створеної БД можна переглянути у Додатку Б, рис. Б.3.

2.2. Проектування БД

З самого початку потрібно створити логічну та фізичну моделі в в AllFusion ERWin Data Modeler. Їх можна побачити в додатку Б.

Після створення моделі в ERWin, потрібно перенести цю модель в робоче середовище sql server. Для цього в ERWin існує потрібний для наших подальших шагів інструментарій.

Після перевірки всіх функцій моделі, потрібно зрозуміти та вибрати де буде створена подальша модель, і вибір пав саме на SQL Server що є продуктом Microsoft.

З початку вибирається в який сервер буде проводиться генерація бази, вікно вибору виглядить ось так:

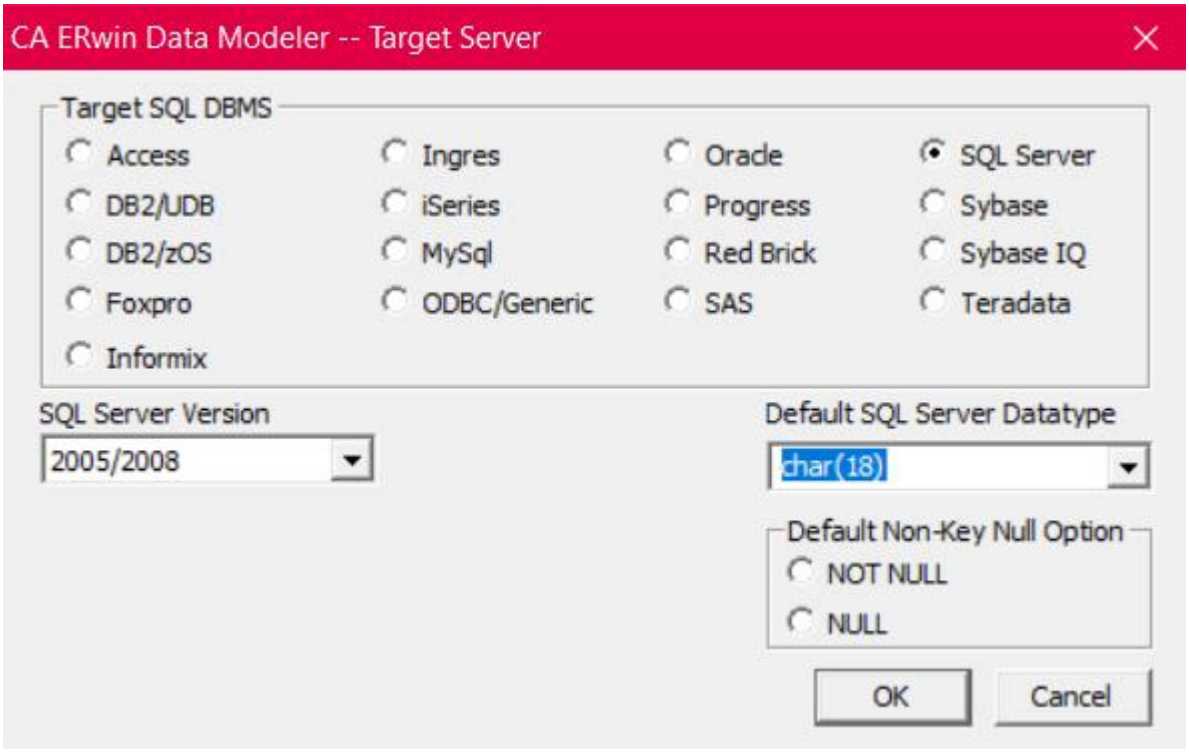


Рисунок 2.1: Вибір сервера в ERWin Data Modeler

Далі потрібно підключитись до бази даних, яка зарані була створена в сервері, за допомогою SQL Server Management Studio, тому що сам інструментарій створює тільки запити на створення. Тобто генерація складається із структурованих SQL запитів, і автоматично все створює.

Для того щоб провести генерацію таблиць, представлень і тому подібне. Вікно в якому здійснюється підключення відображено на рисунку 2.2:

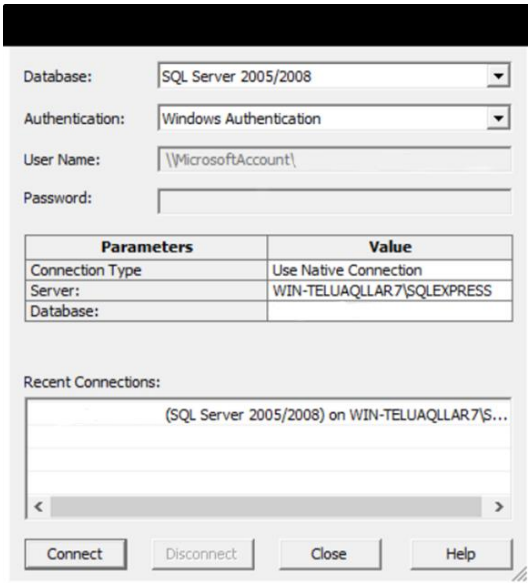


Рисунок 2.2: Під’єднання до SQL

В наступному кроці уже проходить налаштування, що саме потрібно згенерувати, від таблиць до процедур та тригерів БД. Все це відбувається в такому вікні.

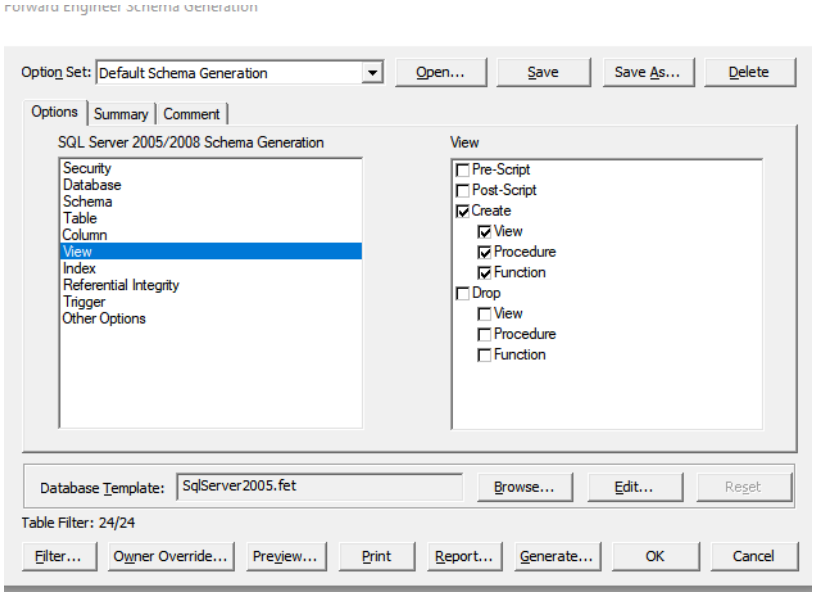


Рисунок 2.3: Налаштування генерації

Останнім кроком є сама генерація. Яка відбувається після натискання кнопки генерації. Саме вікно генерації виглядить ось так:

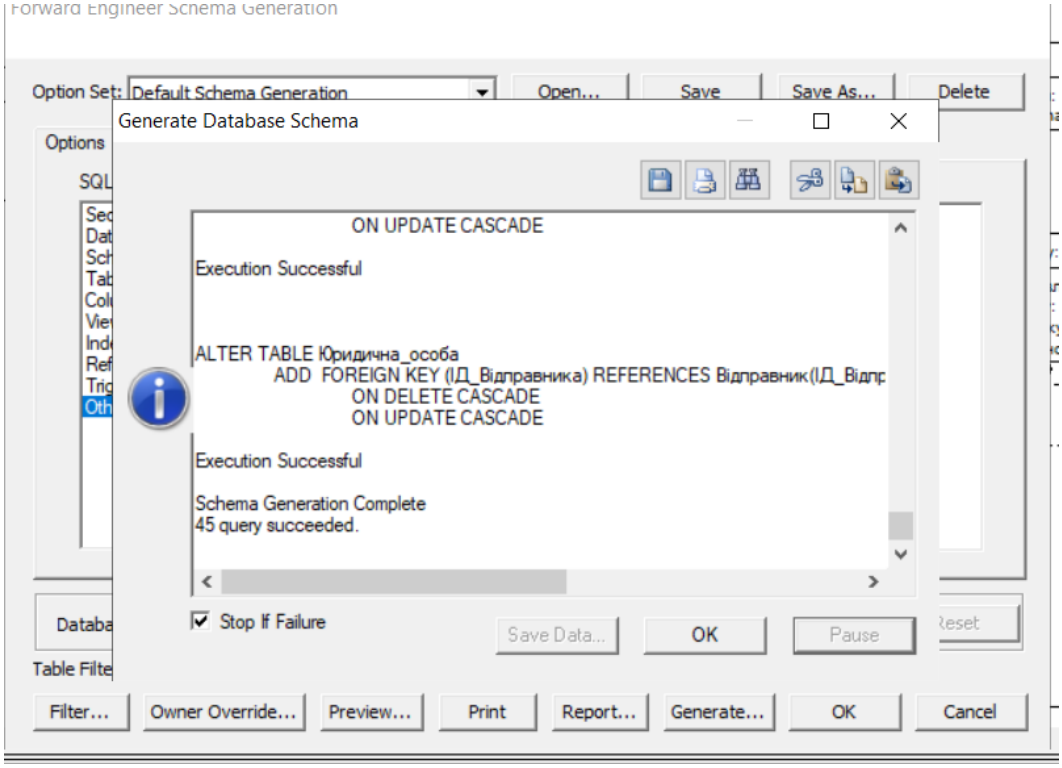


Рисунок 2.4: Генерація бази даних

Результат всіх дій виглядить ось так, а саме в SQL Server Management Studio, відображено на рисунку 2.5.

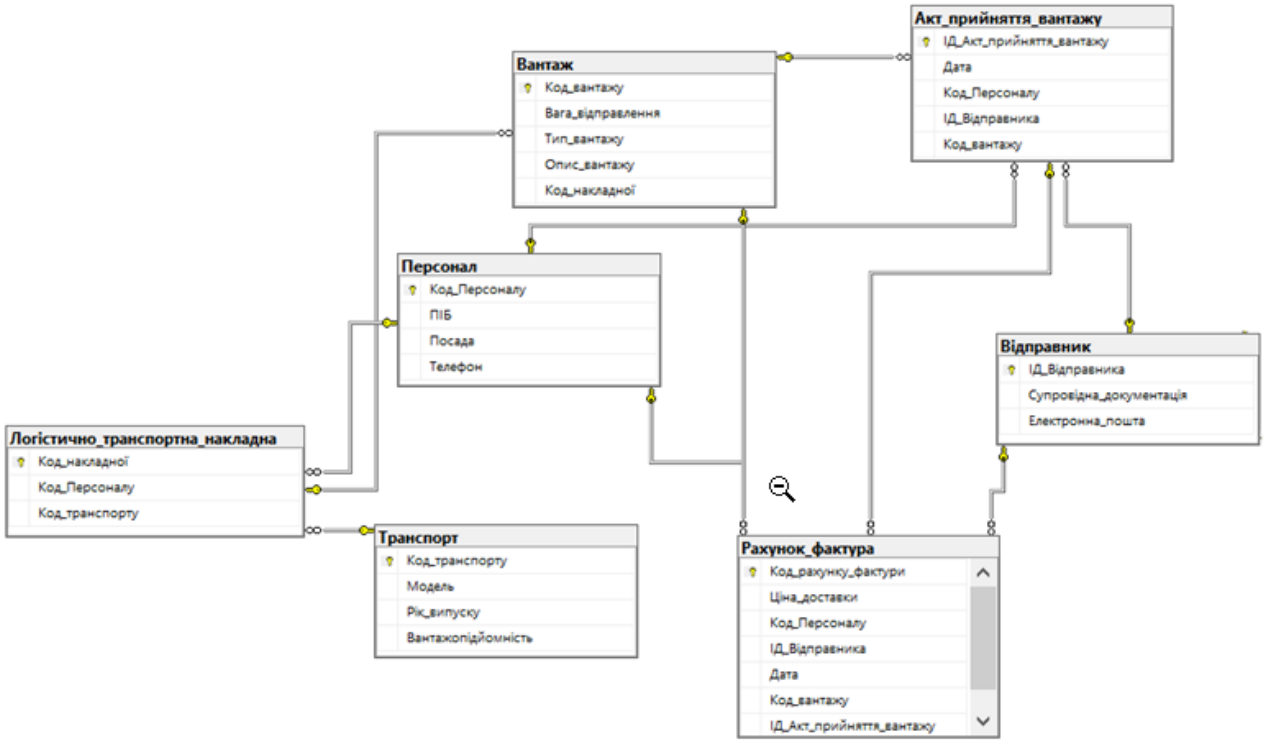


Рисунок 2.5: Вигляд схеми бази даних в SQL Server Management Studio

Далі починається робота зі створенням ІС. Тому першим кроком відбувається створення проекту:

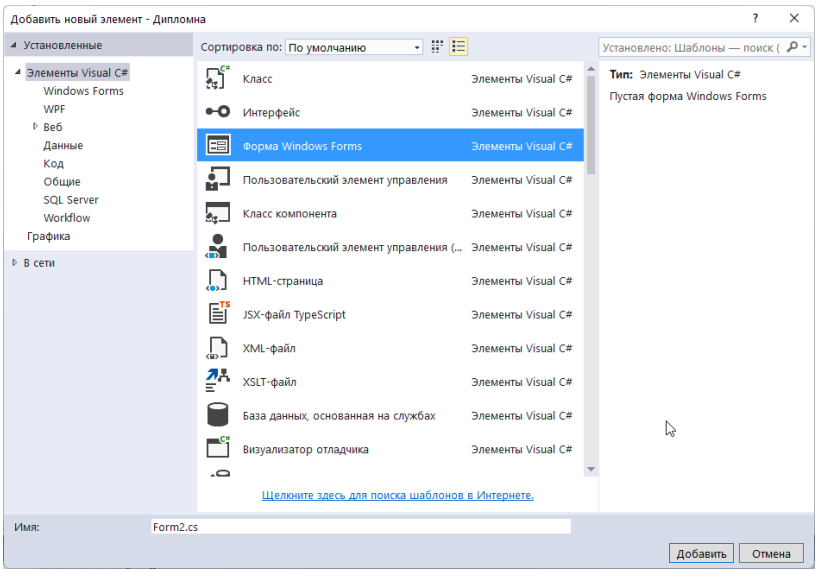


Рисунок 2.6: Вікно створення проекту

Підключення БД до SQL Server відбувається за допомогою рядка підключення:

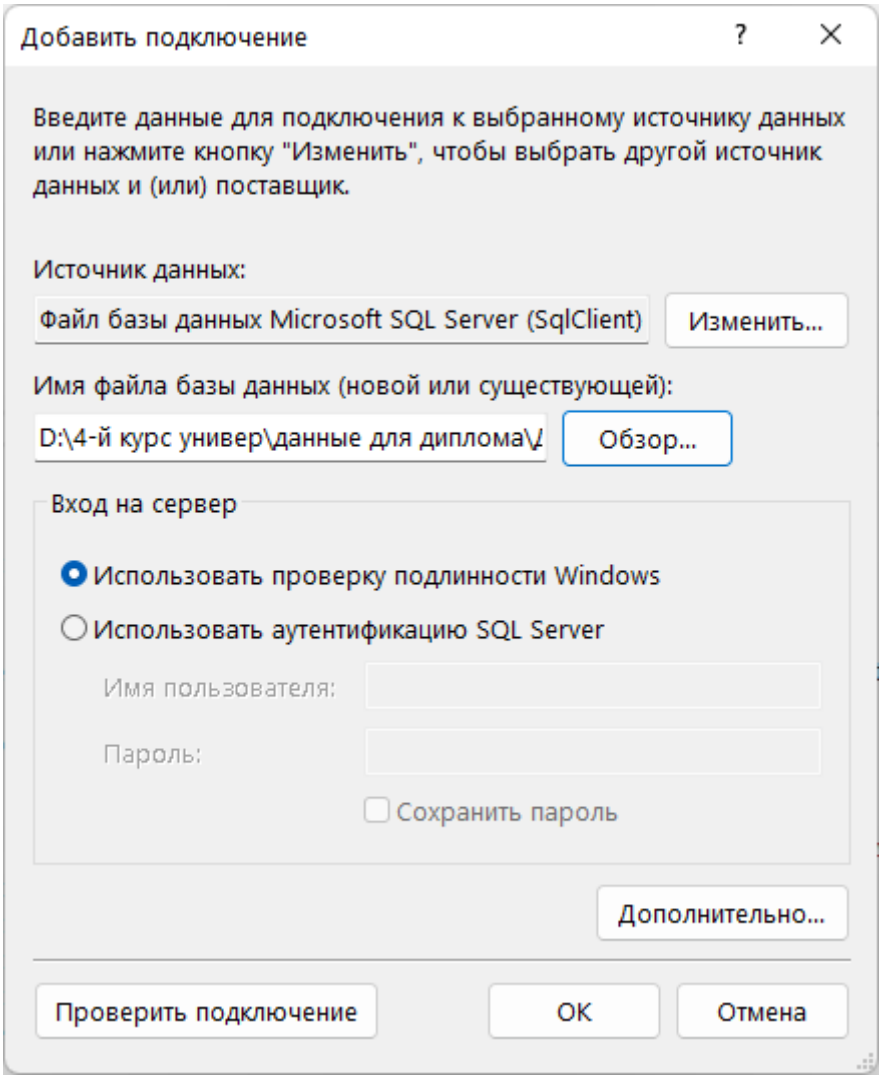


Рисунок 2.7: Підключення до сервера

Результат підключення бази даних до проекту, відображається в оглядачу серверів:

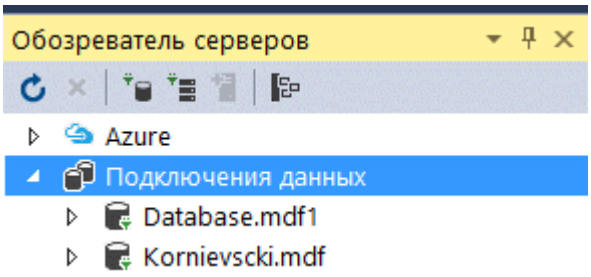


Рисунок 2.8: Оглядач серверів на якому відображена підключена БД

Щоб побачити чи правильно всі зв’язки перенесено, є можливість переглянути структуру таблиць розгорнувши підпункт «Таблиці» у оглядачі серверов:

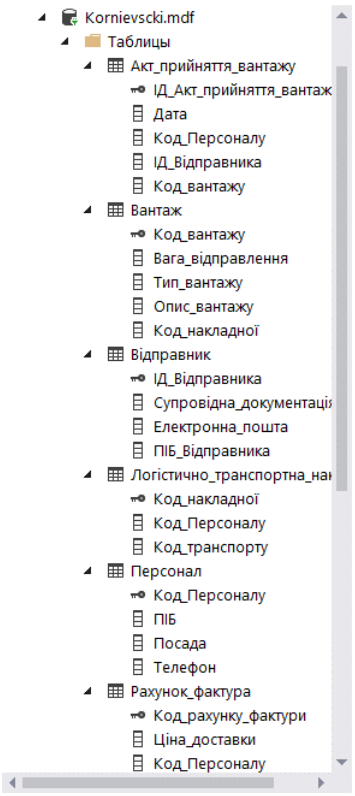


Рисунок 2.9: Структура таблиць

2.3 Формування форм введення даних

Далі маючи проект с підключеною до нього базою даних, створимо інтерфейс через який користувач буде здійснювати введення даних, запити до даних та інші роботу з ними.

Почати потрібно з форм авторизації та реєстрації з яких при наявності облікового запису можна перейти до головного меню та компонентами зв’язку між всіма вікнами існуючими формами у проекті.

На формі авторизації розмішуємо поля TextBox для зчитування введених даних і подальшої авторизації

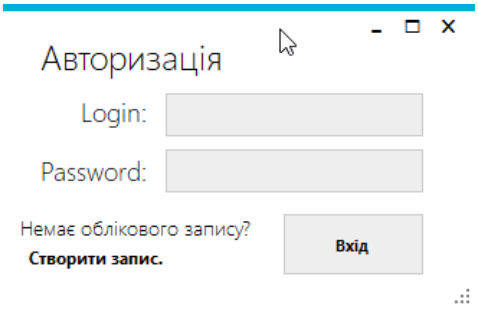


Рисунок 2.10 Вікно авторизації

Якщо користувач не має облікового запису то потрібно дати змогу створити новий обліковий запис натиснувши на кнопку «Створити запис»

На формі з реєстрацією також добавляємо TextBox для заповнення даних які потрібні для реєстрації нового облікового запису.

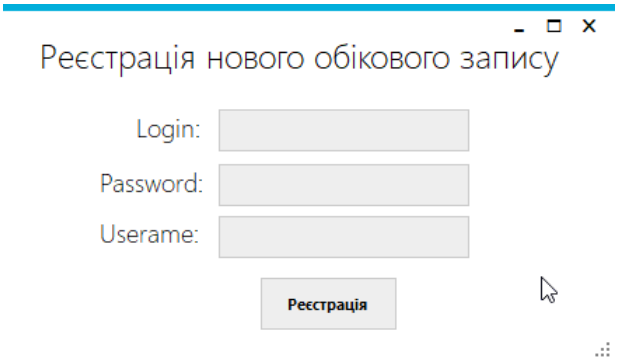


Рисунок 2.11 Вікно реєстрації

Після успішної реєстрації та авторизації ми переходимо до головної форми.

На головній формі розміщуємо елемент MenuStrip, щоб організувати управління за допомогою меню, який має в собі можливість робити пошук. Далі додаємо елемент DataGridWiev для відображення даних та ComboBox як елемент для вибору інформації яка буде відображена в DataGridWiev.

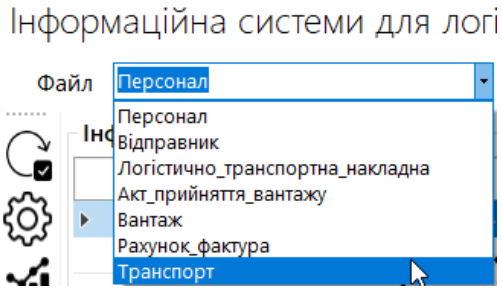


Рисунок 2.12 Вибір таблиці яка буде подалі відображена

При виборі таблиці для якої є можливість робити запити автоматично буде з’являтись необхідні для цього інструменти.

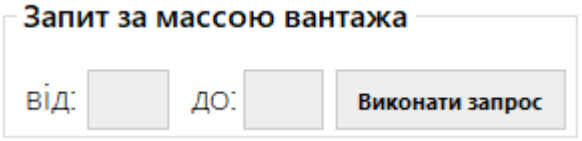


Рисунок 2.13 Приклад інтерфейсу який з’явитися при виборі таблиці для якої передбачені запити

Також користувач може зробити рахунок фактуру за умови що вибрана коректна для цього таблиця, для цього він потрібен обрати запис за яким буде сформовано документ та натиснути на кнопку «Сформувати рахунок-фактуру»

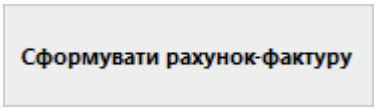


Рисунок 2.14 Кнопка для формування рахунку-фактури

Далі було добавлено та облаштовано форми в зручний та належний вигляд та добавлено функціонал за допомогою таких компонентів TextBox, ComboBox і так далі для введення або редагування даних.

Типовий вигляд вікна додавання даних після його оформлення та добавлення функціонала приймає такий вигляд:

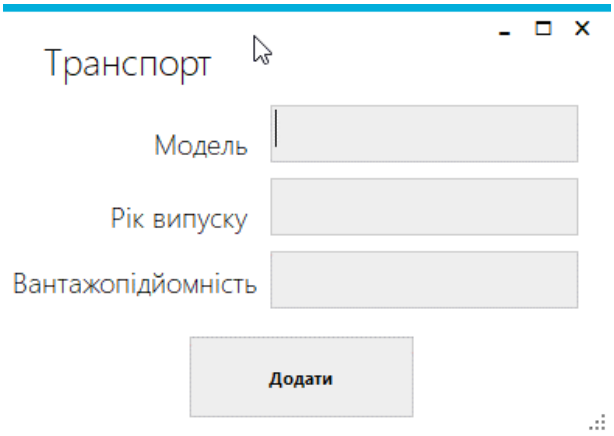


Рисунок 2.15 Вікно вводу даних для таблиці «Транспорт»

Переглянуть всі створені форми можливо в оглядачу всього рішення:

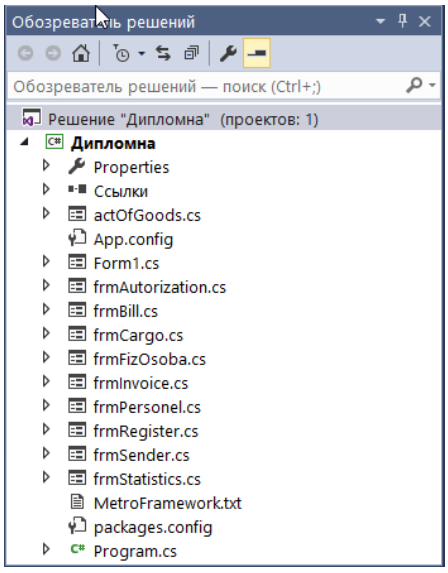


Рисунок 2.16. Основна частина вікон/форм ІС

Для деяких вікон було здійснення підстановку зі зв’язаних таблиць значень, та підставлених ключів.

```
SqlDataReader dr = comand.ExecuteReader();
var vis1 = new List<Tuple<int, string>>();
while (dr.Read())
{
    int sid = (Int32)dr["Код_Персоналу"];
    string spib = (string)dr["ПІБ"];
    vis1.Add(Tuple.Create(Convert.ToInt32(sid), spib.ToString()));
}
metroComboBox1.DataSource = vis1;
metroComboBox1.ValueMember = "Item1";
metroComboBox1.DisplayMember = "Item2";
dr.Close();
```

Рисунок 2.17. Приклад коду забезпечення даних

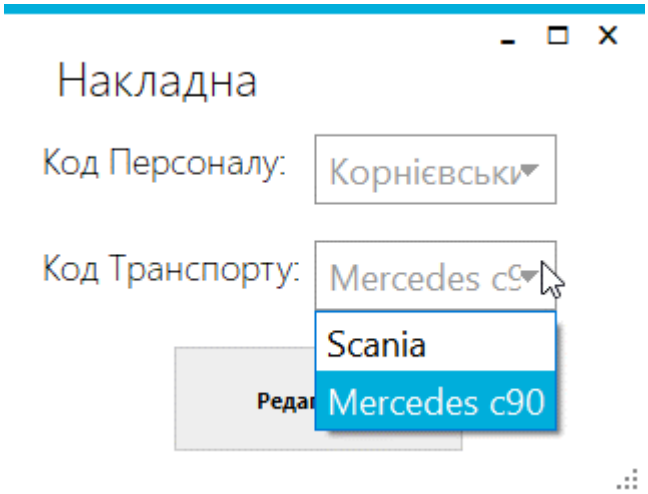


Рисунок 2.18 Результат забезпечення підстановки даних

2.4 Перевірка введення

Було також реалізовано перевірку введених даних, які реагують та показують помилку або вже існуючі дані, що дає змогу виправити введені дані, і не перериває виконання програми.

```
dataadapter.Fill(table);
if (table.Rows.Count > 0)
{
    MessageBox.Show("Такий обліковий запис вже існує! Введіть інший логін, або авторизуйтесь.");
    return true;
}
else
{
    return false;
}
```

Рисунок 2.19 Перевірка на вже існуючі дані

Також було додано перевірку на наявність введених даних:

```
if (String.IsNullOrWhiteSpace(textLogin.Text) != true && String.IsNullOrWhiteSpace(textPassword.Text) != true && String.IsNullOrWhiteSpace(textUsername.Text) != true)
{
```

Рисунок 2.20 Перевірка на вже введені дані

Також для полів там де потрібно вводити тільки цифри, стоїть обмеження на ввід символів окрім цифр

```
ссылка: 1
private void metroTextBox1_KeyPress(object sender, KeyPressEventArgs e)
{
    char number = e.KeyChar;
    if ((e.KeyChar <= 47 || e.KeyChar >= 58) && number != 8 && (e.KeyChar <= 39 || e.KeyChar >= 46) && number != 47 && number != 61)
    {
        e.Handled = true;
    }
}
```

Рисунок 2.21 Обмеження на ввід будь якого символу окрім цифр

2.5 Налаштування функцій пошуку та фільтрації.

Для реалізації пошуків та фільтрацій було створено рядок меню, а саме випадаючий список, текстове поле, та кнопку «Пошук». За допомогою цих елементів буде відбуватись пошук та його параметри.

Для того щоб почати щось шукати треба вибрати таблицю та поле в якому буде вестись пошук. Далі в текстове поле потрібно вписати що ми шукаємо і натиснути кнопку «Пошук»

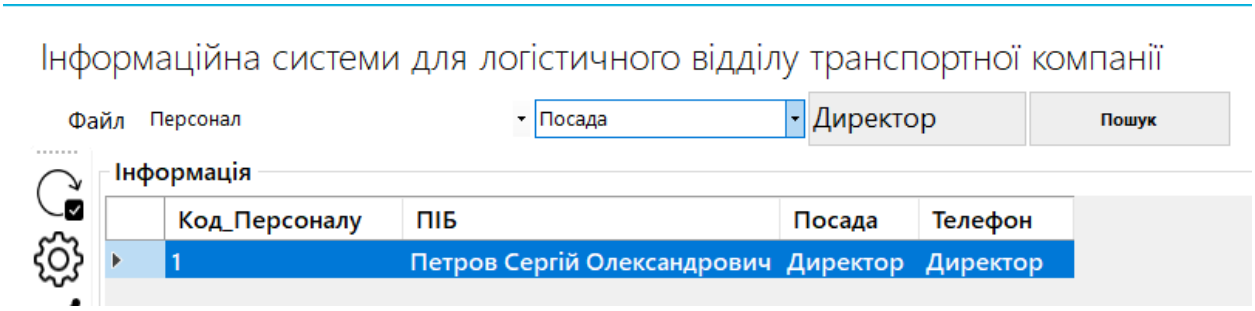


Рисунок 2.22. Результат запиту пошуку по посаді персоналу.

Завдяки цьому коду було здійснено пошук:

```
- ссылки
private void btnSearch_Click(object sender, EventArgs e)
{
    connection = new SqlConnection(connectionString);
    string table = "@";
    string data = "SELECT * FROM [";
    data += toolStripComboBox1.Text;
    data += "] WHERE ";
    data += toolStripComboBox2.Text;
    data += " LIKE @";
    data += toolStripComboBox2.Text;
    table += toolStripComboBox2.Text;
    SqlCommand comm = new SqlCommand(data, connection);

    comm.Parameters.AddWithValue(table, '%' + textVvod.Text + '%');
    SqlDataAdapter dataadapter = new SqlDataAdapter(comm);
    try
    {
        DataSet ds = new DataSet();
        dataadapter.Fill(ds, "Customer");
        dataGridView1.DataSource = ds;
        dataGridView1.DataMember = "Customer";
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(), MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
    finally
    {
        connection.Close();
        textVvod.Text = null;

    }
}
```

Рисунок 2.23 Код пошуку

2.6 Реалізація автоматизованого формування документів та експорту даних.

Для реалізації документів прийшлось встановити не існуючий в даному Visual Studio пакет NuGet, а саме ці що відображені на рисунку.

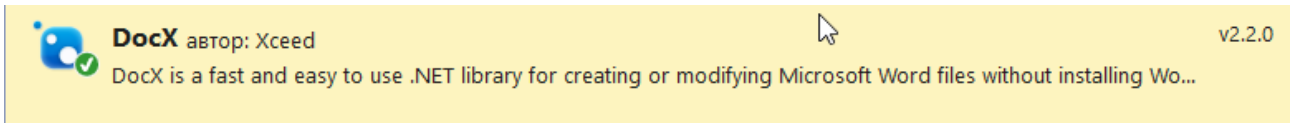


Рисунок 2.24. Потрібний пакет для формування документів

Далі було вибрано строку з таблиці, та побудовано документ на основі обраної строки з таблиці, при цьому, документ автоматично сформується в папці «Documents», що знаходиться в кореневій папці проекту.

Логістично-транспортний відділ
+38044228322
ФОП Мудрук, м Житомир. вул Київська 1
IBAN UA43242425564565464 ЄДРПОУ 344234

Номер документу: 9
Ціна до сплати: 500
Обробив замовлення: Корнієвський Андрій
Відправник: ОАО Атлант
Код вантажу: 6

Дата: 30.05.2022
_____(підпис)

Рисунок 2.25. Приклад сформованого в середовищі World документу.

Код завдяки якому здійснюється формування документів наведено на рис. 2.26

```
string filePath = Application.StartupPath + "/Documents/ID" + a.ToString() + ".docx";
DocX doc = DocX.Create(filePath);


string title = "Логістично-транспортний відділ" + Environment.NewLine + "+38044228322" + Environment.NewLine + "ФОП Мудрук, м Житомир. вул Київська 1" + Environment.NewLine + "IBAN UA43242425564565464 ЄДРПОУ 344234";
Formatting titleFormat = new Formatting();
//Specify font family
titleFormat.FontFamily = new Xceed.Document.NET.Font("Arial");
//Specify font size
titleFormat.Size = 12;
Paragraph paragraphTitle = doc.InsertParagraph(title, false, titleFormat);
paragraphTitle.Alignment = Alignment.right;
string text = Environment.NewLine + "Номер документу: " + a + Environment.NewLine + "Ціна до сплати: " + price + Environment.NewLine + "Обробив замовлення: " + personel + Environment.NewLine + "Відправник: " + vidpranyk + Environment.NewLine
Formatting textParagraphFormat = new Formatting();
textParagraphFormat.FontFamily = new Xceed.Document.NET.Font("Arial");
//Specify font size
textParagraphFormat.Size = 14;
Paragraph paragraphtext = doc.InsertParagraph(text, false, textParagraphFormat);
paragraphtext.Alignment = Alignment.left;
string parTime = Environment.NewLine + Environment.NewLine + "Дата: " + time;
Formatting footer = new Formatting();
footer.Size = 14;
footer.FontFamily = new Xceed.Document.NET.Font("Arial");
Paragraph footertext = doc.InsertParagraph(parTime, false, footer);
footertext.Alignment = Alignment.left;
string pidpis = "_____(підпис)";
Paragraph footepidpis = doc.InsertParagraph(pidpis, false, footer);
footepidpis.Alignment = Alignment.right;
doc.Save(); // save changes to file
```

Рисунок 2.26. Код формування документації

Також можна створити звіт, який в свою чергу, можна перенести в середовище Excel та оформити у вигляді стандартного документу. Це реалізовано через кнопку під назвою «Експортувати дані» код якої можна побачити на рисунку 2.27

```
private void ToExcel(DataGridView dGV, string filename)
{
    string stOutput = "";
    string stHeaders = "";
    for (int j = 0; j < dGV.Columns.Count; j++)
        stHeaders = stHeaders.ToString() + Convert.ToString(dGV.Columns[j].HeaderText) + "\t";
    stOutput += stHeaders + "\r\n";

    for (int i = 0; i < dGV.RowCount - 1; i++)
    {
        string stline = "";
        for (int j = 0; j < dGV.Rows[i].Cells.Count; j++)
            stline = stline.ToString() + Convert.ToString(dGV.Rows[i].Cells[j].Value) + "\t";
        stOutput += stline + "\r\n";
    }

    Encoding utf16 = Encoding.GetEncoding(1251);
    byte[] output = utf16.GetBytes(stOutput);
    FileStream fs = new FileStream(filename, FileMode.Create);
    BinaryWriter bw = new BinaryWriter(fs);
    bw.Write(output, 0, output.Length);
    bw.Flush();
    bw.Close();
    fs.Close();
}
private void metroButton1_Click(object sender, EventArgs e)
{
    SaveFileDialog sfd = new SaveFileDialog();
    sfd.Filter = "Excel Document (*.xls)|*.xls";
    if (sfd.ShowDialog() == DialogResult.OK)
    {
        ToExcel(dataGridView1, sfd.FileName);
    }
}
```

Рисунок 2.27 Код експорту даних в Excel

Вигляд самого звіту у MS Excel:

	A	B	C	D	E
1	Код_тран	Модель	Рік_випус	Вантажопідйомність	
2	1	Scania	2013	40	
3	2	Mercedes	2018	35	

Рисунок 2.28 Сформований звіт

2.7 Реалізація автоматичного обрахунку статистики

За допомогою запиту саме в проекті було створено форму яка автоматично без яких ніби то дій, обраховує статистику виконуючи sql-запити

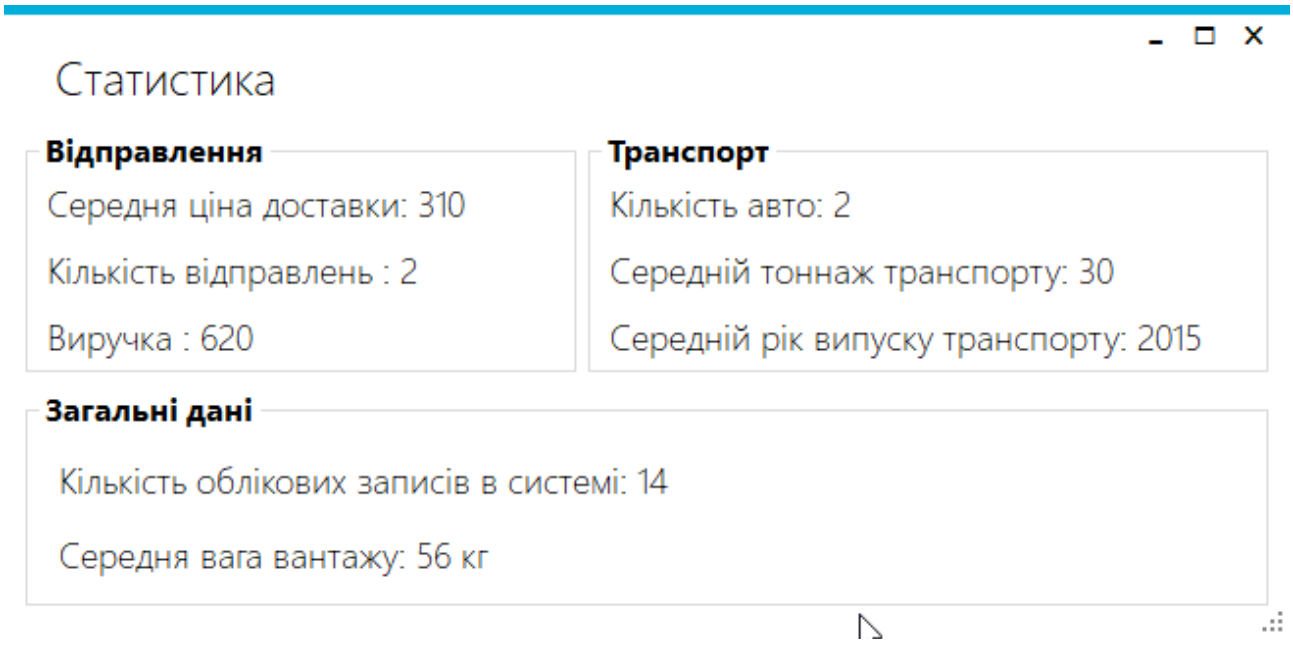


Рисунок 2.29. Статистика

Запити за допомогою якого робиться статистика виглядить так:

```
ссылка: 1
private void frmStatistics_Load(object sender, EventArgs e)
{
    connection = new SqlConnection(connectionString);
    SqlCommand comm = new SqlCommand("SELECT AVG(Ціна_доставки) FROM [Рахунок_фактура] ", connection);
    connection.Open();
    object result = comm.ExecuteScalar();
    txtAVG.Text = "Середня ціна доставки: " + Convert.ToString(result);
    SqlCommand command = new SqlCommand("SELECT COUNT(Код_рахунку_фактури) FROM [Рахунок_фактура] ", connection);
     result = command.ExecuteScalar();
    txtCountOrders.Text = "Кількість відправлень : " + Convert.ToString(result);
    SqlCommand summa = new SqlCommand("SELECT SUM(Ціна_доставки) FROM [Рахунок_фактура] ", connection);
    result = summa.ExecuteScalar();
    txtSum.Text = "Виручка : " + Convert.ToString(result);
    SqlCommand countavto = new SqlCommand("SELECT COUNT(Код_транспорту) FROM [Транспорт] ", connection);
    result = countavto.ExecuteScalar();
    txtCountVehicle.Text = "Кількість авто: " + Convert.ToString(result);
    SqlCommand avgtons = new SqlCommand("SELECT AVG(Вантажопідйомність) FROM [Транспорт] ", connection);
    result = avgtons.ExecuteScalar();
    txtAVGtons.Text = "Середній тоннаж транспорту: " + Convert.ToString(result);
    SqlCommand avgyears = new SqlCommand("SELECT AVG(Рік_випуску) FROM [Транспорт] ", connection);
    result = avgyears.ExecuteScalar();
    txtAVGyear.Text = "Середній рік випуску транспорту: " + Convert.ToString(result);
    connectionUser = new SqlConnection(connectionStringUser);
    SqlCommand countUser = new SqlCommand("SELECT COUNT(Id) FROM [users] ", connectionUser);
    connectionUser.Open();
    result = countUser.ExecuteScalar();
    txtCountUsers.Text = "Кількість облікових записів в системі: " + Convert.ToString(result);
    connectionUser.Close();
    SqlCommand avgmass = new SqlCommand("SELECT AVG(Вага_відправлення) FROM [Вантаж] ", connection);
    result = avgmass.ExecuteScalar();
    txtAVGmass.Text = "Середня вага вантажу: " + Convert.ToString(result) + " кг";


    connection.Close();
}
```

Рисунок 2.30 Код формування статистики

2.8 Реалізація авторизації та реєстрації працівника в інформаційну систему

Останнім кроком було створення таблиці «users», яка заповнюється адміністратором або розробником програмного забезпечення.

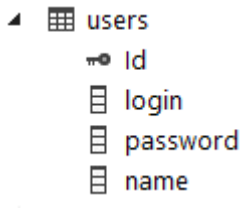


Рисунок 2.31. Таблиця «users»

Та буле створена форма для авторизації самого користувача.

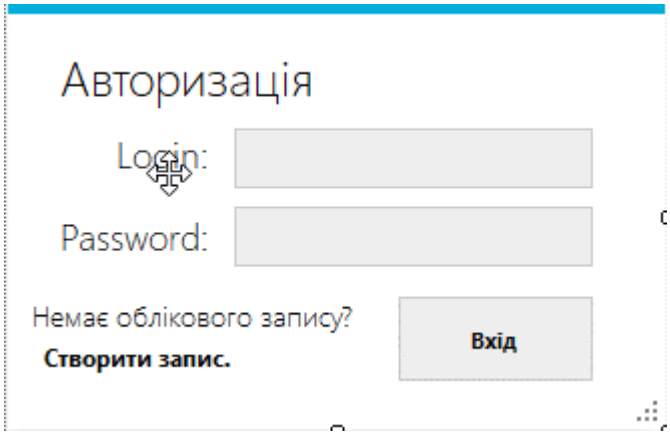


Рисунок 2.32. Форма авторизації

Функціонал авторизації було здійснено за допомогою ось такого кода, який робив запит до таблиці, і при отриманні тільки одного запита який збігається із паролем та ім’ям користувача давало доступ надалі:

```
private void BtnAutorization_Click(object sender, EventArgs e)
        {
            string userlogin = textLogin.Text;
            string userpassword = textPassword.Text;
            DataTable table = new DataTable();
            connection = new SqlConnection(connectionString);
            SqlCommand comm = new SqlCommand("SELECT * FROM [users] WHERE login =@userLogin
AND password =@userPassword", connection);
            comm.Parameters.AddWithValue("@userLogin", userlogin);
            comm.Parameters.AddWithValue("@userPassword", userpassword);
            SqlDataAdapter dataadapter = new SqlDataAdapter(comm);
            dataadapter.Fill(table);
```

```
comm.Connection.Close();

if (table.Rows.Count > 0)
{
    frmMain mainform = new frmMain();
    mainform.Show();
    this.Hide();
}
else MessageBox.Show("Данні введено некорректно!");
```

Якщо користувач не має облікового запису у системі, то його можливо создати натиснувши на поле «Створити запис» після чого з'явится вікно реєстрації (див. рис.2.33)

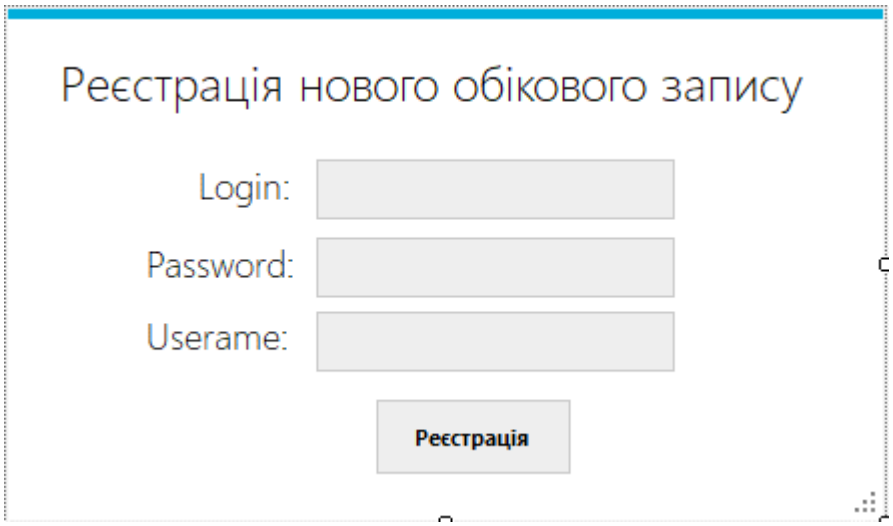


Рисунок 2.33 Вікно реєстрації

2.9. Інструкція користувача

При відкриті програмного продукту з’являється меню авторизації на якому ми можемо авторизуватись або перейти на форму реєстрації облікового запису якщо у користувача немає аккаунту в системі:

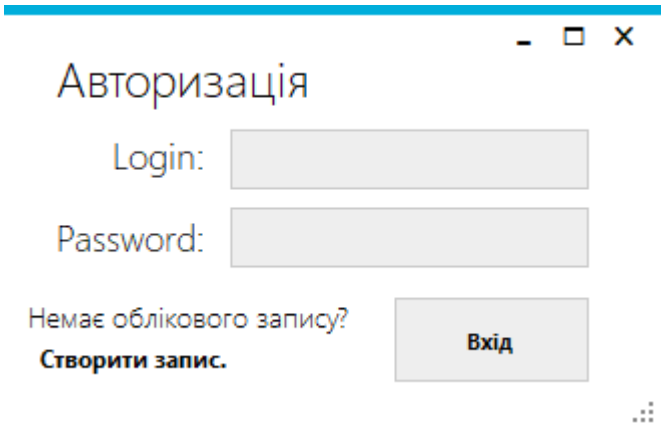


Рисунок 2.34. Форма авторизації

При переході на форму реєстрації користувачу потрібно буде ввести свої дані на створити обліковий запис:

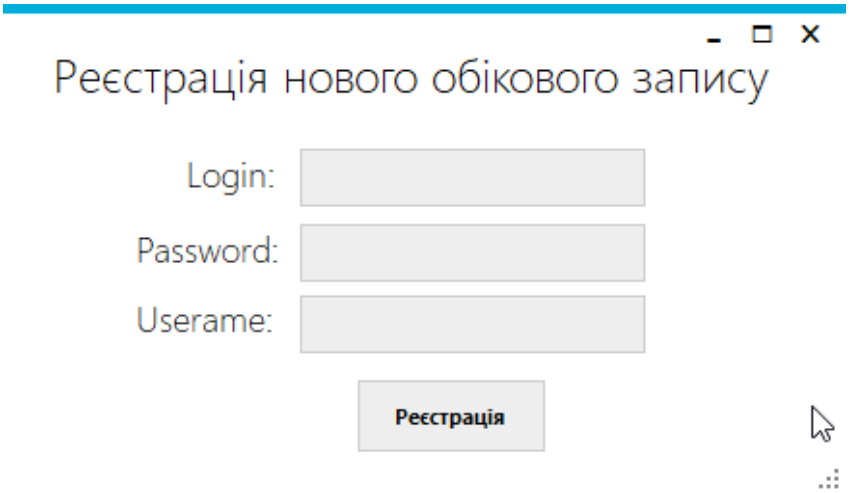


Рисунок 2.35. Форма реєстрації

Після успішної авторизації відкривається головне меню, що забезпечує навігацію по формам, необхідно лиш обрати функцію, яку необхідно виконати: внесення даних, редагування даних, видалення даних, налаштування таблиць, оновлення даних, пошуку даних та формування файлу Експорту даних:

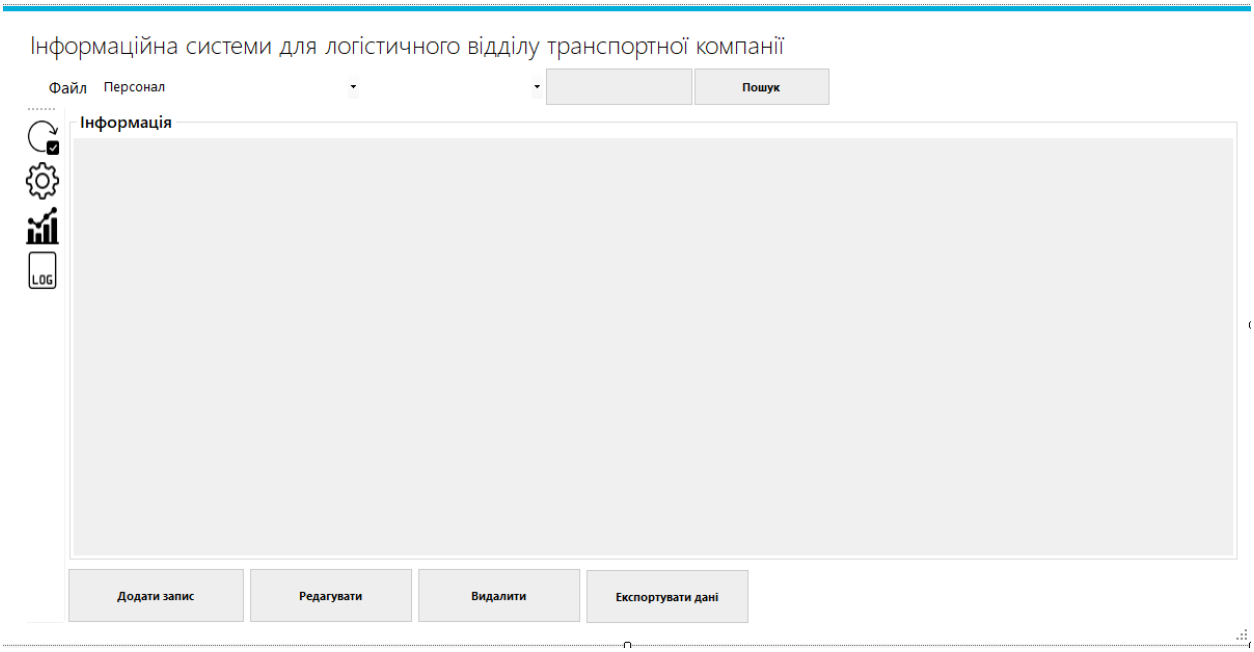


Рисунок 2.36 Головне меню

Далі користувач має змогу додати запис до БД, для цього потрібно обрати потрібну таблицю в виїжджаючому списку та натиснути на кнопку «Додати»:

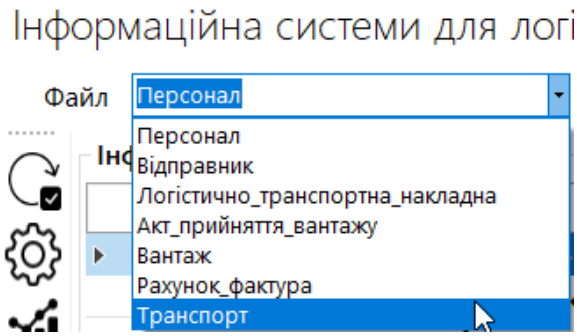


Рис 2.37 Виїжджаючий список для обрання таблиці

Після цього відкриється форма з текстовими полями на виїжджаючими списками які потрібно заповнити для додавання даних до таблиці:

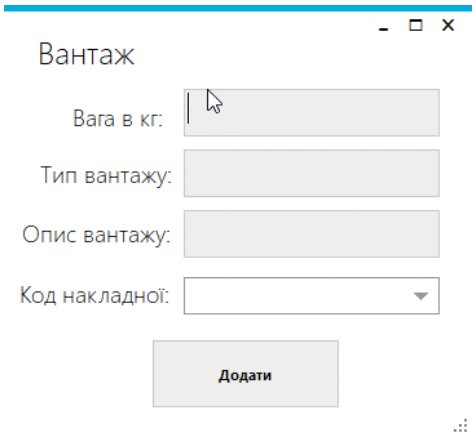


Рис 2.38 Форма додавання даних до таблиці

Також користувач має змогу редагувати обрані записи які він може вибрати в таблиці де відображуються дані з таблиць:

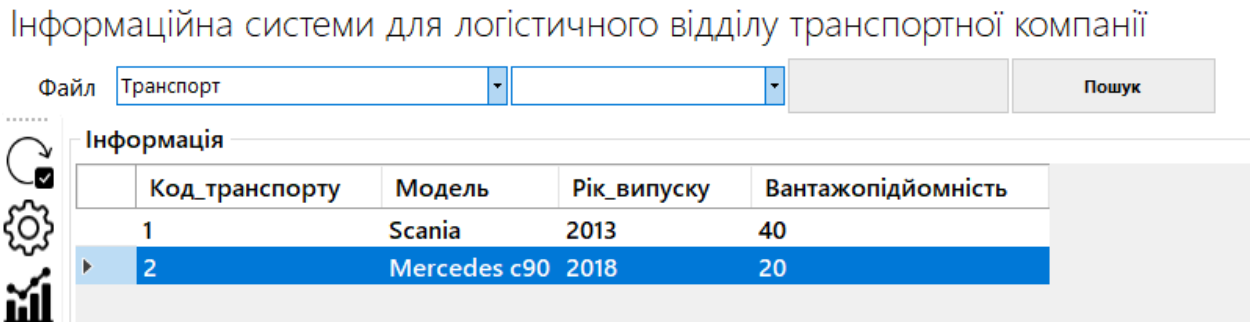
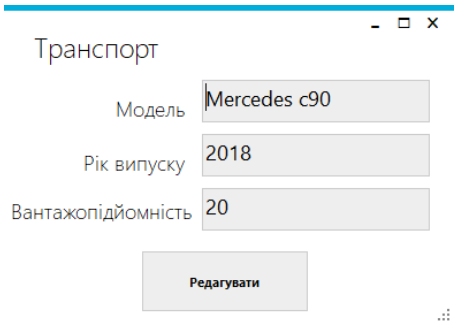


Рисунок 2.39 Обирання запису з таблиці для редагування

Після того як користувач обрав потрібний рядок для редагування даних йому потрібно натиснути кнопку «Редагувати» після чого буде відкрито форма редагування даних в поля якої автоматично буде загружено дані для подальшого редагування:



Рис*унок 2*.40 Форма редагування даних

Кнопка видалення рядку дає змогу видаляти не потрібні, або не актуальні рядки, для цього потрібно обрати запис в таблиці та натиснути кнопку «Видалити»

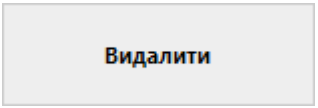


Рис. 2.41 Кнопка видалення рядку

Кнопка оновлення потрібна для оновлення.

Рис. 2.42 Кнопка збереження нового або оновленого рядка

В головному меню можливо здійснювати пошук за допомогою випадаючих списків. Потрібно вибрати, що шукати і де шукати і при натисканні на кнопку здійснюється пошук і відображення даних у таблицю.

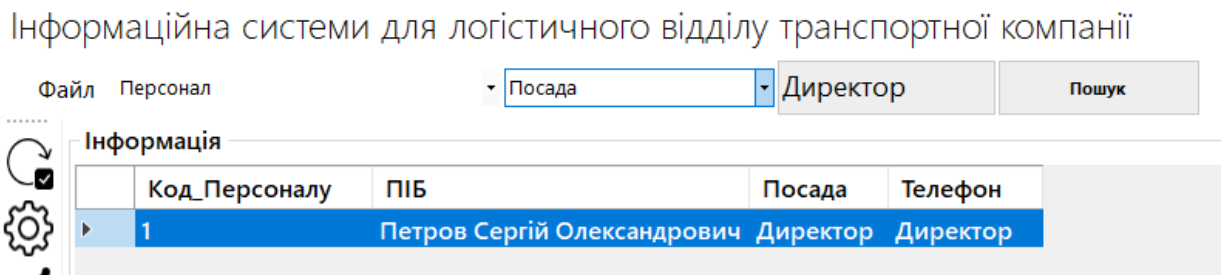


Рис. 2.43 Результат пошуку через випадаючий список

Для формування звітних таблиць та їх експорту потрібно вибрати потрібну таблицю через випадаючий список та натиснути кнопку «Експортувати дані» а потім вибрати путь збереження згенерованого файлу:

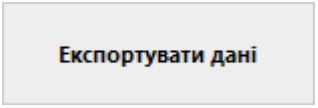


Рисунок 2.44 Кнопка Експорту даних з обраної таблиці

Результат Експорту даних:

	A	B	C	D	E
1	Код_тран	Модель	Рік_випус	Вантажопідйомність	
2	1	Scania	2013	40	
3	2	Mercedes	2018	35	

Рис. 2.45 Згенерований файл звіту

Якщо користувачу потрібно буде змінити налаштування таблиці, він, в свою чергу може натиснути на кнопку «Налаштування» та у вікні яке відкриється після цього – поміняти налаштування.

Рисунок 2.46 Кнопка «Налаштування»

Саме вікно налаштувань має такий вигляд:

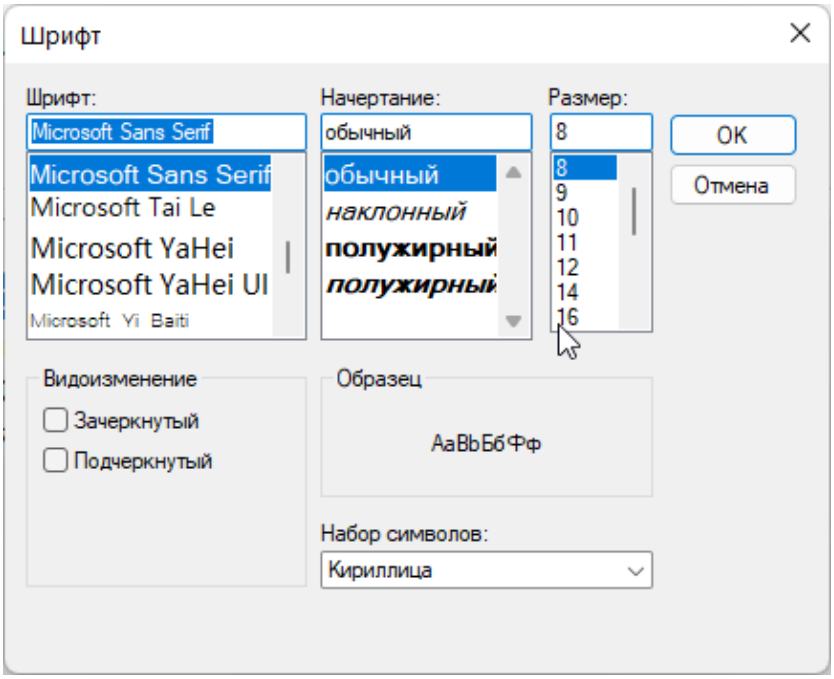


Рисунок 2.47 Вікно налаштувань

Якщо користувач хоче здійснити запити до бази даних, то він потрібен в виїжджаючому меню вибрати таблицю для якої можливість запиту буде доступна.

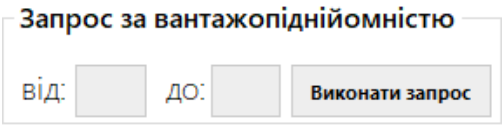


Рисунок 2.48 Приклад інтерфейсу який з’явитися при виборі таблиці для якої передбачені запити

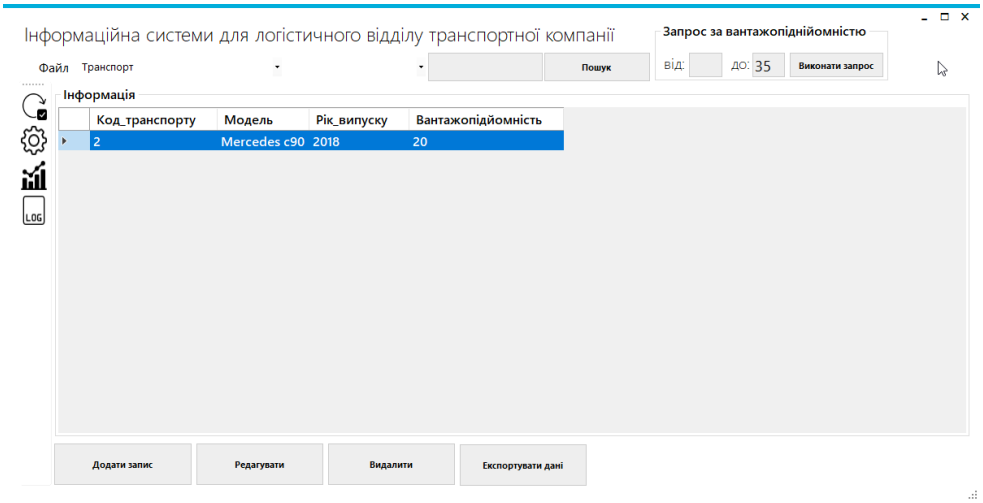


Рисунок 2.49 Приклад виконання запиту

Також користувач має змогу сформувати рахунок-фактуру, для цього потрібно вибрати потрібний запис у таблиці «рахунок-фактура» і натиснувши кнопку «сформувати рахунок-фактуру» документ сформується в папці «Documents», яка в свою чергу знаходиться в кореневій папці проекту.

Логістично-транспортний відділ
+38044228322
ФОП Мудрук, м Житомир. вул Київська 1
IBAN UA43242425564565464 ЄДРПОУ 344234

Номер документу: 9
Ціна до сплати: 500грн
Обробив замовлення: Корнієвський Андрій
Відправник: ОАО «Атлант»
Код вантажу: 9 - Бандероль

Дата: 30.05.2022

_____(підпис)

Рисунок 2.50. Приклад сформованого в середовищі World документу.

2.10. Технічне та системне забезпечення розробки

2.10.1. Обґрунтування вибору технічних засобів

Для розробки інформаційної системи, було використано велику кількість технологій та програмних засобів. Було вибрано із переліку існуючого інструментарію та використано такі програмні засоби для створення функціональної моделі:

- CASE-засіб AllFusion Process Modeler — це модуль програмного забезпечення AllFusion, який має ціль в створенні функціональних моделей.

Було вибрані із переліку існуючого інструментарію, та використано такі програмні засоби для створення бази даних:

- CASE-засіб Erwin Data Modeler — це модуль програмного забезпечення AllFusion, який має ціль в проектуванні бази даних, та їх генерація в сервера.
- MS SQL Server 2008 це є сервер, від корпорації Microsoft, який розвертається, та забезпечує зберігання даних та їх надавання.

Було вибрані із переліку існуючого інструментарію та технологій, та використано такий програмний засіб та такі технології для створення самого програмного забезпечення, форми інакше кажучи вікна:

- MS Visual Studio 2015 –професійний інструмент розробки програмного забезпечення для Windows. [7]
- C# — це мова програмування, призначена для розробки найрізноманітніших додатків. Мова C #, розроблена компанією Microsoft. [11]

2.10.2. Визначення топології комп’ютерної мережі

В проекті використовується звернення до MS SQL Server який є локальним так як, його функція це є зберігання даних, і інформація що містить база даних, не повинна бути отримана сторонньою особою, яка не має потрібного допуску, для того щоб не здійснився витік даних.

2.10.3. Обґрунтування вибору ОС

Розробка здійснювалась під операційну систему Windows, а саме Windows 11.

Дана операційна система є найновітнішою системою. На разі вона є менш стабільнішою ніж Windows 10, однак як на момент створення проекту, операційна система отримала велику кількість переправлень багів, та отримала оптимізацію, що в свою чергу, знизила мінімальні вимоги до ПК

Вибрано вона була по причині того що, на думку компанії, треба йти з ногою у часі, та, відповідно розроблювати програмне забезпечення для найновітніших систем.

2.10.4. Заходи захисту від несанкціонованого доступу до системи

Самою основною метою захисту є захист від зливу даних, їх видалення та/або зіпсування даних. Захист від несанкціонованого доступу до інформаційної системи здійснюється за допомогою авторизації.

ВИСНОВКИ

Під час виконання дипломної роботи була проведена робота з вивчення технологій розробки Windows-додатків з використанням технологій C#, Sql Server, Erwin Data Modeler. За підсумками виконання роботи було розроблено інформаційну систему для логістичного відділу транспортної компанії.

Реалізовано функції формування рахунку-фактури та експорту даних, що скоротить робочий час роботи персоналу, та значно спростить облік та формування документів для логістичного відділу.

Створена система дозволяє користувачу використовувати пошук по любим критеріям, що в свою чергу спростить користування програмним продуктом, коли він буде містити в собі багато записів. При створенні системи були враховані всі плюси та мінуси систем-аналогів, вибраних для порівняння.

Всі функції відповідають вимогам, тому створене програмне забезпечення готове до повної функціональної роботі, тобто система для впровадження, яка може бути використана бізнесом для підвищення продуктивності праці персоналу.

В даній кваліфікаційній роботі було реалізовано:

- Формування рахунку-фактури
- Експорт даних в формі файлу Excel
- Пошук даних

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. М’якшило О.М. CASE-технології у проектуванні інформаційних систем: [електронний ресурс] навч. посібник для студентів вищих навчальних закладів / О.М. М’якшило, Л.Г. Загоровська,– К.: НУХТ, 2017. – 190 с.

2. М’якшило О.М. Моделювання баз даних засобами CASE – технології ERWin: конспект лекцій/ О.М. М’якшило – К.:НУХТ, 2007 – 60 с.

3. Проектування інформаційних систем: лабораторний практикум для студ. освіт. ступ. "Бакалавр" спец. 122 "Комп'ютерні науки " ден. і заоч. форм навч. Частина 2 "Проектування клієнтського додатку" / уклад. : О. М. М'якшило, О. В. Харкянен; Нац. ун-т харч. технол. - Київ : НУХТ, 2017. - 33 с.

4. Методичні рекомендації до виконання кваліфікаційної роботи на здобуття освітнього ступеня «Бакалавр» спеціальності 122 «Комп’ютерні науки» освітньо-професійної програми «Комп’ютерні науки» денної та заочної форм навчання / уклад. : Л. Г. Загоровська, О. М. М’якшило, М. П. Костіков. – К. : НУХТ, 2020. – 30 с.

5. Управління ІТ-проектами: методичні рекомендації до виконання курсового проекту для студентів освітнього ступеня «Бакалавр» спеціальності 122 «Комп’ютерні науки» денної та заочної форм навчання. Уклад: М. В. Гладка, О. А. Хлобистова: К. НУХТ, 2014.– 91 с.

6. Microsoft SQL Server 2017 [Електронний ресурс] – Режим доступу до ресурсу: https://www.microsoft.com/ru-ru/sql-server/sql-server-2017-editions

7. Microsoft Visual Studio [Електронний ресурс] – Режим доступу до ресурсу: https://visualstudio.microsoft.com/ru/

8. 1С:Торговля и Склад 7.7 [Електронний ресурс] – Режим доступу до ресурсу: https://1c.ru/rus/products/1c/predpr/torg77.htm

9. Brokkeeper [Електронний ресурс] – Режим доступу до ресурсу: https://bookkeeper.kiev.ua/pro-proekt/

10. Ананас [Електронний ресурс] – Режим доступу до ресурсу: https://ananas.su/wiki/%D0%97%D0%B0%D0%B3%D0%BB%D0%B0%D0%B2%

D0%BD%D0%B0%D1%8F_%D1%81%D1%82%D1%80%D0%B0%D0%BD%D0%B8%D1%86%D0%B0

11. C# [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/C_Sharp

ДОДАТКИ

ДОДАТОК А - ФУНКЦІОНАЛЬНІ МОДЕЛІ

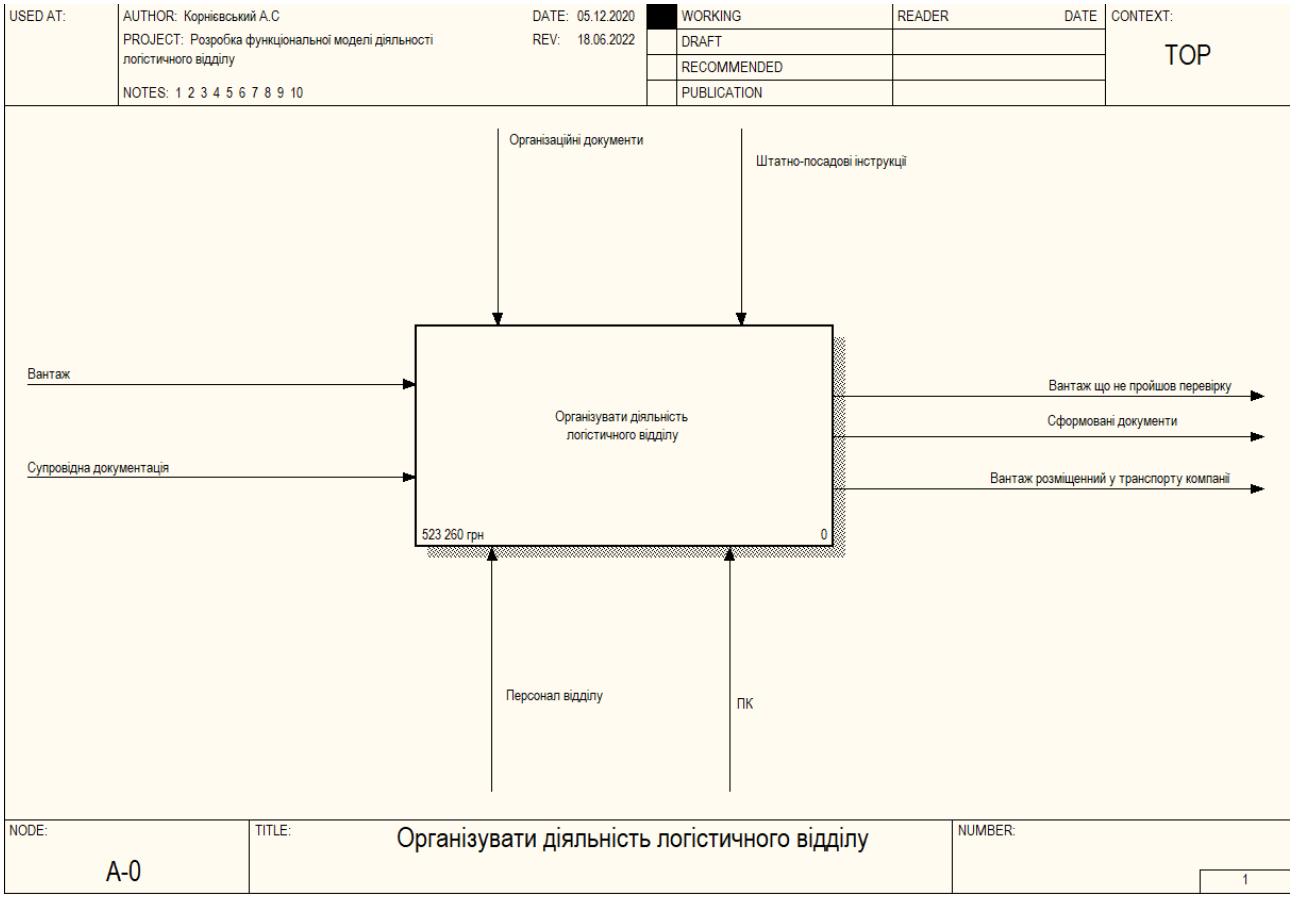


Рисунок А.1 - Контекстна діаграма AS-IS функціональної моделі

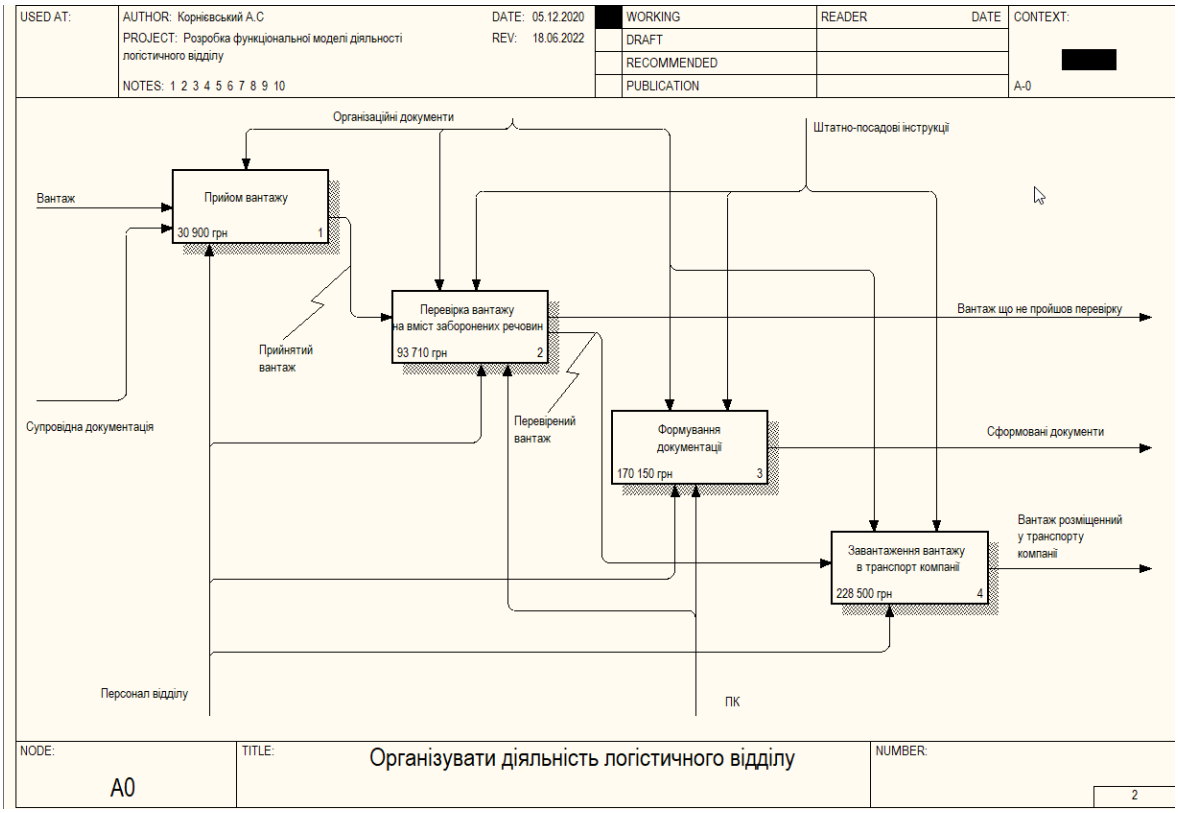


Рисунок А.2 - Діаграма декомпозиції на першому рівні

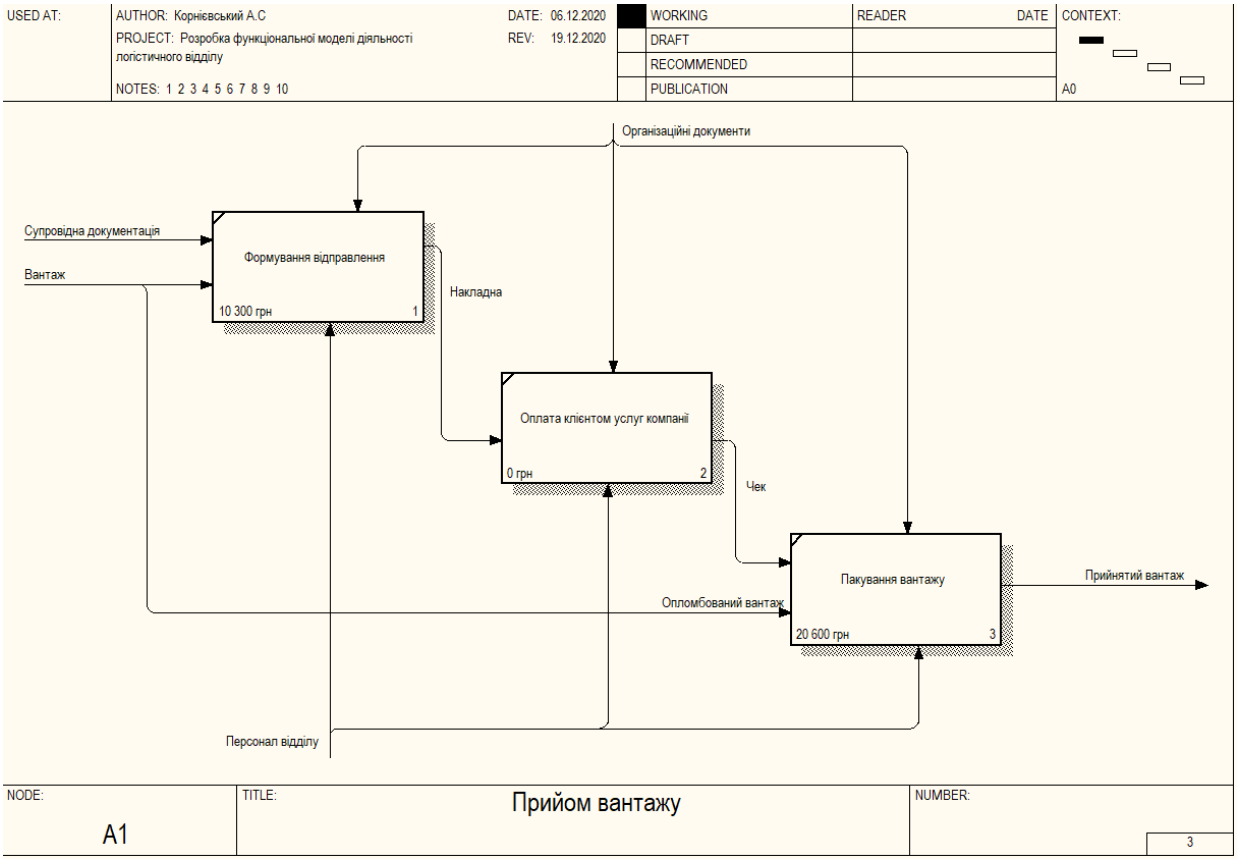


Рисунок А.3 - Діаграма декомпозиції блоку «Прийом вантажу»

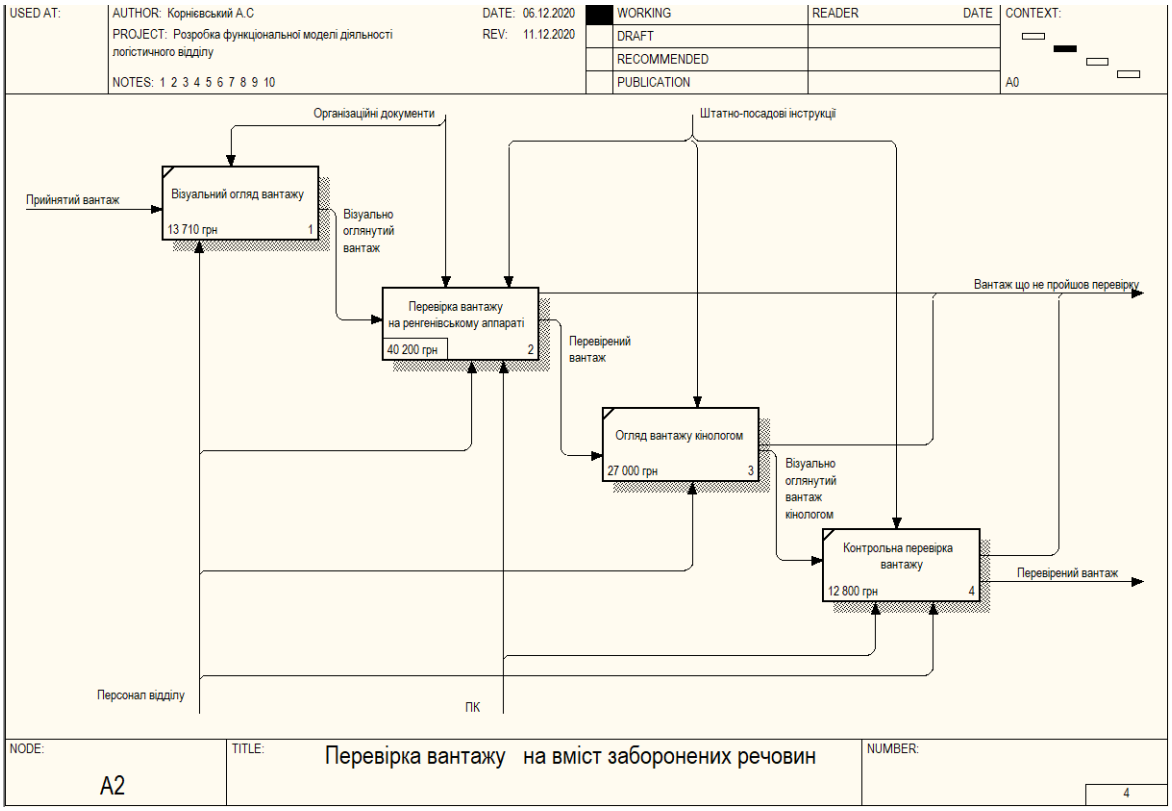


Рисунок А.4 - Діаграма декомпозиції блоку «Перевірка вантажу на вміст заборонених речовин»

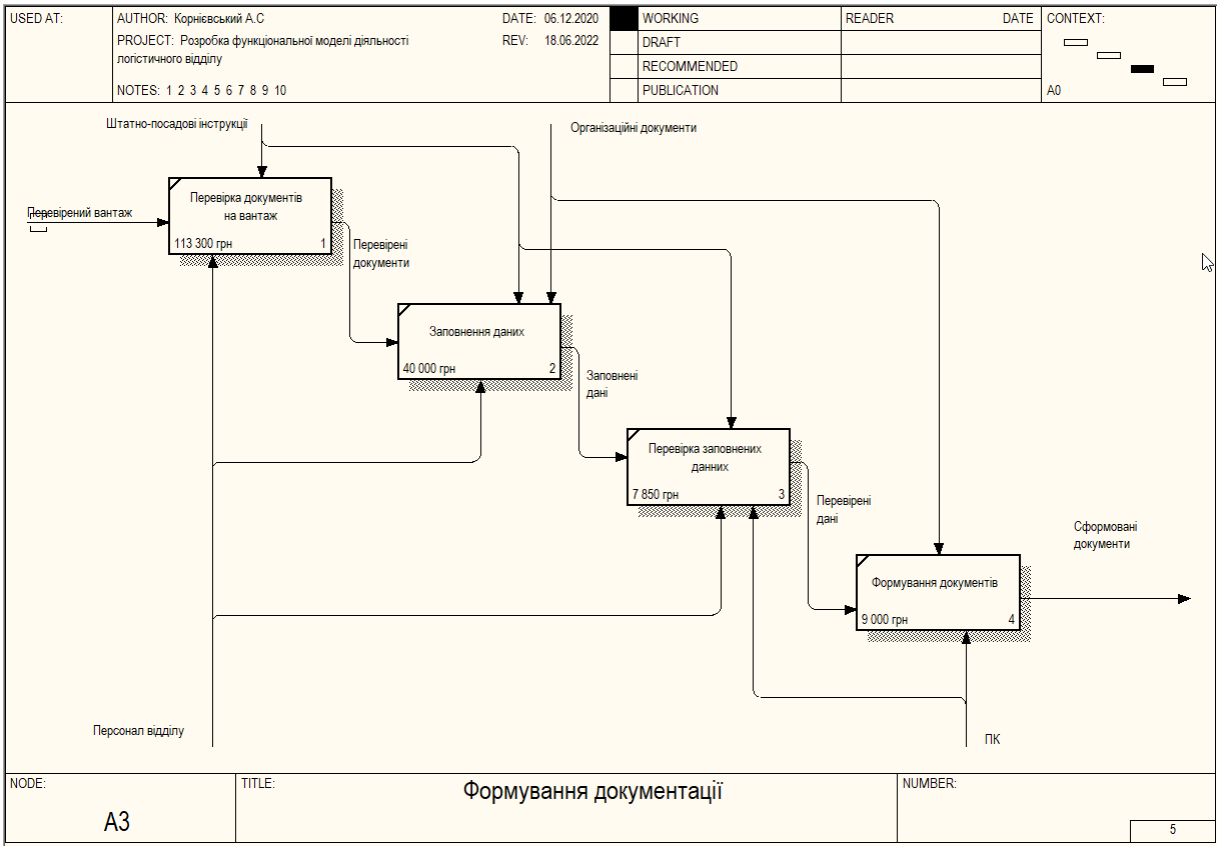


Рисунок А.5 - Діаграма декомпозиції блоку «Формування документації»

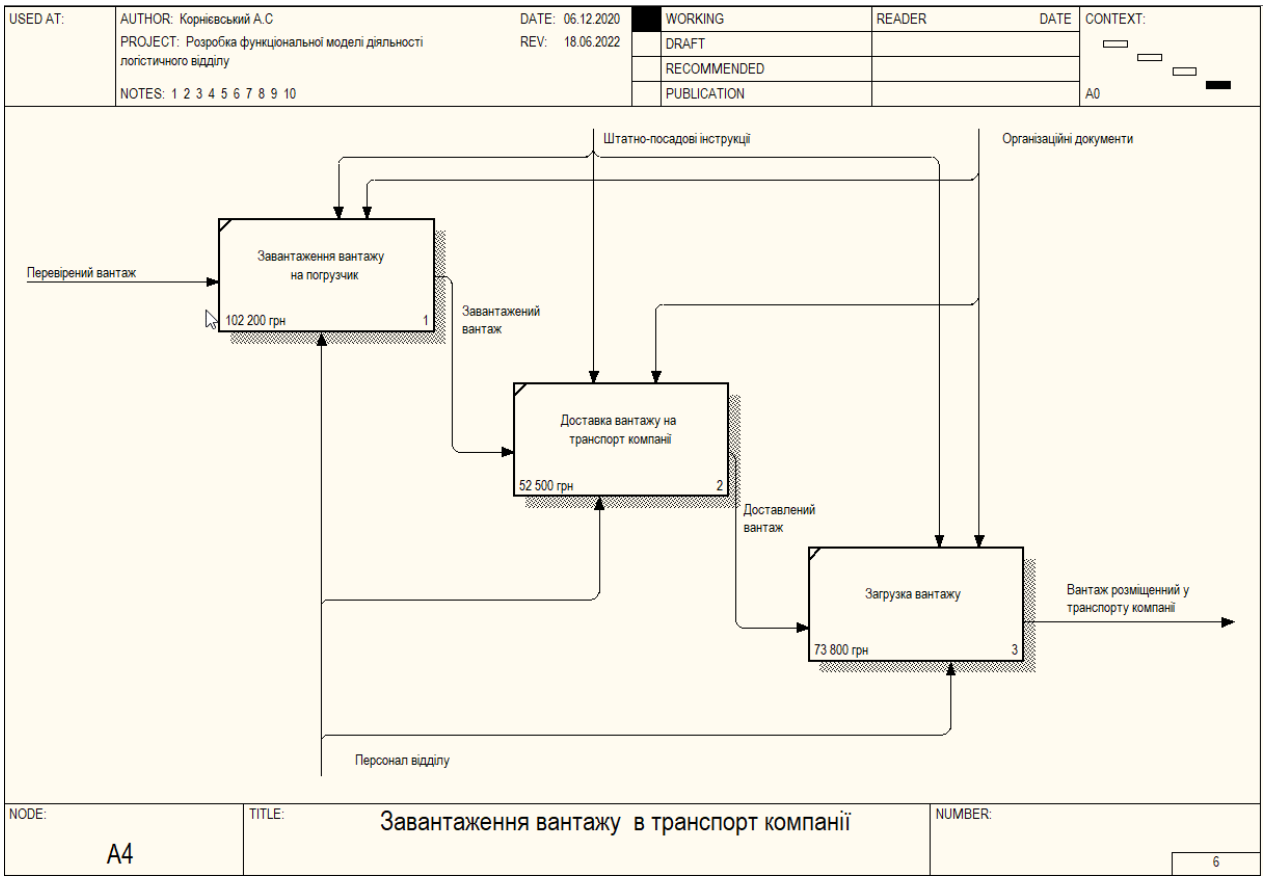


Рисунок А.6 - Діаграма декомпозиції блоку «Завантаження вантажу в транспорт компанії»

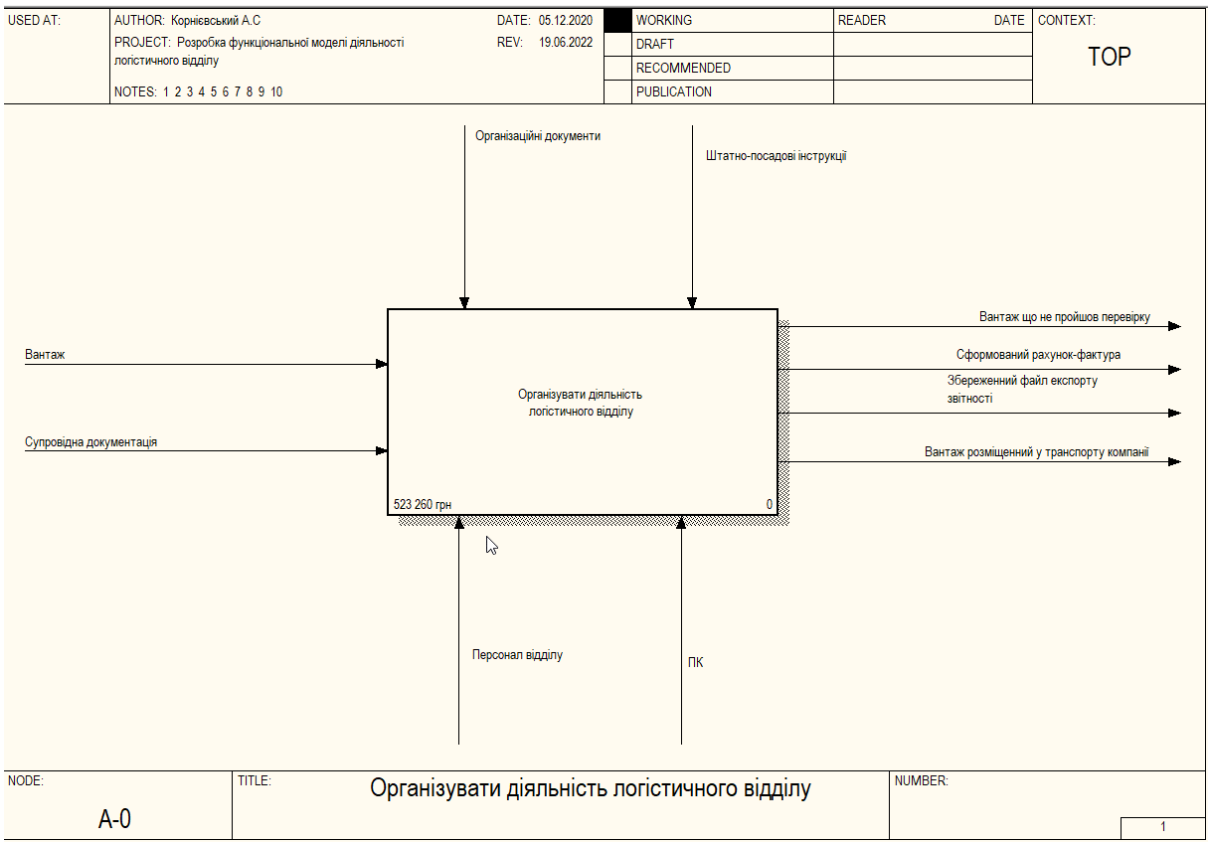


Рисунок А.7 - Контекстна діаграма ТО-ВЕ функціональної моделі

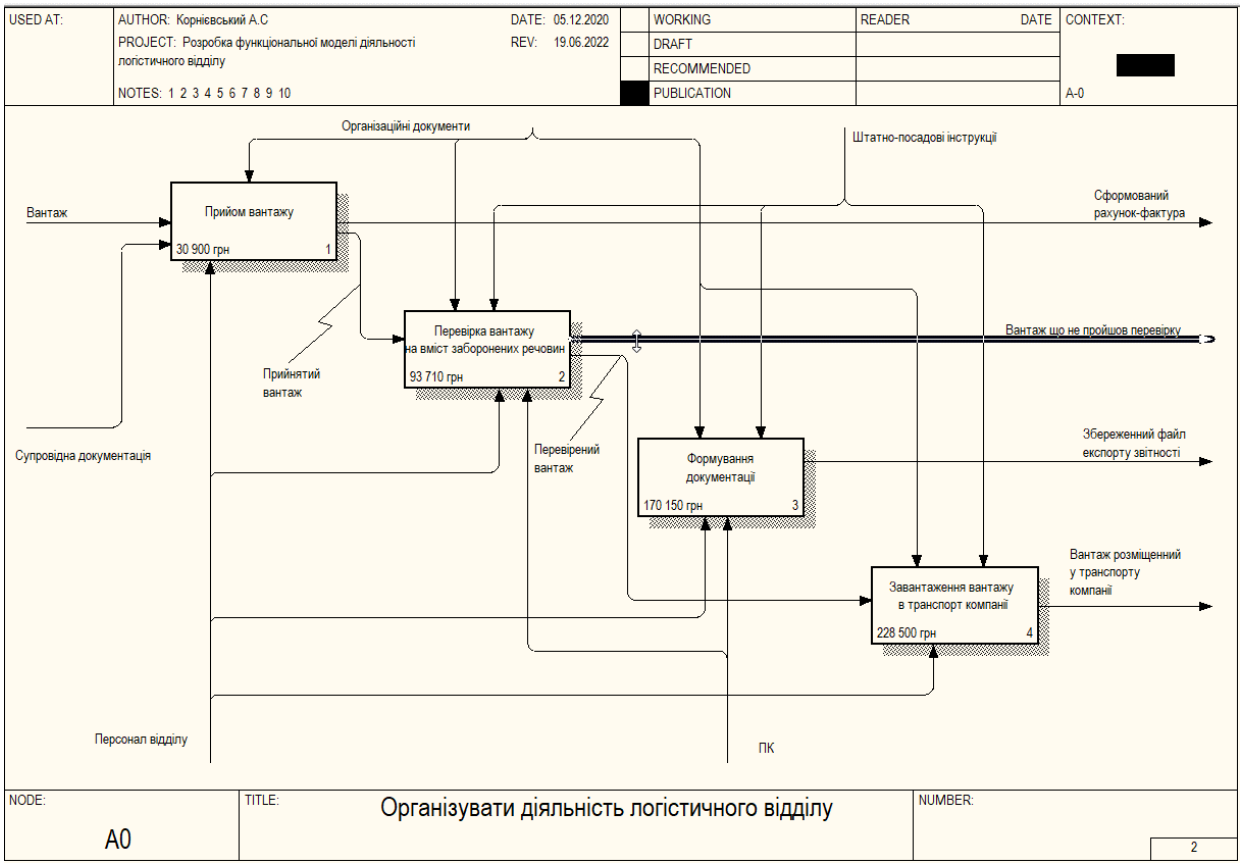


Рисунок А.8 - Діаграма декомпозиції на першому рівні

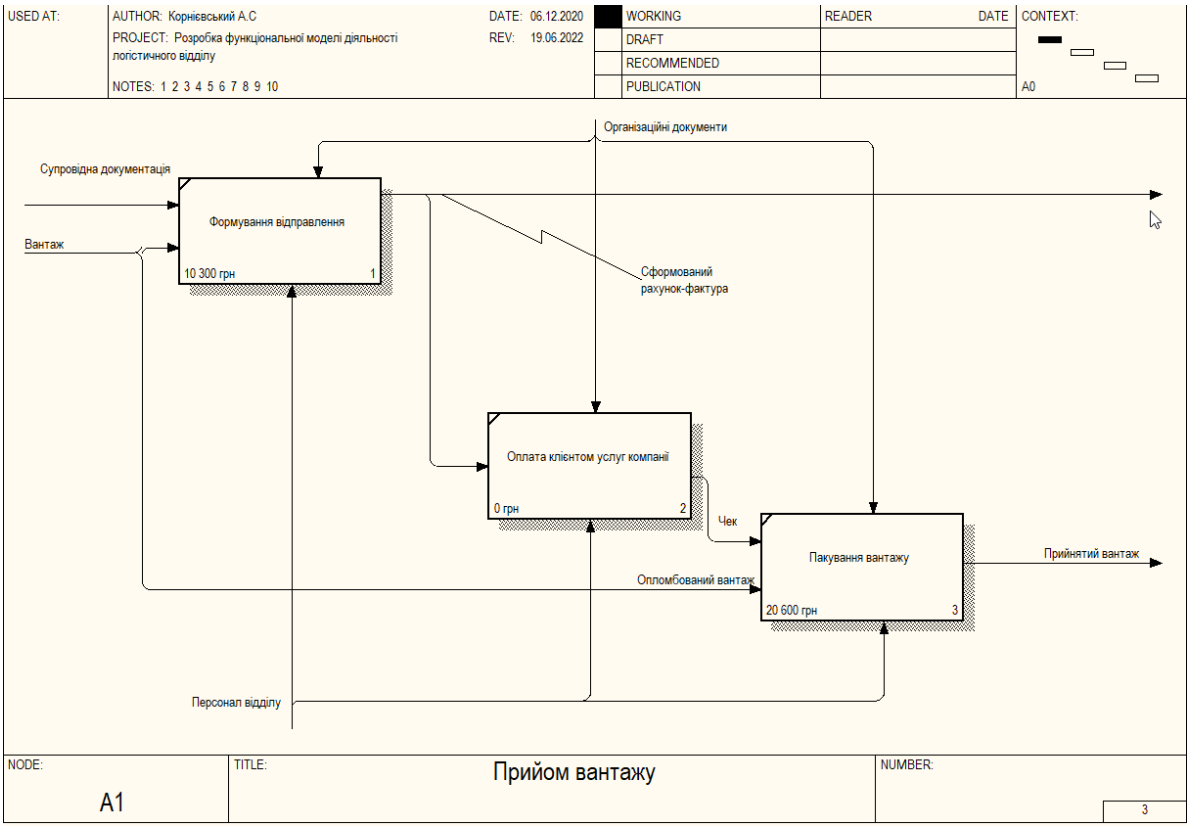


Рисунок А.9 - Діаграма декомпозиції блоку «Прийом вантажу»

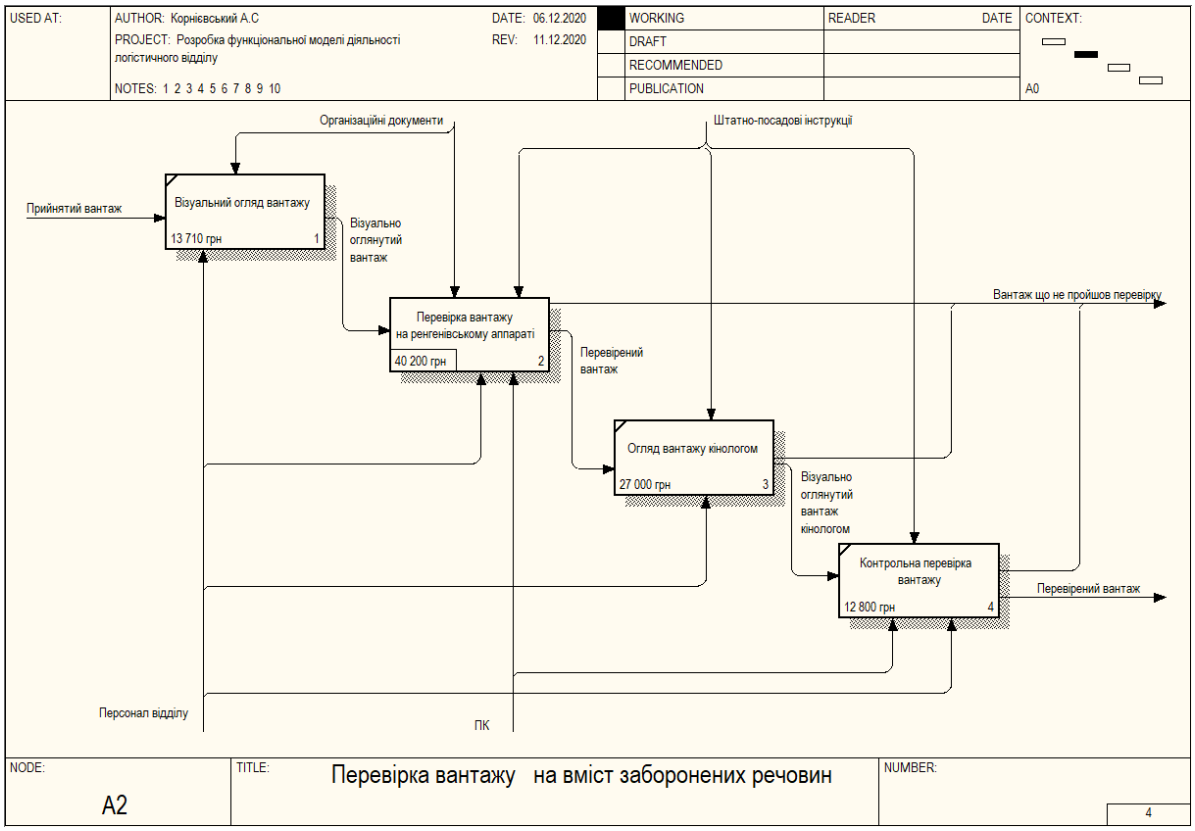


Рисунок А.10 - Діаграма декомпозиції блоку «Перевірка вантажу на вміст заборонених речовин»

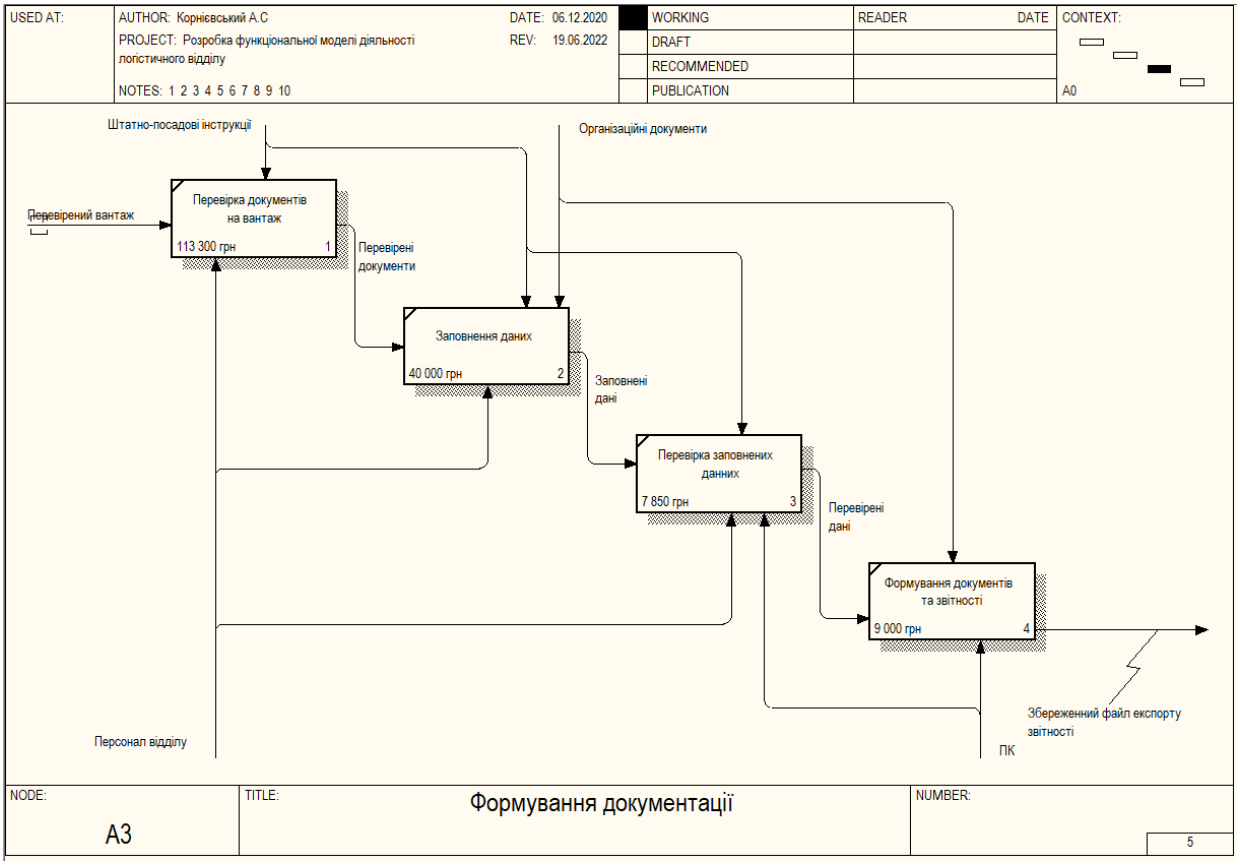


Рисунок А.11 - Діаграма декомпозиції блоку «Формування документації»

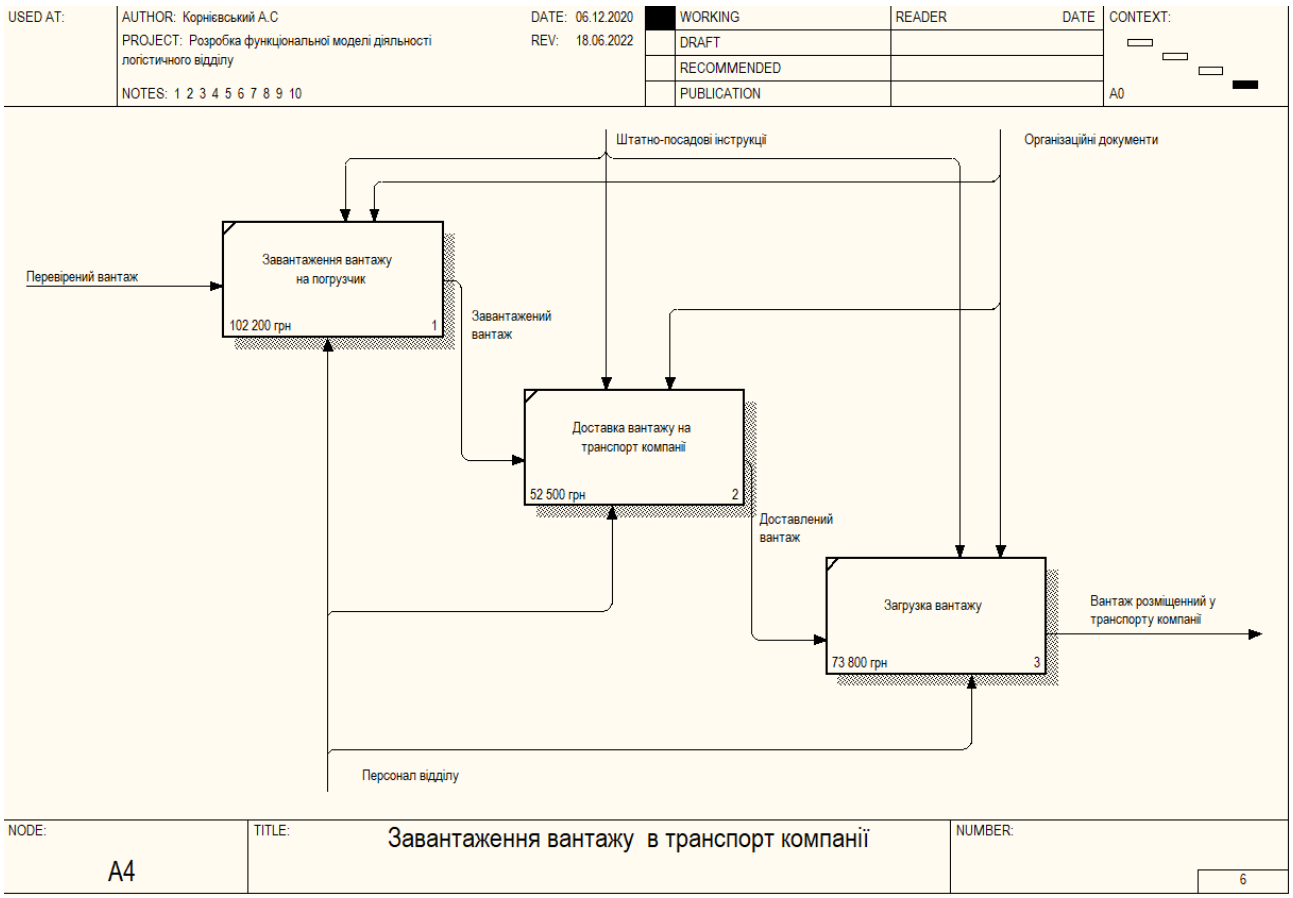
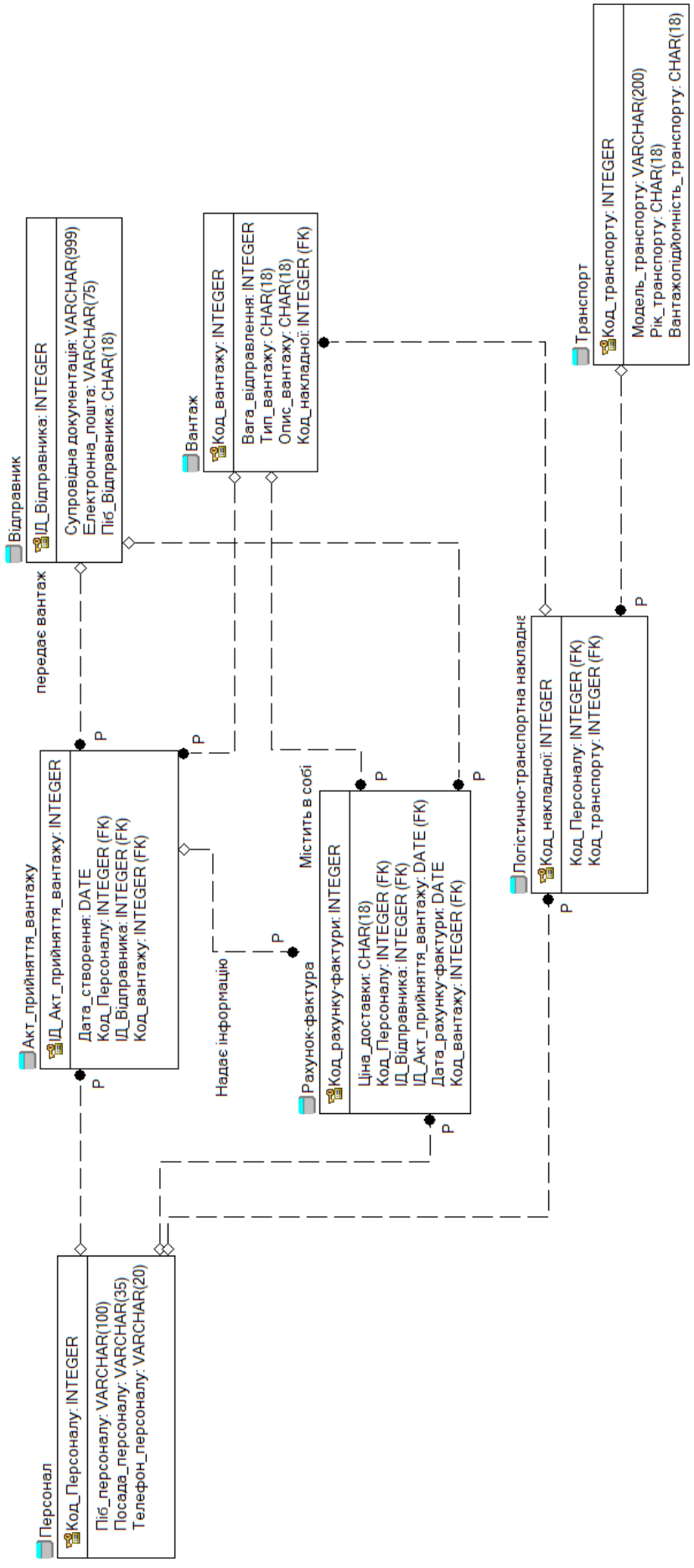
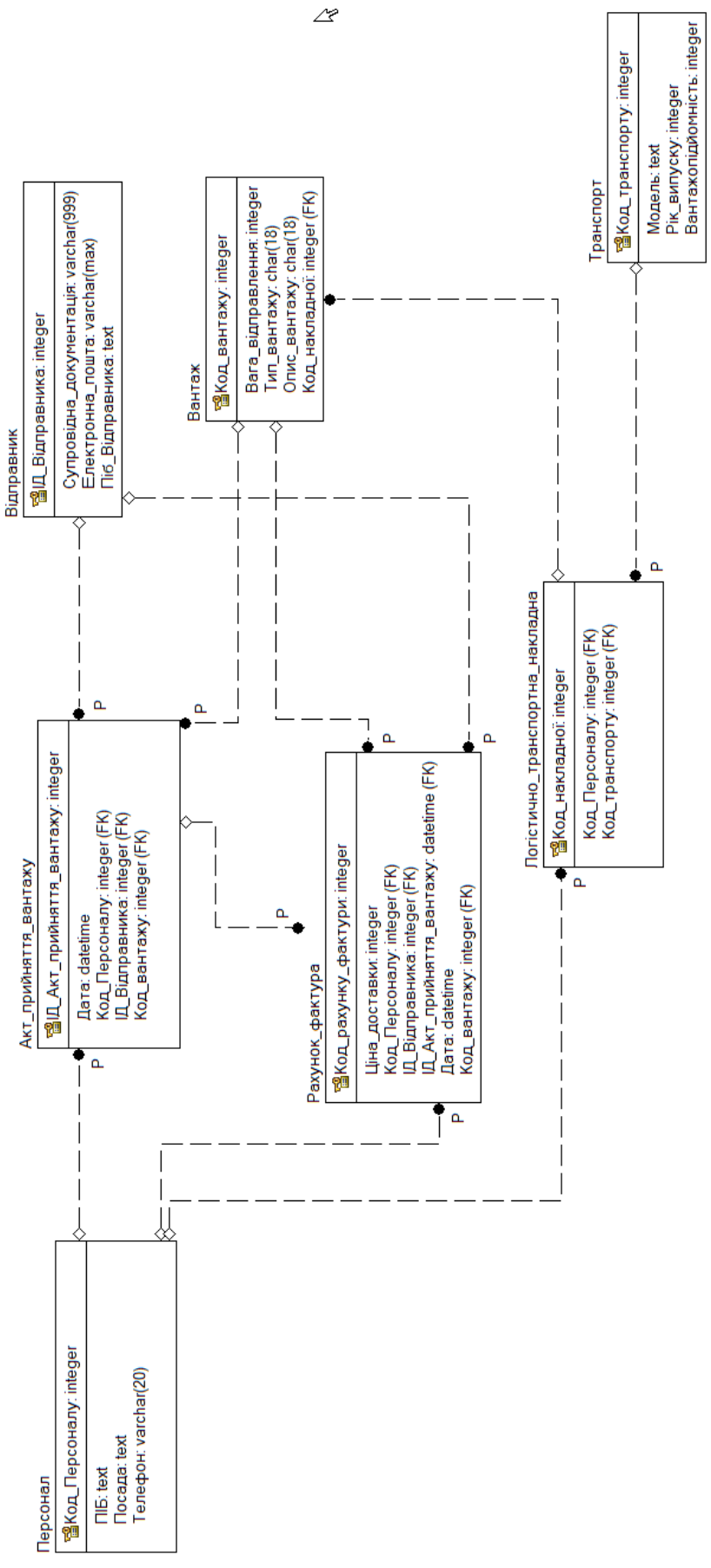


Рисунок А.13 - Діаграма декомпозиції блоку «Завантаження вантажу в транспорт компанії»

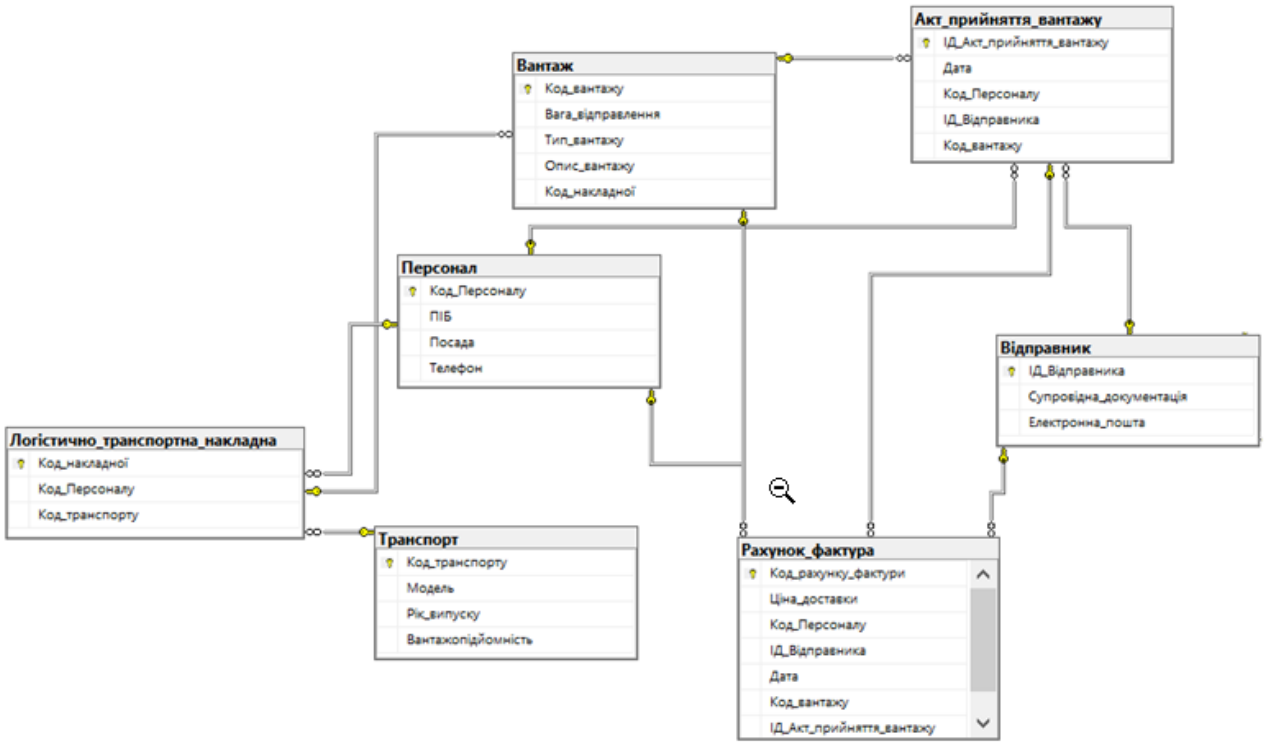
ДОДАТОК Б – ЛОГІЧНА МОДЕЛЬ БАЗИ ДАНИХ



ДОДАТОК В – ФІЗИЧНА МОДЕЛЬ БАЗИ ДАНИХ



ДОДАТОК Г – ЗГЕНЕРОВАНА БАЗА ДАНИХ У MICROSOFT SQL SERVER



ДОДАТОК Д – ВІКНА ТА КОД ПРОГРАМИ

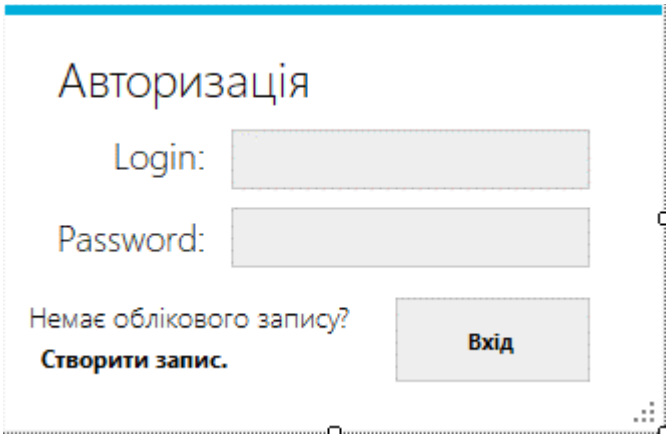


Рисунок Д.1. Форма авторізації

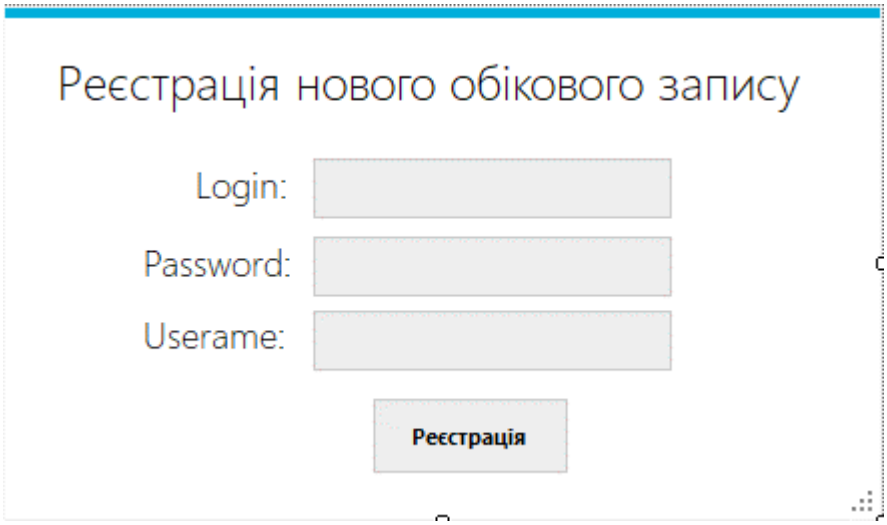


Рисунок Д.2. Форма реєстрації

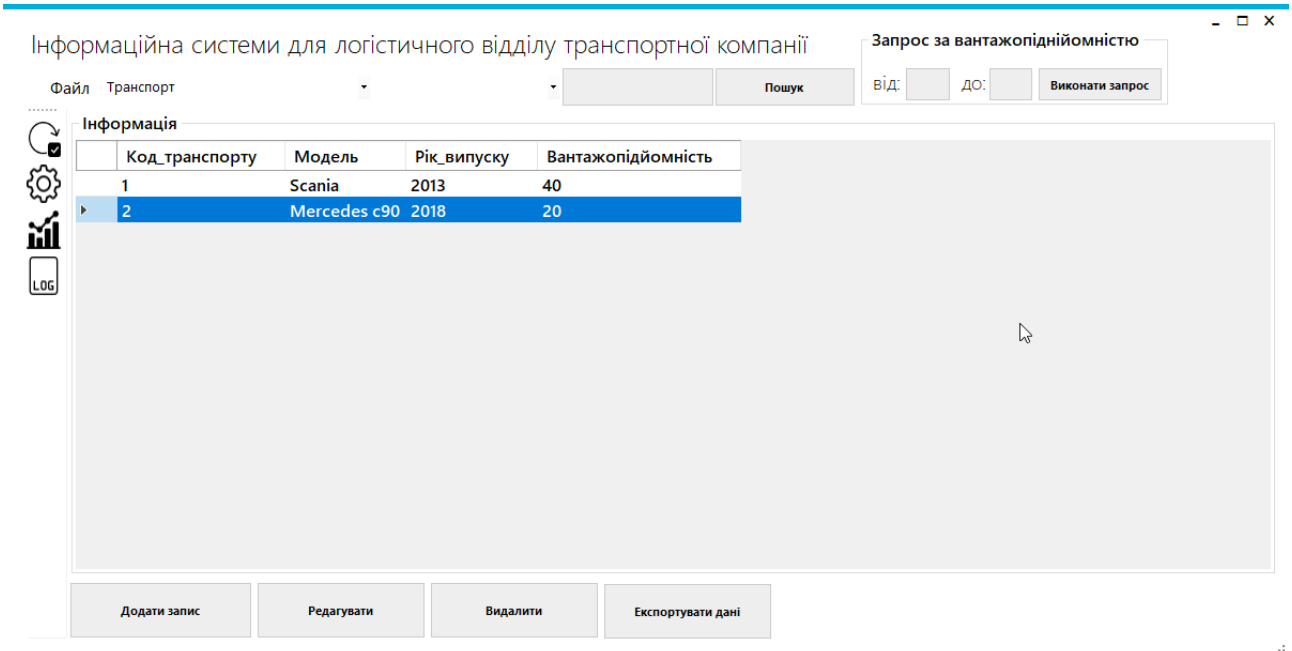


Рисунок Д.3. Головне меню

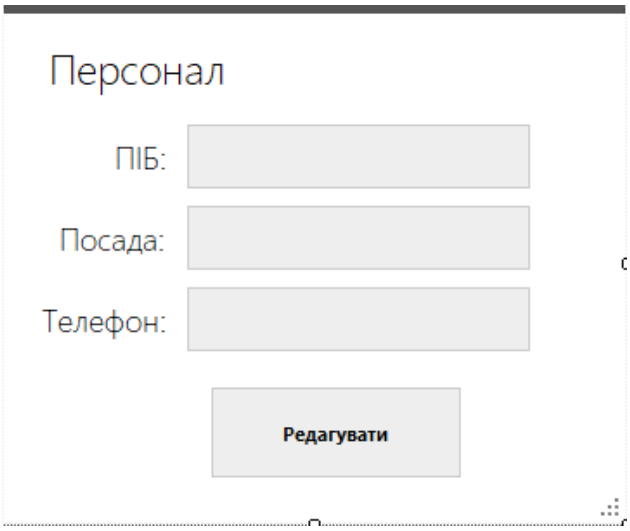


Рисунок Д.4. Вікно “Персонал”

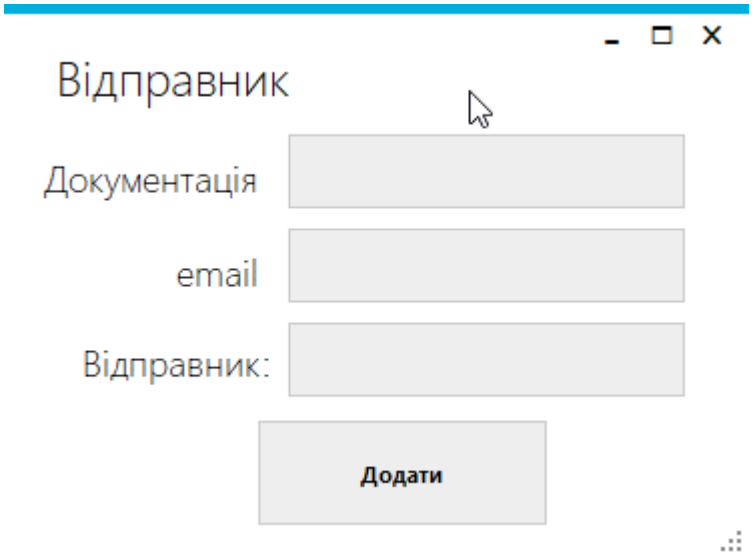


Рисунок Д.5. Вікно “Відправник”

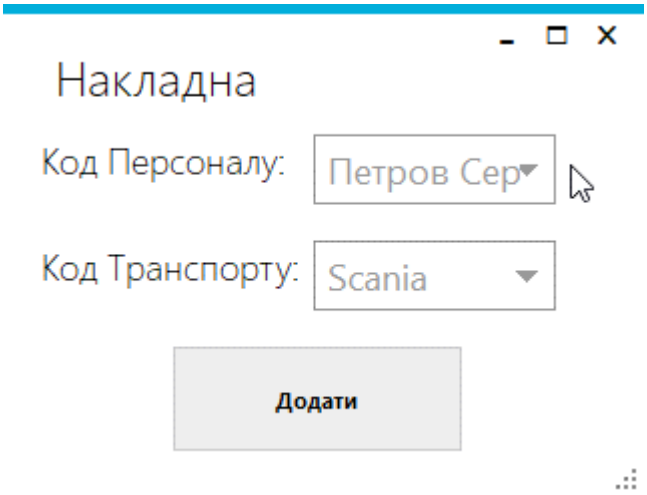


Рисунок Д.6. Вікно “Накладна”

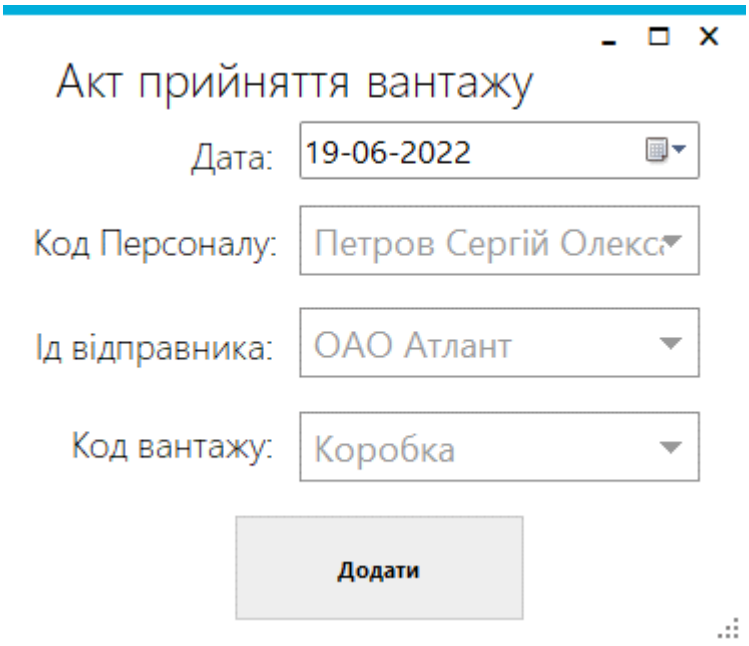


Рисунок Д.7. Вікно “Акт прийняття вантажу”

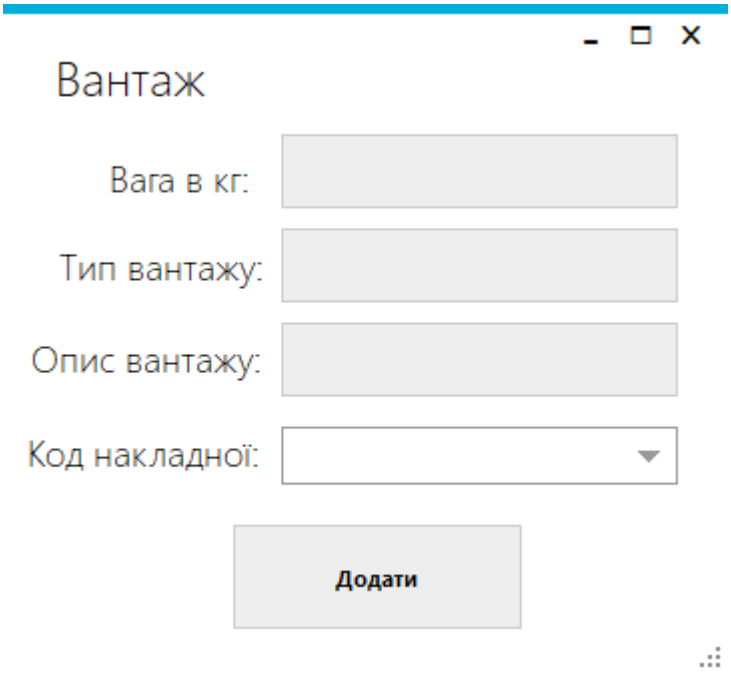


Рисунок Д.8. Вікно “Вантаж”

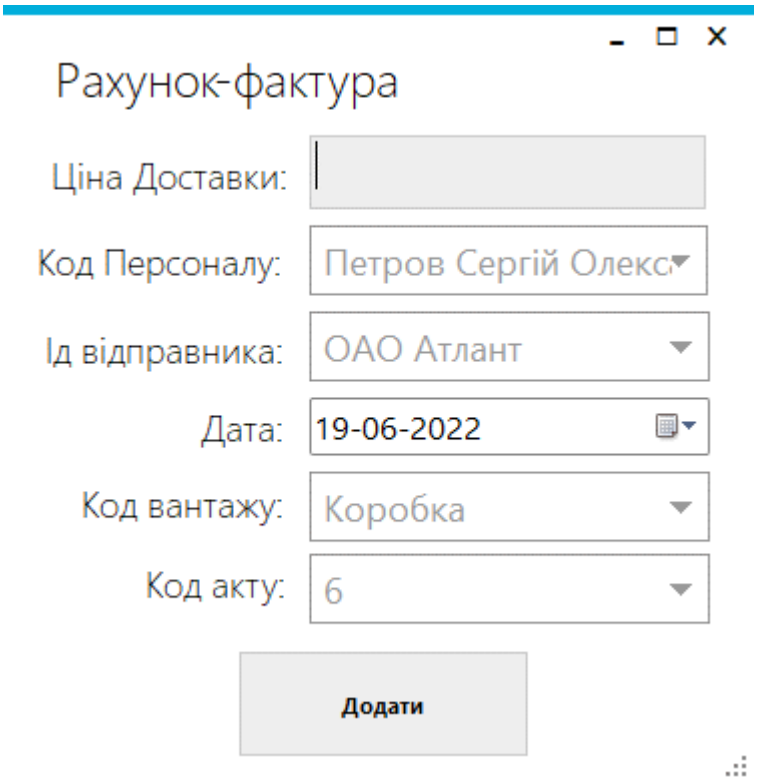


Рисунок Д.8. Вікно “Рахунок-фактура”

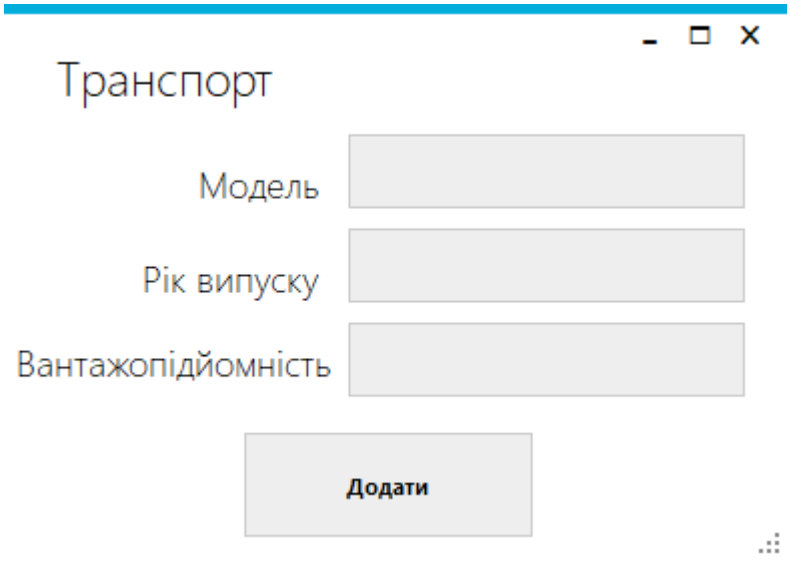


Рисунок Д.9. Вікно “Транспорт”

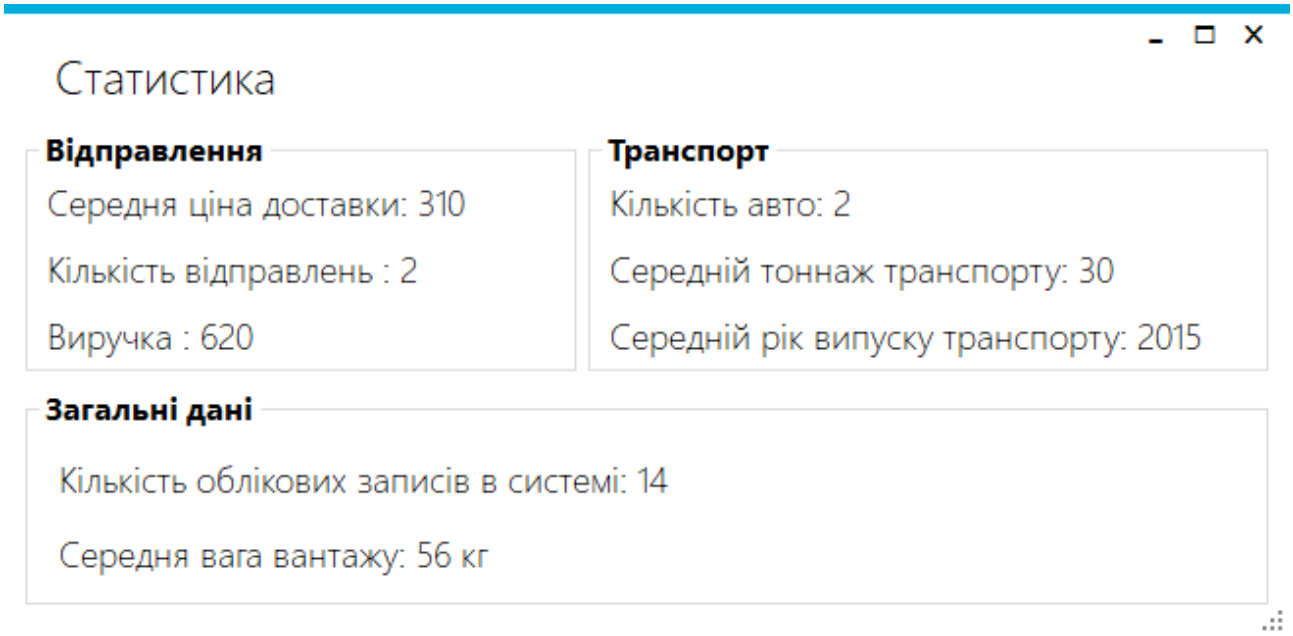


Рисунок Д.10. Вікно “Статистика”

Код форми авторизація

```
using System;
using System.Data;
using System.Data.SqlClient;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Дипломна
{
    public partial class frmAutorization : MetroFramework.Forms.MetroForm
    {
        string connectionString = @"Data Source = (LocalDB)\MSSQLLocalDB; AttachDbFilename
=" + Application.StartupPath + @"\Database.mdf; Integrated Security = True; Connect
Timeout=30;User Instance=False";
        SqlConnection connection;
        DataSet ds = new DataSet();
        public frmAutorization()
        {
            InitializeComponent();
        }
        private void btnRegistration_Click(object sender, EventArgs e)
        {
            this.Hide();
            frmRegister frm = new frmRegister();
            frm.Show();
        }

        private void BtnAutorization_Click(object sender, EventArgs e)
        {
            string userlogin = textLogin.Text;
            string userpassword = textPassword.Text;
            DataTable table = new DataTable();
            connection = new SqlConnection(connectionString);

            SqlCommand comm = new SqlCommand("SELECT * FROM [users] WHERE login =@userLogin
AND password =@userPassword", connection);
```

```
            comm.Parameters.AddWithValue("@userLogin", userlogin);
            comm.Parameters.AddWithValue("@userPassword", userpassword);
            SqlDataAdapter dataadapter = new SqlDataAdapter(comm);
            dataadapter.Fill(table);
            comm.Connection.Close();
            //  connection.Close();

            if (table.Rows.Count > 0)
            {
                frmMain mainform = new frmMain();
                mainform.Show();
                this.Hide();
            }
            else MessageBox.Show("Данні введено некорректно!");
        }
    }
    }
```

Код форми реєстрації

```
using System;
using System.Data.SqlClient;
using System.Drawing;
using System.Linq;
using System.Windows.Forms;

namespace Дипломна
{
    public partial class frmRegister : MetroFramework.Forms.MetroForm
    {
        string connectionString = @"Data Source = (LocalDB)\MSSQLLocalDB; AttachDbFilename
=" + Application.StartupPath + @"\Database.mdf; Integrated Security = True";
        SqlConnection sqlConnection;
        public frmRegister()
        {
            InitializeComponent();
        }
        public Boolean isUserExists()
        {
            DataTable table = new DataTable();
            SqlDataAdapter dataadapter = new SqlDataAdapter();
            sqlConnection = new SqlConnection(connectionString);
            SqlCommand command = new SqlCommand("SELECT * FROM[users] WHERE login =
@userLogin", sqlConnection);
            command.Parameters.AddWithValue("@userLogin", textLogin.Text);
            dataadapter.SelectCommand = command;
            dataadapter.Fill(table);
            if (table.Rows.Count > 0)
            {
                MessageBox.Show("Такий обліковий запис вже існує! Введіть інший логін, або
авторизуйтесь.");
                return true;
            }
            else
            {
                return false;
            }
        }
        private void frmRegister_Load(object sender, EventArgs e)
        {

        }

        private void BtnRegistration_Click(object sender, EventArgs e)
```

```
        {
            if (isUserExists()) { return; }
            if (String.IsNullOrWhiteSpace(textLogin.Text) != true &&
String.IsNullOrWhiteSpace(textPassword.Text) != true &&
String.IsNullOrWhiteSpace(textUsername.Text) != true)
            {
                sqlConnection = new SqlConnection(connectionString);


                SqlCommand command = new SqlCommand("INSERT INTO [users] (login, password,
name) VALUES (@login,@password,@name)", sqlConnection);
                if (command.Connection.State != ConnectionState.Open)
command.Connection.Open();
                command.Parameters.AddWithValue("@login", textLogin.Text);
                command.Parameters.AddWithValue("@password", textPassword.Text);
                command.Parameters.AddWithValue("@name", textUsername.Text);
                if (command.ExecuteNonQuery() == 1)
                {
                    MessageBox.Show("Обліковий запис створено!");
                }
                else
                {
                    MessageBox.Show("Обліковий не було створено!");
                }
                if (command.Connection.State != ConnectionState.Closed)
command.Connection.Close();
                sqlConnection.Close();
                frmAutorization form = new frmAutorization();
                form.Show();
                this.Close();
            }
        }

        private void frmRegister_FormClosed(object sender, FormClosedEventArgs e)
        {
            frmAutorization frm = new frmAutorization();
            frm.Show();
        }
    }
}
```

Код форми FrmMain.cs інакше вікно головного меню

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.SqlClient;
using System.Drawing;
using System.IO;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using Xceed.Document.NET;
using Xceed.Words.NET;

namespace Дипломна
{

    public partial class frmMain : MetroFramework.Forms.MetroForm
```

```
   {
      string connectionString = @"Data Source = (LocalDB)\MSSQLLocalDB; AttachDbFilename
=" + Application.StartupPath + @"\Kornievscki.mdf; Integrated Security = True";
      SqlConnection connection;
      DataSet ds = new DataSet();
      string a;
      bool mark;
      public frmMain()
      {
         InitializeComponent();
      }
      void updatebase()
      {
         string data = "SELECT * FROM [";
         data += toolStripComboBox1.Text;
         data += "]";
         connection = new SqlConnection(connectionString);
         SqlCommand comm = new SqlCommand(data, connection); // обнеовление базы данных

         SqlDataAdapter dataadapter = new SqlDataAdapter(comm);

         try
         {
            DataSet ds = new DataSet();

            dataadapter.Fill(ds, "Customers");
            dataGridView1.DataSource = ds;
            dataGridView1.DataMember = "Customers";
         }
         catch (Exception ex)
         {
            MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
MessageBoxButtons.OK, MessageBoxIcon.Error);
         }
         finally
         {
            connection.Close();
         }
      }
      private void frmMain_Load(object sender, EventArgs e)
      {
         dataGridView1.SelectionMode = DataGridViewSelectionMode.FullRowSelect;
         dataGridView1.AllowUserToAddRows = false;
         updatebase();
         toolStripComboBox2.Items.Add("Код_Персоналу");
         toolStripComboBox2.Items.Add("ПІБ");
         toolStripComboBox2.Items.Add("Посада");
         toolStripComboBox2.Items.Add("Телефон");
      }

      private void файлToolStripMenuItem_Click(object sender, EventArgs e)
      {
```

```
}

private void toolStripUpdate_Click(object sender, EventArgs e)
{
   updatebase();
}

private void toolStripComboBox1_TextChanged(object sender, EventArgs e)
{
   updatebase();
   switch (toolStripComboBox1.Text)
   {
      case "Персонал":
         toolStripComboBox2.Text = String.Empty;
         toolStripComboBox2.Items.Clear();
         toolStripComboBox2.Items.Add("Код_Персоналу");
         toolStripComboBox2.Items.Add( "ПІБ");
         toolStripComboBox2.Items.Add( "Посада");
         toolStripComboBox2.Items.Add( "Телефон");
         groupBox2.Visible = false;
         grbZaprosCargo.Visible = false;
         btnSaveDoc.Visible = false;
         break;
      case "Відправник":
         toolStripComboBox2.Text = String.Empty;
         toolStripComboBox2.Items.Clear();
         toolStripComboBox2.Items.Add("ІД_Відправника");
         toolStripComboBox2.Items.Add("Супровідна_документація");
         toolStripComboBox2.Items.Add("Електронна_пошта");
         toolStripComboBox2.Items.Add("ПІБ_Відправника");
         groupBox2.Visible = false;
         grbZaprosCargo.Visible = false;
         btnSaveDoc.Visible = false;
         break;
      case "Транспорт":
         toolStripComboBox2.Text = String.Empty;
         toolStripComboBox2.Items.Clear();
         toolStripComboBox2.Items.Add("Код_транспорту");
         toolStripComboBox2.Items.Add("Модель");
         toolStripComboBox2.Items.Add("Рік_випуску");
         toolStripComboBox2.Items.Add("Вантажопідйомність");
         groupBox2.Visible = true;
         grbZaprosCargo.Visible = false;
         btnSaveDoc.Visible = false;
         break;
      case "Вантаж":
         toolStripComboBox2.Text = String.Empty;
         toolStripComboBox2.Items.Clear();
         toolStripComboBox2.Items.Add("Код_вантажу");
         toolStripComboBox2.Items.Add("Вага_відправлення");
         toolStripComboBox2.Items.Add("Тип_вантажу");
```

```
            toolStripComboBox2.Items.Add("Опис_вантажу");
            toolStripComboBox2.Items.Add("Код_накладної");
            groupBox2.Visible = false;
            grbZaprosCargo.Visible = true;
            btnSaveDoc.Visible = false;
            break;
        case "Логістично_транспортна_накладна":
            toolStripComboBox2.Text = String.Empty;
            toolStripComboBox2.Items.Clear();
            toolStripComboBox2.Items.Add("Код_накладної");
            toolStripComboBox2.Items.Add("Код_Персоналу");
            toolStripComboBox2.Items.Add("Код_транспорту");
            groupBox2.Visible = false;
            grbZaprosCargo.Visible = false;
            btnSaveDoc.Visible = false;
            break;
        case "Акт_прийняття_вантажу":
            toolStripComboBox2.Text = String.Empty;
            toolStripComboBox2.Items.Clear();
            toolStripComboBox2.Items.Add("ІД_Акт_прийняття_вантажу");
            toolStripComboBox2.Items.Add("Дата");
            toolStripComboBox2.Items.Add("Код_Персоналу");
            toolStripComboBox2.Items.Add("ІД_Відправника");
            toolStripComboBox2.Items.Add("Код_вантажу");
            groupBox2.Visible = false;
            grbZaprosCargo.Visible = false;
            btnSaveDoc.Visible = false;
            break;
        case "Рахунок_фактура":
            toolStripComboBox2.Text = String.Empty;
            toolStripComboBox2.Items.Clear();
            toolStripComboBox2.Items.Add("Код_рахунку_фактури");
            toolStripComboBox2.Items.Add("Ціна_доставки");
            toolStripComboBox2.Items.Add("Код_Персоналу");
            toolStripComboBox2.Items.Add("ІД_Відправника");
            toolStripComboBox2.Items.Add("Дата");
            toolStripComboBox2.Items.Add("Код_вантажу");
            toolStripComboBox2.Items.Add("ІД_Акт_прийняття_вантажу");
            groupBox2.Visible = false;
            grbZaprosCargo.Visible = false;
            btnSaveDoc.Visible = true;
            break;
    }
}

private void btnAdd_Click(object sender, EventArgs e)
{

}

private void metroToolTip1_Popup(object sender, PopupEventArgs e)
{
```

```
}

private void btnAdd_Click_1(object sender, EventArgs e)
{
   a = null;
   mark = true;
   switch (toolStripComboBox1.Text)
   {
      case "Персонал":
         Form personel = new frmPersonel(mark,a);
         personel.Show();
         break;
      case "Відправник":
         Form Sender = new frmSender(mark, a);
         Sender.Show();
         break;
      case "Транспорт":
         Form transport = new frmTransport(mark, a);
         transport.Show();
         break;
      case "Вантаж":
         Form cargo = new frmCargo(mark, a);
         cargo.Show();
         break;
      case "Логістично_транспортна_накладна":
         Form invoice = new frmInvoice(mark, a);
         invoice.Show();
         break;
      case "Акт_прийняття_вантажу":
         Form act = new actOfGoods(mark, a);
         act.Show();
         break;
      case "Рахунок_фактура":
         Form bill = new frmBill(mark, a);
         bill.Show();
         break;
   }
}

private void btnUpdateInfo_Click(object sender, EventArgs e)
{
   mark = false;
   int index = dataGridView1.CurrentCell.RowIndex;
   string[] myArray = new string[dataGridView1.SelectedCells.Count + 1];
   dataGridView1.Rows[index].Selected = true;

   for (int j = 0; j < dataGridView1.Columns.Count; j++)
   {
      myArray[j] = dataGridView1.Rows[index].Cells[0].Value.ToString(); //получення
      a = myArray[j];
```

```
            }
            switch (toolStripComboBox1.Text)
            {
                case "Персонал":
                    Form personel = new frmPersonel(mark,a);
                    personel.Show();
                    break;
                case "Відправник":
                    Form Sender = new frmSender(mark, a);
                    Sender.Show();
                    break;
                case "Транспорт":
                    Form transport = new frmTransport(mark, a);
                    transport.Show();
                    break;
                case "Вантаж":
                    Form cargo = new frmCargo(mark, a);
                    cargo.Show();
                    break;
                case "Логістично_транспортна_накладна":
                    Form invoice = new frmInvoice(mark, a);
                    invoice.Show();
                    break;
                case "Акт_прийняття_вантажу":
                    Form act = new actOfGoods(mark, a);
                    act.Show();
                    break;
                case "Рахунок_фактура":
                    Form bill = new frmBill(mark, a);
                    bill.Show();
                    break;
            }
        }

        private void menuStrip1_ItemClicked(object sender, ToolStripItemClickedEventArgs e)
        {

        }
        private void ToExcel(DataGridView dGV, string filename)
        {
            string stOutput = "";
            string stHeaders = "";
            for (int j = 0; j < dGV.Columns.Count; j++)
                stHeaders = stHeaders.ToString() + Convert.ToString(dGV.Columns[j].HeaderText) +
"\t";
            stOutput += stHeaders + "\r\n";

            for (int i = 0; i < dGV.RowCount - 1; i++)
            {
                string stline = "";
                for (int j = 0; j < dGV.Rows[i].Cells.Count; j++)
                    stline = stline.ToString() + Convert.ToString(dGV.Rows[i].Cells[j].Value) + "\t";
```

```
            stOutput += stline + "\r\n";
        }

        Encoding utf16 = Encoding.GetEncoding(1251);
        byte[] output = utf16.GetBytes(stOutput);
        FileStream fs = new FileStream(filename, FileMode.Create);
        BinaryWriter bw = new BinaryWriter(fs);
        bw.Write(output, 0, output.Length);
        bw.Flush();
        bw.Close();
        fs.Close();
    }

    private async void btnDelete_Click(object sender, EventArgs e)
    {
        if (dataGridView1.SelectedCells != null)
        {
            try
            {

                if (dataGridView1.SelectedRows.Count > 0)
                {

                    int index = dataGridView1.CurrentCell.RowIndex;
                    string[] myArray = new string[dataGridView1.SelectedCells.Count + 1];
                    dataGridView1.Rows[index].Selected = true;

                    for (int j = 0; j < dataGridView1.Columns.Count; j++)
                    {
                        myArray[j] = dataGridView1.Rows[index].Cells[0].Value.ToString(); //
получаем id вибраного елементу
                        a = myArray[j];
                    }
                    string data = "DELETE FROM [";
                    data += toolStripComboBox1.Text;
                    switch (toolStripComboBox1.Text)
                    {
                        case "Персонал":
                        data += "] WHERE [Код_Персоналу]= @ID";
                            break;
                        case "Відправник":
                            data += "] WHERE [ІД_Відправника]= @ID";
                            break;
                        case "Транспорт":
                            data += "] WHERE [Код_транспорту]= @ID";
                            break;
                        case "Вантаж":
                            data += "] WHERE [Код_вантажу]= @ID";
                            break;
                        case "Логістично_транспортна_накладна":
                            data += "] WHERE [Код_накладної]= @ID";
                            break;
```

```
                    case "Акт_прийняття_вантажу":
                        data += "] WHERE [ІД_Акт_прийняття_вантажу]= @ID";
                        break;
                    case "Рахунок_фактура":
                        data += "] WHERE [Код_рахунку_фактури]= @ID";
                        break;
                }
                SqlCommand command = new SqlCommand(data, connection);
                if (command != null || command.Connection.State != ConnectionState.Open)
command.Connection.Open();
                DialogResult DR = MessageBox.Show("Запис буде видалено \n Продовжити?",
"Видалення записи", MessageBoxButtons.OKCancel, MessageBoxIcon.Question,
MessageBoxDefaultButton.Button2);
                if (DR == DialogResult.OK)
                {
                    command.Parameters.AddWithValue("@ID", a); // сама процедура удаления
                    await command.ExecuteNonQueryAsync();
                    connection.Close(); // закриття підключення

                }

            }

        }


        catch (Exception ex)
        {
            MessageBox.Show("Можлива втрата даних! Видалення неможливе!", "Помилка!",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
        }
        finally
        {
            updatebase();
        }
    }
}

private void btnSearch_Click(object sender, EventArgs e)
{
    connection = new SqlConnection(connectionString);
    string table = "@";
    string data = "SELECT * FROM [";
    data += toolStripComboBox1.Text;
    data += "] WHERE ";
    data += toolStripComboBox2.Text;
    data += " LIKE @";
    data += toolStripComboBox2.Text;
    table += toolStripComboBox2.Text;
    SqlCommand comm = new SqlCommand(data, connection);

    comm.Parameters.AddWithValue(table, '%' + textVvod.Text + '%');
```

```
            SqlDataAdapter dataadapter = new SqlDataAdapter(comm);
            try
            {
                DataSet ds = new DataSet();
                dataadapter.Fill(ds, "Customer");
                dataGridView1.DataSource = ds;
                dataGridView1.DataMember = "Customer";
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
            finally
            {
                connection.Close();
                textVvod.Text = null;

            }
        }

        private void metroButton1_Click(object sender, EventArgs e)
        {
            SaveFileDialog sfd = new SaveFileDialog();
            sfd.Filter = "Excel Document (*.xls)|*.xls";
            if (sfd.ShowDialog() == DialogResult.OK)
            {
                ToExcel(dataGridView1, sfd.FileName);
            }
        }
        private void OnlyDigit(KeyPressEventArgs e)
        {
            char number = e.KeyChar;
            if ((e.KeyChar <= 47 || e.KeyChar >= 58) && number != 8 && (e.KeyChar <= 39 ||
e.KeyChar >= 46) && number != 47 && number != 61) //калькулятор
            {
                e.Handled = true;
            }
        }
        private void txtVid_KeyPress(object sender, KeyPressEventArgs e)
        {
            OnlyDigit(e);
        }

        private void txtDO_KeyPress(object sender, KeyPressEventArgs e)
        {
            OnlyDigit(e);
        }

        private void metroButton2_Click(object sender, EventArgs e)
        {
```

```
        if (String.IsNullOrWhiteSpace(txtDO.Text) != true &
String.IsNullOrWhiteSpace(txtVid.Text) != true)
        {
            string data = "SELECT * FROM [Транспорт] Where Вантажопідйомність BETWEEN
@txtvid AND @txtdo";
            connection = new SqlConnection(connectionString);
            SqlCommand comm = new SqlCommand(data, connection); // обнеовление базы
данных
            SqlDataAdapter dataadapter = new SqlDataAdapter(comm);
            comm.Parameters.AddWithValue("txtvid", txtVid.Text);
            comm.Parameters.AddWithValue("txtdo", txtDO.Text);
            try
            {
               DataSet ds = new DataSet();
               dataadapter.Fill(ds, "Customers");
               dataGridView1.DataSource = ds;
               dataGridView1.DataMember = "Customers";
            }
            catch (Exception ex)
            {
               MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
            finally
            {
               connection.Close();
            }
        }
        if (String.IsNullOrWhiteSpace(txtDO.Text) != false &
String.IsNullOrWhiteSpace(txtVid.Text) != true)
        {
            string data = "SELECT * FROM [Транспорт] Where Вантажопідйомність >=
@txtvid";
            connection = new SqlConnection(connectionString);
            SqlCommand comm = new SqlCommand(data, connection); // обнеовление базы
данных
            SqlDataAdapter dataadapter = new SqlDataAdapter(comm);
            comm.Parameters.AddWithValue("txtvid", txtVid.Text);

            try
            {
               DataSet ds = new DataSet();
               dataadapter.Fill(ds, "Customers");
               dataGridView1.DataSource = ds;
               dataGridView1.DataMember = "Customers";
            }
            catch (Exception ex)
            {
               MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
            finally
```

```
            {
                connection.Close();
            }
        }
        if (String.IsNullOrWhiteSpace(txtDO.Text) != true &
String.IsNullOrWhiteSpace(txtVid.Text) != false)
        {
            string data = "SELECT * FROM [Транспорт] Where Вантажопідйомність <= @txtdo";
            connection = new SqlConnection(connectionString);
            SqlCommand comm = new SqlCommand(data, connection); // обнеовление базы
данных
            SqlDataAdapter dataadapter = new SqlDataAdapter(comm);
            comm.Parameters.AddWithValue("txtdo", txtDO.Text);
            try
            {
                DataSet ds = new DataSet();
                dataadapter.Fill(ds, "Customers");
                dataGridView1.DataSource = ds;
                dataGridView1.DataMember = "Customers";
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
            finally
            {
                connection.Close();
            }
        }
    }

    private void toolStripSettngs_Click(object sender, EventArgs e)
    {
        if (fontDialog1.ShowDialog() == DialogResult.Cancel)
            return;
        // установка шрифта

        dataGridView1.DefaultCellStyle.Font = fontDialog1.Font;

        dataGridView1.ForeColor = fontDialog1.Color;
    }

    private void toolStripStat_Click(object sender, EventArgs e)
    {
        Form stat = new frmStatistics();
        stat.Show();
    }

    private void metroButton3_Click(object sender, EventArgs e)
    {
```

```
        if (String.IsNullOrWhiteSpace(txtDOcargo.Text) != true &
String.IsNullOrWhiteSpace(txtVidCargo.Text) != true)
        {
            string data = "SELECT * FROM [Вантаж] Where Вага_відправлення BETWEEN
@txtvid AND @txtdo";
            connection = new SqlConnection(connectionString);
            SqlCommand comm = new SqlCommand(data, connection); // обнеовление базы
данных
            SqlDataAdapter dataadapter = new SqlDataAdapter(comm);
            comm.Parameters.AddWithValue("txtvid", txtVidCargo.Text);
            comm.Parameters.AddWithValue("txtdo", txtDOcargo.Text);
            try
            {
               DataSet ds = new DataSet();
               dataadapter.Fill(ds, "Customers");
               dataGridView1.DataSource = ds;
               dataGridView1.DataMember = "Customers";
            }
            catch (Exception ex)
            {
               MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
            finally
            {
               connection.Close();
            }
        }
        if (String.IsNullOrWhiteSpace(txtDOcargo.Text) != false &
String.IsNullOrWhiteSpace(txtVidCargo.Text) != true)
        {
            string data = "SELECT * FROM [Вантаж] Where Вага_відправлення >= @txtvid";
            connection = new SqlConnection(connectionString);
            SqlCommand comm = new SqlCommand(data, connection); // обнеовление базы
данных
            SqlDataAdapter dataadapter = new SqlDataAdapter(comm);
            comm.Parameters.AddWithValue("txtvid", txtVidCargo.Text);

            try
            {
               DataSet ds = new DataSet();
               dataadapter.Fill(ds, "Customers");
               dataGridView1.DataSource = ds;
               dataGridView1.DataMember = "Customers";
            }
            catch (Exception ex)
            {
               MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
            finally
            {
```

```
            connection.Close();
        }
    }
    if (String.IsNullOrWhiteSpace(txtDOcargo.Text) != true &
String.IsNullOrWhiteSpace(txtVidCargo.Text) != false)
    {
        string data = "SELECT * FROM [Вантаж] Where Вага_відправлення <= @txtdo";
        connection = new SqlConnection(connectionString);
        SqlCommand comm = new SqlCommand(data, connection); // обнеовление базы
данных
        SqlDataAdapter dataadapter = new SqlDataAdapter(comm);
        comm.Parameters.AddWithValue("txtdo", txtDOcargo.Text);
        try
        {
            DataSet ds = new DataSet();
            dataadapter.Fill(ds, "Customers");
            dataGridView1.DataSource = ds;
            dataGridView1.DataMember = "Customers";
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
        finally
        {
            connection.Close();
        }
    }
}

private async void metroButton4_Click(object sender, EventArgs e)
{
    string price = null;
    string time = null;
    string personel = null;
    string vidpranyk = null;
    string cargo = null;
    SqlConnection sqlConnection;
    sqlConnection = new SqlConnection(connectionString);
    SqlDataReader sqlReader = null;
    try
    {

        if (dataGridView1.SelectedRows.Count > 0)
        {

            int index = dataGridView1.CurrentCell.RowIndex;
            string[] myArray = new string[dataGridView1.SelectedCells.Count + 1];
            dataGridView1.Rows[index].Selected = true;

            for (int j = 0; j < dataGridView1.Columns.Count; j++)
```

```
            {
                myArray[j] = dataGridView1.Rows[index].Cells[0].Value.ToString(); // получаем
id вибраного елементу
                a = myArray[j];
            }
            if (a != null)
            {


                await sqlConnection.OpenAsync();
                SqlCommand command = new SqlCommand("SELECT *
FROM[Рахунок_фактура] WHERE [Код_рахунку_фактури] = @id", sqlConnection); // запрос
на виборку даних
                command.Parameters.AddWithValue("@id",a);
                try
                {
                    sqlReader = await command.ExecuteReaderAsync();
                    await sqlReader.ReadAsync();
                    price = Convert.ToString(sqlReader["Ціна_доставки"]); //считує конкретні
поля
                    personel = Convert.ToString(sqlReader["Код_Персоналу"]);
                    time = Convert.ToString(sqlReader["Дата"]);
                    vidpranyk = Convert.ToString(sqlReader["ІД_Відправника"]);
                    cargo = Convert.ToString(sqlReader["Код_вантажу"]);

                }
                catch (Exception ex)
                {
                    MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
MessageBoxButtons.OK, MessageBoxIcon.Error);
                }
                finally

                {
                    if (sqlReader != null)
                    {
                        sqlReader.Close();
                    }
                }
            }

            SqlDataReader sqlReader1 = null;

            SqlCommand command1 = new SqlCommand("SELECT * FROM[Персонал]
WHERE [Код_Персоналу] = @id", sqlConnection); // запрос на виборку даних
            command1.Parameters.AddWithValue("@id", personel);
            sqlReader1 = await command1.ExecuteReaderAsync();
            await sqlReader1.ReadAsync();
            personel = Convert.ToString(sqlReader1["ПІБ"]);
            if (sqlReader1 != null)
            {
                sqlReader1.Close();
```

```
            }
            SqlDataReader sqlReader2 = null;
            SqlCommand command2 = new SqlCommand("SELECT * FROM[Відправник]
WHERE [ІД_Відправника] = @id", sqlConnection); // запрос на виборку даних
            command2.Parameters.AddWithValue("@id", vidpranyk);
            sqlReader2 = await command2.ExecuteReaderAsync();
            await sqlReader2.ReadAsync();
            vidpranyk = Convert.ToString(sqlReader2["ПІБ_Відправника"]);
            if (sqlReader2 != null)
            {
               sqlReader2.Close();
            }


            string filePath = Application.StartupPath + "/Documents/ID" + a.ToString() + ".docx";
            DocX doc = DocX.Create(filePath);


            string title = "Логістично-транспортний відділ" + Environment.NewLine +
"+38044228322" + Environment.NewLine + "ФОП Мудрук, м Житомир. вул Київська 1" +
Environment.NewLine + "IBAN UA43242425564565464 ЄДРПОУ 344234";
            Formatting titleFormat = new Formatting();
            //Specify font family
            titleFormat.FontFamily = new Xceed.Document.NET.Font("Arial");
            //Specify font size
            titleFormat.Size = 12;
            Paragraph paragraphTitle = doc.InsertParagraph(title, false, titleFormat);
            paragraphTitle.Alignment = Alignment.right;
            string text = Environment.NewLine + "Номер документу: " + a +
Environment.NewLine + "Ціна до сплати: " + price + Environment.NewLine + "Обробив
замовлення: " + personel + Environment.NewLine + "Відправник: " + vidpranyk +
Environment.NewLine + "Код вантажу: " + cargo;
            Formatting textParagraphFormat = new Formatting();
            textParagraphFormat.FontFamily = new Xceed.Document.NET.Font("Arial");
            //Specify font size
            textParagraphFormat.Size = 14;
            Paragraph paragraphtext = doc.InsertParagraph(text, false, textParagraphFormat);
            paragraphtext.Alignment = Alignment.left;
            string parTime = Environment.NewLine + Environment.NewLine + "Дата: " + time;
            Formatting footer = new Formatting();
            footer.Size = 14;
            footer.FontFamily = new Xceed.Document.NET.Font("Arial");
            Paragraph footertext = doc.InsertParagraph(parTime, false, footer);
            footertext.Alignment = Alignment.left;
            string pidpis = "_____(підпис)";
            Paragraph footepidpis = doc.InsertParagraph(pidpis, false, footer);
            footepidpis.Alignment = Alignment.right;
            doc.Save(); // save changes to file
```

```
            }

        }


        catch (Exception ex)
        {
            MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
        finally
        {

        }
    }


  }
}
```

Код форми ActofGods.cs інакше вікно Акту прийняття вантажу

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

namespace Дипломна
{
    public partial class actOfGoods : MetroFramework.Forms.MetroForm
    {
        string a;
        bool mark;
        SqlConnection sqlConnection;
        string connectionString = @"Data Source = (LocalDB)\MSSQLLocalDB; AttachDbFilename
=" + Application.StartupPath + @"\Kornievscki.mdf; Integrated Security = True";
        public actOfGoods(bool mark, string a)
        {
            this.mark = mark;
            this.a = a;
            InitializeComponent();
        }

        private async void actOfGoods_Load(object sender, EventArgs e)
        {
            DateTime myDateTime = dateTimePicker1.Value; // налаштування поля дата\час
            myDateTime.ToLongTimeString();
            dateTimePicker1.Format = DateTimePickerFormat.Custom;
            dateTimePicker1.CustomFormat = "dd-MM-yyyy";
            dateTimePicker1.MaxDate = DateTime.Now.Date;
            dateTimePicker1.Value = DateTime.Today;
            sqlConnection = new SqlConnection(connectionString);
            SqlCommand comand = new SqlCommand("SELECT * FROM [Персонал]", sqlConnection);
            SqlCommand transport = new SqlCommand("SELECT * FROM [Відправник]",
sqlConnection);
            SqlCommand cargo = new SqlCommand("SELECT * FROM [Вантаж]", sqlConnection);
```

```
            await sqlConnection.OpenAsync();
            SqlDataReader dr = comand.ExecuteReader();
            var vis1 = new List<Tuple<int, string>>();
            while (dr.Read())
            {
                int sid = (Int32)dr["Код_Персоналу"];
                string spib = (string)dr["ПІБ"];
                vis1.Add(Tuple.Create(Convert.ToInt32(sid), spib.ToString()));
            }
            metroComboBox1.DataSource = vis1;
            metroComboBox1.ValueMember = "Item1";
            metroComboBox1.DisplayMember = "Item2";
            dr.Close();
            SqlDataReader druser = transport.ExecuteReader();
            var sender1 = new List<Tuple<int, string>>();
            while (druser.Read())
            {
                int sid = (Int32)druser["ІД_Відправника"];
                string sname = (string)druser["ПІБ_Відправника"];
                sender1.Add(Tuple.Create(Convert.ToInt32(sid), sname.ToString()));
            }
            metroComboBox2.DataSource = sender1;
            metroComboBox2.ValueMember = "Item1";
            metroComboBox2.DisplayMember = "Item2";
            druser.Close();
            SqlDataReader cg = cargo.ExecuteReader();
            var cargo1 = new List<Tuple<int, string, string>>();
            while (cg.Read())
            {
                int cgid = (Int32)cg["Код_вантажу"];
                string cargoAbout = (string)cg["Опис_вантажу"];
                int cargoWeight = (Int32)cg["Вага_відправлення"];
                cargo1.Add(Tuple.Create(Convert.ToInt32(cgid), cargoAbout.ToString(),
cargoWeight.ToString()));
            }
            metroComboBox3.DataSource = cargo1;
            metroComboBox3.ValueMember = "Item1";
            metroComboBox3.DisplayMember = "Item2";
            cg.Close();
            await comand.ExecuteNonQueryAsync();
            if (mark == true)
            {
                btnAdd.Visible = true;
            }
            else
            {
                btnUpdate.Visible = true;
                if (a != null)
                {
                    sqlConnection = new SqlConnection(connectionString);
                    SqlDataReader sqlReader = null;
                    await sqlConnection.OpenAsync();
                    SqlCommand command = new SqlCommand("SELECT *
FROM[Акт_прийняття_вантажу] WHERE [ІД_Акт_прийняття_вантажу] = @id", sqlConnection); //
запрос на виборку даних
                    command.Parameters.AddWithValue("@id", a);
                    try
                    {
                        sqlReader = await command.ExecuteReaderAsync();
                        await sqlReader.ReadAsync();
                        dateTimePicker1.Value = Convert.ToDateTime(sqlReader["Дата"]);
//считує конкретні поля
                        metroComboBox1.Text = Convert.ToString(sqlReader["Код_Персоналу"]);
                        metroComboBox2.Text = Convert.ToString(sqlReader["ІД_Відправника"]);
```

```
                    metroComboBox3.Text = Convert.ToString(sqlReader["Код_вантажу"]);

                }
                catch (Exception ex)
                {
                    MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
MessageBoxButtons.OK, MessageBoxIcon.Error);
                }
                finally

                {
                    if (sqlReader != null)
                    {
                        sqlReader.Close();
                    }
                }
            }
        }
    }

    private async void btnAdd_Click(object sender, EventArgs e)
    {
        if (String.IsNullOrWhiteSpace(metroComboBox1.Text) != true &&
String.IsNullOrWhiteSpace(metroComboBox2.Text) != true &&
String.IsNullOrWhiteSpace(metroComboBox3.Text) != true)
        {
            sqlConnection = new SqlConnection(connectionString);
            SqlCommand command = new SqlCommand("INSERT INTO [Акт_прийняття_вантажу]
(Дата, Код_Персоналу, ІД_Відправника,Код_вантажу) VALUES
(@Дата,@Код_Персоналу,@ІД_Відправника,@Код_вантажу)", sqlConnection);
            try // запрос на додавання запису
            {
                await sqlConnection.OpenAsync();
                command.Parameters.AddWithValue("@Дата", dateTimePicker1.Value);
                command.Parameters.AddWithValue("@Код_Персоналу",
metroComboBox1.SelectedValue);
                command.Parameters.AddWithValue("@ІД_Відправника",
metroComboBox2.SelectedValue);
                command.Parameters.AddWithValue("@Код_вантажу",
metroComboBox3.SelectedValue);
            }

            catch (Exception ex)
            {
                MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
            finally
            {
                await command.ExecuteNonQueryAsync();
                this.Close();
            }
        }
        else
        {
            MessageBox.Show("Заповніть корректно дані!", "Попередження!",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
        }
    }

    private async void btnUpdate_Click(object sender, EventArgs e)
    {
        sqlConnection = new SqlConnection(connectionString);
        await sqlConnection.OpenAsync();
```

```
            SqlCommand command = new SqlCommand("UPDATE [Акт_прийняття_вантажу] SET
[Дата]=@Дата,  [Код_Персоналу]=@Код_Персоналу, [ІД_Відправника]=@ІД_Відправника,
[Код_вантажу]=@Код_вантажу WHERE [ІД_Акт_прийняття_вантажу]=@ID", sqlConnection);
            command.Parameters.AddWithValue("Id", a); // редагування поля де ID = a;
            try
            {
                if (String.IsNullOrWhiteSpace(metroComboBox1.Text) != true &&
String.IsNullOrWhiteSpace(metroComboBox2.Text) != true &&
String.IsNullOrWhiteSpace(metroComboBox3.Text) != true)
                { // захист

                    command.Parameters.AddWithValue("@Дата", dateTimePicker1.Value);
                    command.Parameters.AddWithValue("@Код_Персоналу",
metroComboBox1.SelectedValue);
                    command.Parameters.AddWithValue("@ІД_Відправника",
metroComboBox2.SelectedValue);
                    command.Parameters.AddWithValue("@Код_вантажу",
metroComboBox3.SelectedValue);

                }
                else
                {
                    MessageBox.Show("Дані було введено некорректно", "Помилка",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
            finally
            {

                await command.ExecuteNonQueryAsync();
                this.Close();
            }
        }
    }
}
```

Код форми frmBill.cs

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

namespace Дипломна
{
    public partial class frmBill : MetroFramework.Forms.MetroForm
    {
        string a;
        bool mark;
        SqlConnection sqlConnection;
        string connectionString = @"Data Source = (LocalDB)\MSSQLLocalDB; AttachDbFilename
=" + Application.StartupPath + @"\Kornievscki.mdf; Integrated Security = True";
        public frmBill(bool mark, string a)
        {
            this.mark = mark;
```

```
            this.a = a;
            InitializeComponent();
        }

        private async void frmBill_Load(object sender, EventArgs e)
        {
            DateTime myDateTime = dateTimePicker1.Value; // налаштування поля дата\час
            myDateTime.ToLongTimeString();
            dateTimePicker1.Format = DateTimePickerFormat.Custom;
            dateTimePicker1.CustomFormat = "dd-MM-yyyy";
            dateTimePicker1.MaxDate = DateTime.Now.Date;
            dateTimePicker1.Value = DateTime.Today;
            sqlConnection = new SqlConnection(connectionString);
            SqlCommand comand = new SqlCommand("SELECT * FROM [Персонал]", sqlConnection);
            SqlCommand transport = new SqlCommand("SELECT * FROM [Відправник]",
sqlConnection);
            SqlCommand cargo = new SqlCommand("SELECT * FROM [Вантаж]", sqlConnection);
            SqlCommand bill = new SqlCommand("SELECT * FROM [Акт_прийняття_вантажу]",
sqlConnection);
            await sqlConnection.OpenAsync();
            SqlDataReader dr = comand.ExecuteReader();
            var vis1 = new List<Tuple<int, string>>();
            while (dr.Read())
            {
                int sid = (Int32)dr["Код_Персоналу"];
                string spib = (string)dr["ПІБ"];
                vis1.Add(Tuple.Create(Convert.ToInt32(sid), spib.ToString()));
            }
            metroComboBox1.DataSource = vis1;
            metroComboBox1.ValueMember = "Item1";
            metroComboBox1.DisplayMember = "Item2";
            dr.Close();
            SqlDataReader druser = transport.ExecuteReader();
            var sender1 = new List<Tuple<int, string>>();
            while (druser.Read())
            {
                int sid = (Int32)druser["ІД_Відправника"];
                string sname = (string)druser["ПІБ_Відправника"];
                sender1.Add(Tuple.Create(Convert.ToInt32(sid), sname.ToString()));
            }
            metroComboBox2.DataSource = sender1;
            metroComboBox2.ValueMember = "Item1";
            metroComboBox2.DisplayMember = "Item2";
            druser.Close();
            SqlDataReader cg = cargo.ExecuteReader();
            var cargo1 = new List<Tuple<int, string, string>>();
            while (cg.Read())
            {

                int cgid = (Int32)cg["Код_вантажу"];
                string cargoAbout = (string)cg["Опис_вантажу"];
                int cargoWeight = (Int32)cg["Вага_відправлення"];
                cargo1.Add(Tuple.Create(Convert.ToInt32(cgid), cargoAbout.ToString(),
cargoWeight.ToString()));
            }
            metroComboBox3.DataSource = cargo1;
            metroComboBox3.ValueMember = "Item1";
            metroComboBox3.DisplayMember = "Item2";
            cg.Close();
            SqlDataReader bil = bill.ExecuteReader();
            while (bil.Read())
            {
                int intbil = (Int32)bil["ІД_Акт_прийняття_вантажу"];
```

```
                metroComboBox4.Items.Add(intbil);
            }
            bil.Close();
            await comand.ExecuteNonQueryAsync();
            if (mark == true)
            {
                btnAdd.Visible = true;
            }
            else
            {
                btnUpdate.Visible = true;
                if (a != null)
                {
                    sqlConnection = new SqlConnection(connectionString);
                    SqlDataReader sqlReader = null;
                    await sqlConnection.OpenAsync();
                    SqlCommand command = new SqlCommand("SELECT * FROM[Рахунок_фактура]
WHERE [Код_рахунку_фактури] = @id", sqlConnection); // запрос на виборку даних
                    command.Parameters.AddWithValue("@id", a);
                    try
                    {
                        sqlReader = await command.ExecuteReaderAsync();
                        await sqlReader.ReadAsync();
                        metroTextBox1.Text = Convert.ToString(sqlReader["Ціна_доставки"]);
//считує конкретні поля

                        dateTimePicker1.Text = Convert.ToString(sqlReader["Дата"]);


                    }
                    catch (Exception ex)
                    {
                        MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
MessageBoxButtons.OK, MessageBoxIcon.Error);
                    }
                    finally

                    {
                        if (sqlReader != null)
                        {
                            sqlReader.Close();
                        }
                    }
                }

            }
        }

        private async void btnAdd_Click(object sender, EventArgs e)
        {
            if (String.IsNullOrWhiteSpace(metroTextBox1.Text) != true &&
String.IsNullOrWhiteSpace(metroComboBox1.Text) != true &&
String.IsNullOrWhiteSpace(metroComboBox2.Text) != true &&
String.IsNullOrWhiteSpace(metroComboBox3.Text) != true &&
String.IsNullOrWhiteSpace(metroComboBox4.Text) != true)
            {
                sqlConnection = new SqlConnection(connectionString);
                SqlCommand command = new SqlCommand("INSERT INTO [Рахунок_фактура]
(Ціна_доставки, Код_Персоналу, ІД_Відправника,Дата, Код_вантажу,ІД_Акт_прийняття_вантажу)
VALUES
(@Ціна_доставки,@Код_Персоналу,@ІД_Відправника,@Дата,@Код_вантажу,@ІД_Акт_прийняття_вантажу)
", sqlConnection);
                try // запрос на додавання запису
                {
```

```
                    await sqlConnection.OpenAsync();
                    command.Parameters.AddWithValue("@Ціна_доставки", metroTextBox1.Text);
                    command.Parameters.AddWithValue("@Код_Персоналу",
metroComboBox1.SelectedValue);
                    command.Parameters.AddWithValue("@ІД_Відправника",
metroComboBox2.SelectedValue);
                    command.Parameters.AddWithValue("@Дата", dateTimePicker1.Value);
                    command.Parameters.AddWithValue("@Код_вантажу",
metroComboBox3.SelectedValue);
                    command.Parameters.AddWithValue("@ІД_Акт_прийняття_вантажу",
metroComboBox4.Text);
                }

                catch (Exception ex)
                {
                    MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
MessageBoxButtons.OK, MessageBoxIcon.Error);
                }
                finally
                {
                    await command.ExecuteNonQueryAsync();
                    this.Close();
                }
            }
            else
            {
                MessageBox.Show("Заповніть корректно дані!", "Попередження!",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
            }
        }

        private async void btnUpdate_Click(object sender, EventArgs e)
        {
            sqlConnection = new SqlConnection(connectionString);
            await sqlConnection.OpenAsync();
            SqlCommand command = new SqlCommand("UPDATE [Рахунок_фактура] SET
[Ціна_доставки]=@Ціна_доставки,  [Код_Персоналу]=@Код_Персоналу,
[ІД_Відправника]=@ІД_Відправника, [Дата]=@Дата, [Код_вантажу]=@Код_вантажу,
[ІД_Акт_прийняття_вантажу]=@ІД_Акт_прийняття_вантажу WHERE [Код_рахунку_фактури]=@ID",
sqlConnection);
            command.Parameters.AddWithValue("Id", a); // редагування поля де ID = a;
            try
            {
                if (String.IsNullOrWhiteSpace(metroTextBox1.Text) != true &&
String.IsNullOrWhiteSpace(metroComboBox1.Text) != true &&
String.IsNullOrWhiteSpace(metroComboBox2.Text) != true &&
String.IsNullOrWhiteSpace(metroComboBox3.Text) != true &&
String.IsNullOrWhiteSpace(metroComboBox4.Text) != true)
                { // захист

                    command.Parameters.AddWithValue("@Ціна_доставки", metroTextBox1.Text);
                    command.Parameters.AddWithValue("@Код_Персоналу",
metroComboBox1.SelectedValue);
                    command.Parameters.AddWithValue("@ІД_Відправника",
metroComboBox2.SelectedValue);
                    command.Parameters.AddWithValue("@Дата", dateTimePicker1.Value);
                    command.Parameters.AddWithValue("@Код_вантажу",
metroComboBox3.SelectedValue);
                    command.Parameters.AddWithValue("@ІД_Акт_прийняття_вантажу",
metroComboBox4.Text);

                }
                else
                {
```

```
                    MessageBox.Show("Дані було введено некорректно", "Помилка",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
            finally
            {

                await command.ExecuteNonQueryAsync();
                this.Close();
            }
        }

        private void metroTextBox1_KeyPress(object sender, KeyPressEventArgs e)
        {
            char number = e.KeyChar;
            if ((e.KeyChar <= 47 || e.KeyChar >= 58) && number != 8 && (e.KeyChar <= 39 ||
e.KeyChar >= 46) && number != 47 && number != 61) //калькулятор
            {
                e.Handled = true;
            }
        }
    }
}
```

Код форми frmCargo.cs інакше вікно вантаж

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

namespace Дипломна
{
    public partial class frmCargo : MetroFramework.Forms.MetroForm
    {
        string a;
        bool mark;
        SqlConnection sqlConnection;
        string connectionString = @"Data Source = (LocalDB)\MSSQLLocalDB; AttachDbFilename
=" + Application.StartupPath + @"\Kornievscki.mdf; Integrated Security = True";
        public frmCargo(bool mark, string a)
        {
            this.mark = mark;
            this.a = a;
            InitializeComponent();
        }

        private async void frmCargo_Load(object sender, EventArgs e)
        {
            // TODO: данная строка кода позволяет загрузить данные в таблицу
"database.Логістично_транспортна_накладна". При необходимости она может быть перемещена или
удалена.

            sqlConnection = new SqlConnection(connectionString);
```

```
            SqlCommand comand = new SqlCommand("SELECT * FROM
[Логістично_транспортна_накладна]", sqlConnection);
            await sqlConnection.OpenAsync();
            SqlDataReader dr = comand.ExecuteReader();
            while (dr.Read())
            {
                int sid = (Int32)dr["Код_накладної"];

                metroTextBox4.Items.Add(sid);
            }
            dr.Close();
            await comand.ExecuteNonQueryAsync();


            if (mark == true)
            {
                btnAdd.Visible = true;
            }
            else
            {
                btnUpdate.Visible = true;
                if (a != null)
                {
                    sqlConnection = new SqlConnection(connectionString);
                    SqlDataReader sqlReader = null;
                    await sqlConnection.OpenAsync();
                    SqlCommand command = new SqlCommand("SELECT * FROM[Вантаж] WHERE
[Код_вантажу] = @id", sqlConnection); // запрос на виборку даних
                    command.Parameters.AddWithValue("@id", a);
                    try
                    {
                        sqlReader = await command.ExecuteReaderAsync();
                        await sqlReader.ReadAsync();
                        metroTextBox1.Text =
Convert.ToString(sqlReader["Вага_відправлення"]); //считує конкретні поля
                        metroTextBox2.Text = Convert.ToString(sqlReader["Тип_вантажу"]);
                        metroTextBox3.Text = Convert.ToString(sqlReader["Опис_вантажу"]);
                        metroTextBox4.Text = Convert.ToString(sqlReader["Код_накладної"]);

                    }
                    catch (Exception ex)
                    {
                        MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
MessageBoxButtons.OK, MessageBoxIcon.Error);
                    }
                    finally

                    {
                        if (sqlReader != null)
                        {
                            sqlReader.Close();
                        }
                    }
                }
            }
        }

        private async void btnAdd_Click(object sender, EventArgs e)
        {
            if (String.IsNullOrWhiteSpace(metroTextBox1.Text) != true &&
String.IsNullOrWhiteSpace(metroTextBox2.Text) != true &&
String.IsNullOrWhiteSpace(metroTextBox3.Text) != true &&
String.IsNullOrWhiteSpace(metroTextBox4.Text) != true)
            {
```

```
                sqlConnection = new SqlConnection(connectionString);
                SqlCommand command = new SqlCommand("INSERT INTO [Вантаж]
(Вага_відправлення, Тип_вантажу, Опис_вантажу,Код_накладної) VALUES
(@Вага_відправлення,@Тип_вантажу,@Опис_вантажу,@Код_накладної)", sqlConnection);
                try // запрос на додавання запису
                {
                    await sqlConnection.OpenAsync();
                    command.Parameters.AddWithValue("@Вага_відправлення",
metroTextBox1.Text);
                    command.Parameters.AddWithValue("@Тип_вантажу", metroTextBox2.Text);
                    command.Parameters.AddWithValue("@Опис_вантажу", metroTextBox3.Text);
                    command.Parameters.AddWithValue("@Код_накладної", metroTextBox4.Text);
                }

                catch (Exception ex)
                {
                    MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
MessageBoxButtons.OK, MessageBoxIcon.Error);
                }
                finally
                {
                    await command.ExecuteNonQueryAsync();
                    this.Close();
                }
            }
            else
            {
                MessageBox.Show("Заповніть корректно дані!", "Попередження!",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
            }
        }

        private void вантажBindingNavigatorSaveItem_Click(object sender, EventArgs e)
        {


        }

        private void вантажBindingNavigatorSaveItem_Click_1(object sender, EventArgs e)
        {


        }

        private async void btnUpdate_Click(object sender, EventArgs e)
        {
            sqlConnection = new SqlConnection(connectionString);
            await sqlConnection.OpenAsync();
            SqlCommand command = new SqlCommand("UPDATE [Вантаж] SET
[Вага_відправлення]=@Вага_відправлення,  [Тип_вантажу]=@Тип_вантажу,
[Опис_вантажу]=@Опис_вантажу, [Код_накладної]=@Код_накладної WHERE [Код_вантажу]=@ID",
sqlConnection);
            command.Parameters.AddWithValue("Id", a); // редагування поля де ID = a;
            try
            {
                if (String.IsNullOrWhiteSpace(metroTextBox1.Text) != true &&
String.IsNullOrWhiteSpace(metroTextBox2.Text) != true &&
String.IsNullOrWhiteSpace(metroTextBox3.Text) != true)
                { // захист

                    command.Parameters.AddWithValue("@Вага_відправлення",
metroTextBox1.Text);
                    command.Parameters.AddWithValue("@Тип_вантажу", metroTextBox2.Text);
                    command.Parameters.AddWithValue("@Опис_вантажу", metroTextBox3.Text);
```

```
                    command.Parameters.AddWithValue("@Код_накладної", metroTextBox4.Text);

                }
                else
                {
                    MessageBox.Show("Дані було введено некорректно", "Помилка",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
            finally
            {

                await command.ExecuteNonQueryAsync();
                this.Close();
            }
        }

        private void metroTextBox1_KeyPress(object sender, KeyPressEventArgs e)
        {
            char number = e.KeyChar;
            if ((e.KeyChar <= 47 || e.KeyChar >= 58) && number != 8 && (e.KeyChar <= 39 ||
e.KeyChar >= 46) && number != 47 && number != 61)
            {
                e.Handled = true;
            }
        }
    }
}
```

Код форми frmTransport.cs інакше вікно Транспорт

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

namespace Дипломна
{
    public partial class frmTransport : MetroFramework.Forms.MetroForm
    {
        string a;
        bool mark;
        SqlConnection sqlConnection;
        string connectionString = @"Data Source = (LocalDB)\MSSQLLocalDB; AttachDbFilename
=" + Application.StartupPath + @"\Kornievscki.mdf; Integrated Security = True";
        public frmTransport(bool mark, string a)
        {
            this.mark = mark;
            this.a = a;
            InitializeComponent();
        }

        private async void frmFizOsoba_Load(object sender, EventArgs e)
        {
```

```
            if (mark == true)
            {
                btnAdd.Visible = true;
            }
            else
            {
                btnUpdate.Visible = true;
                if (a != null)
                {
                    sqlConnection = new SqlConnection(connectionString);
                    SqlDataReader sqlReader = null;
                    await sqlConnection.OpenAsync();
                    SqlCommand command = new SqlCommand("SELECT * FROM[Транспорт] WHERE
[Код_Транспорту] = @id", sqlConnection); // запрос на виборку даних
                    command.Parameters.AddWithValue("@id", a);
                    try
                    {
                        sqlReader = await command.ExecuteReaderAsync();
                        await sqlReader.ReadAsync();
                        metroTextBox1.Text = Convert.ToString(sqlReader["Модель"]); //считує
конкретні поля
                        metroTextBox2.Text = Convert.ToString(sqlReader["Рік_випуску"]);
                        metroTextBox3.Text =
Convert.ToString(sqlReader["Вантажопідйомність"]);

                    }
                    catch (Exception ex)
                    {
                        MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
MessageBoxButtons.OK, MessageBoxIcon.Error);
                    }
                    finally

                    {
                        if (sqlReader != null)
                        {
                            sqlReader.Close();
                        }
                    }
                }
            }
        }

        private async void btnAdd_Click(object sender, EventArgs e)
        {
            if (String.IsNullOrWhiteSpace(metroTextBox1.Text) != true &&
String.IsNullOrWhiteSpace(metroTextBox2.Text) != true &&
String.IsNullOrWhiteSpace(metroTextBox3.Text) != true)
            {
                sqlConnection = new SqlConnection(connectionString);
                SqlCommand command = new SqlCommand("INSERT INTO [Транспорт] (Модель,
Рік_випуску, Вантажопідйомність) VALUES (@Модель,@Рік_випуску,@Вантажопідйомність)",
sqlConnection);
                try // запрос на додавання запису
                {

                    await sqlConnection.OpenAsync();


                    command.Parameters.AddWithValue("@Модель", metroTextBox1.Text);
                    command.Parameters.AddWithValue("@Рік_випуску", metroTextBox2.Text);
                    command.Parameters.AddWithValue("@Вантажопідйомність",
metroTextBox3.Text);
```

```
                }

                catch (Exception ex)
                {
                    MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
MessageBoxButtons.OK, MessageBoxIcon.Error);
                }
                finally
                {
                    await command.ExecuteNonQueryAsync();
                    this.Close();
                }
            }
            else
            {
                MessageBox.Show("Заповніть корректно дані!", "Попередження!",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
            }
        }

        private async void btnUpdate_Click(object sender, EventArgs e)
        {
            sqlConnection = new SqlConnection(connectionString);
            await sqlConnection.OpenAsync();
            SqlCommand command = new SqlCommand("UPDATE [Транспорт] SET [Модель]=@Модель,
[Рік_випуску]=@Рік_випуску, [Вантажопідйомність]=@Вантажопідйомність WHERE
[Код_транспорту]=@ID", sqlConnection);
            command.Parameters.AddWithValue("Id", a); // редагування поля де ID = a;
            try
            {
                if (String.IsNullOrWhiteSpace(metroTextBox1.Text) != true &&
String.IsNullOrWhiteSpace(metroTextBox2.Text) != true &&
String.IsNullOrWhiteSpace(metroTextBox3.Text) != true)
                { // захист

                    command.Parameters.AddWithValue("@Модель", metroTextBox1.Text);
                    command.Parameters.AddWithValue("@Рік_випуску", metroTextBox2.Text);
                    command.Parameters.AddWithValue("@Вантажопідйомність",
metroTextBox3.Text);

                }
                else
                {
                    MessageBox.Show("Дані було введено некорректно", "Помилка",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
            finally
            {

                await command.ExecuteNonQueryAsync();
                this.Close();
            }
        }

        private void metroTextBox2_KeyPress(object sender, KeyPressEventArgs e)
        {
            char number = e.KeyChar;
```

```
            if ((e.KeyChar <= 47 || e.KeyChar >= 58) && number != 8 && (e.KeyChar <= 39 ||
e.KeyChar >= 46) && number != 47 && number != 61) //калькулятор
            {
                e.Handled = true;
            }
        }

        private void metroTextBox3_KeyPress(object sender, KeyPressEventArgs e)
        {
            char number = e.KeyChar;
            if ((e.KeyChar <= 47 || e.KeyChar >= 58) && number != 8 && (e.KeyChar <= 39 ||
e.KeyChar >= 46) && number != 47 && number != 61) //калькулятор
            {
                e.Handled = true;
            }
        }
    }
}
```

Код форми frmInvoice.cs інакше вікно Накладна

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

namespace Дипломна
{
    public partial class frmInvoice : MetroFramework.Forms.MetroForm
    {
        string a;
        bool mark;
        SqlConnection sqlConnection;
        string connectionString = @"Data Source = (LocalDB)\MSSQLLocalDB; AttachDbFilename
=" + Application.StartupPath + @"\Kornievscki.mdf; Integrated Security = True";
        public frmInvoice(bool mark, string a)
        {
            this.mark = mark;
            this.a = a;
            InitializeComponent();
        }

        private async void frmInvoice_Load(object sender, EventArgs e)
        {
            sqlConnection = new SqlConnection(connectionString);
            SqlCommand comand = new SqlCommand("SELECT * FROM [Персонал]", sqlConnection);
            SqlCommand transport = new SqlCommand("SELECT * FROM [Транспорт]",
sqlConnection);
            await sqlConnection.OpenAsync();
            SqlDataReader dr = comand.ExecuteReader();
            var vis1 = new List<Tuple<int, string>>();
            while (dr.Read())
            {
                int sid = (Int32)dr["Код_Персоналу"];
                string spib = (string)dr["ПІБ"];
                vis1.Add(Tuple.Create(Convert.ToInt32(sid), spib.ToString()));
            }
            metroComboBox1.DataSource = vis1;
            metroComboBox1.ValueMember = "Item1";
            metroComboBox1.DisplayMember = "Item2";
```

```
            dr.Close();
            SqlDataReader druser = transport.ExecuteReader();
            var vis2 = new List<Tuple<int, string>>();
            while (druser.Read())
            {

                int sname = (Int32)druser["Код_транспорту"];
                string svehicle = (string)druser["Модель"];
                vis2.Add(Tuple.Create(Convert.ToInt32(sname), svehicle.ToString()));

            }
            metroComboBox2.DataSource = vis2;
            metroComboBox2.ValueMember = "Item1";
            metroComboBox2.DisplayMember = "Item2";
            druser.Close();
            await comand.ExecuteNonQueryAsync();
            if (mark == true)
            {
                btnAdd.Visible = true;
            }
            else
            {
                btnUpdate.Visible = true;


            }
        }

        private async void btnAdd_Click(object sender, EventArgs e)
        {
            if (String.IsNullOrWhiteSpace(metroComboBox1.Text) != true &&
String.IsNullOrWhiteSpace(metroComboBox2.Text) != true)
            {
                sqlConnection = new SqlConnection(connectionString);
                SqlCommand command = new SqlCommand("INSERT INTO
[Логістично_транспортна_накладна] (Код_Персоналу, Код_транспорту) VALUES
(@Код_Персоналу,@Код_транспорту)", sqlConnection);
                try // запрос на додавання запису
                {
                    await sqlConnection.OpenAsync();
                    command.Parameters.AddWithValue("@Код_Персоналу",
metroComboBox1.SelectedValue);
                    command.Parameters.AddWithValue("@Код_транспорту",
metroComboBox2.SelectedValue);
                }

                catch (Exception ex)
                {
                    MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
MessageBoxButtons.OK, MessageBoxIcon.Error);
                }
                finally
                {
                    await command.ExecuteNonQueryAsync();
                    this.Close();
                }
            }
            else
            {
                MessageBox.Show("Заповніть корректно дані!", "Попередження!",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
            }
        }
```

```
        private async void btnUpdate_Click(object sender, EventArgs e)
        {
            sqlConnection = new SqlConnection(connectionString);
            await sqlConnection.OpenAsync();
            SqlCommand command = new SqlCommand("UPDATE [Логістично_транспортна_накладна]
SET [Код_Персоналу]=@Код_Персоналу,  [Код_транспорту]=@Код_транспорту WHERE
[Код_накладної]=@ID", sqlConnection);
            command.Parameters.AddWithValue("Id", a); // редагування поля де ID = a;
            try
            {
                if (String.IsNullOrWhiteSpace(metroComboBox1.Text) != true &&
String.IsNullOrWhiteSpace(metroComboBox2.Text) != true)
                { // захист

                    command.Parameters.AddWithValue("@Код_Персоналу",
metroComboBox1.SelectedValue);
                    command.Parameters.AddWithValue("@Код_транспорту",
metroComboBox2.SelectedValue);

                }
                else
                {
                    MessageBox.Show("Дані було введено некорректно", "Помилка",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
            finally
            {

                await command.ExecuteNonQueryAsync();
                this.Close();
            }
        }
    }
}
```

Код форми frmPersonel.cs інакше вікно персонал

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

namespace Дипломна
{
    public partial class frmPersonel : MetroFramework.Forms.MetroForm
    {
        string a;
        bool mark;
        SqlConnection sqlConnection;
        string connectionString = @"Data Source = (LocalDB)\MSSQLLocalDB; AttachDbFilename
=" + Application.StartupPath + @"\Kornievscki.mdf; Integrated Security = True";
        public frmPersonel(bool mark, string a)
        {
            this.mark = mark;
```

```
            this.a = a;
            InitializeComponent();
        }

        private async void frmPersonel_Load(object sender, EventArgs e)
        {
            if (mark == true)
            {
                btnAdd.Visible = true;
            }
            else
            {
                btnUpdate.Visible = true;
                if (a != null)
                {
                    sqlConnection = new SqlConnection(connectionString);
                    SqlDataReader sqlReader = null;
                    await sqlConnection.OpenAsync();
                    SqlCommand command = new SqlCommand("SELECT * FROM[Персонал] WHERE
[Код_Персоналу] = @id", sqlConnection); // запрос на виборку даних
                    command.Parameters.AddWithValue("@id", a);
                    try
                    {
                        sqlReader = await command.ExecuteReaderAsync();
                        await sqlReader.ReadAsync();
                        metroTextBox1.Text = Convert.ToString(sqlReader["ПІБ"]); //считує
конкретні поля
                        metroTextBox2.Text = Convert.ToString(sqlReader["Посада"]);
                        metroTextBox3.Text = Convert.ToString(sqlReader["Телефон"]);

                    }
                    catch (Exception ex)
                    {
                        MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
MessageBoxButtons.OK, MessageBoxIcon.Error);
                    }
                    finally

                    {
                        if (sqlReader != null)
                        {
                            sqlReader.Close();
                        }
                    }
                }
            }

        }

        private async void btnUpdate_Click(object sender, EventArgs e)
        {
            sqlConnection = new SqlConnection(connectionString);
            await sqlConnection.OpenAsync();
            SqlCommand command = new SqlCommand("UPDATE [Персонал] SET [ПІБ]=@ПІБ,
[Посада]=@Посада, [Телефон]=@Телефон WHERE [Код_Персоналу]=@ID", sqlConnection);
            command.Parameters.AddWithValue("Id", a); // редагування поля де ID = a;
            try
            {
                if (String.IsNullOrWhiteSpace(metroTextBox1.Text) != true &&
String.IsNullOrWhiteSpace(metroTextBox2.Text) != true &&
String.IsNullOrWhiteSpace(metroTextBox3.Text) != true)
                { // захист

                    command.Parameters.AddWithValue("@ПІБ", metroTextBox1.Text);
```

```
                    command.Parameters.AddWithValue("@Посада", metroTextBox2.Text);
                    command.Parameters.AddWithValue("@Телефон", metroTextBox3.Text);

                }
                else
                {
                    MessageBox.Show("Дані було введено некорректно", "Помилка",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
            finally
            {

                await command.ExecuteNonQueryAsync();
                this.Close();
            }
        }

        private async void btnAdd_Click(object sender, EventArgs e)
        {
            if (String.IsNullOrWhiteSpace(metroTextBox1.Text) != true &&
String.IsNullOrWhiteSpace(metroTextBox2.Text) != true &&
String.IsNullOrWhiteSpace(metroTextBox3.Text) != true)
            {
                sqlConnection = new SqlConnection(connectionString);
                SqlCommand command = new SqlCommand("INSERT INTO [Персонал] (ПІБ, Посада,
Телефон) VALUES (@ПІБ,@Посада,@Телефон)", sqlConnection);
                try // запрос на додавання запису
                {

                    await sqlConnection.OpenAsync();


                    command.Parameters.AddWithValue("@ПІБ", metroTextBox1.Text);
                    command.Parameters.AddWithValue("@Посада", metroTextBox2.Text);
                    command.Parameters.AddWithValue("@Телефон", metroTextBox3.Text);

                }

                catch (Exception ex)
                {
                    MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
MessageBoxButtons.OK, MessageBoxIcon.Error);
                }
                finally
                {
                    await command.ExecuteNonQueryAsync();
                    this.Close();
                }
            }
            else
            {
                MessageBox.Show("Заповніть корректно дані!", "Попередження!",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
            }
        }

        private void metroTextBox3_KeyPress(object sender, KeyPressEventArgs e)
        {
```

```
            char number = e.KeyChar;
            if ((e.KeyChar <= 47 || e.KeyChar >= 58) && number != 8 && (e.KeyChar <= 39 ||
e.KeyChar >= 46) && number != 47 && number != 61) //калькулятор
            {
                e.Handled = true;
            }
        }
    }
}
```

Код форми frmSender.cs інакше вікно Відправник

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

namespace Дипломна
{
    public partial class frmSender : MetroFramework.Forms.MetroForm
    {
        string a;
        bool mark;
        SqlConnection sqlConnection;
        string connectionString = @"Data Source = (LocalDB)\MSSQLLocalDB; AttachDbFilename
=" + Application.StartupPath + @"\Kornievscki.mdf; Integrated Security = True";
        public frmSender(bool mark, string a)
        {
            this.mark = mark;
            this.a = a;
            InitializeComponent();
        }

        private async void frmSender_Load(object sender, EventArgs e)
        {
            if (mark == true)
            {
                btnAdd.Visible = true;
            }
            else
            {
                btnUpdate.Visible = true;
                if (a != null)
                {
                    sqlConnection = new SqlConnection(connectionString);
                    SqlDataReader sqlReader = null;
                    await sqlConnection.OpenAsync();
                    SqlCommand command = new SqlCommand("SELECT * FROM[Відправник] WHERE
[ІД_Відправника] = @id", sqlConnection); // запрос на виборку даних
                    command.Parameters.AddWithValue("@id", a);
                    try
                    {
                        sqlReader = await command.ExecuteReaderAsync();
                        await sqlReader.ReadAsync();
                        metroTextBox1.Text =
Convert.ToString(sqlReader["Супровідна_документація"]); //считує конкретні поля
                        metroTextBox2.Text =
Convert.ToString(sqlReader["Електронна_пошта"]);
                        metroTextBox3.Text = Convert.ToString(sqlReader["ПІБ_Відправника"]);
```

```
                    }
                    catch (Exception ex)
                    {
                        MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
MessageBoxButtons.OK, MessageBoxIcon.Error);
                    }
                    finally

                    {
                        if (sqlReader != null)
                        {
                            sqlReader.Close();
                        }
                    }
                }

            }
        }

        private async void btnAdd_Click(object sender, EventArgs e)
        {
            if (String.IsNullOrWhiteSpace(metroTextBox1.Text) != true &&
String.IsNullOrWhiteSpace(metroTextBox2.Text) != true &&
String.IsNullOrWhiteSpace(metroTextBox3.Text) != true)
            {
                sqlConnection = new SqlConnection(connectionString);
                SqlCommand command = new SqlCommand("INSERT INTO [Відправник]
(Супровідна_документація, Електронна_пошта, ПІБ_Відправника) VALUES
(@Супровідна_документація,@Електронна_пошта,@ПІБ_Відправника)", sqlConnection);
                try // запрос на додавання запису
                {
                    await sqlConnection.OpenAsync();
                    command.Parameters.AddWithValue("@Супровідна_документація",
metroTextBox1.Text);
                    command.Parameters.AddWithValue("@Електронна_пошта",
metroTextBox2.Text);
                    command.Parameters.AddWithValue("@ПІБ_Відправника", metroTextBox3.Text);
                }

                catch (Exception ex)
                {
                    MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
MessageBoxButtons.OK, MessageBoxIcon.Error);
                }
                finally
                {
                    await command.ExecuteNonQueryAsync();
                    this.Close();
                }
            }
            else
            {
                MessageBox.Show("Заповніть корректно дані!", "Попередження!",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
            }
        }

        private async void btnUpdate_Click(object sender, EventArgs e)
        {
            sqlConnection = new SqlConnection(connectionString);
            await sqlConnection.OpenAsync();
            SqlCommand command = new SqlCommand("UPDATE [Відправник] SET
[Супровідна_документація]=@Супровідна_документація,  [Електронна_пошта]=@Електронна_пошта,
[ПІБ_Відправника]=@ПІБ_Відправника WHERE [ІД_Відправника]=@ID", sqlConnection);
```

```
            command.Parameters.AddWithValue("Id", a); // редагування поля де ID = a;
            try
            {
                if (String.IsNullOrWhiteSpace(metroTextBox1.Text) != true &&
String.IsNullOrWhiteSpace(metroTextBox2.Text) != true &&
String.IsNullOrWhiteSpace(metroTextBox3.Text) != true)
                { // захист
                    command.Parameters.AddWithValue("@Супровідна_документація",
metroTextBox1.Text);
                    command.Parameters.AddWithValue("@Електронна_пошта",
metroTextBox2.Text);
                    command.Parameters.AddWithValue("@ПІБ_Відправника", metroTextBox3.Text);
                }
                else
                {
                    MessageBox.Show("Дані було введено некорректно", "Помилка",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(),
MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
            finally
            {
                await command.ExecuteNonQueryAsync();
                this.Close();
            }
        }
    }
}
```

Код форми frmStatistics.cs інакше вікно Статистика

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

namespace Дипломна
{
    public partial class frmStatistics : MetroFramework.Forms.MetroForm
    {
        string connectionString = @"Data Source = (LocalDB)\MSSQLLocalDB; AttachDbFilename
=" + Application.StartupPath + @"\Kornievscki.mdf; Integrated Security = True";
        SqlConnection connection;
        string connectionStringUser = @"Data Source = (LocalDB)\MSSQLLocalDB;
AttachDbFilename =" + Application.StartupPath + @"\Database.mdf; Integrated Security =
True";
        SqlConnection connectionUser;
        DataSet ds = new DataSet();
        public frmStatistics()
        {
            InitializeComponent();
        }

        private void frmStatistics_Load(object sender, EventArgs e)
        {
            connection = new SqlConnection(connectionString);
```

```
            SqlCommand comm = new SqlCommand("SELECT AVG(Ціна_доставки) FROM
[Рахунок_фактура] ", connection);
            connection.Open();
            object result = comm.ExecuteScalar();
            txtAVG.Text = "Середня ціна доставки: " + Convert.ToString(result);
            SqlCommand command = new SqlCommand("SELECT COUNT(Код_рахунку_фактури) FROM
[Рахунок_фактура] ", connection);
             result = command.ExecuteScalar();
            txtCountOrders.Text = "Кількість відправлень : " + Convert.ToString(result);
            SqlCommand summa = new SqlCommand("SELECT SUM(Ціна_доставки) FROM
[Рахунок_фактура] ", connection);
            result = summa.ExecuteScalar();
            txtSum.Text = "Виручка : " + Convert.ToString(result);
            SqlCommand countavto = new SqlCommand("SELECT COUNT(Код_транспорту) FROM
[Транспорт] ", connection);
            result = countavto.ExecuteScalar();
            txtCountVehicle.Text = "Кількість авто: " + Convert.ToString(result);
            SqlCommand avgtons = new SqlCommand("SELECT AVG(Вантажопідйомність) FROM
[Транспорт] ", connection);
            result = avgtons.ExecuteScalar();
            txtAVGtons.Text = "Середній тоннаж транспорту: " + Convert.ToString(result);
            SqlCommand avgyears = new SqlCommand("SELECT AVG(Рік_випуску) FROM [Транспорт]
", connection);
            result = avgyears.ExecuteScalar();
            txtAVGyear.Text = "Середній рік випуску транспорту: " +
Convert.ToString(result);
            connectionUser = new SqlConnection(connectionStringUser);
            SqlCommand countUser = new SqlCommand("SELECT COUNT(Id) FROM [users] ",
connectionUser);
            connectionUser.Open();
            result = countUser.ExecuteScalar();
            txtCountUsers.Text = "Кількість облікових записів в системі: " +
Convert.ToString(result);
            connectionUser.Close();
            SqlCommand avgmass = new SqlCommand("SELECT AVG(Вага_відправлення) FROM [Вантаж]
", connection);
            result = avgmass.ExecuteScalar();
            txtAVGmass.Text = "Середня вага вантажу: " + Convert.ToString(result) + " кг";


            connection.Close();
        }

    }
}
```